Modeling Word Forms Using Latent Underlying Morphs and Phonology

Ryan Cotterell and **Nanyun Peng** and **Jason Eisner**
Department of Computer Science, Johns Hopkins University
{ryan.cotterell,npeng1,eisner}@jhu.edu

Abstract

The observed pronunciations or spellings of words are often explained as arising from the "underlying forms" of their morphemes. These forms are latent strings that linguists try to reconstruct by hand. We propose to reconstruct them automatically at scale, enabling generalization to new words. Given some surface word types of a concatenative language along with the abstract morpheme sequences that they express, we show how to recover consistent underlying forms for these morphemes, together with the (stochastic) phonology that maps each concatenation of underlying forms to a surface form. Our technique involves loopy belief propagation in a natural directed graphical model whose variables are unknown strings and whose conditional distributions are encoded as finite-state machines with trainable weights. We define training and evaluation paradigms for the task of surface word prediction, and report results on subsets of 7 languages.

1 Introduction

How is plurality expressed in English? Comparing `cats` ([kæts]), `dogs` ([dɔgz]), and `quizzes` ([kwɪzɪz]), the plural morpheme evidently has at least three pronunciations ([s], [z], [ɪz]) and at least two spellings (`-s` and `-es`). Also, considering singular `quiz`, perhaps the "short exam" morpheme has multiple spellings (`quizz-`, `quiz-`).

Fortunately, languages are systematic. The realization of a morpheme may vary by context but is largely predictable from context, in a way that generalizes across morphemes. In fact, generative linguists traditionally posit that each morpheme of a language has a *single* representation shared across all contexts (Jakobson, 1948; Kenstowicz and Kisseberth, 1979, chapter 6). However, this string is a latent variable that is never observed. Variation appears when the *phonology* of the language maps these *underlying representations* (URs)—in context—to *surface representations* (SRs) that may be easier to pronounce. The phonology is usually described by a grammar that may consist of either rewrite rules (Chomsky and Halle, 1968) or ranked constraints (Prince and Smolensky, 2004).

We will review this framework in section 2. The upshot is that the observed words in a language are supposed to be explainable in terms of a smaller underlying lexicon of morphemes, plus a phonology. Our goal in this paper is to recover the lexicon and phonology (enabling generalization to new words). This is difficult even when we are told which morphemes are expressed by each word, because the unknown underlying forms of the morphemes must cooperate properly with one another and with the unknown phonological rules to produce the observed results. Because of these interactions, we must reconstruct everything jointly. We regard this as a problem of inference in a directed graphical model, as sketched in Figure 1.

This is a natural problem for computational linguistics. Phonology students are trained to puzzle out solutions for small datasets by hand. Children apparently solve it at the scale of an entire language. Phonologists would like to have grammars for many languages, not just to study each language but also to understand universal principles and differences among related languages. Automatic procedures would recover such grammars. They would also allow comprehensive evaluation and comparison of different phonological theories (i.e., what inductive biases are useful?), and would suggest models of human language learning.

Solving this problem is also practically important for NLP. What we recover is a *model* that can generate and help analyze novel word forms,[1] which abound in morphologically complex languages. Our approach is designed to model sur-

[1] An analyzer would require a prior over possible analyses. Our present model defines just the corresponding likelihoods, i.e., the probability of the observed word *given* each analysis.

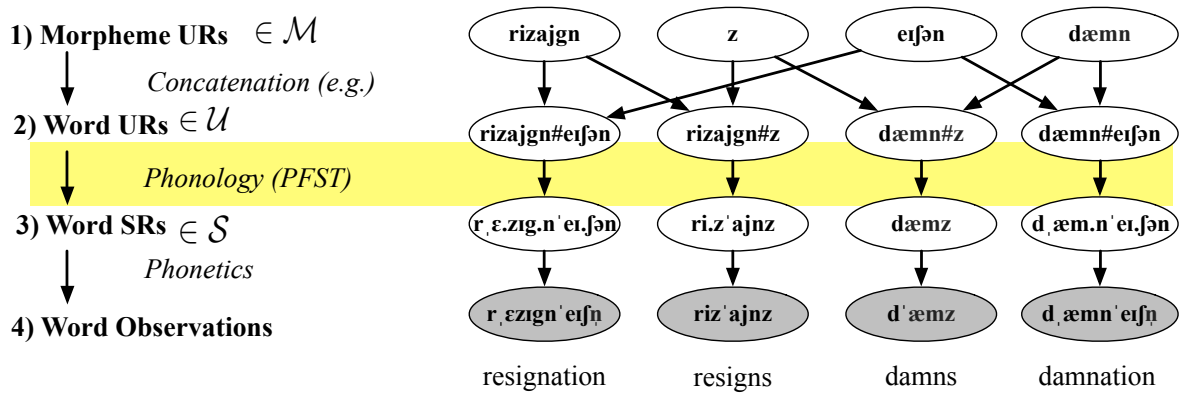

Figure 1: Our model as a Bayesian network, in which surface forms arise from applying phonology to a concatenation of underlying forms. Shaded nodes show the observed surface forms for four words: *resignation*, *resigns*, *damns*, and *damnation*. The graphical model encodes their morphological relationships using latent forms. Each morpheme UR at layer 1 is generated by the lexicon model M_ϕ (a probabilistic finite-state automaton). These are concatenated into various word URs at layer 2. Each SR at layer 3 is generated using the phonology model S_θ (a probabilistic finite-state transducer). Layer 4 derives observable phonetic forms from layer 3. This deletes unpronounced symbols such as syllable boundaries, and translates the phonemes into an observed phonetic, articulatory, or acoustic representation. However, our present paper simply merges layers 3 and 4: our layer 3 does not currently make use of any unpronounced symbols (e.g., syllable boundaries) and we observe it directly.

face *pronunciations* (as needed for text-to-speech and ASR). It might also be applied in practice to model surface *spellings* (as needed for MT on text). Good morphological analysis has been used to improve NLP tasks such as machine translation, parsing, and NER (Fraser et al., 2012; Hohensee and Bender, 2012; Yeniterzi, 2011).

Using loopy belief propagation, this paper attacks larger-scale learning problems than prior work on this task (section 8). We also develop a new evaluation paradigm that examines how well an inferred grammar predicts *held-out* SRs. Unlike previous algorithms, we do not pre-restrict the possible URs for each morpheme to a small or structured finite set, but use weighted finite-state machines to reason about the infinite space of all strings. Our graphical model captures the standard assumption that each morpheme has a single UR, unlike some probabilistic learners. However, we do not try to learn traditional ordered rules or constraint rankings like previous methods. We just search directly for a probabilistic finite-state transducer that captures likely UR-to-SR mappings.

2 Formal Framework

We urge the reader to begin by examining Figure 1, which summarizes our modeling approach through an example. The upcoming sections then give a formal treatment with details and discussion. Section 2 describes the random variables in Figure 1's Bayesian network, while section 3 describes its conditional probability distributions. Sections 4–5 give inference and learning methods.

A **morpheme** is a lexical entry that pairs form with content (Saussure, 1916). Its *form* is a **morph**—a string of phonemes. Its *content* is a bundle of syntactic and/or semantic properties.[2]

Note that in this paper, we are nonstandardly using "morph" to denote an *underlying* form. We assume that all underlying and surface representations can be encoded as strings, over respective alphabets Σ_u and Σ_s. This would be possible even for autosegmental representations (Kornai, 1995).

A language's phonological system thus consists of the following components. We denote each important set by a calligraphic letter. We use the corresponding uppercase letter to denote a function to that set, the corresponding lowercase letter as a variable that ranges over the set's elements, and a distinguished typeface for specific elements.

