Variational Bayesian Inference for Unsupervised Lexicon Discovery

Elias Stengel-Eskin, BA&Sc., Honours Cognitive Science
Joint work with Emily Kellison-Linn[*] and Prof. Timothy O'Donnell[†]

1 Abstract

Speech is a defining human trait, yet much of our mental framework for producing and processing spoken language remains a mystery. Computational models can help elucidate some these internal structures while allowing us to engineer improved speech technology. We present a variational Bayesian inference model for the fully unsupervised discovery of phones, sub-word units, and words directly from an acoustic input. Extending the state-of-the-art model for unsupervised lexicon discovery introduced by Lee et al. (2015), which relied on sampling, our framework permits parallelization and distribution to multiple cores, promising speed improvements and scalability. We give an introduction to variational Bayesian methodology and use it to re-frame the original model. We highlight some results from Lee et al. (2015) which underscore the capabilities of the model, and discuss improvements made to similar models through the application of variational Bayesian methods. With these advances in mind, we consider future experiments made feasible by our variational system and suggest a range potential uses for completely unsupervised language-independent models such as ours.

2 Introduction

2.1 Background

Spoken language allows humans to communicate effectively with each other, making it a fundamental characteristic of our species. Studying the computational underpinnings of speech production and perception affords us a more extensive understanding of the phenomenon (and of language as a whole) while helping us to develop improved computer-human interactions. Many current state-of-the-art spoken language systems focus on supervised learning—that is, training a system on copious amounts of labeled data. Producing such data is costly and time-consuming, meaning that sufficient quantities exist only for a small fraction of the world's languages, and even then, the quality is often inadequate. This contributes to the underrepresentation of many world languages and language families, particularly those spoken in the developing world. In addition, a supervised approach does not offer an entirely accurate model of human language capabilities. Language acquisition is unsupervised in the

[*]Department of Linguistics - McGill University. emily.kellison-linn@mail.mcgill.ca
[†]Department of Linguistics - McGill University. tim.odonnell@mcgill.ca

machine-learning sense; we infer linguistic structure and rules implicitly from unlabeled data. These factors motivate an unsupervised approach which incorporates existing theories about language and cognition in order to eliminate the need for labeled training data. The practically unbounded amount of unlabeled speech data which exists in the form of videos, audiobooks, and archival recordings (to name a few sources) particularly incentivizes unsupervised algorithms for speech processing, especially ones that can interface directly with an acoustic signal.

2.2 The model

The specific language phenomenon we model is lexicon discovery, for which we implement an unsupervised learning algorithm. Extending the state-of-the-art framework introduced by Lee et al. (2015), our model constructs a complete hierarchy of linguistic units (phonemes, morphemes, and words) directly from an acoustic input. The *unsupervised lexicon discovery* model (ULD) presented by Lee et al. (2015) was the first to jointly model the induction of a full stack of hierarchical components, combining and building on earlier work in phoneme discovery (Lee and Glass, 2012) and in unsupervised syntactic and morphemic structure inference (Johnson et al., 2007; O'Donnell, 2015) . ULD is composed of three main components: a Dirichlet process hidden Markov model (DPHMM) for segmenting continuous audio input and hypothesizing a sequence of reusable phone-like units (PLUs), an adaptor grammar which recognizes and stores frequently reused syntactic structures based on an underlying grammar—in this case, grouping PLUs into morphemes and words—and finally a noisy channel model, which allows substitutions, insertions, and deletions to occur between the inputs and outputs of the DPHMM and adaptor grammar components, approximating a phonological system. The noisy channel is crucial to the *joint learning* nature of the model in that it allows the DPHMM and adaptor grammar to constrain one another. This type of joint learning framework, whereby multiple linguistic phenomena are modeled simultaneously, has been shown to improve accuracy (Johnson, 2008).

ULD uses nonparametric Bayesian inference to implement a fully unsupervised learning model. In Bayesian modeling, we posit unobserved (latent) variables which are conditionally dependent on the observed data. Defining the model in this way allows for the dynamic updating of the latent variables (the hypothesis) according to the data (the evidence) while incorporating prior theoretical assumptions about the problem, yielding a powerful method for unsupervised learning.

A Bayesian model is defined as follows: let Z be the set of latent variables, let X be the set of observed data, and let Φ be a set of static model hyperparameters specified by the user. Then by Bayes' rule we have:

$$P(Z|X,\Phi) = \frac{P(X|Z,\Phi)P(Z,\Phi)}{\sum\limits_{z\in Z} P(X|Z,\Phi)P(Z,\Phi)}$$

The numerator is often called the *generative model*, and gives a joint distribution on data and latent variables. In general, we define Bayesian models in terms of a generative model, where draws from the latent random variables are used to generate data. It consists of the conditional probability of the data given the latent variables defining the model (referred to as the *likelihood*), multiplied by the prior probability of the latent variables (known as the *prior*). This gives us an unnormalized measure of the likelihood of generating the data from the model. However, in order to obtain a proper probability we need to divide by a normalizing constant—the probability of the data. We obtain this probability by integrating (or summing, in the discrete case) over all possible ways of obtaining the data—that is, all the latent variables. It is easy to see why, given a sufficiently complex model and a large amount of data, evaluating this integral becomes computationally intractable.

In absence of a method to compute the marginal probability of the data directly, there are two common approaches to doing Bayesian inference. The first, used in the original ULD model, is sampling, which capitalizes on the fact that given a generative model, we can approximate the *posterior* (the left-hand side of Bayes' rule) by randomly sampling from the generative model, eliminating the need to calculate the *marginal likelihood*. This technique has played a major role in Bayesian inference, but is difficult to parallelize. This makes it nearly impossible to scale such algorithms to the types of large speech datasets available (Blei et al., 2017). Variational Bayesian inference presents an often faster alternative for approximating the posterior distribution.

2.3 Variational Bayesian inference

Variational Bayesian inference re-casts the challenge of computing the posterior distribution on latent variables as an optimization problem. By iteratively maximizing a lower bound on the (incomputable) marginal likelihood, framed as a *variational distribution* over latent variables, the algorithm yields an approximation of the posterior. This intuition is clarified by (3). This strategy lends itself well to parallelization across multiple cores and interfacing with tools such as the MapReduce framework for cluster computation (Zhai et al., 2012). In order obtain the approximation of the posterior, we introduce a family of variational distributions $q_\nu(Z)$ which have the same support as the posterior ($p(Z|X, \Phi)$) indexed by variational parameter ν, used to adjust the distribution q.

2.3.1 Computing the ELBO

Our goal is to find the $q_\nu(Z)$ which minimizes the Kullback-Liebler (KL) divergence between $q_\nu(Z)$ and $p(Z|X, \Phi)$, or $D_{KL}(q_\nu(Z) \mid\mid p(Z|X, \Phi))$, where KL-divergence is a measure of the difference between to probability distributions. KL-divergence is given by

$$\begin{aligned} D_{KL}(q_\nu(Z) \mid\mid p(Z \mid X, \Phi)) &= \mathbb{E}_q[\log \ \frac{q_\nu(Z)}{p(Z \mid X, \Phi)}] \\ &= \mathbb{E}_q[\log \ q_\nu(Z)] - \mathbb{E}_q[\log \ p(Z, X \mid \Phi)] + \log \ p(X \mid \Phi) \end{aligned} \tag{1}$$

where $\mathbb{E}_q$ indicates taking the expected value with respect to q (see Appendix A for further explanation of this notation). Unfortunately, the third value, $\log \ p(X \mid \Phi)$, is the marginal likelihood, requiring the intractable computation we seek to avoid, so we cannot directly compute KL divergence. However, this equation does generate a valuable result: a lower bound on the marginal called the evidence lower bound (ELBO). To obtain this result, first note that because of what KL divergence represents, it can never be negative. This gives us:

$$\begin{aligned} 0 &\leq \mathbb{E}_q[\log\ q(Z)] - \mathbb{E}_q[\log\ p(Z, X \mid \Phi)] + \log\ p(X \mid \Phi) \\ -\log\ p(X \mid \Phi) &\leq \mathbb{E}_q[\log\ q(Z)] - \mathbb{E}_q[\log\ p(Z, X \mid \Phi)] \\ \log\ p(X \mid \Phi) &\geq \mathbb{E}_q[\log\ p(Z, X \mid \Phi)] - \mathbb{E}_q[\log\ q(Z)] \end{aligned} \tag{2}$$

Thus

$$\begin{aligned} ELBO(q) &= \mathbb{E}_q[\log\ p(Z, X \mid \Phi)] - \mathbb{E}_q[\log\ q(Z)] \\ &= \mathbb{E}_q[\log\ p(Z, X \mid \Phi)] + H(q) \end{aligned}$$

where $H(q)$ is the entropy of the distribution q. This derivation yields an important fact:

$$\log\ p(X \mid \Phi) - D_{KL}(q(Z) \mid\mid p(Z \mid X \mid \Phi)) = ELBO(q) \tag{3}$$

(3) provides the explanation to the previous intuition that maximizing the ELBO allows us to minimize the KL divergence—the maximal ELBO is $\log\ p(X)$. When $ELBO(q) = \log\ p(X \mid \Phi)$ the KL-divergence must be 0 (Blei et al., 2017). For an expanded derivation, as well as an equivalent derivation using Jensen's inequality, see Appendix A.

