Bootstrapping a Unified Model of Lexical and Phonetic Acquisition

Micha Elsner
melsner0@gmail.com
ILCC, School of Informatics
University of Edinburgh
Edinburgh, EH8 9AB, UK

Sharon Goldwater
sgwater@inf.ed.ac.uk
ILCC, School of Informatics
University of Edinburgh
Edinburgh, EH8 9AB, UK

Jacob Eisenstein
jacobe@gmail.com
School of Interactive Computing
Georgia Institute of Technology
Atlanta, GA, 30308, USA

Abstract

During early language acquisition, infants must learn both a lexicon and a model of phonetics that explains how lexical items can vary in pronunciation—for instance "the" might be realized as [ði] or [ðə]. Previous models of acquisition have generally tackled these problems in isolation, yet behavioral evidence suggests infants acquire lexical and phonetic knowledge simultaneously. We present a Bayesian model that clusters together phonetic variants of the same lexical item while learning both a language model over lexical items and a log-linear model of pronunciation variability based on articulatory features. The model is trained on transcribed surface pronunciations, and learns by bootstrapping, without access to the true lexicon. We test the model using a corpus of child-directed speech with realistic phonetic variation and either gold standard or automatically induced word boundaries. In both cases modeling variability improves the accuracy of the learned lexicon over a system that assumes each lexical item has a unique pronunciation.

1 Introduction

Infants acquiring their first language confront two difficult cognitive problems: building a lexicon of word forms, and learning basic phonetics and phonology. The two tasks are closely related: knowing what sounds can substitute for one another helps in clustering together variant pronunciations of the same word, while knowing the environments in which particular words can occur helps determine which sound changes are meaningful and which are not (Feldman et al., 2009). For instance, if an infant who already knows the word [ju] "you" encounters a new word [jə], they must decide whether it is a new lexical item or a variant of the word they already know. Evidence for the correct conclusion comes from the pronunciation (many English vowels are reduced to [ə] in unstressed positions) and the context—if the next word is "want", "you" is a plausible choice.

(a)	intended:	/ju want wʌn/	/want e kʊki/
(b)	surface:	[jə wɑ̃ʔ wʌn]	[wan ə kʊki]
(c)	unsegmented:	[jəwɑ̃ʔwʌn]	[wanəkʊki]
(d)	idealized:	/juwantwʌn/	/wantekʊki/

Figure 1: The utterances *you want one? want a cookie?* represented (a) using a canonical phonemic encoding for each word and (b) as they might be pronounced phonetically. Lines (c) and (d) remove the word boundaries (but not utterance boundaries) from (b) and (a), respectively.

To date, most models of infant language learning have focused on either lexicon-building or phonetic learning in isolation. For example, many models of word segmentation implicitly or explicitly build a lexicon while segmenting the input stream of phonemes into word tokens; in nearly all cases the phonemic input is created from an orthographic transcription using a phonemic dictionary, thus abstracting away from any phonetic variability (Brent, 1999; Venkataraman, 2001; Swingley, 2005; Goldwater et al., 2009, among others). As illustrated in Figure 1, these models attempt to infer line (a) from line (d). However, (d) is an idealization: real speech has variability, and behavioral evidence suggests that infants are still learning about the phonetics and phonology of their language even after beginning to segment words, rather than learning to neutralize

the variations first and acquiring the lexicon afterwards (Feldman et al., 2009, and references therein).

Based on this evidence, a more realistic model of early language acquisition should propose a method of inferring the *intended* forms (Figure 1a) from the *unsegmented surface* forms (1c) while also learning a model of phonetic variation relating the intended and surface forms (a) and (b). Previous models with similar goals have learned from an artificial corpus with a small vocabulary (Driesen et al., 2009; Räsänen, 2011) or have modeled variability only in vowels (Feldman et al., 2009); to our knowledge, this paper is the first to use a naturalistic infant-directed corpus while modeling variability in all segments, and to incorporate word-level context (a bigram language model). Our main contribution is a joint lexical-phonetic model that infers intended forms from *segmented* surface forms; we test the system using input with either gold standard word boundaries or boundaries induced by an existing unsupervised segmentation model (Goldwater et al., 2009). We show that in both cases modeling variability improves the accuracy of the learned lexicon over a system that assumes each intended form has a unique surface form.

Our model is conceptually similar to those used in speech recognition and other applications: we assume the intended tokens are generated from a bigram language model and then distorted by a noisy channel, in particular a log-linear model of phonetic variability. But unlike speech recognition, we have no ⟨intended-form, surface-form⟩ training pairs to train the phonetic model, nor even a dictionary of intended-form strings to train the language model. Instead, we initialize the noise model using feature weights based on universal linguistic principles (e.g., a surface phone is likely to share articulatory features with the intended phone) and use a bootstrapping process to iteratively infer the intended forms and retrain the language model and noise model. While we do not claim that the particular inference mechanism we use is cognitively plausible, our positive results further support the claim that infants can and do acquire phonetics and the lexicon in concert.

