Journal of Machine Learning Research 12 (2011) 2335-2382 Submitted 1/10; Revised 8/10; Published 7/11

Producing Power-Law Distributions and Damping Word Frequencies with Two-Stage Language Models

Sharon Goldwater SGWATER@INF.ED.AC.UK
School of Informatics
10 Crichton Street
Edinburgh, EH8 9AB
United Kingdom

Thomas L. Griffiths TOM_GRIFFITHS@BERKELEY.EDU
Department of Psychology
University of California, Berkeley
3210 Tolman Hall, MC 1650
Berkeley, CA 94720-1650
USA

Mark Johnson MJOHNSON@SCIENCE.MQ.EDU.AU
Department of Computing
Macquarie University
Sydney, NSW 2109
Australia

Editor: Fernando Pereira

Abstract

Standard statistical models of language fail to capture one of the most striking properties of natural languages: the power-law distribution in the frequencies of word tokens. We present a framework for developing statistical models that can generically produce power laws, breaking generative models into two stages. The first stage, the generator, can be any standard probabilistic model, while the second stage, the adaptor, transforms the word frequencies of this model to provide a closer match to natural language. We show that two commonly used Bayesian models, the Dirichlet-multinomial model and the Dirichlet process, can be viewed as special cases of our framework. We discuss two stochastic processes—the Chinese restaurant process and its two-parameter generalization based on the Pitman-Yor process—that can be used as adaptors in our framework to produce power-law distributions over word frequencies. We show that these adaptors justify common estimation procedures based on logarithmic or inverse-power transformations of empirical frequencies. In addition, taking the Pitman-Yor Chinese restaurant process as an adaptor justifies the appearance of type frequencies in formal analyses of natural language and improves the performance of a model for unsupervised learning of morphology.

Keywords: nonparametric Bayes, Pitman-Yor process, language model, unsupervised

1. Introduction

It is important for models used in unsupervised learning to be able to describe the gross statistical properties of the data they are intended to learn from, otherwise these properties may distort inferences about the parameters of the model. One of the most striking statistical properties of nat-

ural languages is that the distribution of word frequencies is closely approximated by a power law. That is, the probability that a word w will occur with frequency n_w in a sufficiently large corpus is proportional to n_w^{-g}. This observation—usually attributed to Zipf (1932), though it enjoys a long and detailed history (Mitzenmacher, 2004)—stimulated intense research in the 1950s (e.g., Simon, 1955) but has largely been ignored in modern machine learning and computational linguistics.

By developing models that can generically exhibit power laws, it may be possible to improve methods for identifying structure in linguistic data. In particular, postulating a separate mechanism within the model that accounts for the skewed distribution of word frequencies takes the burden of explaining this distribution off the other components of the model, effectively reducing the frequencies of those words. Such "damping" of word frequencies can often be desirable. It is commonly observed in applications of statistical natural language processing that reducing the counts of word tokens, typically by taking their logarithms or inverse powers, can improve performance (Salton and Buckley, 1988).

An extreme version of damping frequencies forms part of a tension exhibited by formal approaches to natural language: whether explanations should be based upon the distinct *types* of words that languages exhibit, or the frequencies with which *tokens* (instances) of those words occur. One place where this tension manifests is in accounts of morphology (the substructure of words), where formal linguists develop accounts of why particular words appear in the lexicon (e.g., Pierrehumbert, 2003), while computational linguists focus on statistical models of the frequencies of tokens of those words (e.g., Hakkani-Tür et al., 2002). The same tension arises in various areas of statistical natural language processing and related fields. For example, one of the most successful forms of smoothing used in statistical language models, Kneser-Ney smoothing, explicitly interpolates between type and token frequencies (Ney et al., 1994; Kneser and Ney, 1995; Chen and Goodman, 1998). Information retrieval systems can also differ in whether they use binary vectors indicating the presence or absence of words in a document or a full vector of word frequencies (Baeza-Yates and Ribeiro-Neto, 1999), and the same distinction appears in machine learning methods applied to text (e.g., Blei et al., 2003; Thibaux and Jordan, 2007).