2.3.2 Mean-field approximation

One of the fundamental reasons why we cannot compute the posterior directly is the presence of conditional dependencies between latent variables in it—variables that are independent in the generative process may become conditionally dependent in the posterior. Since the variational distribution need only be an approximation, we make a mean-field assumption—that is, we assume that the variational distribution $q_\nu(Z)$ has none of these conditional dependencies, or

$$q_\nu(Z) = \prod_{z_i \in Z} q_{\nu_i}(z_i)$$

This is a powerful assumption allowing us to optimize each variational distribution iteratively. While holding all other variational distributions constant, we can find the variational parameters for $q_{\nu_i}(z_i)$ that maximize the marginal likelihood. By the chain rule, we derive the following lower bound for each variational distribution as:

$$\mathcal{L}_i(q_{\nu_i}(z_i)) = \mathbb{E}_q[\log\ p(z_i | Z_{-i}, X, \Phi)] - \mathbb{E}_q[\log\ q_{\nu_i}(z_i)] \tag{4}$$

where Z_{-i} indicates the set of all latent variables in Z which are not z_i.

2.3.3 Updates

Recall that the goal is to maximize the lower bound on the variational distribution over each latent variable $\mathcal{L}_i$, which is accomplished by adjusting each ν_i. The value for ν_i that locally maximizes the function $\mathcal{L}_i$ is found by setting the first derivative with respect to ν_i equal to 0 and solving for ν_i. Taking the derivative of the objective function is costly; this cost is compounded by the need to recompute the derivative at every parameter update, and for every variational distribution parameter. Using exponential family random variables allows us to take advantage of some convenient mathematical facts and avoid this costly computation entirely. When each $q_{\nu_i}(z_i)$ and each distribution in the generative model are in the exponential family, we obtain the following closed-form update for each ν_i:

$$\nu_i = \mathbb{E}_q[g_i(Z_{-i}, X, \Phi)] = \mathbb{E}_q \begin{bmatrix} \phi_1 + \sum\limits_{z_n \in Z_{-i}} t(x_n, z_n) \\ \phi_2 + N \end{bmatrix} \tag{5}$$

where $g_i(Z_{-i}, X, \Phi)$ is a function which gives the natural parameters of the exponential family distribution *in the posterior*, ϕ_1 and ϕ_2 are the parameters for the exponential family distribution in the *prior*, N is the total number of datapoints, and $t(x_n, z_n)$ is the *sufficient statistic* of the prior distribution—in many cases, this is simply a count of occurrences of z_n. For a more in-depth explanation of this result, see Appendix B.

2.3.4 Coordinate ascent

We now have a way of optimizing each variational distribution by setting the parameters to the expected value of the natural parameters in the posterior, conditioned on the other latent variables and the data. If the objective function could be formulated as a strictly convex function, then a single update of the variational parameters would be sufficient to find a solution, since a local optimum in a convex function is a global one. However, given the composite nature of most Bayesian models, this is seldom the case, and we use a non-convex optimization algorithm to iteratively find local maxima, with the ultimate goal of converging on the global maximum. The Coordinate Ascent Variational Inference (CAVI) provides an interface for optimizing the ELBO. It is worth noting that the CAVI algorithm is a generalization of the well-known Expectation-Maximization algorithm (Dempster et al., 1977); where the latter gives a point estimate of the posterior, the former returns an approximation of the full distribution—a more data-rich representation. In CAVI, we alternate between computing the objective function (analogous to the expectation step) and updating the variational parameters (analogous to the maximization step) (Neal and Hinton, 1998).

Algorithm 1: The CAVI algorithm

```
initialize each $\nu_i$
while not ELBO converged do
    for each variational parameter $\nu_i$ do
        $\nu_i = \mathbb{E}_q[g_i(Z_{-i}, X, \Phi)]$
    re-compute ELBO $\mathcal{L}(q) = \mathbb{E}_q[\log\ p(Z, X)] + H(q)$
```

The initialization step for each ν_i can be random, but this is not required. Often, we let $q_{\nu_i}(z_i)$ be a distribution of the same type as in the generative model, and initialize it with uniform or random parameters. However, we may choose the initial parameters more deliberately and encode some bias in the variational distribution, with the caveat (and occasionally the benefit) that varying initializations can lead to convergence on different local optima (Blei et al., 2017)

3 The Generative Model

The ULD generative model can be broken up into roughly three parts: the adaptor grammar, the noisy channel, and the Dirichlet process hidden Markov model (DPHMM). From a top-down perspective, the adaptor grammar parses a sequence of top-level phone-like units (PLUs) into morphemes and words, building a syntactic tree. The noisy channel, using a set of edit operations (insertion, deletion, and substitution) maps the yield of this tree (top-level PLUs) to bottom-level PLUs, modeling some of the phonological processes which occur during speech production. Finally, the DPHMM takes these bottom-level PLUs and finds an acoustic signal (represented by 39-dimensional Mel frequency cepstral coefficient (MFCC) vectors) that could have generated them. It is necessary to point out that this top-down view of the model does not accurately reflect the complete flow of information in it. Due to its joint learning properties, every component affects every other—for example, the DPHMM first infers the PLU inventory which the adaptor grammar needs to generate trees, and makes adjustments to PLU boundaries which can affect the adaptor grammar's parses. This complex generative model contains a multitude of inference steps, making it an ideal candidate for the application of variational Bayesian methods. The following sections offer more detailed descriptions of each model and their respective latent variables.

3.1 Adaptor grammars

First developed by Johnson et al. (2007), adaptor grammars take as input some context-free grammar and a set of strings which can be parsed by that grammar. Using a non-parametric distribution, they adjust the probability of different rule expansions in the context free grammar, thereby storing derivational trees while biasing the reuse of frequently occurring ones; these stored fragments reveal patterns in the linguistic structure of the data. By increasing the likelihood of reusing a tree according to its frequency, adaptor grammars instantiate a "rich get richer" dynamic where common trees become more likely than rare ones. In ULD, we use adaptor grammars to group discovered PLUs into morphemes and words. Before formally defining adaptor grammars, we need to define context free grammars, probabilistic context free grammars, and the Pitman-Yor Process.

3.1.1 Context-free Grammars and probabilistic context-free grammars

A context-free grammar (CFG) is a tuple (N, E, R, S) where N is a set of nonterminals symbols, E is a set of terminal symbols disjoint from N, and R is a set of rules of the form $A \to \beta$ where $A \in N$ and $\beta \in (N \cup E)*$ (i.e. any concatenation of symbols in N and E). We constrain the CFGs to be in Chomsky normal form, meaning every rule is either of form $A \to a$ where $a \in E$ or $A \to BC$ where $B \in N$ and $C \in N$. Note that any CFG without epsilon productions (rules that go to the empty string) can be rewritten in Chomsky normal form. (Hopcroft et al., 2006)

Similar to a CFG, a probabilistic context-free grammar (PCFG) is a tuple (N, E, R, S, θ) where N, E, R, S are the same as in a CFG, and θ is a set of probability vectors such that $\sum\limits_{A \to \beta \in R_A} \theta_{A \to \beta} = 1$, where R_A is the set of rules which have nonterminal A on the left-hand side.

3.1.2 Pitman-Yor Process

The Pitman-Yor process (Pitman and Yor, 1997) can be thought of as a distribution on infinite-sided dice, or as generating a partition of integers. Perhaps more intuitively, a Pitman-Yor process defines a distribution over distributions—each draw from a Pitman-Yor process is itself a countably infinite distribution. There are multiple equivalent ways of defining a Pitman-Yor process; we use the stick-breaking construction, as it provides an iterative definition. Given a scale parameter a, a discount factor b and a base distribution G_0, a Pitman-Yor process which partitions $[0, 1]$ into countably infinite segments is defined by algorithm 2:

Algorithm 2: The Pitman-Yor process

```
for $i \in \{1, \ldots\}$ do
    draw $\nu_i \sim Beta(1 - b, a + ib)$
    sample atom $z_i \sim G_0$
    define $\pi_i \triangleq \nu_i \prod_{j=1}^{i-1} (1 - v_j)$
define $G(z) \triangleq \sum_{i=1}^{\infty} \pi_i \delta(z_i, z)$ where $\delta(z_i, z) = 1$ if $z_i = z$, $0$ otherwise
```

(Sethuraman, 1994)

Recall that a draw from a Beta distribution is a biased coin. Intuitively, each ν_i is a coin that gives the probability of stopping at that stick, and $1 - \nu_i$ is the probability of continuing to the next stick. Thus ν_i is the portion of the stick that we "break off" and $1 - \nu_i$ is the remaining length of the stick, from which we break off ν_{i+1} (hence the name of this representation). π_i, the probability of being at stick i, is equivalent to the product of the probability of having passed sticks $1, ..., i - 1$ and the probability of stopping at stick i. The parameters a, b control the concentration and spread of the distribution, determining whether most of the mass will be concentrated on a few sticks or spread out over many. $G(z)$ then gives the probability of a distribution z with the support as the base distribution G_0.