- $\mathcal{A}$ is a set of abstract morphemes such as *quiz* and *plural*. These are atoms, not strings.
- $\mathcal{M} = \Sigma_u^*$ is the space of possible morphs: concrete UR strings such as /kwɪz/ or /z/.
- $M : \mathcal{A} \to \mathcal{M}$ is the lexicon that maps each morpheme a to an underlying morph $m = M(a)$. We will find $M(a)$ for each a.
- $\mathcal{U} = (\Sigma_u \cup \{\#\})^*$ is the space of underlying representations for words, such as /kwɪz#z/.
- $U : \mathcal{M}^* \to \mathcal{U}$ combines morphs. A word is specified by a sequence of morphemes $\vec{a} = a_1, a_2, \ldots$, with concrete forms $m_i = M(a_i)$.

[2]This paper does not deal with the content. However, note that a *single* morpheme might specify a conjunction or disjunction of *multiple* properties, leading to morphological phenomena such as fusion, suppletion, or syncretism.

That word's underlying form is then $u = U(m_1, m_2, \ldots) \in \mathcal{U}$.

- $\mathcal{S} = \Sigma_s^*$ is the space of surface representations for words, such as [kwɪzɪz].
- $S : \mathcal{U} \rightarrow \mathcal{S}$ is the phonology. It maps an underlying form u to its surface form s. We will find this function S along with M.

We assume in this paper that U simply concatenates the sequence of morphs, separating them by the morph boundary symbol #: $u = U(m_1, m_2, \ldots) = m_1\#m_2\#\cdots$. However, see section 4.3 for generalizations.

The overall system serves to map an (abstract) morpheme sequence $\vec{a} \in \mathcal{A}^*$ to a surface word $s \in \mathcal{S}$. Crucially, S acts on the underlying form u of the entire word, not one morph at a time. Hence its effect on a morph may depend on context, as we saw for English pluralization. For example, S(/kwɪz#s/) = [kwɪzɪz]—or if we were to apply our model to orthography, S(/quiz#s/) = [quizzes]. S produces a single well-formed surface form, which is not arbitrarily segmented as [quiz-zes] or [quizz-es] or [quizze-s].

3 Probability Model

Our goal is to reconstruct the lexicon M and morphophonology S for a given language. We therefore define prior probability distributions over them. (We assume $\Sigma_u, \Sigma_s, \mathcal{A}, U$ are given.)

For each morpheme $a \in \mathcal{A}$, we model the morph $M(a) \in \mathcal{M}$ as an IID sample from a probability distribution $M_\phi(m)$.[3] This model describes what sort of underlying forms appear in the language's lexicon.

The phonology is probabilistic in a similar way. For a word with underlying form $u \in \mathcal{U}$, we presume that the surface form $S(u)$ is a sample from a conditional distribution $S_\theta(s \mid u)$. This single sample appears in the lexical entry of the word *type* and is reused for all tokens of that word.

The parameter vectors ϕ and θ are specific to the language being generated. Thus, under our generative story, a language is created as follows:

1. Sample ϕ and θ from priors (see section 3.4).
2. For each $a \in \mathcal{A}$, sample $M(a) \sim M_\phi$.
3. Whenever a new abstract word $\vec{a} = a_1, a_2 \cdots$ must be pronounced for the first time, construct u as described in section 2, and sample $S(u) \sim S_\theta(\cdot \mid u)$. Reuse this $S(u)$ in future.

[3] See section 3.3 for a generalization to $M_\phi(m \mid a)$.

Note that we have not specified a probability distribution over abstract words $\vec{a}$, since in this paper, these sequences will always be observed. Such a distribution might be influenced by the semantic and syntactic content of the morphemes. We would need it to recover the abstract words if they were *unobserved*, e.g., when analyzing novel word forms or attempting unsupervised training.

3.1 Discussion: Why probability?

A language's lexicon M and morphophonology S are deterministic, in that each morpheme has a single underlying form and each word has a single surface form. The point of the language-specific distributions M_ϕ and S_θ is to aid recovery of these forms by capturing *regularities* in M and S.

In particular, S_θ constitutes a theory of the regular phonology of the language. Its high-probability sound changes are the "regular" ones, while irregularities and exceptions can be explained as occasional lower-probability choices. We prefer a theory S_θ that has high likelihood, i.e., it assigns high probability (≈ 1) to each observed form s given its underlying u. In linguistic terms, we prefer predictive theories that require few exceptions.

In the linguistic community, the primary motivation for probabilistic models of phonology (Pierrehumbert, 2003) has been to explain "soft" phenomena: synchronic variation (Sankoff, 1978; Boersma and Hayes, 2001) or graded acceptability judgments on novel surface forms (Hayes and Wilson, 2008). These applications are orthogonal to our motivation, as we do not observe any variation or gradience in our present experiments. Fundamentally, we use probabilities to measure irregularity—which simply means unpredictability and is a matter of degree. Our objective function will quantitatively favor explanations that show greater regularity (Eisner, 2002b).

A probabilistic treatment also allows relatively simple learning methods (e.g., Boersma and Hayes (2001)) since inference never has to backtrack from a contradiction. Our method searches a continuous space of phonologies S_θ, all of which are consistent with every mapping S. That is, we always have $S_\theta(s \mid u) > 0$ for all u, s, so our current guess of S_θ is always capable of explaining the observed words, albeit perhaps with low probability. Our EM learner tunes S_θ (and M_ϕ) so as to raise the probability of the observed surface forms, marginalizing over the reconstructed lexicon M of underlying forms. We do warn that

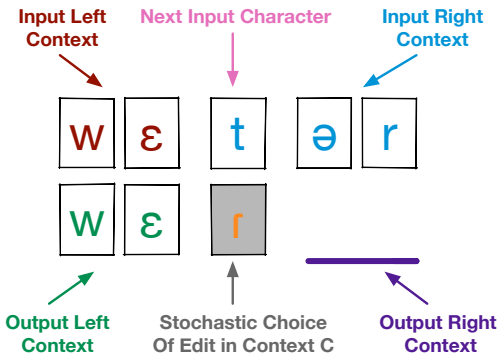

Figure 2: Illustration of a contextual edit process as it pronounces the English word *wetter* by transducing the underlying /wɛt#ər/ (after erasing #) to the surface [wɛɾər]. At the point shown, it is applying the "intervocalic alveolar flapping" rule, replacing /t/ in this context by applying SUBST(ɾ).

EM can get stuck at a local optimum; random restarts and simulated annealing are ways to escape such low-likelihood solutions, much as backtracking escapes zero-likelihood solutions.

3.2 Mapping URs to SRs: The phonology S_θ

We currently model $S_\theta(s \mid u)$ as the probability that a left-to-right *stochastic contextual edit process* (Figure 2) would edit u into s. This probability is a sum over all edit sequences that produce s from u—that is, all s-to-u alignments.

Stochastic contextual edit processes were described by Cotterell et al. (2014). Such a process writes surface string $s \in \Sigma_s^*$ while reading the underlying string $u \in \Sigma_u^*$. If the process has so far consumed some prefix of the input and produced some prefix of the output, it will next make a stochastic choice among $2|\Sigma_s| + 1$ possible edits. Edits of the form SUBST(c) or INSERT(c) (for $c \in \Sigma_s$) append c to the output string. Edits of the form SUBST(c) or DELETE will (also) consume the next input phoneme; if no input phonemes remain, the only possible edits are INSERT(c) or HALT.

The stochastic choice of edit, *given context*, is governed by a conditional log-linear distribution with feature weight vector θ. The feature functions may look at a bounded amount of left and right input context, as well as left output context. Our feature functions are described in section 6.

Our *normalized* probabilities $S_\theta(s \mid u)$ can be computed by a weighted finite-state transducer, a crucial computational property that we will exploit in section 4.2. As Cotterell et al. (2014) explain, the price is that our model is left/right-asymmetric. The inability to condition directly on the right output context arises from local normalization, just like "label bias" in maximum entropy Markov models (McCallum et al., 2000). With certain fancier approaches to modeling S_θ, which we leave to future work, this effect could be mitigated while preserving the transducer property.