2 Related work

Our work is inspired by the lexical-phonetic model of Feldman et al. (2009). They extend a model for clustering acoustic tokens into phonetic categories (Vallabha et al., 2007) by adding a lexical level that simultaneously clusters word tokens (which contain the acoustic tokens) into lexical entries. Including the lexical level improves the model's phonetic categorization, and a follow-up study on artificial language learning (Feldman, 2011) supports the claim that human learners use lexical knowledge to distinguish meaningful from unimportant phonetic contrasts. Feldman et al. (2009) use a real-valued representation for vowels (formant values), but assume no variability in consonants, and treat each word token independently. In contrast, our model uses a symbolic representation for sounds, but models variability in *all* segment types and incorporates a bigram word-level language model.

To our knowledge, the only other lexicon-building systems that also learn about phonetic variability are those of Driesen et al. (2009) and Räsänen (2011). These systems learn to represent lexical items and their variability from a discretized representation of the speech stream, but they are tested on an artificial corpus with only 80 vocabulary items that was constructed so as to "avoid strong word-to-word dependencies" (Räsänen, 2011). Here, we use a naturalistic corpus, demonstrating that lexical-phonetic learning is possible in this more general setting and that word-level context information is important for doing so.

Several other related systems work directly from the acoustic signal and many of these do use naturalistic corpora. However, they do not learn at both the lexical and phonetic/acoustic level. For example, Park and Glass (2008), Aimetti (2009), Jansen et al. (2010), and McInnes and Goldwater (2011) present lexicon-building systems that use hard-coded acoustic similarity measures rather than *learning* about variability, and they only extract and cluster a few frequent words. On the phonetic side, Varadarajan et al. (2008) and Dupoux et al. (2011) describe systems that learn phone-like units but without the benefit of top-down information.

A final line of related work is on word segmentation. In addition to the models mentioned in Section 1, which use phonemic input, a few models of word segmentation have been tested using phonetic input (Fleck, 2008; Rytting, 2007; Daland and Pierrehumbert, 2011). However, they do not cluster segmented

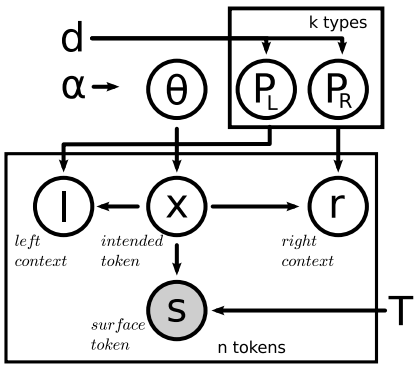

Figure 2: Our generative model of the surface tokens s from intended tokens x, which occur with left and right contexts l and r.

word tokens into lexical items (none of these models even maintains an explicit lexicon), nor do they model or learn from phonetic variation in the input.

3 Lexical-phonetic model

Our lexical-phonetic model is defined using the standard noisy channel framework: first a sequence of intended word tokens is generated using a language model, and then each token is transformed by a probabilistic finite-state transducer to produce the observed surface sequence. In this section, we present the model in a hierarchical Bayesian framework to emphasize its similarity to existing models, in particular those of Feldman et al. (2009) and Goldwater et al. (2009). In our actual implementation, however, we use approximation and MAP point estimates to make our inference process more tractable; we discuss these simplifications in Section 4.

Our observed data consists of a (segmented) sequence of surface words $s_1 \dots s_n$. We wish to recover the corresponding sequence of intended words $x_1 \dots x_n$. As shown in Figure 2, s_i is produced from x_i by a transducer T: $s_i \sim T(x_i)$, which models phonetic changes. Each x_i is sampled from a distribution θ which represents word frequencies, and its left and right context words, l_i and r_i, are drawn from distributions conditioned on x_i, in order to capture information about the environments in which x_i appears: $l_i \sim P_L(x_i), r_i \sim P_R(x_i)$. Because the number of word types is not known in advance, θ is drawn from a Dirichlet process $DP(\alpha)$, and $P_L(x)$ and $P_R(x)$ have Pitman-Yor priors with concentration parameter 0 and discount d (Teh, 2006).

Our generative model of x_i is unusual for two reasons. First, we treat each x_i independently rather than linking them via a Markov chain. This makes the model deficient, since l_i overlaps with x_{i-1} and so forth, generating each token twice. During inference, however, we will never compute the joint probability of all the data at once, only the probabilities of subsets of the variables with particular intended word forms u and v. As long as no two of these words are adjacent, the deficiency will have no effect. We make this independence assumption for computational reasons—when deciding whether to merge u and v into a single lexical entry, we compute the change in estimated probability for their contexts, but not the effect on other words for which u and v themselves appear as context words.