3.1.3 Adaptor grammar definition

With these components, we can formally define an adaptor grammar as a tuple (G, M, a, b, α) where G is a CFG, M is a set of *adapted nonterminals*, a and b are Pitman-Yor process parameters, and α is a set of Dirichlet distribution parameters indexed by each nonterminal in N. The adaptor grammar employs the Pitman-Yor process to generate a distribution over tree fragments (called grammatons) which biases the reuse of common fragments, increasing the probability of stopping at the stick associated with the grammaton when it appears. To formally define an adaptor grammar, let $A_1, \ldots, A_k$ be a reverse topological sorting of the adapted nonterminals in M, such that for all $A_j \in M$ the children of A_j come before A_j in the ordering. Relying on this ordering, algorithms 3 and 4 give a formal definition of the adaptor grammar generative process.

This definition of adaptor grammars gives us the following latent variables:

- z_i: the full derivational trees that yielded the data.

Algorithm 3: Building the grammar

```
# constructing the PCFG
foreach nonterminal $A$ in $N$ do
draw rule weights $\theta_A \sim Dir(\alpha_A)$ ;
# constructing the grammatons $G_A$
for $A \in A_1, ..., A_k$ do
    draw $\pi_A \sim PYP(a_A, b_A)$
    # construct tree $z_{A,i}$
    for $i \in \{1, \ldots\}$ do
        draw rule $A \to B_1 \ldots B_n$ from $R_A$
        set $z_{A,i}$ =
                         $A$
                        /   \
                    $B_1$ ··· $B_n$
        while $z_{A,i}$ has nonterminals in leaves do
            choose a $B$ from $B_1 \ldots B_n$
            if $B$ is non-adapted nonterminal then
                expand $B$ using the PCFG
            else
                expand $B$ using $G_B$
                # guaranteed to exist because of topological ordering
        for $i \in \{1, \ldots\}$ do
            set $G_A(z_{A,i}) = \pi_{A_i}$
```

Algorithm 4: Generating data

```
# generating derivation trees $z_i$
for $i \in \{1, \ldots\}$ do
    if $S$ is adapted nonterminal then
        draw $z_i \sim G_S$
    else
        draw $S \to B_1 \ldots B_n$ from $R_S$
        set $z_i$ =
                         $S$
                        /   \
                    $B_1$ ··· $B_n$
    while $z_i$ has nonterminals in leaves do
        choose a $B$ from $B_1 \ldots B_n$
        if $B$ is non-adapted nonterminal then
            expand $B$ using the PCFG
        else
            expand $B$ using $G_B$
set $x_i$ to the yield of tree $z_i$
```

- $z_{A,i}$: the stored sub-trees headed by adapted non-terminals
- ν: the set of stick-weight proportions for the Pitman-Yor process
- θ: the set of PCFG rule probabilities

Our inference problem can be formalized as finding the posterior distribution on full derivational trees z_i—these depend on all the other latent variables, and reveal the inferred underlying linguistic structure. However, this inference is over an extremely large set of latent variables. In fact, in the current formalization, we cannot do inference over this set of latent variables, since some are countably infinite. To make this problem finite, we use a truncated stick-breaking representation, where after a sufficiently large i we let $\nu_i = 1$, so that the probability of continuing past that stick is 0. Beyond the large number of latent variables, we need to take into account the potentially exponential number of possible parses for each sentence given the grammar. Indeed, averaging rule probabilities over all of these parses is the most costly portion of the algorithm.

3.2 Dirichlet process hidden Markov model

The goal of the DPHMM is to jointly learn the phonetic boundaries of the speech input, clusters of acoustically similar segments, and PLU identities. By using a Dirichlet process, we do not bound the number of possible PLUs, but the reuse of existing PLUs is preferred (as in the adaptor grammar model). In the original model, defined by Lee and Glass (2012), a sampling approach was used. Extending this work, Ondel et al. (2016) implemented the model using variational Bayesian techniques. To describe the model, we first need to provide background on hidden Markov models and Gaussian mixture models.

3.2.1 Hidden Markov Models

A hidden Markov model (HMM) consists of a finite number of states combined with probability distribution over transitioning between states, dependent on the previous state. Observations are generated by such transitions, and the probability of emitting a certain observation is defined by a distribution which depends on the current state (Rabiner and Juang, 1986). In the case of the DPHMM model, each PLU is modeled by its own three-state HMM, corresponding to the start, middle, and end of a phone. Each emission distribution is modeled by a Gaussian mixture model.

3.2.2 Gaussian Mixture Models

A multimodal continuous distribution can be modeled by a linear combination (mixture) of Gaussian distributions. Each Gaussian is known as a *component* with a mean μ_k and a covariance Σ_k. In order to combine several Gaussians, we need *mixing coefficients* π_k such that $\sum_{k=1}^{K} \pi_k = 1$. Using these coefficients, we choose a Gaussian distribution and then sample a datapoint from its probability density function. The probability of datapoint x in a GMM is

$$p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \Sigma_k)$$

(Bishop, 2006)
From this, it is clear that the greater the mixing coefficient, the more often that component is chosen.

3.2.3 The DPHMM generative model

The DPHMM generative model first chooses a PLU label, or cluster, from the Dirichlet process. This cluster is associated with an 3-state HMM giving the start, beginning, and end of the phone. Sampling from the Gaussian mixture models at each state of the HMM produces a vector of MFCCs corresponding to the acoustic signal of the chosen PLU. Algorithm 5 gives a formal representation of this process:

Algorithm 5: Defining the DPHMM

```
# Defining the mixture models
choose the GMM mixture weights $\pi \sim Dir(\eta_0^{gmm})$
choose mean $\mu$, and covariance matrix $\Sigma$ with diagonal $\lambda$ by drawing from the
  Normal-Gamma distribution parametrized by normal distribution hyper-parameters
  $\mu_0$ and $(\kappa_0 \lambda)^{-1}$, and Gamma distribution hyper-parameters $\alpha_0, \beta_0$.
# Defining the HMM transition matrix
choose the rows of the transition matrix $r_i \sim Dir(\eta_0^{hmm,i})$
# Sampling M possible PLUs
for $i \in \{1, \dots, M\}$ do
    sample $\nu_i \sim Beta(1, \gamma)$
    sample cluster label $\theta_i \sim G_0$, the Dirichlet process base distribution
# Sampling N datapoints
for $i \in \{1, \dots, N\}$ do
    choose an cluster label $\theta_i$ with probability $\pi_i(\nu) = \nu_i \prod_{j=1}^{i-1} (1 - v_j)$
    # cluster labels correspond to HMMs
    sample a path $S = s_1, \dots, s_n$ from the transition probability distribution
    for $s_j \in S$ do
        choose a Gaussian component from mixture model
        sample datapoint from Gaussian density function
```

This model has three sets of latent variables:

- c_i: the cluster assignment of the i^{th} segment in the dataset
- s_{ij}: the HMM state of the j^{th} frame for the i^{th} segment
- m_{ij}: the GMM component of the j^{th} frame for the i^{th} segment

3.3 The noisy channel

The final component to the joint model, the noisy channel, allows the DPHMM and adaptor grammar to interface by rewriting each other's outputs. For example, if there is a mistake in the acoustic input or in the DPHMM cluster assignment, the adaptor grammar could fix this by substituting in a more probable PLU. The ability to make such edits is crucial when the acoustic signal is coupled with some amount of noise. The full set of operations the noisy channel allows is: substitution, insertion, and deletion. These are the same exact operations as in the well-known *Levenshtein distance* algorithm (Levenshtein, 1966), a dynamic programming algorithm typically used to find the minimum edit distance between two strings—that is, the minimum number of insertions, deletions, and substitutions required to rewrite one string as the other. To define a noisy channel, we leave the second string unspecified, and use the Levenshtein framework to enumerate all possible strings that the first string could be edited into. To maintain the plausibility of our model, we limit the number of consecutive insertions and deletions at 2, and strongly bias making no edit at all. This makes intuitive sense, as all communication would break down were a speaker to substitute every phoneme he intended to produce, insert an arbitrary number of extra ones, or delete all of them. The following prior probabilities are needed to define this model:

- operation probabilities o: this is a vector specifying the probability of doing an insertion, deletion, or substitution at all. It is drawn from a Dirichlet distribution.
- probability of inserting each phone I: given an alphabet of length k, we draw a vector from a Dirichlet distribution which specifies the probability of inserting that phone into the produced string (assuming that insertion has already been picked)
- probability of substitution ζ: since each phone can be substituted for each other phone, this is a $k \times k$ matrix where each row sums to 1. Thus, we draw each row from a k-dimensional Dirichlet distribution.

These three random variables also comprise the set of latent variables in the noisy channel model.