In this paper, we present a framework for developing generative models for language that produce power-law distributions. Our framework is based upon the idea of specifying these models in terms of two components: a *generator*, an underlying generative model for words which need not (and usually does not) produce a power-law distribution, and an *adaptor*, which transforms the stream of words produced by the generator into one whose frequencies obey a power-law distribution. This framework is extremely general: any generative model for language can be used as a generator, with the power-law distribution being produced as the result of making an appropriate choice for the adaptor.

Adopting this two-stage framework divides responsibility for the appearance of the tokens in the corpus between the generator and the adaptor, with only a subset of the tokens being produced by the generator. The parameters of the generator will be estimated based only on the tokens for which the generator is considered responsible, rather than on the full set of tokens in the corpus. By explaining away the presence of some of the tokens, the adaptor effectively damps the word counts used to estimate the parameters of the generator. Estimation of these parameters will thus be affected by assumptions about the form of the adaptor. We consider several adaptor-generator pairs, focusing especially on the Chinese restaurant process (Aldous, 1985) and its two-parameter generalization, derived from the Pitman-Yor process (Pitman, 1995; Pitman and Yor, 1997; Ishwaran and James, 2003), as adaptors. We show that using these stochastic processes as adaptors can

produce appropriate power-law distributions while implementing different forms of damping. We also show that the Pitman-Yor generalization of the Chinese restaurant process can be used to justify parameter estimation based purely on type frequencies, and demonstrate that using this adaptor improves the performance of a simple two-stage model applied to learning morphology.

Our work contributes to a growing body of research on Bayesian approaches to modeling and learning language. This paper is not the first to propose the use of the Chinese restaurant process or Pitman-Yor process for modeling language, and some of the models we discuss have been used in previous work by ourselves and others (Goldwater et al., 2006a; Teh, 2006b). However, considering these models in greater depth allows us to make several novel contributions. First, we show how the two-stage framework makes it possible to unify a variety of Bayesian models of language. This unified picture offers us a way to concisely summarize existing Bayesian language models, and to identify the mathematical relationships between these models. Second, we provide a quantitative argument that these models are a good fit for language by virtue of the power-law distributions they produce, detailing the differences between the distributions produced by different adaptors, and discussing the use of different approximations. Third, we present new empirical studies that provide insight into the practical effects of different approximations and parameter choices. Finally, we expand on the idea, introduced by Goldwater et al. (2006a), that these models provide a way to understand and model the relationship between linguistic types and tokens, and a mathematical justification for commonly used smoothing and damping techniques.

In addition to considering the general properties of models developed in our two-stage framework, we provide a detailed case study of applying this approach to an unsupervised learning problem: morphological segmentation. In this problem, the goal is to identify the meaningful components from which words are comprised. This problem is challenging because natural languages possess both regular and irregular morphology, with only a subset of words following regular morphological rules. Linguists have long noted a strong relationship between frequency and regularity in language, with irregular forms often being among the most frequent (Greenberg, 1966; Bybee, 1985). Without accounting for this fact, an unsupervised learning system is likely to be misled by the very frequent irregular forms, and fail to appropriately model the regular patterns that are needed to account for infrequent forms, which will comprise most unseen data. We show that the two-stage framework proposed here can explain the relationship between frequency and regularity and thus leads to better learning of regular patterns.

The morphological segmentation task is a good example of a situation where appropriately modeling word frequencies can significantly affect the outcome of unsupervised learning. While we explore this case in detail, the goal of this paper is not to develop state-of-the-art models for any particular application. Rather, we hope to strengthen intuitions and insights into how nonparametric Bayesian models of language behave in general, in order to give other researchers a better sense of when these tools may be helpful and how to use them. We consider other promising applications of this approach, and ways in which it can be extended, in Section 9.