Also unusual is that we factor the joint probability (l, x, r) as $p(x)p(l|x)p(r|x)$ rather than as a left-to-right chain $p(l)p(x|l)p(r|x)$. Given our independence assumption above, these two quantities are mathematically equivalent, so the difference matters only because we are using smoothed estimates. Our factorization leads to a symmetric treatment of left and right contexts, which simplifies implementation: we can store all the context parameters locally as $P_L(\cdot|x)$ rather than distributed over various $P(x|\cdot)$.

Next, we explain our transducer T. A weighted finite-state transducer (WFST) is a variant of a finite-state automaton (Pereira et al., 1994) that reads an input string symbol-by-symbol and probabilistically produces an output string; thus it can be used to specify a conditional probability on output strings given an input. Our WFST (Figure 3) computes a weighted edit distance, and is implemented using OpenFST (Allauzen et al., 2007). It contains a state for each triplet of *(previous, current, next)* phones; conditioned on this state, it emits a character *output* which can be thought of as a possible surface realization of *current* in its particular environment. The *output* can be the empty string ϵ, in which case *current* is deleted. The machine can also insert characters at any point in the string, by transitioning to an insert state *(previous, ϵ, current)* and then returning while emitting some new character.

The transducer is parameterized by the probabilities of the arcs. For instance, all arcs leaving the state (•, ð, i) consume the character ð and emit some character c with probability $p(c|\bullet, \text{ð}, \text{i})$. Following

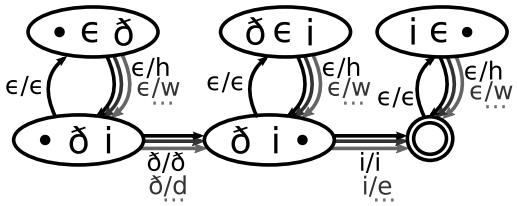

Figure 3: The fragment of the transducer responsible for input string [ði] "the". "..." represents an output arc for each possible character, including the empty string ϵ; • is the word boundary marker.

Dreyer et al. (2008), we parameterize these distributions with a log-linear model. The model features are based on articulatory phonetics and distinguish three dimensions of sound production: voicing, place of articulation and manner of articulation.

Features are generated from four positional templates (Figure 4): *(curr)→out*, *(prev, curr)→out*, *(curr, next)→out* and *(prev, curr, next)→out*. Each template is instantiated once per articulatory dimension, with *prev*, *curr*, *next* and *out* replaced by their values for that dimension: for instance, there are two voicing values, *voiced* and *unvoiced*[1] and the *(curr)→out* template for [ð] producing [d] would be instantiated as *(voiced)→voiced*. To capture trends specific to particular sounds, each template is instantiated again using the actual symbol for *curr* and articulatory values for everything else (e.g., [ð]→*unvoiced*). An additional template, →*out*, captures the marginal frequency of the output symbol. There are also faithfulness features, *same-sound*, *same-voice*, *same-place* and *same-manner* which check if *curr* is exactly identical to *out* or shares the exact value of a particular feature.

Our choice of templates and features is based on standard linguistic principles: we expect that changing only a single articulatory dimension will be more acceptable than changing several, and that the articulatory dimensions of context phones are important because of assimilatory and dissimilatory processes (Hayes, 2011). In modern phonetics and phonology, these generalizations are usually expressed as Optimality Theory constraints; log-linear models such as ours have previously been used to implement stochastic Optimality Theory models (Goldwater and Johnson, 2003; Hayes and Wilson, 2008).

[1]We use seven place values and five manner values (stop, nasal stop, fricative, vowel, other). Empty segments like ϵ and • are assigned a special value "no-value" for all features.

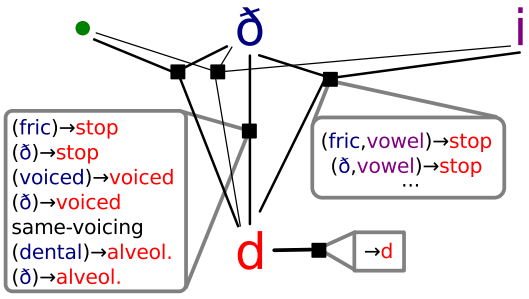

Figure 4: Some features generated for $(\bullet, \text{ð}, \text{i}) \rightarrow d$. Each black factor node corresponds to a positional template. The features instantiated for the *(curr)→out* and →*out* template are shown in full, and we show some of the features for the *(curr,next)→out* template.

4 Inference

Global optimization of the model posterior is difficult; instead we use Viterbi EM (Spitkovsky et al., 2010; Allahverdyan and Galstyan, 2011). We begin with a simple initial transducer and alternate between two phases: clustering together surface forms, and reestimating the transducer parameters. We iterate this procedure until convergence (when successive clustering phases find nearly the same set of merges); this tends to take about 5 or 6 iterations.