4 Variational Updates

Having defined the model, we need to construct variational distributions which approximate the posterior for each latent variable. The full set of latent variables, listed with the hyperparameters of their prior distributions and the name of the variational parameter indexing its variational distribution q (if applicable) is given in Table 1 . Note that the variational implementation of the DPHMM follows a slightly different paradigm, so we refer the reader to Ondel et al. (2016) for a full description of the variational parameter updates and derivations.

Latent variable	Description	variational parameter	hyperparameter
Adaptor grammar			
z_i	the full derivational trees that yielded the data	ϕ	
$z_{A,i}$	the stored sub-trees headed by adapted non-terminals	ϕ_A	
ν	the set of stick-weight proportions for the Pitman-Yor process	γ^1, γ^2	a, b
θ	the set of PCFG rule probabilities	τ	α
DPHMM			
c_i	the cluster assignment of the i^{th} segment in the dataset		γ
s_{ij}	the HMM state of the j^{th} frame for the i^{th} segment		η_0^{hmm}, G_0
m_{ij}	the GMM component of the j^{th} frame for the i^{th} segment		$\mu_0, (\kappa_0\lambda)^{-1}, \eta_0^{gmm}$
Noisy channel			
o	the operation probabilities	ξ^{ops}	ε^{ops}
I	the insertion probabilities	φ^{ins}	ς^{ins}
ζ	the substitution probabilities	σ	ρ

Table 1: latent variables with their respective variational parameters and hyperparameters

4.1 Adaptor grammar updates

The updates for the adaptor grammar are given as in Cohen et al. (2010):

$$\gamma^1_{A,i} = 1 - b_A + \sum_{B \in M} \sum_{k=1}^{N_B} \tilde{f}\Big(A \xrightarrow{*} s_{A,k}, s_{B,k}\Big)$$

$$\gamma^2_{A,i} = a_A + ib_A + \sum_{j=1}^{i-1} \sum_{B \in M} \sum_{k=1}^{N_B} \tilde{f}\Big(A \xrightarrow{*} s_{A,j}, s_{B,k}\Big)$$

$$\tau_{A,A\to\beta} = \sum_{B \in M} \sum_{k=1}^{N_B} \tilde{f}\big(A \to \beta, s_{B,k}\big)$$

$$\phi_{A,A\xrightarrow{*}s_{A,k}} = \Phi(\gamma^1_{A,i}) - \Phi(\gamma^1_{A,i} + \gamma^2_{A,i}) + \sum_{j=1}^{i-1} \big(\Phi(\gamma^1_{A,i}) - \Phi(\gamma^1_{A,i} + \gamma^2_{A,i})\big)$$

$$\phi_{A,A\to\beta} = \Phi(\tau_{A,A\to\beta}) - \Phi(\sum_{\beta} \tau_{A,A\to\beta})$$

where $\tilde{f}\Big(r, s_{B,k}\Big)$ is the expected count of rule r in the derivation trees of string $s_{B,k}$ which is headed by nonterminal B and spans k units, and $A \xrightarrow{*} s_{A,k}$ indicates that non-terminal A expands to the string spanning i and corresponding to the yield of the grammaton headed by A. In Cohen et al. (2010) the value $\tilde{f}\Big(r, s_{B,k}\Big)$ is computing using the inside-outside algorithm and a preprocessing step to determine $s_{B,k}$. However, in the implementation proposed in Zhai et al. (2014), this preprocessing step is avoided by sampling an approximating PCFG. In fact, the Zhai et al. (2014) model uses sampling to approximate both the tree fragments $z_{A,i}$ and the full tree derivations z_i. Counter-intuitively, this speeds up the model, despite the sampling approach being slower in the general case. This speed increase emerges from the fact that the expectation and maximization steps of the CAVI algorithm can be equivalently defined in terms of local and global latent variables. Local variables, such as the stick-weight proportions and rule weights, must be computed for each data point. Global variables, like the set of derivation trees z_i, need to take all of these variables into account. The expectation step involves optimizing the global variables, while the maximization step optimizes the local variables. This second optimization can be easily distributed across multiple cores. However, the optimized local variables need to be collected again in order to recalculate the global variables in the expectation step; this portion of the algorithm is not easily parallelizable. Furthermore, in the original variational model, the run time was dominated by the Inside-Outside algorithm for calculating expected values of rule counts, which has a time-complexity of $O(|N|^2|x_i|^3 + |N|^3|x_i|^2)$ where $|x_i|$ is the length of the i^{th} input sequence (Cohen et al., 2010). By using sampling, Zhai et al. (2014) avoid some of the cost involved in this computation. We incorporate this faster implementation into the ULD framework.

For an example derivation of a variational update see Appendix C.

4.2 Noisy channel updates

Let S be the set of all input strings to the noisy channel, let $PLU(i)$ indicate the PLU with index i, and let $O(i)$ indicate the operation indexed by i. Let $\tilde{g}(op[p], s_n)$ be the expected number of times an operation *op* (which can be insertion or substitution) is applied to PLU parameter(s) p in the string s_n. Note the overloaded call to $op[p]$ in the case of substitution, where it takes two parameters. The expected count $\tilde{g}$ can be computing using a Forward-Backward style algorithm which sums over all entries in the expanded Levenshtein chart. With this value, we can derive the updates for the noisy channel's variational distributions, using (29). They are:

$$\begin{aligned}
\xi_i^{ops} &= \varepsilon_i^{ops} + \sum_{s_n \in S} \sum_{l=1}^{k} \tilde{g}\Big(\big(O(i)[PLU(l)]\big), s_n\Big) \\
\phi_i^{ins} &= \varsigma_i^{ins} + \sum_{s_n \in S} \tilde{g}\Big(ins\big[PLU(i)\big], s_n\Big) \\
\sigma_{i,j} &= \rho_{i,j} + \sum_{s_n \in S} \tilde{g}\Big(sub\big[PLU(i), PLU(j)\big], s_n\Big)
\end{aligned}$$

5 Summary of previous results

Lee et al. (2015) ran several variants of the ULD model on a set of lecture recordings from the MIT lecture corpus. These were: a full model where the number of distinct PLU types was inferred from the data, a truncated model where the PLU inventory size was upper-bounded by 50, a lesioned version where the acoustic model (the DPHMM component) was removed after discovering the initial PLU labels and boundaries, meaning that the joint model could no longer relabel or re-segment PLUs, and finally a version were the noisy channel and acoustic model were removed, splitting the joint model into two separate parts.

5.1 Phone segmentation results

The phone segmentation produced by the joint model was evaluated against forced alignments of each lecture, with a $20ms$ tolerance margin (i.e. anything within $20ms$ of the force-aligned gold standard would be considered correct). Note that forced alignment—aligning a transcription of an audio recording with the actual audio by determining word and phone boundaries—is error-prone, so the gold standard against which Lee et al. (2015) evaluated their results most likely contained misaligned segments. The F1-score values reported by Lee et al. (2015) for phone segmentation, which can be seen in Table 2, show similar values for both the inferred PLU inventory system (FullDP) and the limited PLU inventory system (Full50), as well as the DPHMM system used to initialize the phone boundaries in the FullDP system, and the hierarchical hidden Markov model (HHMM) used to initialize the Full50 system.

Lecture topic	Full50	HHMM	FullDP	DPHMM
Economics	74.4	74.6	74.6	75.0
Signal processing	76.2	76.0	76.0	76.3
Clustering	76.6	76.6	77.0	76.9
Speaker adaptation	76.5	76.9	76.7	76.9
Physics	75.9	74.9	75.7	75.8
Linear algebra	75.5	73.8	75.5	75.7

Table 2: F1 scores for phone segmentation for each system and their respective initialization systems (Lee et al., 2015)

Lecture topic	Full50	-AM	-NC	FullDP	-AM	-NC
Economics	15.4	15.4	14.5	16.1	14.9	13.8
Signal processing	17.5	16.4	12.1	18.3	17.0	14.5
Clustering	16.7	18.1	15.9	18.4	16.9	15.2
Speaker adaptation	17.3	17.4	15.4	18.7	17.6	16.2
Physics	17.7	17.9	15.6	20.0	18.0	15.2
Linear algebra	17.9	17.5	15.4	20.0	17.0	15.6

Table 3: F1 scores for word segmentation by each system and its lesioned versions(Lee et al., 2015)

5.2 Word segmentation

As Lee et al. (2015) mention, due to the lack of a gold standard alignment of the audio used in the experiments, defining and measuring word segmentation presents its own challenges. Table 3 shows F1 scores for the word segmentation task for both the truncated and the full PLU inventory systems run by Lee et al. (2015). These results show that the noisy channel was important for word segmentation—intuitively, this makes sense, as words of the same type but with different surface realizations cannot be labeled as the same if the noisy channel is not able to make edits accommodating the variation. The 1.6% average improvement between the full system and the -AM lesioned version suggests that the joint learning nature of the model has a small positive effect on word segmentation.