The plan of the paper is as follows. Section 2 summarizes related work. Section 3 discusses stochastic processes that can produce power-law distributions and introduces the generic two-stage modeling framework. Section 4 presents models based on the Chinese restaurant process and Pitman-Yor Chinese restaurant process, stochastic processes from nonparametric Bayesian statistics that produce power-law distributions. Section 5 shows how some other Bayesian language models can be viewed as special cases of our two-stage framework. Section 6 examines some of the consequences of using the adaptors introduced in Section 4: Section 6.1 discusses the implications

of using these models for estimation of the parameters of the generator, Section 6.2 shows that estimation based on type and token frequencies are special cases of a two-stage language model, and Section 6.3 uses these results to provide a novel justification for the use of Kneser-Ney smoothing. Section 7 describes a two-stage model for unsupervised learning of the morphological structure of words, and Section 8 presents the results of some experiments with this model demonstrating that its performance improves as we move from estimation based upon tokens to types. Section 9 discusses additional applications and extensions of our approach, and Section 10 concludes.

2. Related Work

Our two-stage approach fits within a more general trend of using Bayesian models for linguistic data. Previous work has used Bayesian models in two ways: to understand and justify approaches to smoothing, or as a method of unsupervised structure discovery and learning. Since we will touch upon both of these topics in this paper, we now present a brief review of related work in each area.

Smoothing methods are schemes for regularizing empirical estimates of the probabilities of words, with the goal of improving the predictive performance of language models. The simplest kind of smoothing involves adding a small constant to the empirical frequencies of words prior to normalizing those frequencies (Chen and Goodman, 1998). This approach can be shown to be equivalent to Bayesian estimation of a multinomial distribution using a Dirichlet prior (MacKay and Peto, 1994), a method that has more recently evolved into the use of compound Dirichlet-multinomial models for text (Elkan, 2006; Madsen et al., 2005). The observation of a correspondence between smoothing methods and Bayesian inference has been used to define more complex smoothing schemes based on hierarchical Bayesian models (MacKay and Peto, 1994). The connection between Pitman-Yor processes and Kneser-Ney smoothing is one instance of this broader correspondence, and was independently pointed out by Teh (2006a,b) following our own work on this topic (Goldwater et al., 2006a). More recently, Wood and Teh (2008, 2009) have developed more sophisticated cross-domain smoothing models by combining multiple hierarchical Pitman-Yor processes.

Another strand of work on Bayesian models of language aims to improve unsupervised (or semi-supervised) learning of linguistic structure. Much of this work can be traced back to the latent Dirichlet allocation (LDA) model and related work on document clustering and topic modeling by Blei and colleagues (Blei et al., 2002, 2003, 2004). While LDA takes a bag-of-words approach to language modeling, recent research in the computational linguistics community has focused on using similar Bayesian techniques to develop models of linguistic structure with more sophisticated intra- and inter-word dependencies. For example, Goldwater et al. (2006b) presented a model based on the hierarchical Dirichlet process (Teh et al., 2005) to identify word boundaries in unsegmented text. This model is very similar to the hierarchical Pitman-Yor language model described in Section 6.3 as well as in Teh (2006a). Finkel et al. (2007) and Liang et al. (2007) introduced models for learning better syntactic categories for parsing by extending the idea of the infinite hidden Markov model (Beal et al., 2002; Teh et al., 2005) to probabilistic context-free grammars (PCFGs) and dependency trees. Johnson et al. (2007) described a different kind of infinite Bayesian model for learning grammatical structure, the adaptor grammar, which is more directly based on the two-stage framework presented here. An adaptor grammar can be seen as a two-stage model in which the generator is a PCFG. Adaptor grammars have since been used for learning word segmentation, syllable structure, and morphology in English and Sesotho (Johnson et al., 2007; Johnson, 2008a,b),

as well as for named-entity clustering (Elsner et al., 2009). They have also been extended by Cohn et al. (2009), Post and Gildea (2009), and O'Donnell et al. (2009), who independently proposed very similar generalizations of the adaptor grammar for learning tree substitution grammars.

Finally, although this paper focuses primarily on the general Bayesian framework rather than the specific application to morphological learning that we discuss in Sections 7 and 8, it is worth mentioning a few other notable approaches to the unsupervised learning of morphology. Probably the most well-known systems are Linguistica (Goldsmith, 2001, 2006) and Morfessor (Creutz and Lagus, 2004, 2005), both of which are based on probabilistic models using maximum *a posteriori* estimation, and are freely available for download. A number of other systems use more heuristic approaches; Goldsmith (2001) provides a thorough review. An interesting recent approach uses sentence-aligned multilingual texts to perform simultaneous morphological segmentation on multiple languages (Snyder and Barzilay, 2008). The Bayesian model used in that work can be viewed as an extension of the word segmentation model of Goldwater et al. (2006b) described above.