In our clustering phase, we improve the model posterior as much as possible by greedily making *type merges*, where, for a pair of intended word forms u and v, we replace all instances of $x_i = u$ with $x_i = v$. We maintain the invariant that each intended word form's most common surface form must be itself; this biases the model toward solutions with low distortion in the transducer.

4.1 Scoring merges

We write the change in the log posterior probability of the model resulting from a type merge of u to v as $\Delta(u, v)$, which factors into two terms, one depending on the surface string and the transducer, and the other depending on the string of intended words. In order to ensure that each intended word form's most common surface form is itself, we define $\Delta(u, v) = -\infty$ if u is more common than v.

We write the log probability of x being transduced to s as $T(s|x)$. If we merge u into v, we no longer

need to produce any surface forms from u, but instead we must derive them from v. If $\#(\cdot)$ counts the occurrences of some event in the current state of the model, the transducer component of Δ is:

$$\Delta_T = \sum_s \#(x_i{=}u, s_i{=}s)(T(s|v) - T(s|u)) \quad (1)$$

This term is typically negative, voting against a merge, since u is more similar to itself than to v.

The language modeling term relating to the intended string again factors into multiple components. The probability of a particular l_i, x_i, r_i can be broken into $p(x_i)p(l_i|x_i)p(r_i|x_i)$ according to the model. We deal first with the $p(x_i)$ unigram term, considering all tokens where $x_i \in \{u, v\}$ and computing the probability $p_u = p(x_i = u | x_i \in \{u, v\})$. By definition of a Dirichlet process, the marginal over a subset of the variables will be Dirichlet, so for $\alpha > 1$ we have the MAP estimate:

$$p_u = \frac{\#(x_i{=}u) + \alpha - 1}{\#(x_i \in \{u, v\}) + 2(\alpha - 1)} \quad (2)$$

$p_v = p(x_i = v | x_i \in \{u, v\})$ is computed similarly. If we decide to merge u into v, however, the probability $p(x_i = v | x_i \in \{u, v\})$ becomes 1. The change in log-probability resulting from the merge is closely related to the entropy of the distribution:

$$\Delta_U = -\#(x_i{=}u)\log(p_u) - \#(x_i{=}v)\log(p_v) \quad (3)$$

This change must be positive and favors merging.

Next, we consider the change in probability from the left contexts (the derivations for right contexts are equivalent). If u and v are separate words, we generate their left contexts from different distributions $p(l|u)$ and $p(l|v)$, while if they are merged, we must generate all the contexts from the same distribution $p(l|\{u, v\})$. This change is:

$$\Delta_L = \sum_l \#(l, u)\{\log(p(l|\{u, v\})) - \log(p(l|u)\} + \sum_l \#(l, v)\{\log(p(l|\{u, v\})) - \log(p(l|v)\}$$

In a full Bayesian model, we would integrate over the parameters of these distributions; instead, we use Kneser-Ney smoothing (Kneser and Ney, 1995) which has been shown to approximate the MAP solution of a hierarchical Pitman-Yor model (Teh, 2006; Goldwater et al., 2006). The Kneser-Ney discount[2] d is a tunable parameter of our system, and controls whether the term favors merging or not. If d is small, $p(l|u)$ and $p(l|v)$ are close to their maximum-likelihood estimates, and Δ_L is similar to a Jensen-Shannon divergence; it is always negative and discourages mergers. As d increases, however, $p(l|u)$ for rare words approaches the prior distribution; in this case, merging two words may result in better posterior parameters than estimating both separately, since the combined estimate loses less mass to discounting.

Because neither the transducer nor the language model are perfect models of the true distribution, they can have incompatible dynamic ranges. Often, the transducer distribution is too peaked; to remedy this, we downweight the transducer probability by λ, a parameter of our model, which we set to .5. Downweighting of the acoustic model versus the LM is typical in speech recognition (Bahl et al., 1980).

To summarize, the full change in posterior is:

$$\Delta(u, v) = \Delta_U + \Delta_L + \Delta_R + \lambda\Delta_T \quad (4)$$

There are four parameters. The transducer regularization $r = 1$ and unigram prior $\alpha = 2$, which we set ad-hoc, have little impact on performance. The Kneser-Ney discount $d = 2$ and transducer downweight $\lambda = .5$ have more influence and were tuned on development data.

4.2 Clustering algorithm

In the clustering phase, we start with an initial solution in which each surface form is its own intended pronunciation and iteratively improve this solution by merging together word types, picking (approximately) the best merger at each point.

We begin by computing a set of *candidate* mergers for each surface word type u. This step saves time by quickly rejecting mergers which are certain to get very low transducer scores. We reject a pair u, v if the difference in their length is greater than 4, or if both words are longer than 4 segments, but, when we consider them as unordered bags of segments, the Dice coefficient between them is less than .5.