Lee et al. (2015) also evaluated the number of top 20 *term frequency-inverse document frequency* (TFIDF) words (a commonly-used measure of word importance in a set of documents) that the various systems identified. These values are reported in comparison with the number of terms identified by a baseline system (Park and Glass, 2008) and a state-of-the-art system (Zhang et al., 2013), the latter of which uses a much richer representation for audio data than the MFCCs used in ULD. As can be seen in Table 4, both ULD systems frequently outperformed both the baseline and the state-of-the-art system, despite using a sparser data format to represent the audio than Zhang et al. (2013).

Lecture topic	Full50	-AM	-NC	FullDP	-AM	-NC	Park&Glass	Zhang
Economics	12	4	2	12	9	6	11	14
Signal processing	16	16	5	20	19	14	15	19
Clustering	18	17	9	17	18	13	16	17
Speaker adaptation	14	14	8	19	17	13	13	19
Physics	20	14	12	20	18	16	17	18
Linear algebra	18	16	11	19	17	7	17	16

Table 4: Number of top 20 TFIDF words discovered by each system (Lee et al., 2015)

5.3 Qualitative results

In addition to these quantitative values, Lee et al. (2015) report several qualitative results. For example, the ULD system discovered words such as *globalization* and *collaboration* which occurred frequently in the lectures; for both of these words, the system also discovered the productive *-ation* suffix. Because the purpose of adaptor grammars is to compactly store parse trees, certain frequently occurring morphemes like *-able* and *-ation* were saved. Simultaneously, certain sequences of words, like *the Arab Muslim word*, were identified as lexical items if they were common enough in the data. This calls into question the usefulness of word accuracy in evaluating an unsupervised system like ULD. There are sequences of words (such as some idioms) that almost always occur in that order, especially in a given context. Such collocations might reasonably be considered one lexical item by a language learner presented with only an acoustic input. For example, the grouping of two lexical items into one can be seen in the common malapropism *for all intensive purposes*. It is not impossible then that either through a misunderstanding, or due to the relative frequency of a phrase, we treat a sequence of words as one stored unit. Since our own storage and production process for lexical items is unclear, and, in the case of some idioms and multi-word units, independent of orthographic word boundaries, there is no definitive way of knowing how closely the discovered lexicon corresponds to our internal one.

Such considerations tie closely into the overall linguistic question of balancing productivity and reuse; namely, how much of our language do we compute on the fly (productively) and how much do we store and reuse statically. Both productivity and reuse have their costs and benefits: computing everything is inefficient, especially for high-frequency terms, but it lets us avoid storing anything; storing everything, on the other hand, makes it very efficient to produce sentences and terms that have already been used, but precludes the creation of novel sentences or terms, and entails storing sentences which are never reused. The optimal solution is to reuse those linguistic units which occur often, and compute those larger ones which are rare or unique. By modeling this balancing act mathematically with the Pitman-Yor and Dirichlet processes, ULD offers a rare glimpse at the internal mechanism of a productivity-reuse system. As the storage of super-word units in the results of Lee et al. (2015) shows, the optimal balance may not dovetail perfectly with our conception of the units in question.

6 Variational improvements

Given the faster convergence rate and multiprocessing capabilities of our variational ULD framework, more experiments can be run in a shorter time-frame, and the system scales to large audio corpora. The following data shows the improvements that variational systems made over sampling approaches for both the DPHMM and adaptor grammar components of the ULD model.

6.1 DPHMM improvement

Ondel et al. (2016) found that the variational was both faster and more accurate than the same model using Gibbs sampling. While training the latter took approximately 11 hours on one core, it took less than 30 minutes to train the variational DPHMM on 300 cores. Additionally, the variational model had a better mutual information score between discovered phones and previously labeled phones.

6.2 Adaptor grammar improvement

Cohen (2011) replicated the word-segmentation experiments run by Johnson (2008), and found that the variational system converged in fewer iterations (full passes through the dataset). While the sampling algorithm took 2000 iterations to converge, the variational system only needed 40. In addition, the variational system was faster when run on multiple cores. Inference by sampling took 2 hours and 14 minutes. The variational adaptor grammar needed 2 hours and 34 minutes when run on a single core—however, once distributed to 20 cores, it finished in 47 minutes.

7 Future Work

With this variational implementation of ULD, we plan on running experiments which test lesioning different parts of the model; in Lee et al. (2015), the acoustic model was removed, and then the noisy channel was further removed from that lesioned version. We are particularly interested in ablating the noisy channel but keeping the acoustic model in place, which would allow the DPHMM to continue relabeling and re-segmenting PLUs, but limit its interface with the adaptor grammar. A variational model also inherently creates novel opportunities for experimentation. Recall that in the variational setting, we introduce a family of new distributions indexed by parameters which can be initialized randomly, but can also be given deliberately chosen values. Testing different initializations can potentially reveal more about the phenomena being modeled while providing useful intuitions for future work. Additionally, the empirical Bayesian framework implemented in both Zhai et al. (2014) and Ondel et al. (2016) not only optimizes the variational parameters, but also finds the best value for the hyperparameters of the model, which can play an important role in future models. Thanks to the new multiprocessing capabilities of our ULD model, we will be able to run larger experiments by distributing the computation to a cluster. For example, we will be able to test the full system on large speech databases with gold standard alignments (e.g. the TIMIT corpus), and apply the learning algorithm to a variety of languages.

Given its the language-independent nature and the latent variables it infers, ULD provides a framework for generating linguistic and automatic speech recognition (ASR) resources such as pronunciation dictionaries, particularly for under-resourced languages. Pronunciation dictionaries, which map words to their phonetic transcriptions, are required for forced alignment (which has many research and industrial applications) as well as in most ASR systems (Besacier et al., 2014). With improved accuracy, ULD might in aid in lowering the production cost associated with generating such dictionaries, while simultaneously helping to fill a significant void in resources for specific accents and under-resources languages, whose current scarcity contributes to the under-representation of these languages in some areas of research and industry.

The utility of ULD's complete learning framework is not limited to research and industrial development. In the developing world—where many of under-resourced languages can be found—literacy and computer literacy are major issues facing millions. While ASR applications have been credited with improving literacy (Adams, 2005) and increasing computer accessibility, Plauche et al. (2006) point to the prohibitive cost of producing the requisite resources as the main obstacle to developing these technologies. An unsupervised system such as ULD could break this barrier by increasing the speed at which production can take place and lowering the cost.

8 Conclusion

We have presented a language-independent variational Bayesian inference model for the fully unsupervised induction of a complete hierarchy of linguistic units directly of an acoustic input, based on the sampling-based approach to the same problem by Lee et al. (2015). Our variational model promises significant decreases in amount of time required to train the model by virtue of the ease with which it can be distributed to multiple cores. For the acoustic model and adaptor grammar, we discussed experimental results and speed improvements made by their existing variational Bayesian implementations (Ondel et al., 2016; Cohen et al., 2010; Zhai et al., 2014). These results introduce questions regarding the current methods of evaluating unsupervised models while concomitantly offering a glimpse at a mathematical system for productivity and reuse thought to be similar to our own internal mental representation. Lastly, we discussed future experiments that our variational framework will enable us to conduct, as well as several real-world applications of our model.

A Deriving the ELBO

A.1 The problem

Given our generative model and our data, we would like to find a posterior distribution: $P(Z \mid X)$. Using Bayes Rule, we get: $P(Z \mid X) = \frac{P(X|Z)P(Z)}{P(X)} = \frac{P(X|Z)P(Z)}{\sum\limits_{\forall Z} P(X|Z)P(Z)}$ We call the numerator of the fraction on the right the generative model. It is composed of the product of the likelihood *of the hypothesis* ($P(X \mid Z)$) and the prior probability of the hypothesis ($P(Z)$). Note that the former is not a probability but a measure of how well our hypothesis fits the data. The denominator is the "marginal likelihood" of the data, $P(X)$. To find this, we need to marginalize out (sum over) all possible hypotheses. Because the hypotheses are the range of values for all of the latent variables in our model, this summation is computationally intractable. Instead of explicitly computing the posterior, we are forced to find an approximation of it. Often, a sampling approach is used. However, sampling can be very slow to converge and is not easily parallelizable across multiple cores. The variational Bayesian approach, on the other hand, treats the problem of finding an appropriate posterior distribution as an optimization problem.