3.3 Generating URs: The lexicon model M_ϕ

In our present experiments, we use a very simple lexicon model M_ϕ, so that the burden falls on the phonology S_θ to account for any language-specific regularities in surface forms. This corresponds to the "Richness of the Base" principle advocated by some phonologists (Prince and Smolensky, 2004), and seems to yield good generalization for us. We say all URs of the same length have the same probability, and the length is geometrically distributed with mean $(1/\phi) - 1$. This is a 0-gram model with a single parameter $\phi \in (0, 1]$, namely $M_\phi(m) = ((1-\phi)/|\Sigma_u|)^{|m|} \cdot \phi$.

It would be straightforward to experiment with other divisions of labor between the lexicon model and phonology model. A 1-gram model for M_ϕ would also model which underlying phonemes are common in the lexicon. A 2-gram model would model the "underlying phonotactics" of morphs, though phonological processes would still be needed at morph boundaries. Such models are the probabilistic analogue of morpheme structure constraints. We could further generalize from $M_\phi(m)$ to $M_\phi(m \mid a)$, to allow the shape of the morph m to be influenced by a's content. For example, $M_\phi(m \mid a)$ for English might describe how *nouns* tend to have underlying stress on the first syllable; similarly, $M_\phi(m \mid a)$ for Arabic might capture the fact that underlying *stems* tend to consist of 3 consonants; and across languages, $M_\phi(m \mid a)$ would prefer *affixes* to be short.

Note that we will always learn a language's M_ϕ jointly with its actual lexicon M. Loosely speaking, the parameter vector ϕ is found from easily reconstructed URs in M; then M_ϕ serves as a prior that can help us reconstruct more difficult URs.

3.4 Prior Over the Parameters

For ϕ, which is a scalar under our 0-gram model, our prior is uniform over $(0, 1]$. We place a spherical Gaussian prior on the vector θ, with mean $\vec{0}$ and a variance σ^2 tuned by coarse grid search on dev data (see captions of Figures 3–4).

The Gaussian favors phonologies that are simple in the sense that they have few strongly weighted features. A grammar that refers once to

the natural class of voiced consonants (section 6), which captures a generalization, is preferred to an equally descriptive grammar that refers separately to several specific voiced consonants. If it is hard to tell whether a change applies to round or back vowels (because these properties are strongly correlated in the training data), then the prior resists grammars that make an arbitrary choice. It prefers to "spread the blame" by giving half the weight to each feature. The change is still probable for round back vowels, and moderately probable for other vowels that are *either* round or back.

4 Inference

We are given a training set of surface word forms s that realize *known* abstract words $\vec{a}$. We aim to reconstruct the underlying morphs m and words u, and predict new surface word forms s.

4.1 A Bayesian network

For fixed θ and ϕ, this task can be regarded as marginal inference in a Bayesian network (Pearl, 1988). Figure 1 displays part of a network that encodes the modeling assumptions of section 3. The nodes at layers 1, 2, and 3 of this network represent *string-valued* random variables in $\mathcal{M}, \mathcal{U}$, and $\mathcal{S}$ respectively. Each variable's distribution is conditioned on the values of its parents, if any.

In particular, layer 1 represents the unknown $M(a)$ for various a. Notice that each $M(a)$ is softly constrained by the prior M_ϕ, and also by the fact that it must help produce various observed surface words via S_θ.

Each underlying word u at level 2 is a concatenation of its underlying morphs $M(a_i)$ at level 1. Thus, the topology at levels 1–2 is given by supervision. We would have to learn this topology if the word's morphemes a_i were not known.

Our approach captures the unbounded generative capacity of language. In contrast to Dreyer and Eisner (2009) (see section 8), we have defined a *directed* graphical model. Hence new unobserved descendants can be added without changing the posterior distribution over the existing variables. So our finite network can be viewed as a subgraph of an infinite graph. That is, we make no closed-vocabulary assumption, but *implicitly* include (and predict the surface forms of) any unobserved words that could result from combining morphemes, even morphemes not in our dataset.

While the present paper focuses on word types, we could extend the model to consider tokens as well. In Figure 1, each phonological surface type at layer 3 could be observed to generate 0 or more noisy phonetic tokens at layer 4, in contexts that call for the morphemes expressed by that type.

4.2 Loopy belief propagation

The top two layers of Figure 1 include a long undirected cycle (involving all 8 nodes and all 8 edges shown). On such "loopy" graphical models, exact inference is in general uncomputable when the random variables are string-valued. However, Dreyer and Eisner (2009) showed how to substitute a popular approximate joint inference method, loopy belief propagation (Murphy et al., 1999).

Qualitatively, what does this do on Figure 1?[4] Let u denote the leftmost layer-2 node. Midway through loopy BP, u is not yet sure of its value, but is receiving suggestions from its neighbors. The stem UR immediately above u would like u to *start* with something like /rizajgn#/.[5] Meanwhile, the word SR immediately below u encourages u to be any UR that would have a high probability (under S_θ) of surfacing as [rɛzɪgn#eɪʃən]. So u tries to meet both requirements, guessing that its value might be something like /rizajgn#eɪʃən/ (the product of this string's scores under the two messages to u is relatively high). Now, for U to have produced something like /rizajgn#eɪʃən/ by stem-suffix concatenation, the suffix's UR must have been something like /eɪʃən/. u sends a message saying so to the third node in layer 1. This induces that node (the suffix UR) to inform the rightmost layer-2 node that *it* probably ends in /#eɪʃən/ as well—and so forth, iterating until convergence.

Formally, the loopy BP algorithm iteratively updates *messages* and *beliefs*. Each is a function that scores possible strings (or string tuples). Dreyer and Eisner (2009)'s key insight is that these messages and beliefs can be represented using weighted finite-state machines (WFSMs), and furthermore, loopy BP can compute all of its updates using standard polytime finite-state constructions.

4.3 Discussion: The finite-state requirement

The above results hold when the "factors" that define the graphical model are themselves expressed

[4] Loopy BP actually passes messages on a factor graph *derived from* Figure 1. However, in this informal paragraph we will speak as if it were passing messages on Figure 1 directly.

[5] Because that stem UR thinks its own value is something like /rizajgn/—based on the messages that it is currently receiving from related forms such as /rizajgn#z/, and from M_ϕ.

as WFSMs. This is true in our model. The factors of section 4.1 correspond to the conditional distributions M_ϕ, U, and S_θ that respectively select values for nodes at layers 1, 2, and 3 given the values at their parents. As section 3 models these, for any ϕ and θ, we can represent M_ϕ as a 1-tape WFSM (acceptor), U as a multi-tape WFSM, and S_θ as a 2-tape WFSM (transducer).[6]

Any other WFSMs could be substituted. We are on rather firm ground in restricting to finite-state (regular) models of S_θ. The apparent regularity of natural-language phonology was first observed by Johnson (1972), so computational phonology has generally preferred grammar formalisms that compile into (unweighted) finite-state machines, whether the formalism is based on rewrite rules (Kaplan and Kay, 1994) or constraints (Eisner, 2002a; Riggle, 2004).

Similarly, U could be any multi-tape finite-state relation,[7] not just concatenation as assumed in section 2. This would allow our framework to handle templatic morphology (Hulden, 2009), infixation, or circumfixation.

Although only regular factors are allowed in our graphical model, a loopy graphical model with *multiple* such factors can actually capture non-regular phenomena, for example by using auxiliary variables (Dreyer and Eisner, 2009, §3.4). Approximate inference then proceeds by loopy BP on this model. In particular, reduplication is not regular if unbounded, but we can adopt morphological doubling theory (Inkelas and Zoll, 2005) and model it by having U concatenate two copies of the *same* morph. During inference of URs, this morph exchanges messages with two substrings of the underlying word.

[6] M_ϕ has a single state, with halt probability ϕ and the remaining probability $1 - \phi$ divided among self-loop arcs labeled with the phonemes in Σ_u. U must concatenate k morphs by copying all of tape 1, then tape 2, etc., to tape $k + 1$: this is easily done using $k + 1$ states, and arcs of probability 1. S_θ is constructed as in Cotterell et al. (2014).