For each word u and all its candidates v, we compute $\Delta(u, v)$ as in Equation 4. We keep track of the

[2] We use one discount, rather than several as in modified KN.

Input: vocabulary of surface forms u
Input: $C(u)$: candidate intended forms of u
Output: $intend(u)$: intended form of u
foreach $u \in vocab$ **do**
  // initialization
  $v^*(u) \leftarrow argmax_{v \in C(u)} \Delta(u, v)$;
  $\Delta^*(u) \leftarrow \Delta(u, v^*(u))$
  $intend(u) \leftarrow u$
  add u to queue Q with priority $\Delta^*(u)$)
while $top(Q) > -\infty$ **do**
  $u \leftarrow pop(Q)$
  recompute $v^*(u), \Delta^*(u)$
  if $\Delta^*(u) > 0$ **then**
    // merge u with best merger
    $intend(u) \leftarrow v^*(u)$
    update $\Delta(x, u) \quad \forall x : v^*(x) = u$
    remove u from $C(x) \quad \forall x$
    update $\Delta(x, v) \quad \forall x : v^*(x) = v$
    update $\Delta(v, x) \quad \forall x \in C(v)$
    if *updated* $\Delta > \Delta^*$ *for any words* **then**
      reset Δ^*, v^* for those words
    // (these updates can increase a word's priority from $-\infty$)
  else if $\Delta^*(u) \neq -\infty$ **then**
    // reject but leave in queue
    $\Delta^*(u) \leftarrow -\infty$

Algorithm 1: Our clustering phase.

current best target $v^*(u)$ and best score $\Delta^*(u)$, using a priority queue. At each step of the algorithm, we pop the u with the current best $\Delta^*(u)$, recompute its scores, and then merge it with $v^*(u)$ if doing so would improve the model posterior. In an exact algorithm, we would then need to recompute most of the other scores, since merging u and $v^*(u)$ affects other words for which u and $v^*(u)$ are candidates, and also words for which they appear in the context set. However, recomputing all these scores would be extremely time-consuming.[3] Therefore, we recompute scores for only those words where the previous best merger was either u or $v^*(u)$. (If the best merge would *not* improve the probability, we reject it, but since its score might increase if we merge $v^*(u)$, we leave u in the queue, setting its Δ score to $-\infty$; this score will be updated if we merge $v^*(u)$.)

Since we recompute the exact scores $\Delta(u, v)$ immediately before merging u, the algorithm is guaranteed never to reduce the posterior probability. It can potentially make changes in the wrong order, since not all the Δs are recomputed in each step, but most changes do not affect one another, so performing them out of order has no impact. Empirically, we find that mutually exclusive changes (usually of the form (u, v) and (v, w)) tend to differ enough in initial score that they are evaluated in the correct order.

4.3 Training the transducer

To train the transducer on a set of mappings between surface and intended forms, we find the maximum-probability state sequence for each mapping (another application of Viterbi EM) and extract features for each state and its output. Learning weights is then a maximum-entropy problem, which we solve using Orthant-wise Limited-memory Quasi-Newton.[4]

To construct our initial transducer, we first learn weights for the marginal distribution on surface sounds by training the max-ent system with only the bias features active. Next, we manually set weights (Table 1) for insertions and deletions, which do not appear on the surface, and for faithfulness features. Other features get an initial weight of 0.

5 Experiments

5.1 Dataset

Our corpus is a processed version of the Bernstein-Ratner corpus (Bernstein-Ratner, 1987) from CHILDES (MacWhinney, 2000), which contains orthographic transcriptions of parent-child dyads with infants aged 13-23 months. Brent and Cartwright (1996) created a phonemic version of this corpus by extracting all infant-directed utterances and converted them to a phonemic transcription using a dictionary. This version, which contains 9790 utterances (33399 tokens, 1321 types), is now standard for word segmentation, but contains no phonetic variability.

Since producing a close phonetic transcription of this data would be impractical, we instead construct an approximate phonetic version using information from the Buckeye corpus (Pitt et al., 2007). Buckeye is a corpus of adult-directed conversational American English, and has been phonetically transcribed

[3] The transducer scores can be cached since they depend only on surface forms, but the language model scores cannot.

[4] We use the implementation of Andrew and Gao (2007) with an l_2 regularizer and regularization parameter $r = 1$; although this could be tuned, in practice it has little effect on results.

Feature	Weight
output-is-x	marginal $p(x)$
output-is-ϵ	0
same-sound	5
same-{place,voice, manner}	2
insertion	-3

Table 1: Initial transducer weights.