- Let Z be our set of hidden variable collections:
- Let Φ be the collection of all model parameters (Pitman-Yor parameters a, b and Dirichlet distribution parameter α.
- Let X be the set of observations. In the case of word segmentation, for example, these would be each string of unsegmented phonemes.
- Note that our goal is to find $P(Z \mid X)$, the posterior (where Z is the set of latent variables)
- recall $P(Z \mid X) = \frac{P(X|Z,\Phi)P(Z|\Phi)}{\sum\limits_{\forall Z} P(X|Z,\Phi)P(Z|\Phi)}$

A.2 Important formulae

A.2.1 Jensen's inequality

Jensens inequality states that for a convex function f and random variable X:

$$f(\mathbb{E}[X]) \leq \mathbb{E}[f(X)] \tag{6}$$

We are using the logarithm of the probability, so the function is actually concave. Jensen's inequality works both ways, meaning we switch the direction of the inequality:

$$\log(\mathbb{E}[X]) \geq \mathbb{E}[\log(X)] \tag{7}$$

A.2.2 Expected value

Note that for discrete random variables

$$\mathbb{E}_q(f(x)) = \sum_{\forall x} q(x)f(x) \tag{8}$$

A.2.3 Logarithms

Throughout this derivation (and the variational literature as a whole) the logarithm of the probability is used. There are various reasons to do this. Firstly, logarithms are the foundation of information-theoretic measures such as entropy. Furthermore, they allow us to transform expensive multiplication and division into cheaper addition and subtraction, and help when working with probabilities below the floating-point precision bound. Recall these facts about logarithms:

- $\lim_{n \to 0} \log\ n = -\infty$
- $\log\ AB = \log\ A + \log\ B$
- $\log\ \frac{A}{B} = \log\ A - \log\ B$

A.3 Derivation of variational bound

The value we are looking to approximate is our posterior, which is the likelihood of the latent variables given the data. Recall that our inference problem lies in finding the denominator to the Bayesian equation

$$P(Z \mid X) = \frac{P(X \mid Z)P(Z)}{\int\limits_{\forall Z} P(X \mid Z)P(Z)dZ}$$

Our hypotheses in this case are possible values for the latent variables in the model. This integral (or in the discrete case, summation) is often computationally intractable, so we introduce a variational approximation for it. One way we can do this is by using the Kullback Leibler (KL) divergence between this intractable integral and some variational distribution q.

1. Let $q_\nu(Z)$ be a family of variational distributions with variational parameter ν.
2. to get the marginal likelihood ($\log\ p(X \mid \Phi)$) we take the KL divergence between $q_\nu(Z)$ and $p(Z \mid X, \Phi)$.
3. KL divergence is given by:

$$\begin{aligned} D_{KL}(q_\nu(Z) \mid\mid p(Z \mid X, \Phi)) &= \mathbb{E}_q[\log\ \frac{q_\nu(Z)}{p(Z \mid X, \Phi)}] \\ &= \mathbb{E}_q[\log\ q_\nu(Z) - \log\ p(Z \mid X, \Phi)] \\ &= \mathbb{E}_q[\log\ q_\nu(Z) - \log\ \frac{p(Z, X \mid \Phi)}{p(X \mid \Phi)}] \\ &= \mathbb{E}_q[\log\ q_\nu(Z) - (\log\ p(Z, X \mid \Phi) - \log\ p(X \mid \Phi))] \\ &= \mathbb{E}_q[\log\ q_\nu(Z)] - \mathbb{E}_q[\log\ p(Z, X \mid \Phi)] + \log\ p(X \mid \Phi) \end{aligned} \tag{9}$$

(Blei and Jordan, 2006)

Considering what KL divergence represents, it is easy to understand why it cannot be negative. From here, we can see how minimizing this equation is the same as maximizing the lower bound on log $p(X \mid \Phi)$:

$$\begin{aligned}
0 &\leq \mathbb{E}_q[\log\ q_\nu(Z)] - \mathbb{E}_q[\log\ p(Z, X \mid \Phi)] + \log\ p(X \mid \Phi) \\
-\log\ p(X \mid \Phi) &\leq \mathbb{E}_q[\log\ q_\nu(Z)] - \mathbb{E}_q[\log\ p(Z, X \mid \Phi)] \\
\log\ p(X \mid \Phi) &\geq \mathbb{E}_q[\log\ p(Z, X \mid \Phi)] - \mathbb{E}_q[\log\ q_\nu(Z)] \qquad (10)
\end{aligned}$$

Another method of reaching this same result uses Jensen's inequality. Consider the log marginal likelihood:

$$\log\ p(X \mid \Phi) = \log \sum_{z \in \mathbf{Z}} p(X, z \mid \Phi) \qquad (11)$$

The sum marginalizes out the hidden variables z in the joint probability distribution. Picking any variational distribution $q(z)$ we can multiply by $\frac{q(z)}{q(z)}$:

$$\log \sum_{\forall z \in \mathbf{Z}} (p(x, z \mid \Phi) * \frac{q(z)}{q(z)}) = \log \sum_{\forall z \in \mathbf{Z}} q(z) \frac{p(x, z \mid \Phi)}{q(z)} \qquad (12)$$

Jensen's inequality implies

$$\log \sum_{\forall z \in \mathbf{Z}} q(z) \frac{p(x, z \mid \Phi)}{q(z)} \geq \sum_{\forall z \in \mathbf{Z}} q(z) \log\ \frac{p(x, z \mid \Phi)}{q(z)} \qquad (13)$$

This equation can be broken into:

$$\begin{aligned}
\sum_{\forall z \in \mathbf{Z}} q(z) \log\ \frac{p(x, z \mid \Phi)}{q(z)} = \sum_{\forall z \in \mathbf{Z}} q(z)(\log p(x, z \mid \Phi) - \log q(z)) &= \\
\sum_{\forall z \in \mathbf{Z}} q(z) \log p(x, z \mid \Phi) - \sum_{\forall z \in \mathbf{Z}} q(z) \log\ q(z) &= \\
\sum_{\forall z \in \mathbf{Z}} q(z) \log p(x, z \mid \Phi) + \mathcal{H}(q) & \qquad (14) \\
& \qquad (15)
\end{aligned}$$

where

$$\mathcal{H}(q) = -\sum_{\forall z \in \mathbf{Z}} q(z) \log\ q(z) \qquad (16)$$

(Blei et al., 2017) This first term is of the form of our expected value definition, so our equation becomes:

$$\log\ p(x \mid \Phi) \geq \mathbb{E}_q[\log\ p(x, z \mid \Phi)] + \mathcal{H}(q) \tag{17}$$

This derivation yields an important fact:

$$\log\ p(X \mid \Phi) - KL(q(Z) \mid\mid p(Z \mid X, \Phi)) = \mathbb{E}_q[\log\ p(z, x \mid \Phi)] + H(q) \tag{18}$$

From this equation, we can see why minimizing KL divergence gives us the best possible value for our marginal likelihood.

B Deriving Variational Updates

B.1 Mean Field Approximation

Recall our mean-field assumption was to treat each variational distribution as conditionally independent, i.e. $q(Z) = \prod_i q_i(z_i)$. Also recall that our bound on the log marginal likelihood was:

$$\mathcal{L}(q) \geq \sum_{z_i \in Z} q(Z) \log\ p(X, Z|\Phi) + H(q)$$

Replace $q(Z)$ with this product:

$$\mathcal{L}(q) \geq \sum_{z_i \in Z} \left(\prod_i q_i(z_i) \right) \log\ p(X, Z|\Phi) + H(q)$$
$$\mathcal{L}(q) \geq \mathbb{E}_{\prod_i q_i(z_i)} \log\ p(X, Z|\Phi)[\log\ p(X, Z|\Phi)] + H(q) \tag{19}$$

Using the chain rule and by expanding the entropy term, we can rewrite this expression as

$$\log\ p(X|\Phi) + \sum_{i=1}^{|Z|} \mathbb{E}_q[\log\ p(z_i|X, z_1, ..., z_{i-1}, \Phi)] - \sum_{i=1}^{|Z|} \mathbb{E}_q[\log\ q_{\nu_i}(z_i)] \tag{20}$$

Since $p(X|Phi)$ does not depend on the variational parameter ν_i it factors out as a constant (recall that this is a lower bound, not an exact equality). We can reorder the elements of Z in any way we wish. If we reorder them each time so that z_i comes last, we can say:

$$\mathcal{L}_i = \mathbb{E}_q[\log\ p(z_i|Z_{-i}, X, \Phi)] - \mathbb{E}_q[\log\ q_{\nu_i}(z_i)] \tag{21}$$

(Blei and Jordan, 2006)

Note that for any exponential family distribution q_{ν_i},

$$q_{\nu_i}(z_i) = h(z_i) \exp\left\{\nu_i^T z_i - a(\nu_i)\right\} \tag{22}$$

where $a(\nu_i)$ is the cumulant function, which for the first three derivatives is equivalent to the corresponding derivatives of the same distribution's moment generating function. We can rewrite our equation using this form for $q_{\nu_i}(z_i)$:

$$\begin{aligned}
\mathcal{L}_i &= \mathbb{E}_q[\log\ p(z_i|Z_{-i}, X, \Phi)] - \mathbb{E}_q\left[\log\ \Big(h(z_i)\exp\left\{\nu_i^T z_i - a(\nu_i)\right\}\Big)\right] \\
&= \mathbb{E}_q[\log\ p(z_i|Z_{-i}, X, \Phi)] - \mathbb{E}_q\left[\log\ (h(z_i))) + \nu_i^T z_i - a(\nu_i)\right] \\
&= \mathbb{E}_q[\log\ p(z_i|Z_{-i}, X, \Phi)] - \mathbb{E}_q\big[\log\ (h(z_i)))\big] - \mathbb{E}_q[\nu_i^T z_i] + \mathbb{E}_q[a(\nu_i)] \\
&= \mathbb{E}_q[\log\ p(z_i|Z_{-i}, X, \Phi)] - \mathbb{E}_q\big[\log\ (h(z_i)))\big] - \nu_i^T a'(\nu_i) + a(\nu_i)
\end{aligned} \tag{23}$$

Note that $\mathbb{E}_q[\nu_i^T z_i] = \nu_i^T a'(\nu_i)$ since $E_q(z_i) = a'(\nu_i)$ and ν_i^T factors out as a constant when taking the expectation with respect to q. The goal of variational inference is to cast the intractable calculation of the posterior as an optimization problem. In most optimization problems, there are two general steps: (1) computing an objective function which allows us to (2) optimize the function by adjusting the parameters.