3. The Two-stage Approach

The key idea behind our two-stage framework is to divide the process of generating text into two parts, one of which is sufficient to produce a power-law distribution over word frequencies. In this section we briefly review mechanisms that give rise to power-law distributions and then formally define our framework.

3.1 Producing Power-law Distributions

Assume we want to generate a sequence of n outcomes, $\mathbf{z} = (z_1, \ldots, z_n)$, with each outcome z_i being drawn from a set of (possibly unbounded) size K. Many of the stochastic processes that produce power laws are based upon the principle of *preferential attachment*, where the probability that the ith outcome, z_i, takes on a particular value k depends upon the frequency of k in $\mathbf{z}_{-i} = (z_1, \ldots, z_{i-1})$ (Mitzenmacher, 2004). For example, the number of links pointing to a given web page is sometimes modeled as a power-law distribution, which can be explained by assuming that new web pages are more likely to include links to already-popular pages (Mitzenmacher, 2004). An early preferential attachment process, due to Simon (1955), chooses z_i according to

$$P(z_i = k \,|\, \mathbf{z}_{-i}) = a\frac{1}{K} + (1-a)\frac{n_k^{(\mathbf{z}_{-i})}}{i-1}$$

where $n_k^{(\mathbf{z}_{-i})}$ is the number of times k occurs in $\mathbf{z}_{-i}$, and $0 < a < 1$ is a parameter of the process. This "rich-get-richer" process means that a few outcomes appear with very high frequency in $\mathbf{z}$, while most outcomes appear with low frequency—the key attribute of a power-law distribution. In this case, the power law has parameter $g = 1/(1-a)$.

One problem with this kind of model is that different permutations of the outcomes $\mathbf{z}$ have different probabilities. While this may be appropriate for some settings, the assumption of a temporal ordering restricts the contexts in which such models can be applied. In particular, it is much more restrictive than the assumption of independent sampling that underlies most statistical language models. Consequently, we will focus on a different preferential attachment scheme, based upon the two-parameter species sampling model (Pitman, 1995; Pitman and Yor, 1997) known as the Pitman-Yor process (Ishwaran and James, 2003). We will refer to this scheme as the Pitman-Yor Chinese

restaurant process (PYCRP), as it is a generalization of the more widely known Chinese restaurant process (CRP; Aldous, 1985). Under these schemes, outcomes follow a power-law distribution, but remain *exchangeable*: the probability of a set of outcomes is not affected by their ordering (Aldous, 1985). In addition to its theoretical benefits, the property of exchangeability has practical value in permitting the use of standard sampling algorithms for inference. We return to discussion of the CRP and PYCRP in Section 4 after introducing the basic conceptual framework of the two-stage language model.

3.2 The Generator and Adaptor

In our two-stage modeling framework, a sequence of word tokens $\mathbf{w} = (w_1, \ldots, w_n)$ is generated as follows:

1. Generate a sequence of lexical items $\boldsymbol{\ell} = (\ell_1, \ldots, \ell_K)$ from some probability distribution P_φ parameterized by φ. For example, $(\ell_1, \ldots, \ell_4)$ = (the, dog, a, the). We refer to P_φ as the *lexicon generator* (or simply *generator*). Note that our use of the term *lexical item* is non-standard. Ignoring homophony, a lexicon normally contains one instance of each word type. Here, P_φ is a discrete distribution and the lexical items are generated independently, so the same word type may occur more than once in $\boldsymbol{\ell}$.[1] In the remainder of the paper, we use *lexical item* to refer to the items produced by the generator, *word type* to refer to unique wordforms, and *word* or *token* to refer to word tokens.