[7] In general, a U factor enforces $u = U(m_1, \ldots, m_k)$, so it is a degree-$(k + 1)$ factor, represented by a $(k + 1)$-tape WFSM connecting these variables (Dreyer and Eisner, 2009). If one's finite-state library is limited to 2-tape WFSMs, then one can simulate the U factor using (1) an auxiliary string variable π encoding the path through U, (2) a unary factor weighting π according to U, (3) a set of binary factors relating π to each of $u, m_1, \ldots, m_k$. The standard case $u = m_1\#\ldots\#m_k$ can be handled more easily. Given factor U's incoming messages $\mu_{\cdot \to U}$, each being a 1-tape WFSM, compute its loopy BP outgoing messages $\mu_{U \to u} = \mu_{m_1 \to U}\#\cdots\#\mu_{m_k \to U}$ and (e.g.) $\mu_{U \to m_2} = \text{range}(\mu_{u \to U} \circ ((\mu_{m_1 \to U}\# \times \epsilon)\ \Sigma_u^*\ (\#\mu_{m_3 \to U}\#\cdots\#\mu_{m_k \to U} \times \epsilon)))$.

Similarly, we can manipulate the graphical model structure to encode cyclic phonology—i.e., concatenating a word SR with a derivational affix UR and passing the result through S_θ once again. An alternative is to encode this hierarchical structure into the word UR u, by encoding level-1 and level-2 boundaries with different symbols. A single application of S_θ can treat these boundaries differently: for example, by implementing cyclic phonology as a composition of two transductions.

4.4 Loopy BP implementation details

Each loopy BP message to or from a random variable is a 1-tape WFSM (acceptor) that scores all possible values of that variable (given by the set $\mathcal{M}$, $\mathcal{U}$, or $\mathcal{S}$: see section 2). We initialized each message to the uniform distribution.[8] We then updated the messages serially, alternating between upward and downward sweeps through the Bayesian network. After 10 iterations we stopped and computed the final belief at each variable.

A complication is that a popular affix such as /z/ (in layer 1) receives messages from hundreds of words that realize that affix. Loopy BP obtains that affix's belief and outgoing messages by intersecting all these WFSMs—which can lead to astronomically large results and runtimes. We address this for now with a simple pruning approximation where at each variable m, we dynamically restrict to a finite *support set* of plausible values for m. We take this to be the union of the 20-best lists of all messages sent to m.[9] We modify those messages so that strings in m's support set have unchanged weight, but all other strings have weight 0. As a result, m's outgoing messages and belief are also confined to its support set. Note that the support set is not hand-specified, but determined automatically by taking the best hypotheses under the probability model.

Improved approaches with *no* pruning are possible. After submitting this paper, we developed a penalized expectation propagation method (Cotterell and Eisner, 2015). It approximates the messages using log-linear functions (based on variable-order n-gram features) whose support is the *entire* space Σ^* . We also developed a dual

[8] This is standard—although the uniform distribution over the space of strings is actually an *improper* distribution. It is expressed by a single-state WFSM whose arcs have weight 1.

[9] In general, we should update this support set dynamically as inference and learning improve the messages. But in our present experiments, that appears unnecessary, since the initial support set always appears to contain the "correct" UR.

decomposition method (Peng et al., 2015), which if it converges, exactly recovers the single most probable explanation of the data[10] given ϕ and θ.

5 Parameter Learning

We employ MAP-EM as the learning algorithm. The E-step is approximated by the loopy BP algorithm of section 4. The M-step takes the resulting beliefs, together with the prior of section 3.4, and uses them to reestimate the parameters θ and ϕ.

If we knew the true UR u_k for each observed word type s_k, we would just do supervised training of θ, using L-BFGS (Liu and Nocedal, 1989) to locally maximize θ's posterior log-probability

$$(\sum_k \log S_\theta(s_k \mid u_k)) + \log p_{\text{prior}}(\theta)$$

Cotterell et al. (2014) give the natural dynamic programming algorithm to compute each summand and its gradient w.r.t. θ. The gradient is the difference between observed and expected feature vectors of the contextual edits (section 3.2), averaged over edit contexts in proportion to how many times those contexts were likely encountered. The latent alignment makes the objective non-concave.

In our EM setting, u_k is not known. So our M-step replaces $\log S_\theta(s_k \mid u_k)$ with its expectation, $\sum_{u_k} b_k(u_k) \log S_\theta(s_k \mid u_k)$, where b_k is the normalized belief about u_k computed by the previous E-step. Since b_k and S_θ are both represented by WFSMs (with 1 and 2 tapes respectively), it is possible to compute this quantity and its gradient exactly, using finite-state composition in a second-order expectation semiring (Li and Eisner, 2009). For speed, however, we currently prune b_k back to the 5-best values of u_k. This lets us use a simpler and faster approach: a weighted average over 5 runs of the Cotterell et al. (2014) algorithm.

Our asymptotic runtime benefits from the fact that our graphical model is directed (so our objective does not have to contrast with all other values of u_k) and the fact that S_θ is locally normalized (so our objective does not have to contrast with all other values of s_k for each u_k). In practice we are far faster than Dreyer and Eisner (2009).

We initialized the parameter vector θ to $\vec{0}$, except for setting the weight of the COPY feature (section 6) such that the probability of a COPY edit is 0.99 in every context other than end-of-string. This encourages URs to resemble their SRs.

[10] That is, a lexicon of morphs together with contextual edit sequences that will produce the observed word SRs.

BIGRAM(strident,strident)	adjacent surface stridents
BIGRAM(ϵ,uvular)	surface uvular
EDIT([s],[z])	/s/ became [z]
EDIT(coronal,labial)	coronal became labial
EDIT(ϵ, phoneme)	phoneme was inserted
EDIT(consonant,ϵ)	consonant was deleted

Table 1: Examples of markedness and faithfulness features that fire in our model. They have a natural interpretation as Optimality-Theoretic constraints. ϵ denotes the empty string. The natural classes were adapted from (Riggle, 2005).

To reestimate ϕ, the M-step does not need to use L-BFGS, for section 3.3's simple model of M_ϕ and uniform prior over $\phi \in (0, 1]$. It simply sets $\phi = 1/(\ell + 1)$ where ℓ is the average expected length of a UR according to the previous E-step. The expected length of each u_k is extracted from the WFSM for the belief b_k, using dynamic programming (Li and Eisner, 2009). We initialized ϕ to 0.1; experiments on development data suggested that the choice of initializer had little effect.

6 Features of the Phonology Model

Our stochastic edit process $S_\theta(s \mid u)$ assigns a probability to each possible u-to-s edit sequence. This edit sequence corresponds to a character-wise alignment of u to s. Our features for modeling the contextual probability of each edit are loosely inspired by constraints from Harmonic Grammar and Optimality Theory (Smolensky and Legendre, 2006). Such constraints similarly evaluate a u-to-s alignment (or "correspondence"). They are traditionally divided into markedness constraints that encourage a well-formed s, and faithfulness constraints that encourage phonemes of s to resemble their aligned phonemes in u.

Our EDIT faithfulness features evaluate an edit's (input, output) phoneme pair. Our BIGRAM markedness features evaluate an edit that emits a new phoneme of s. They evaluate the surface bigram it forms with the *previous* output phoneme.[11] Table 1 shows example features. Notice that these features back off to various natural classes of phonemes (Clements and Hume, 1995).