Recall that to avoid expensive computations, we employ exponential family distributions which allow us to simplify the problem. Our goal is to optimize the function by adjusting the variational parameters, so we take the partial derivative of our function with respect to ν_i:

$$\begin{aligned}
\frac{\delta}{\delta\nu_i}\mathcal{L}_i &= \frac{\delta}{\delta\nu_i}\big(\mathbb{E}_q[\log\ p(z_i|Z_{-i}, X, \Phi)] - \mathbb{E}_q\big[\log\ (h(z_i)))\big] - \nu_i^T a'(\nu_i) + a(\nu_i)\big) \\
&= \frac{\delta}{\delta\nu_i}\left(\mathbb{E}_q[\log\ p(z_i|Z_{-i}, X, \Phi)] - \mathbb{E}_q[\log\ h(z_i))]\right) - \big(\nu_i^T a''(\nu_i) + a''(\nu_i)\big) + a''(\nu_i) \\
&= \frac{\delta}{\delta\nu_i}\left(\mathbb{E}_q[\log\ p(z_i|Z_{-i}, X, \Phi)] - \mathbb{E}_q[\log\ h(z_i))]\right) - \nu_i^T a''(\nu_i)
\end{aligned} \tag{24}$$

Setting this to 0 we get:

$$\begin{aligned}
0 &= \frac{\delta}{\delta\nu_i}\left(\mathbb{E}_q[\log\ p(z_i|Z_{-i}, X, \Phi)] - \mathbb{E}_q[\log\ h(z_i))]\right) - \nu_i^T a''(\nu_i) \\
\nu_i^T a''(\nu_i) &= \frac{\delta}{\delta\nu_i}\left(\mathbb{E}_q[\log\ p(z_i|Z_{-i}, X, \Phi)] - \mathbb{E}_q[\log\ h(z_i))]\right) \\
\nu_i &= \left(\frac{\delta}{\delta\nu_i}\mathbb{E}_q[\log\ p(z_i|Z_{-i}, X, \Phi)] - \frac{\delta}{\delta\nu_i}\mathbb{E}_q[\log\ h(z_i))]\right)\left(a''(\nu_i)\right)^{-1}
\end{aligned} \tag{25}$$

If $p(z_i|Z_{-i}, X, \Phi)$ is also a member of the exponential family, it can be rewritten:

$$p(z_i|Z_{-i}, X, \Phi) = h(z_i) \exp\left\{ g_i(Z_{-i}, X, \Phi)^T z_i - a\left(g_i(Z_{-i}, X, \Phi)\right) \right\} \tag{26}$$

where $g_i(Z_{-i}, X, \Phi)$ is the natural parameter of distribution p. Replacing $p(z_i|Z_{-i}, X, \Phi)$ (first in the expected values for the sake of readability) and taking the derivative gives us

$$\mathbb{E}_q[\log\ p(z_i|Z_{-i}, X, \Phi)] = \mathbb{E}_q[\log\ h(z_i)] + \mathbb{E}_q[g_i(Z_{-i}, X, \Phi)]^T a'(\nu_i) - \mathbb{E}_q\left[a\left(g_i(Z_{-i}, X, \Phi)\right)\right] \tag{27}$$

$$\begin{aligned}
\frac{\delta}{\delta\nu_i}\mathbb{E}_q[p(z_i|Z_{-i}, X, \Phi)] &= \frac{\delta}{\delta\nu_i}\mathbb{E}_q[\log\ h(z_i)] \\
&+ \left(\frac{\delta}{\delta\nu_i}\left(\mathbb{E}_q[g_i(Z_{-i}, X, \Phi)]^T\right) a'(\nu_i) + \mathbb{E}_q[g_i(Z_{-i}, X, \Phi)]^T a''(\nu_i)\right) - \frac{\delta}{\delta\nu_i}\mathbb{E}_q\left[a\left(g_i(Z_{-i}, X, \Phi)\right)\right] \\
&= \frac{\delta}{\delta\nu_i}\mathbb{E}_q[\log\ h(z_i)] + \mathbb{E}_q[g_i(Z_{-i}, X, \Phi)]^T a''(\nu_i)
\end{aligned} \tag{28}$$

Notice that many of the expectations drop out. Substituting this for $\frac{\delta}{\delta\nu_i}\left(\mathbb{E}_q[\log\ p(z_i|Z_{-i}, X, \Phi)]\right)$ in our first differentiation, we get:

$$\begin{aligned}
\nu_i &= \left(\frac{\delta}{\delta\nu_i}\mathbb{E}_q[\log\ h(z_i)] + \mathbb{E}_q[g_i(Z_{-i}, X, \Phi)]^T a''(\nu_i) - \frac{\delta}{\delta\nu_i}\mathbb{E}_q[\log\ h(z_i))]\right)\left(a''(\nu_i)\right)^{-1} \\
&= \left(\mathbb{E}_q[g_i(Z_{-i}, X, \Phi)]^T a''(\nu_i)\right)\left(a''(\nu_i)\right)^{-1} \\
&= \mathbb{E}_q[g_i(Z_{-i}, X, \Phi)]
\end{aligned} \tag{29}$$

So the optimal value (when the derivative is 0) of ν_i is $\nu_i = \mathbb{E}_q[g_i(Z_{-i}, X, \Phi)]$.

B.2 Conjugacy

Now we need to obtain a closed-form expression for $\mathbb{E}_q[g_i(Z_{-i}, X, \Phi)$. First, let Φ becomposed of 2 parts ϕ_1 and ϕ_2, where ϕ_1 is the number of observations contributed by the prior, and ϕ_2 corresponds to the total effect of the observations on the sufficient statistic. Because of the factorization and exponential family assumptions we made earlier, we can say:

$$\begin{aligned}
P_\pi(z_i|\phi_1, \phi_2) &= f(\phi_1, \phi_2) \exp\left\{\eta^T \phi_1 - \phi_2 a(\eta)\right\} \\
&= f(\phi_1, \phi_2) g(\eta)^{\phi_2} \exp\left\{\eta^T \phi_1\right\} \\
&\propto g(\eta)^{\phi_2} \exp\left\{\eta^T \phi_1\right\}
\end{aligned} \tag{30}$$

where η are the natural parameters for the distribution, $a(\eta)$ is the cumulant function, and $f(\phi_1, \phi_2)$ is a normalizing function.

Assuming the posterior over data and local hidden variables $P(X, Z_{-i} \mid z_i)$ is also in the exponential family and factorizes, we can say that for one data point x_n

$$P(x_n, z_n \mid z_i) = h(x_n, z_n) g(z_i) \exp\left\{z_i^T t(x_n, z_n)\right\}$$
$$\Rightarrow P(X, Z_{-i} \mid z_i) = \prod_{z_n \in Z_{-i}} h(x_n, z_n) g(z_i)^N \exp\left\{z_i^T t(x_n, z_n)\right\} \tag{31}$$

where $t(x_n, z_n)$ is the sufficient statistic, which in most cases is simply the count of occurrences of x_n or z_n. By Bayes rule, the distribution over the selected hidden variable rewrites as

$$\begin{aligned} P(z_i|X, Z_{-i}, \Phi) &\propto P(X, Z_{-i}|z_i) P(z_i|\Phi) \\ = \prod_{n=0}^{N} h(x_n, z_n) g(z_i) &\exp\left\{z_i^T t(x_n, z_n)\right\} g(\eta)^{\phi_2} \exp\left\{\eta^T \phi_1\right\} \\ &\propto g(z_i)^N \exp\left\{z_i^T t(x_n, z_n)\right\} g(\eta)^{\phi_2} \exp\left\{\eta^T \phi_1\right\} \\ &\propto g(z_i)^{N+\phi_2} \exp\left\{z_i^T \big(\phi_1 + \sum_{z_n \in Z_{-i}} t(x_n, z_n)\big)\right\} \end{aligned} \tag{32}$$

Because all exponential family distributions have conjugate priors, this result implies that the posterior $P(z_i \mid X, Z_{-i}, \Phi)$ is the same type of distribution as the prior with parameters:

$$P(z_i|X, Z_{-i}, \Phi) = P_\pi\big(z_i|\phi_1 + \sum_{z_n \in Z_{-i}} t(x_n, z_n), \phi_2 + N\big) \tag{33}$$