"about"	ahbawt:15, bawt:9, ihbawt:4, ahbawd:4, ihbawd:4, ahbaat:2, baw:1, ahbaht:1, erbawd:1, bawd:1, ahbaad:1, ahpaat:1, bah:1, baht:1, ah:1, ahbahd:1, ehbaat:1, ahbaed:1, ihbaht:1, baot:1
"wanna"	waanah:94, waanih:37, wahnah:16, waan:13, wahneh:8, wahnih:5, wahney:3, waanlih:3, wehnih:2, waaneh:2, waonih:2, waaah:1, wuhnih:1, wahn:1, waantah:1, waanaa:1, wowiy:1, waaih:1, wah:1, waaniy:1

Table 2: Empirical distribution of pronunciations of "about" and "wanna" in our dataset.

by hand to indicate realistic pronunciation variability. To create our phonetic corpus, we replace each phonemic word in the Bernstein-Ratner-Brent corpus with a phonetic pronunciation of that word sampled from the empirical distribution of pronunciations in Buckeye (Table 2). If the word never occurs in Buckeye, we use the original phonemic version.

Our corpus is not completely realistic as a sample of child-directed speech. Since each pronunciation is sampled independently, it lacks coarticulation and prosodic effects, and the distribution of pronunciations is derived from adult-directed rather than child-directed speech. Nonetheless, it represents phonetic variability more realistically than the Bernstein-Ratner-Brent corpus, while still maintaining the lexical characteristics of infant-directed speech (as compared to the Buckeye corpus, with its much larger vocabulary and more complex language model).

We conduct our development experiments on the first 8000 input utterances, holding out the remaining 1790 for evaluation. For evaluation experiments, we run the system on all 9790 utterances, reporting scores on only the last 1790.

5.2 Metrics

We evaluate our results by generalizing the three segmentation metrics from Goldwater et al. (2009): word boundary F-score, word token F-score, and lexicon (word type) F-score.

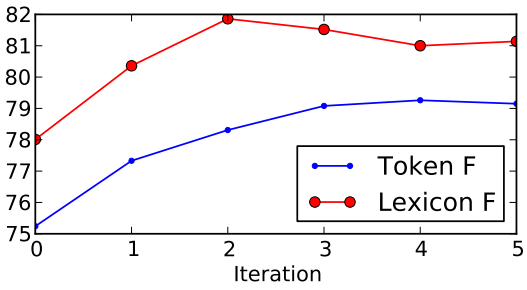

Figure 5: System scores over 5 iterations.

In our first set of experiments we evaluate how well our system clusters together surface forms derived from the same intended form, assuming gold standard word boundaries. We do not evaluate the induced intended forms directly against the gold standard intended forms—we want to evaluate cluster memberships and not labels. Instead we compute a one-to-one mapping between our induced lexical items and the gold standard, maximizing the agreement between the two (Haghighi and Klein, 2006). Using this mapping, we compute *mapped token F-score*[5] and *lexicon F-score*.

In our second set of experiments, we use unknown word boundaries and evaluate the segmentations. We report the standard word boundary F and *unlabeled word token F* as well as mapped F. The unlabeled token score counts correctly segmented tokens, whether assigned a correct intended form or not.

5.3 Known word boundaries

We first run our system with known word boundaries (Table 3). As a baseline, we treat every surface token as its own intended form (*none*). This baseline has fairly high accuracy; 65% of word tokens receive the most common pronunciation for their intended form.[6] As an upper bound, we find the best intended form for each surface type (*type ubound*). This correctly resolves 91% of tokens; the remaining error is due to homophones (surface types corresponding to more than one intended form). We also test our sys-

[5]When using the gold word boundaries, the precision and recall are equal and this is is the same as the accuracy; in segmentation experiments the two differ, because with fewer segmentation boundaries, the system proposes fewer tokens. Only correctly segmented tokens which are also mapped to the correct form count as matches.

[6]The lexicon recall is not quite 100% because one rare word appears only as a homophone of another word.

System	Tok F	Lex P	Lex R	Lex F
none	65.4	50.2	99.7	66.7
initializer	75.2	83.2	73.3	78.0
system	79.2	87.1	75.9	81.1
oracle trans.	82.7	88.7	83.8	86.2
type ubound	91.0	97.5	98.0	97.7

Table 3: Results on 1790 utterances (known boundaries).

	Boundaries			Unlabeled Tokens		
	P	R	F	P	R	F
no var.	90.1	80.3	84.9	74.5	68.7	71.5
w/var.	70.4	93.5	80.3	56.5	69.7	62.4

Table 4: Degradation in `dpseg` segmentation performance caused by pronunciation variation.

	Mapped Tokens			Lexicon (types)		
	P	R	F	P	R	F
none	39.8	49.0	43.9	37.7	49.1	42.6
init	42.2	52.0	56.5	50.1	40.8	45.0
sys	44.2	54.5	48.8	48.6	43.1	45.7

Table 5: Results on 1790 utterances (induced boundaries).

tem using an oracle transducer (*oracle trans.*)—the transducer estimated from the upper-bound mapping. This scores 83%, showing that our articulatory feature set captures most, but not all, of the available information. At the beginning of bootstrapping, our system (*init*) scores 75%, but this improves to 79% after five iterations of reestimation (*system*). Most learning occurs in the first two or three iterations (Figure 5).