These features of an edit need to examine at most (0,1,1) phonemes of (left input, right input, left output) context respectively (see Figure 2). So the PFST that implements S_θ should be able to use what Cotterell et al. (2014) calls a (0,1,1) topology. However, we actually used a (0,2,1) topology,

[11] At beginning-of-string, the previous "phoneme" is the special symbol BOS. For the HALT edit at end-of-string, which copies the symbol EOS, the new "phoneme" is EOS.

to allow features that also look at the "upcoming" input phoneme that immediately follows the edit's input (/ə/ in Figure 2). Specifically, for each natural class, we also included *contextual* versions of each EDIT or BIGRAM feature, which fired only if the "upcoming" input phoneme fell in that natural class. Contextual BIGRAM features are our approximation to surface trigram features that look at the edit's output phoneme together with the previous *and next* output phonemes. (A PFST cannot condition its edit probabilities on the next output phoneme because that has not been generated yet—see section 3.2—so we are using the upcoming *input* phoneme as a proxy.) Contextual EDIT features were cheap to add once we were using a (0,2,1) topology, and in fact they turned out to be helpful for capturing processes such as Catalan's deletion of the underlyingly *final* consonant.

Finally, we included a COPY feature that fires on any edit where surface and underlying phonemes are exactly equal. (This feature resembles Optimality Theory's IDENT-IO constraint, and ends up getting the strongest weight.) In total, our model has roughly 50,000 binary features.

Many improvements to this basic feature set would be possible in future. We cannot currently express implications such as "adjacent obstruents must also agree in voicing," "a vowel that surfaces must preserve its height," or "successive vowels must also agree in height." We also have not yet designed features that are sensitive to surface prosodic boundaries or underlying morph boundaries. (Prosodic structure and autosegmental tiers are absent from our current representations, and we currently simplify the stochastic edit process's feature set by having S_θ erase the # morph boundaries *before* applying that process.)

Our standard prior over θ (section 3.4) resists overfitting in a generic way, by favoring phonologies that are "simple to describe." Linguistic improvements are possible here as well. The prior should arguably discourage positive weights more than negative ones, since our features detect constraint violations that ordinarily reduce probability. It should also be adjusted to mitigate the current structural bias against deletion edits, which arises because the single deletion possible in a context must compete on equal footing with $|\Sigma_s|$ insertions and $|\Sigma_s| - 1$ substitutions. More ambitiously, a linguistically plausible prior should prefer phonologies that are conservative ($s \approx u$) and have low conditional entropies $H(s \mid u), H(u \mid s)$ to facilitate communication.

7 Experimental Design

We objectively evaluate our learner on its ability to predict *held-out surface forms*. This blind testing differs from traditional practice by linguists, who evaluate a manual or automatic analysis (= URs + phonology) on whether it describes the *full* dataset in a "natural" way that captures "appropriate" generalizations. We avoid such theory-internal evaluation by simply quantifying whether the learner's analysis *does* generalize (Eisner, 2015).

To avoid tailoring to our training/test data, we developed our method, code, features, and hyperparameters using only two development languages, English and German. Thus, our learner was not engineered to do well on the other 5 languages below: the graphs below show its first attempt to learn those languages. We do also evaluate our learners on English and German, using *separate* training/test data.

We provide all our data (including citations, development data, training-test splits, and natural classes) at `http://hubal.cs.jhu.edu/tacl2015/`, along with brief sketches of the phonological phenomena in the datasets, the "gold" stem URs we assumed for evaluation, and our learner's predictions and error patterns.

7.1 Evaluation methodology

Given a probability distribution p over surface word types of a language, we sample a training set of N types without replacement. This simulates reading text until we have seen N distinct types. For each of these frequent words, we observe the SR s and the morpheme sequence $\vec{a}$.

After training our model, we evaluate its beliefs b about the SRs s on a disjoint set of test words whose $\vec{a}$ are observed. To improve interpretability of the results, we limit the test words to those whose morphemes have all appeared at least once in the training set. (*Any* method would presumably get other words badly wrong, just as it would get the training words right.)

To evaluate our belief b about the SR of a test word $(\vec{a}, s^*)$, we use three measures for which "smaller is better." First, 0-1 loss asks whether $s^* \neq \operatorname{argmax}_s b(s)$. This could be compared with non-probabilistic predictors. Second, the surprisal $-\log_2 b(s^*)$ is low if the model finds it *plausible*

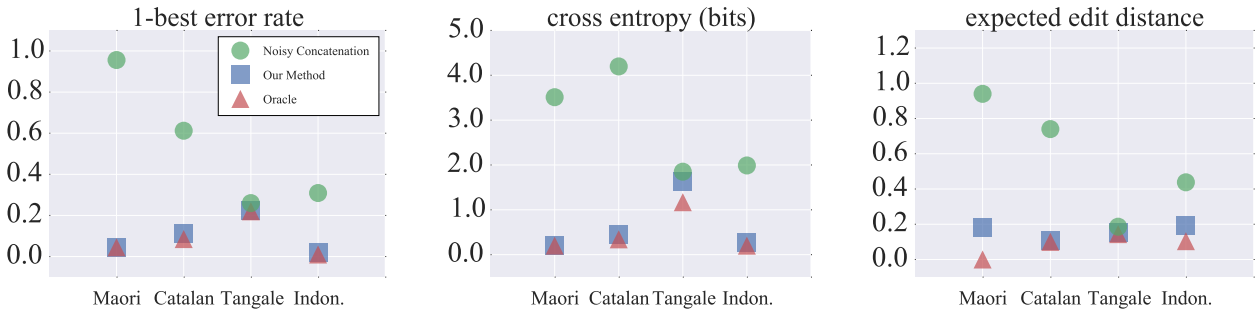

Figure 3: Results on the small phonological exercise datasets ($\approx$ 100 word types). Smaller numbers are better. Preliminary tests suggested that the variance of the prior (section 3.4) did not strongly affect the results, so we took $\sigma^2 = 5$ for all experiments.

that s^* realizes $\vec{a}$. If so, this holds out promise for future work on analyzing or learning from *unannotated* tokens of s^*. Third, we evaluate the *whole* distribution b in terms of $\sum_s b(s)L(s^*, s)$ where L is unweighted Levenshtein distance.

We take the average of each measure over test words, weighting those words according to p. This yields our three reported metrics: *1-best error rate*, *cross-entropy*, and *expected edit distance*. Each metric is the expected value of some measure on a random test token.

These metrics are actually random variables, since they depend on the randomly sampled training set and the resulting test distribution. We report the expectations of these random variables by running many training-test splits (see section 7.2).

7.2 Datasets

To test discovery of interesting patterns from limited data, we ran our learner on 5 "exercises" drawn from phonology textbooks (102 English nouns, 68 Maori verbs, 72 Catalan adjectives, 55 Tangale nouns, 44 Indonesian nouns), exhibiting a diverse range of phenomena. In each case we took p to be the uniform distribution over the provided word types. We took N to be one less than the number of provided types. So to report our expected metrics, we ran all $N + 1$ experiments where we trained jointly on N forms and tested on the 1 remaining form. This is close to linguists' practice of fitting an analysis on the entire dataset, yet it is a fair test.

To test on larger, naturally occurring datasets, we ran our learner on subsets of the CELEX database (Baayen et al., 1995), which provides surface phonological forms and token counts for German, Dutch, and English words. For each language, we constructed a coherent subcorpus of 1000 nouns and verbs, focusing on inflections with common phonological phenomena. These turned out to involve mainly voicing: final obstruent devoicing (German 2nd-person present indicative verbs, German nominative singular nouns, Dutch infinitive verbs, Dutch singular nouns) and voicing assimilation (English past tense verbs, English plural nouns). We were restricted to relatively simple phenomena because our current representations are simple segmental strings that lack prosodic and autosegmental structure. In future we plan to consider stress, vowel harmony, and templatic morphology.

We constructed the distribution p in proportion to CELEX's token counts. In each language, we trained on $N = 200$, 400, 600, or 800 forms sampled from p. To estimate the expectation of each metric over *all* training sets of size N, we report the sample mean and bootstrap standard error over 10 random training sets of size N.

Except in Indonesian, every word happens to consist of at most two morphemes (one stem plus one possibly empty suffix). In all experiments, we take the phonological inventories Σ_u and Σ_s to be given as the set of all surface phonemes observed in training $\cup$ test.