This means that the natural parameters of the posterior distribution on global hidden variables has the natural parameters $\phi_2 + \sum_{z_n \in Z_{-i}} t(x_n, z_n)$ and $\phi_2 + N$, giving us a closed form for our expectation in (29):

$$\mathbb{E}_q[g_i(Z_{-i}, X, \Phi)] = \mathbb{E}_q \begin{bmatrix} \phi_1 + \sum_{z_n \in Z_{-i}} t(x_n, z_n) \\ \phi_2 + N \end{bmatrix} \tag{34}$$

(Hoffman et al., 2013)

B.3 Alternative Form

We can derive an alternative form for an optimal setting of q without the exponential family requirements by following the method described in Bishop (2006). Recall that

$$\mathcal{L}(q) = \sum_{z_i \in Z} \Big(\Big(\prod_i q_i(z_i) \Big) \log\ p(X, Z|\Phi) + \sum_i q_i(z_i) \Big)$$

This can be rewritten as

$$\begin{aligned}\mathcal{L}(q) &= \sum_{\forall z_j} q_j(z_j)\Big(\sum_{z_i \neq z_j} \log\ p(X,Z)\prod_{i\neq j} q_i(z_i)\Big) - \sum_{\forall z_j} q_j(z_j)\log\ q_j + const \\ &= \sum_{\forall z_j} q_j(z_j)\log\ \tilde{p}(X,z_j) - \sum_{\forall z_j} q_j(z_j)\log\ q_j(z_j) + const \end{aligned} \tag{35}$$

where $\log \tilde{p}(X,z_j) = \mathbb{E}_{i\neq j}[\log\ p(X,Z)] + const$ and $\mathbb{E}_{i\neq j}$ is the expectation taken with respect to all distributions q except q_i. Maximizing (35) is equivalent to minimizing the KL divergence, with the minimum occurring when $q_j(z_j) = \tilde{p}(X,z_j)$. Thus the optimal distribution $q_j^*(z_j)$ can be written:

$$\log q_j^*(z_j) = \mathbb{E}_{i\neq j}[\log\ p(X,Z)] + const \tag{36}$$

(Bishop, 2006)

C Derivation of Updates

As an example of how variational updates can be found, we show the explicit derivation of the Beta distribution updates for the adaptor grammar portion of the ULD model.

Recall that in the generative process, each stick-weight proportion ν_i is drawn from a Beta distribution prior parametrized as $Beta(1-b_A, a_A - ib_A)$. Recall also that our updates take the general form

$$\mathbb{E}_q[g_i(Z_{-i},X,\Phi)] = \mathbb{E}_q\begin{bmatrix}\phi_1 + \sum\limits_{z_n\in Z_{-i}} t(x_n,z_n) \\ \phi_2 + N\end{bmatrix} \tag{37}$$

where ϕ_1 and ϕ_2 are the parameters of the prior. This implies that in this case, $\phi_1 = 1-b_A$ and $\phi_2 = a_A - ib_A$. The sufficient statistic $t(x_n,z_n)$ is the number of times stick i was the final stick (i.e. the process did not continue after i). N is the total number of times any stick was the final one, which is equivalent to the sum over all sticks of $t(x_n,z_n)$. Formally,

$$\sum_{z_n\in Z_{-i}} t(x_n,z_n) = \sum_{B\in M}\sum_{k=1}^{N_B} f(A \overset{*}{\to} s_{A,k}, z_k) \tag{38}$$

which is the sum over all adapted nonterminals of the sum up to the truncation level of that nonterminal (the maximal stick index) of the count of the grammaton expansion of A to the substring $s_{A,k}$ in the derivation tree z_k. As mentioned before,

$$N = \sum_{j=1}^{i-1}\sum_{B\in M}\sum_{k=1}^{N_B} f(A \overset{*}{\to} s_{A,j}, z_k) \tag{39}$$

Putting (37),(38), and (39) together, the full update becomes

$$\begin{aligned}\begin{bmatrix}\gamma_{A,i}^{1}\\ \gamma_{A,i}^{2}\end{bmatrix} &= \mathbb{E}_q\begin{bmatrix} 1-b_A+\sum_{B\in M}\sum_{k=1}^{N_B} f(A \overset{*}{\to} s_{A,k}, z_k) \\ a_A - ib_A + \sum_{j=1}^{i-1}\sum_{B\in M}\sum_{k=1}^{N_B} f(A \overset{*}{\to} s_{A,j}, z_k)\end{bmatrix} \\ &= \begin{bmatrix} 1-b_A+\sum_{B\in M}\sum_{k=1}^{N_B} \tilde{f}\Big(A \overset{*}{\to} s_{A,k}, s_{B,k}\Big) \\ a_A + ib_A + \sum_{j=1}^{i-1}\sum_{B\in M}\sum_{k=1}^{N_B} \tilde{f}\Big(A \overset{*}{\to} s_{A,j}, s_{B,k}\Big)\end{bmatrix}\end{aligned} \tag{40}$$

where $\tilde{f}\Big(r, s_{B,k}\Big)$ is the expected count of rule r in the derivation trees of string $s_{B,k}$ which is headed by nonterminal B and spans k units, and $A \overset{*}{\to} s_{A,k}$ indicates that non-terminal A expands to the string headed by A and spanning k in its grammaton form (Cohen et al., 2010; Zhai et al., 2014).

References

Adams, M. (2005). The promise of automatic speech recognition for fostering literacy growth in children and adults. *Handbook of literacy and technology*, 2:109–128.

Besacier, L., Barnard, E., Karpov, A., and Schultz, T. (2014). Automatic speech recognition for under-resourced languages: A survey. *Speech Communication*, 56:85–100.

Bishop, C. M. (2006). *Pattern recognition and machine learning.* Springer.

Blei, D. M. and Jordan, M. I. (2006). Variational inference for Dirichlet process mixtures. *Bayesian Analysis*, 1(1):121–143.

Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. (2017). Variational inference: A review for statisticians. *Journal of the American Statistical Association*, (just-accepted).

Cohen, S. (2011). *Computational Learning of Probabilistic Grammars in the Unsupervised Setting.* PhD thesis, Ph. D. thesis, Carnegie Mellon University.

Cohen, S. B., Blei, D. M., and Smith, N. A. (2010). Variational inference for adaptor grammars. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 564–572. Association for Computational Linguistics.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38.

Hoffman, M. D., Blei, D. M., Wang, C., and Paisley, J. (2013). Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347.

Hopcroft, J. E., Motwani, R., and Ullman, J. D. (2006). *Introduction to Automata Theory, Languages, and Computation (3rd Edition).* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Johnson, M. (2008). Using Adaptor Grammars to Identify Synergies in the Unsupervised Acquisition of Linguistic Structure. In *ACL*, pages 398–406.

Johnson, M., Griffiths, T. L., and Goldwater, S. (2007). Adaptor grammars: A framework for specifying compositional nonparametric Bayesian models. In *Advances in neural information processing systems*, pages 641–648.

Lee, C.-y. and Glass, J. (2012). A nonparametric Bayesian approach to acoustic model discovery. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 40–49. Association for Computational Linguistics.

Lee, C.-y., O'Donnell, T. J., and Glass, J. (2015). Unsupervised lexicon discovery from acoustic input. *Transactions of the Association for Computational Linguistics*, 3:389–403.

Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710.

Neal, R. M. and Hinton, G. E. (1998). A view of the EM algorithm that justifies incremental, sparse, and other variants. In *Learning in graphical models*, pages 355–368. Springer.

O'Donnell, T. J. (2015). *Productivity and reuse in language: A theory of linguistic computation and storage.* MIT Press.

Ondel, L., Burget, L., and Černockỳ, J. (2016). Variational inference for acoustic unit discovery. *Procedia Computer Science*, 81:80–86.

Park, A. S. and Glass, J. R. (2008). Unsupervised pattern discovery in speech. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(1):186–197.

Pitman, J. and Yor, M. (1997). The two-parameter Poisson-Dirichlet distribution derived from a stable subordinator. *The Annals of Probability*, pages 855–900.

Plauche, M., Nallasamy, U., Pal, J., Wooters, C., and Ramachandran, D. (2006). Speech recognition for illiterate access to information and technology. In *Information and Communication Technologies and Development, 2006. ICTD'06. International Conference on*, pages 83–92. IEEE.

Rabiner, L. and Juang, B. (1986). An introduction to hidden Markov models. *IEEE ASSP Magazine*, 3(1):4–16.

Sethuraman, J. (1994). A constructive definition of dirichlet priors. *Statistica sinica*, pages 639–650.

Zhai, K., Boyd-Graber, J., Asadi, N., and Alkhouja, M. L. (2012). Mr. LDA: A flexible large scale topic modeling package using variational inference in MapReduce. In *Proceedings of the 21st international conference on World Wide Web*, pages 879–888. ACM.

Zhai, K., Boyd-Graber, J., and Cohen, S. B. (2014). Online adaptor grammars with hybrid inference. *Transactions of the Association for Computational Linguistics*, 2:465–476.

Zhang, Y. et al. (2013). *Unsupervised speech processing with applications to query-by-example spoken term detection.* PhD thesis, Massachusetts Institute of Technology.