To determine the importance of different parts of our system, we run a few ablation tests on development data. Context information is critical to obtain a good solution; setting Δ_L and Δ_R to 0 lowers our dev token F-score from 83% to 75%. Initializing all feature weights to 0 yields a poor initial solution (18% dev token F instead of 75%), but after learning the result is only slightly lower than using the weights in Table 1 (78% rather than 80%), showing that the system is quite robust to initialization.

5.4 Unknown word boundaries

As a simple extension of our model to the case of unknown word boundaries, we interleave it with an existing model of word segmentation, `dpseg` (Goldwater et al., 2009).[7] In each iteration, we run the segmenter, then bootstrap our model for five iterations on the segmented output. We then concatenate the intended word sequence proposed by our model to produce the next iteration's segmenter input.

Phonetic variation is known to reduce the performance of `dpseg` (Fleck, 2008; Boruta et al., 2011) and our experiments confirm this (Table 4). Using induced word boundaries also makes it harder to recover the lexicon (Table 5), lowering the baseline F-score from 67% to 43%. Nevertheless, our system improves the lexicon F-score to 46%, with token F rising from 44% to 49%, demonstrating the system's ability to work without gold word boundaries. Unfortunately, performing multiple iterations between the segmenter and lexical-phonetic learner has little further effect; we hope to address this issue in future.

6 Conclusion

We have presented a noisy-channel model that simultaneously learns a lexicon, a bigram language model, and a model of phonetic variation, while using only the noisy surface forms as training data. It is the first model of lexical-phonetic acquisition to include word-level context and to be tested on an infant-directed corpus with realistic phonetic variability. Whether trained using gold standard or automatically induced word boundaries, the model recovers lexical items more effectively than a system that assumes no phonetic variability; moreover, the use of word-level context is key to the model's success. Ultimately, we hope to extend the model to jointly infer word boundaries along with lexical-phonetic knowledge, and to work directly from acoustic input. However, we have already shown that lexical-phonetic learning from a broad-coverage corpus is possible, supporting the claim that infants acquire lexical and phonetic knowledge simultaneously.

Acknowledgements

This work was supported by EPSRC grant EP/H050442/1 to the second author.

[7] dpseg1.2 from `http://homepages.inf.ed.ac.uk/sgwater/resources.html`

References

Guillaume Aimetti. 2009. Modelling early language acquisition skills: Towards a general statistical learning mechanism. In *Proceedings of the Student Research Workshop at EACL*.

Armen Allahverdyan and Aram Galstyan. 2011. Comparative analysis of Viterbi training and ML estimation for HMMs. In *Advances in Neural Information Processing Systems (NIPS)*.

Cyril Allauzen, Michael Riley, Johan Schalkwyk, Wojciech Skut, and Mehryar Mohri. 2007. OpenFst: A general and efficient weighted finite-state transducer library. In *Proceedings of the Ninth International Conference on Implementation and Application of Automata, (CIAA 2007)*, volume 4783 of *Lecture Notes in Computer Science*, pages 11–23. Springer. `http://www.openfst.org`.

Galen Andrew and Jianfeng Gao. 2007. Scalable training of L1-regularized log-linear models. In *ICML '07*.

Lalit Bahl, Raimo Bakis, Frederick Jelinek, and Robert Mercer. 1980. Language-model/acoustic-channel-model balance mechanism. Technical disclosure bulletin Vol. 23, No. 7b, IBM, December.

Nan Bernstein-Ratner. 1987. The phonology of parent-child speech. In K. Nelson and A. van Kleeck, editors, *Children's Language*, volume 6. Erlbaum, Hillsdale, NJ.

Luc Boruta, Sharon Peperkamp, Benoît Crabbé, and Emmanuel Dupoux. 2011. Testing the robustness of online word segmentation: Effects of linguistic diversity and phonetic variation. In *Proceedings of the 2nd Workshop on Cognitive Modeling and Computational Linguistics*, pages 1–9.

Michael R. Brent and Timothy Cartwright. 1996. Distributional regularity and phonotactic constraints are useful for segmentation. *Cognition*, 61:93–125.

Michael R. Brent. 1999. An efficient, probabilistically sound algorithm for segmentation and word discovery. *Machine Learning*, 34:71–105, February.

Robert Daland and Janet B. Pierrehumbert. 2011. Learning diphone-based segmentation. *Cognitive Science*, 35(1):119–155.

Markus Dreyer, Jason R. Smith, and Jason Eisner. 2008. Latent-variable modeling of string transductions with finite-state methods. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 1080–1089, Stroudsburg, PA, USA. Association for Computational Linguistics.