7.3 Comparison systems

There do not appear to be previous systems that perform our generalization task. Therefore, we compared our own system against variants.

We performed an ablation study to determine whether the learned phonology was helpful. We substituted a simplified phonology model where $S_\theta(s \mid u)$ just decays exponentially with the edit distance between s and u; the decay rate was learned by EM as usual. That is, this model uses *only* the COPY feature of section 6. This baseline system treats phonology as "noisy concatenation" of learned URs, not trying to model its regularity.

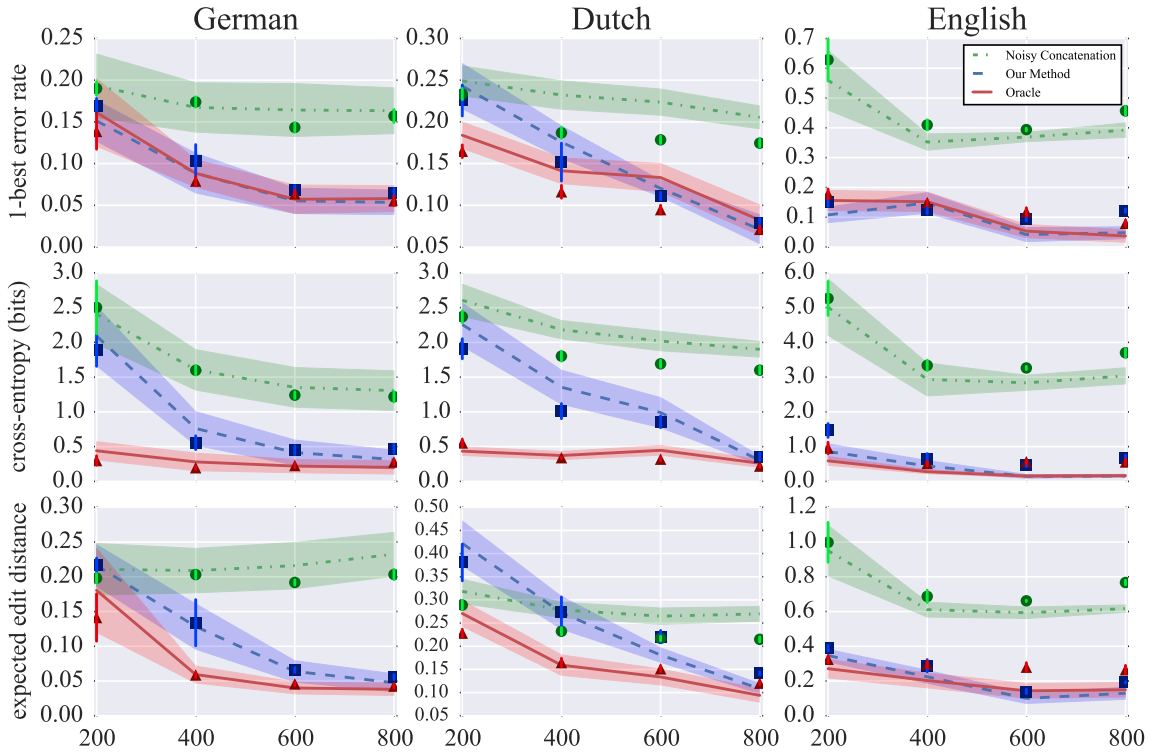

Figure 4: Results on the CELEX datasets (1000 word types) at 4 different training set sizes N. The larger training sets were supersets of the smaller ones, obtained by continuing to sample with replacement from p. For each training set, the *unconnected* points evaluate all words $\notin$ training whose morphemes $\in$ training. Meanwhile, the *connected* points permit comparison across the 4 values of N, by evaluating only on a common test set found by intersecting the 4 unconnected test sets. Each point estimates the metric's expectation over *all* ways of sampling the 4 training sets; specifically, we plot the *sample mean* from 10 such runs, with *error bars* showing a bootstrap estimate of the standard error of the mean. Non-overlapping error bars at a given N always happen to imply that the difference in the two methods' sample means is too extreme to be likely to have arisen by chance (paired permutation test, $p < 0.05$). Each time we evaluated some training-test split on some metric, we first tuned σ^2 (section 3.4) by a coarse grid search where we trained on the first 90% of the training set and evaluated on the remaining 10%.

We considered an additional ablation study to determine whether the learned URs were helpful. However, we did not come up with a plausible heuristic for identifying URs in some simpler way. Thus, instead we asked whether the learned URs were as good as hand-constructed URs. Our "oracle" system was allowed to observe gold-standard URs for stems instead of inferring them. This system is still fallible: it must still infer the *affix* URs by belief propagation, and it must still use MAP-EM to estimate a phonology within our current model family S_θ. Even with supervision, this family will still struggle to model many types of phonology, e.g., ablaut patterns (in Germanic strong verbs) and many stress-related phenomena.

7.4 Results

We graph our results in Figures 3 and 4. When given enough evidence, our method works quite well across the 7 datasets. For 94–98% of held-out words on the CELEX languages (when $N = 800$), and 77–100% on the phonological exercises, our method's top pick is the correct surface form. Further, the other metrics show that it places most of its probability mass on that form,[12] and the rest on highly similar forms. Notably, our method's predictions are nearly as good as if gold stem URs had been supplied (the "oracle" condition). Indeed, it *does* tend to recover those gold URs (Table 2).

Phon. Exercises		**CELEX**	
Maori	95.5	German	99.9
Catalan	99.5	Dutch	86.3
Tangale	79.8	English	82.2
Indonesian	100		

Table 2: Percent of training words, weighted by the distribution p, whose 1-best recovered UR (including the boundary #) exactly matches the manual "gold" analysis. Results are averages over all runs (with $N = 800$ for the CELEX datasets).

Yet there are some residual errors in predicting the SRs. Our phonological learner cannot

[12] Cross-entropy < 1 bit means that the correct form has probability $> 1/2$ on average (using geometric mean).

perfectly learn the UR-to-SR mapping even from many well-supervised pairs (the oracle condition). In the CELEX and Tangale datasets, this is partly due to irregularity in the language itself. However, error analysis suggests we also miss some generalizations due to the imperfections of our current S_θ model (as discussed in sections 3.2 and 6).

When given less evidence, our method's performance is more sensitive to the training sample and is worse on average. This is expected: e.g., a stem's final consonant cannot be reconstructed if it was devoiced (German) or deleted (Maori) in all the training SRs. However, a contributing factor may be the increased error rate of the phonological learner, visible even with oracle data. Thus, we suspect that a S_θ model with better generalization would improve our results at all training sizes. Note that *harming* S_θ—allowing only "noisy concatenation"—clearly harms the method, proving the need for true phonological modeling.

8 Related Work

Jarosz (2013, §2) and Tesar (2014, chapters 5–6) review work on learning the phonology S_θ. Phonologists pioneered stochastic-gradient and passive-aggressive training methods—the Gradual Learning Algorithm (Boersma, 1998) and Error-Driven Constraint Demotion (Tesar and Smolensky, 1998)—for structured prediction of the surface word s from the underlying word u. If s is not fully observed during training (we illustrate why in layer 4 of Figure 1), then it can be imputed, a step known as Robust Interpretive Parsing.

Recent papers consider our setting where $u = m_1\#m_2\#\cdots$ is not observed either. The *contrast analysis* method (Tesar, 2004; Merchant, 2008) in effect uses constraint propagation (Dechter, 2003). That is, it serially eliminates variable values (describing aspects of the URs or the constraint ranking) that are provably incompatible with the data. Constraint propagation is an incomplete method that is not guaranteed to make all logical deductions. We use its probabilistic generalization, loopy belief propagation (Dechter et al., 2010)—which is still approximate but can deal with noise and stochastic irregularity. A further improvement is that we work with string-valued variables, representing uncertainty using WFSMs; this lets us reason about URs of unknown length and unknown alignment to the SRs. (Tesar and Merchant instead used binary variables, one for each segmental feature in each UR, requiring the simplifying assumption that the URs are known except for their segmental features. They assume that SRs are annotated with morph boundaries and that the phonology only changes segmental features, never inserting or deleting segments.) On the other hand, Tesar and Merchant reason globally about the constraint ranking, whereas in this paper, we only locally improve the phonology—we use EM, rather than the full Bayesian approach that treats the parameters $\vec{\theta}$ as variables within BP.