Joris Driesen, Louis ten Bosch, and Hugo Van hamme. 2009. Adaptive non-negative matrix factorization in a computational model of language acquisition. In *Proceedings of Interspeech*.

Emmanuel Dupoux, Guillaume Beraud-Sudreau, and Shigeki Sagayama. 2011. Templatic features for modeling phoneme acquisition. In *Proceedings of the 33rd Annual Cognitive Science Society*.

Naomi Feldman, Thomas Griffiths, and James Morgan. 2009. Learning phonetic categories by learning a lexicon. In *Proceedings of the 31st Annual Conference of the Cognitive Science Society*.

Naomi Feldman. 2011. *Interactions between word and speech sound categorization in language acquisition*. Ph.D. thesis, Brown University.

Margaret M. Fleck. 2008. Lexicalized phonotactic word segmentation. In *Proceedings of ACL-08: HLT*, pages 130–138, Columbus, Ohio, June. Association for Computational Linguistics.

Sharon Goldwater and Mark Johnson. 2003. Learning OT constraint rankings using a maximum entropy model. In J. Spenader, A. Eriksson, and Osten Dahl, editors, *Proceedings of the Stockholm Workshop on Variation within Optimality Theory*, pages 111–120, Stockholm. Stockholm University.

Sharon Goldwater, Tom Griffiths, and Mark Johnson. 2006. Interpolating between types and tokens by estimating power-law generators. In *Advances in Neural Information Processing Systems (NIPS) 18*.

Sharon Goldwater, Thomas L. Griffiths, and Mark Johnson. 2009. A Bayesian framework for word segmentation: Exploring the effects of context. In *In 46th Annual Meeting of the ACL*, pages 398–406.

Aria Haghighi and Dan Klein. 2006. 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. Association for Computational Linguistics.

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

Bruce Hayes. 2011. *Introductory Phonology*. John Wiley and Sons.

Aren Jansen, Ken Church, and Hynek Hermansky. 2010. Towards spoken term discovery at scale with zero resources. In *Proceedings of Interspeech*, pages 1676–1679.

Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for M-gram language modeling. In *Proc. ICASSP '95*, pages 181–184, Detroit, MI, May.

Brian MacWhinney. 2000. *The CHILDES Project: Tools for Analyzing Talk. Vol 2: The Database*. Lawrence Erlbaum Associates, Mahwah, NJ, 3rd edition.

Fergus R. McInnes and Sharon Goldwater. 2011. Unsupervised extraction of recurring words from infant-directed speech. In *Proceedings of the 33rd Annual Conference of the Cognitive Science Society*.

Alex Park and James R. Glass. 2008. Unsupervised pattern discovery in speech. *IEEE Transactions on Audio, Speech and Language Processing*, 16:186–197.

Fernando Pereira, Michael Riley, and Richard Sproat. 1994. Weighted rational transductions and their application to human language processing. In *Workshop on Human Language Technology*.

Mark A. Pitt, Laura Dilley, Keith Johnson, Scott Kiesling, William Raymond, Elizabeth Hume, and Eric Fosler-Lussier. 2007. Buckeye corpus of conversational speech (2nd release).

Okko Räsänen. 2011. A computational model of word segmentation from continuous speech using transitional probabilities of atomic acoustic events. *Cognition*, 120(2):28.

Anton Rytting. 2007. *Preserving Subsegmental Variation in Modeling Word Segmentation (Or, the Raising of Baby Mondegreen)*. Ph.D. thesis, The Ohio State University.

Valentin I. Spitkovsky, Hiyan Alshawi, Daniel Jurafsky, and Christopher D. Manning. 2010. Viterbi training improves unsupervised dependency parsing. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, pages 9–17, Uppsala, Sweden, July. Association for Computational Linguistics.

Daniel Swingley. 2005. Statistical clustering and the contents of the infant vocabulary. *Cognitive Psychology*, 50:86–132.

Yee Whye Teh. 2006. A hierarchical Bayesian language model based on Pitman-Yor processes. In *Proceedings of the Association for Computational Linguistics*, pages 985–992, Sydney, Australia, July.

Gautam K. Vallabha, James L. McClelland, Ferran Pons, Janet F. Werker, and Shigeaki Amano. 2007. Unsupervised learning of vowel categories from infant-directed speech. *Proceedings of the National Academy of Sciences*, 104(33):13273–13278.

Balakrishnan Varadarajan, Sanjeev Khudanpur, and Emmanuel Dupoux. 2008. Unsupervised learning of acoustic sub-word units. In *Proceedings of the Association for Computational Linguistics: Short Papers*, pages 165–168.

Anand Venkataraman. 2001. A statistical model for word discovery in transcribed speech. *Computational Linguistics*, 27(3):351–372.