Jarosz (2006) is closest to our work in that she uses EM, just as we do, to maximize the probability of observed surface forms whose constituent morphemes (but not morphs) are known.[13] Her model is a probabilistic analogue of Apoussidou (2006), who uses a latent-variable structured perceptron. A non-standard aspect of this model (defended by Pater et al. (2012)) is that a morpheme a can stochastically choose *different* morphs $M(a)$ when it appears in different words. To obtain a single shared morph, one could penalize this distribution's entropy, driving it toward 0 as learning proceeds. Such an approach—which builds on a suggestion by Eisenstat (2009, §5.4)—would loosely resemble dual decomposition (Peng et al., 2015). Unlike our BP approach, it would maximize rather than marginalize over possible morphs.

Our work has focused on scaling up inference. For the phonology S, the above papers learn the weights or rankings of just a few plausible constraints (or Jarosz (2006) learns a discrete distribution over all $5! = 120$ rankings of 5 constraints), whereas we use S_θ with roughly 50,000 constraints (features) to enable learning of unknown languages. Our S also allows exceptions. The above papers also consider only very restricted sets of morphs, either identifying a small set of plausible morphs or prohibiting segmental insertion/deletion. We use finite-state methods so that it is possible to consider the space Σ_u^* of all strings.

On the other hand, we are divided from previous work by our inability to use an OT grammar (Prince and Smolensky, 2004), a stochastic OT grammar (Boersma, 1997), or even a maximum entropy grammar (Goldwater and Johnson, 2003; Dreyer et al., 2008; Eisenstat, 2009). The reason is that our BP method *inverts* the phonological mapping S_θ to find possible word URs.

[13] She still assumes that word SRs are annotated with morpheme boundaries, and that a small set of possible morphs is given. These assumptions are relaxed by Eisenstat (2009).

Given a word SR s, we construct a WFSM (message) that scores *every* possible UR $u \in \Sigma_u^*$—the score of u is $S_\theta(s \mid u)$. To accomplish this step without approximation, our method needs S_θ itself to be represented as a WFSM (section 3.2). (The WFSM for a maximum entropy grammar unfortunately does not compute S_θ but only an unnormalized version. A different normalizing constant is needed for each u, akin to the "double intractability" problem in Bayesian learning.)

In the NLP community, Elsner et al. (2013) resembles our work in many respects. Like us, they recover a latent underlying lexicon (using the same simple prior M_ϕ) and use EM to learn a phonology (rather similar to our S_θ, though less powerful).[14] Unlike us, they do not assume annotation of the (abstract) morpheme sequence, but jointly learn a nonparametric bigram model to discover the morphemes. Their evaluation is quite different, as their aim is actually to recover underlying *words* from phonemically transcribed child-directed English utterances. However, nothing in their model distinguishes words from morphemes—indeed, sometimes they do find morphemes instead—so their model could be used in our task. For inference, they invert the finite-state S_θ like us to reconstruct a lattice of possible UR strings. However, they do this not within BP but within a block Gibbs sampler that stochastically reanalyzes utterances one at a time. Whereas our BP tries to find a consensus UR for each given morpheme type, their sampler posits morph tokens while trying to reuse frequent morph types, which are interpreted as the morphemes. With *observed* morphemes (our setting), this sampler would fail to mix.

Dreyer and Eisner (2009, 2011) like us used loopy BP and MAP-EM to predict morphological SRs. Their 2011 paper was also able to exploit raw text without morphological supervision. However, they directly modeled pairwise finite-state relationships among the *surface* word forms without using URs. Their model is a joint distribution over n variables: the word SRs of a single inflectional paradigm. Since it requires a fixed n, it does not directly extend to derivational morphology: deriving new words would require adding new variables, which—for an *undirected* model like theirs—changes the partition function and requires retraining. By contrast, our trained *directed* model is a productive phonological system that can generate unboundedly many new words (see section 4.1). By analogy, n samples from a Gaussian would be described with a directed model, and inferring the Gaussian parameters predicts any number of future samples $n+1, n+2, \ldots$.

Bouchard-Côté et al., in several papers from 2007 through 2013, have used directed graphical models over strings, like ours though without loops, to model *diachronic* sound change. Sometimes they use belief propagation for inference (Hall and Klein, 2010). Their goal is to recover latent *historical* forms (conceptually, surface forms) rather than latent underlying forms. The results are evaluated against manual reconstructions.

None of this work has segmented words into morphs, although Dreyer et al. (2008) did segment surface words into latent "regions." Creutz and Lagus (2005) and Goldsmith (2006) segment an unannotated collection of words into reusable morphs, but without modeling contextual sound change, i.e., phonology.

9 Conclusions and Future Work

We have laid out a probabilistic model for generative phonology. This lets us infer likely explanations of a collection of morphologically related surface words, in terms of underlying morphs and productive phonological changes. We do this by combining well-motivated algorithms for inference in graphical models and MAP estimation from incomplete data, using weighted finite-state machines to encode uncertainty. Throughout our presentation, we were careful to point out various limitations of our setup. But in each case, we also outlined how future work could address these limitations *within the framework we propose here*.

Finally, we proposed a detailed scheme for quantitative evaluation of phonological learners. Across 7 different languages, on both small and larger datasets, our learner was able to predict held-out surface forms with low error rates.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. 1423276, and by a Fulbright grant to the first author. We thank the anonymous reviewers and Reut Tsarfaty for useful discussion of presentation, terminology, and related work.

[14] Elsner et al. (2012) used an S_θ quite similar to ours though lacking bigram well-formedness features. Elsner et al. (2013) simplified this for efficiency, disallowing segmental deletion and no longer modeling the context of changes.

References

Diana Apoussidou. 2006. On-line learning of underlying forms. Technical Report ROA-835, Rutgers Optimality Archive.

R Harald Baayen, Richard Piepenbrock, and Leon Gulikers. 1995. The CELEX lexical database on CD-ROM.

Juliette Blevins. 1994. A phonological and morphological reanalysis of the maori passive. *Te Reo*, 37:29–53.

Paul Boersma and Bruce Hayes. 2001. Empirical tests of the gradual learning algorithm. *Linguistic Inquiry*, 32(1):45–86.

Paul Boersma. 1997. How we learn variation, optionality, and probability. In *Proc. of the Institute of Phonetic Sciences of the University of Amsterdam*, volume 21, pages 43–58.

Paul Boersma. 1998. How we learn variation, optionality, and probability. In *Functional Phonology: Formalizing the Interactions Between Articulatory and Perceptual Drives*, chapter 15. Ph.D. Dissertation, University of Amsterdam. Previously appeared in *IFA Proceedings* (1997), pp. 43–58.

Alexandre Bouchard-Côté, Percy Liang, Thomas L. Griffiths, and Dan Klein. 2007. A probabilistic approach to language change. In *Proc. of NIPS*.

Alexandre Bouchard-Côté, David Hall, Thomas L. Griffiths, and Dan Klein. 2013. Automated reconstruction of ancient languages using probabilistic models of sound change. *Proceedings of the National Academy of Sciences*.

Noam Chomsky and Morris Halle. 1968. *The Sound Pattern of English*. Harper and Row.

George N Clements and Elizabeth V Hume. 1995. The internal organization of speech sounds. In John Goldsmith, editor, *Handbook of Phonological Theory*. Oxford University Press, Oxford.

Ryan Cotterell and Jason Eisner. 2015. Penalized expectation propagation for graphical models over strings. In *Proceedings of NAACL-HLT*, pages 932–942, Denver, June. Supplementary material (11 pages) also available.

Ryan Cotterell, Nanyun Peng, and Jason Eisner. 2014. Stochastic contextual edit distance and probabilistic FSTs. In *Proc. of ACL*.

Mathias Creutz and Krista Lagus. 2005. Inducing the morphological lexicon of a natural language from unannotated text. In *Proc. of the International and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning (AKRR05)*, volume 1.

Rina Dechter, Bozhena Bidyuk, Robert Mateescu, and Emma Rollon. 2010. On the power of belief propagation: A constraint propagation perspective. In Rina Dechter, Hector Geffner, and Joseph Y. Halpern, editors, *Heuristics, Probability and Causality: A Tribute to Judea Pearl*. College Publications.

Rina Dechter. 2003. *Constraint Processing*. Morgan Kaufmann.

Markus Dreyer and Jason Eisner. 2009. Graphical models over multiple strings. In *Proc. of EMNLP*, pages 101–110.

Markus Dreyer and Jason Eisner. 2011. Discovering morphological paradigms from plain text using a dirichlet process mixture model. In *Proc. of EMNLP*, EMNLP '11, pages 616–627.

Markus Dreyer, Jason R. Smith, and Jason Eisner. 2008. Latent-variable modeling of string transductions with finite-state methods. In *Proc. of EMNLP*, pages 1080–1089.

Markus Dreyer. 2011. *A Non-Parametric Model for the Discovery of Inflectional Paradigms from Plain Text Using Graphical Models over Strings*. Ph.D. thesis, Johns Hopkins University, Baltimore, MD, April.

Sarah Eisenstat. 2009. Learning underlying forms with maxent. Master's thesis, Brown University, Providence, RI.

Jason Eisner. 2002a. Comprehension and compilation in Optimality Theory. In *Proc. of ACL*, pages 56–63, Philadelphia, July.

Jason Eisner. 2002b. Discovering syntactic deep structure via Bayesian statistics. *Cognitive Science*, 26(3):255–268, May-June.

Jason Eisner. 2015. Should linguists evaluate grammars or grammar learners? In preparation.

Micha Elsner, Sharon Goldwater, and Jacob Eisenstein. 2012. Bootstrapping a unified model of lexical and phonetic acquisition. In *Proc. of ACL*, pages 184–193.

Micha Elsner, Sharon Goldwater, Naomi Feldman, and Frank Wood. 2013. A joint learning model of word segmentation, lexical acquisition, and phonetic variability. In *Proc. of EMNLP*, pages 42–54.

Alexander M. Fraser, Marion Weller, Aoife Cahill, and Fabienne Cap. 2012. Modeling inflection and word-formation in SMT. In *Proc. of EACL*, pages 664–674.

J. Goldsmith. 2006. An algorithm for the unsupervised learning of morphology. *Natural Language Engineering*, 12(4):353–371.

Sharon Goldwater and Mark Johnson. 2003. Learning OT constraint rankings using a maximum entropy model. In *Proc. of the Workshop on Variation within Optimality Theory*, pages 113–122, Stockholm University.

David Hall and Dan Klein. 2010. Finding cognate groups using phylogenies. In *Proc. of ACL*.

Bruce Hayes and Colin Wilson. 2008. A maximum entropy model of phonotactics and phonotactic learning. *Linguistic Inquiry*, 39(3):379–440.

Matt Hohensee and Emily M. Bender. 2012. Getting more from morphology in multilingual dependency parsing. In *Proc. of NAACL-HLT*, pages 315–326.

Mans Hulden. 2009. Revisiting multi-tape automata for Semitic morphological analysis and generation. In *Proc. of the EACL 2009 Workshop on Computational Approaches to Semitic Languages*, pages 19–26, March.

Sharon Inkelas and Cheryl Zoll. 2005. *Reduplication: Doubling in Morphology*. Number 106 in Cambridge Studies in Linguistics. Cambridge University Press.

Roman Jakobson. 1948. Russian conjugation. *Word*, 4:155–167.

Gaja Jarosz. 2006. Richness of the base and probabilistic unsupervised learning in optimality theory. In *Proc. of the Eighth Meeting of the ACL Special Interest Group on Computational Phonology and Morphology*, pages 50–59.

Gaja Jarosz. 2013. Learning with hidden structure in optimality theory and harmonic grammar: Beyond robust interpretive parsing. *Phonology*, 30(01):27–71.

C. Douglas Johnson. 1972. *Formal Aspects of Phonological Description*. Mouton.

René Kager. 1999. *Optimality Theory*, volume 2. MIT Press.

Ronald M. Kaplan and Martin Kay. 1994. Regular models of phonological rule systems. *Computational Linguistics*, 20(3):331–378.

Michael J Kenstowicz and Charles W Kisseberth. 1979. *Generative Phonology*. Academic Press San Diego.

András Kornai. 1995. *Formal Phonology*. Garland Publishing, New York.

Zhifei Li and Jason Eisner. 2009. First- and second-order expectation semirings with applications to minimum-risk training on translation forests. In *Proc. of EMNLP*, pages 40–51, Singapore, August.

Dong C. Liu and Jorge Nocedal. 1989. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45(1-3):503–528.

Andrew McCallum, Dayne Freitag, and Fernando C. N. Pereira. 2000. Maximum entropy Markov models for information extraction and segmentation. In *Proc. of ICML*, pages 591–598.

Navarré Merchant. 2008. *Discovering Underlying Forms: Contrast Pairs and Ranking*. Ph.D. thesis, Rutgers University. Available on the Rutgers Optimality Archive as ROA-964.

Kevin P. Murphy, Yair Weiss, and Michael I. Jordan. 1999. Loopy belief propagation for approximate inference: An empirical study. In *Proc. of UAI*, pages 467–475.

Joe Pater, Karen Jesney, Robert Staubs, and Brian Smith. 2012. Learning probabilities over underlying representations. In *Proc. of the Twelfth Meeting of the Special Interest Group on Computational Morphology and Phonology*, pages 62–71.

Judea Pearl. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Nanyun Peng, Ryan Cotterell, and Jason Eisner. 2015. Dual decomposition inference for graphical models over strings. In *Proceedings of EMNLP*, Lisbon, September. To appear.

Janet Pierrehumbert. 2003. Probabilistic phonology: Discrimination and robustness. In *Probabilistic Linguistics*, pages 177–228. MIT Press.

Alan Prince and Paul Smolensky. 2004. *Optimality Theory: Constraint Interaction in Generative Grammar*. Wiley-Blackwell.

Jason A. Riggle. 2004. *Generation, Recognition, and Learning in Finite State Optimality Theory*. Ph.D. thesis, University of California at Los Angeles.

Jason Riggle. 2005. Phonological features. Available online at `http://www.mml.cam.ac.uk/dtal/courses/ugrad/paper_support/li8/riggle-feature-chart.pdf` (retrieved 2014-09-28).

David Sankoff. 1978. Probability and linguistic variation. *Synthese*, 37(2):217–238.

Ferdinand de Saussure. 1916. *Course in General Linguistics*. Columbia University Press. English edition of June 2011, based on the 1959 translation by Wade Baskin.

Paul Smolensky and Géraldine Legendre. 2006. *The Harmonic Mind: From Neural Computation to Optimality-Theoretic Grammar (Vol. 1: Cognitive architecture)*. MIT Press.

Bruce Tesar and Paul Smolensky. 1998. Learnability in Optimality theory. *Linguistic Inquiry*, 29(2):229–268.

Bruce Tesar. 2004. Contrast analysis in phonological learning. Technical Report ROA-695, Rutgers Optimality Archive.

Bruce Tesar. 2014. *Output-Driven Phonology: Theory and Learning*. Cambridge University Press.

Reyyan Yeniterzi. 2011. Exploiting morphology in turkish named entity recognition system. In *Proc. of the ACL Student Session*, pages 105–110.