2. Generate a sequence of integers $\mathbf{z} = (z_1, \ldots, z_n)$ with $1 \leq z_i \leq K$, where $z_i = k$ indicates that $w_i = \ell_k$ (that is, z_i is the index of the lexical item corresponding to w_i). For example, $(z_1, \ldots, z_9) = (1,2,1,1,3,1,1,4,3)$, so that, in combination with $(\ell_1, \ldots, \ell_4)$ from above, $(w_1, \ldots, w_9)$ = (the, dog, the, the, a, the, the, the, a). The integers $\mathbf{z}$ are assumed to be generated by some stochastic process P_γ with one or more parameters γ. We refer to this process as the *adaptor*.

We use the notation TwoStage(P_γ, P_φ) to refer to a two-stage model with adaptor P_γ and generator P_φ. A graphical model illustrating the dependencies between the variables in this framework is shown in Figure 1.

The two-stage modeling framework is very general: many different distributions could be used for the generator and adaptor. However, given the discussion above, it is sensible to assume that P_γ is chosen so that the frequencies with which different integer outcomes are produced follow a power-law distribution. In this case, when P_φ is a distribution with infinite support, the power-law distribution over integers produced in Step 2 will result in a power-law distribution over the frequencies in the final sequence of words. Thus, the adaptor "adapts" the word frequencies produced by the generator to fit a power-law distribution. Different choices for the generator model will allow different kinds of linguistic structure to be learned. Here, we show that morphological structure can be learned using a generator that produces words by choosing a stem and suffix and concatenating them together. In other work, we have used different generators to discover word boundaries in unsegmented text (Goldwater et al., 2006b; Johnson, 2008a) and to infer tree substitution grammars from parsed corpora or strings (Cohn et al., 2010).

1. The assumption of independence between lexical items is not strictly necessary, but is mathematically and computationally convenient. An example of a more complex distribution over lexical items that enforces uniqueness is given in Brent (1999).

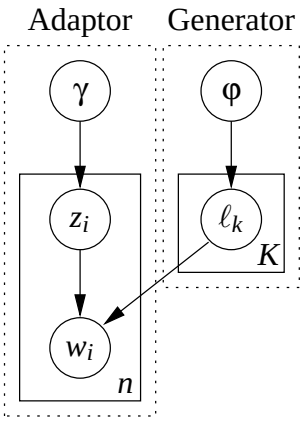


Figure 1: A graphical model representation of the two-stage language modeling framework. Arrows indicate dependencies between variables, and solid-line boxes indicate replicated portions of the model, with the number of copies shown in the lower right hand corner. Variables associated with the generator are on the right; those associated with the adaptor are on the left. Depending on the application, the words w_i may or may not be directly observed.

4. Chinese Restaurant Processes as Adaptors

While any stochastic process that results in a power-law distribution over word frequencies can be used as an adaptor, the choice of adaptor will have significant implications for the resulting model. In this section, we discuss two stochastic processes that are particularly suitable as adaptors in the two-stage framework: the Chinese restaurant process (Aldous, 1985; Pitman, 1995; Griffiths, 2006) and the Pitman-Yor Chinese restaurant process (Pitman, 1995; Pitman and Yor, 1997; Ishwaran and James, 2003). Both the CRP and PYCRP are used in nonparametric Bayesian statistics, with the more widely known CRP arising as the distribution over the sizes of mixture components in infinite mixture models (Rasmussen, 2000). We review the definitions of these processes, discuss the properties that make them useful as adaptors, and define the two-stage models that result from using CRP or PYCRP adaptors.

4.1 The Chinese Restaurant Process

The Chinese restaurant process is a simple stochastic process that can be described using the analogy of a restaurant with an infinite number of tables, each of which has an infinite seating capacity. Customers enter the restaurant one at a time, and choose a table at which to sit. The probability of choosing an occupied table is proportional to the number of people already sitting there, and the probability of choosing an unoccupied table is proportional to some constant parameter α. That is, if z_i is the index of the table chosen by the ith customer, then

$$P(z_i = k \,|\, \mathbf{z}_{-i}, \alpha) = \begin{cases} \frac{n_k^{(\mathbf{z}_{-i})}}{i-1+\alpha} & 1 \le k \le K(\mathbf{z}_{-i}) \\ \frac{\alpha}{i-1+\alpha} & k = K(\mathbf{z}_{-i})+1 \end{cases}$$

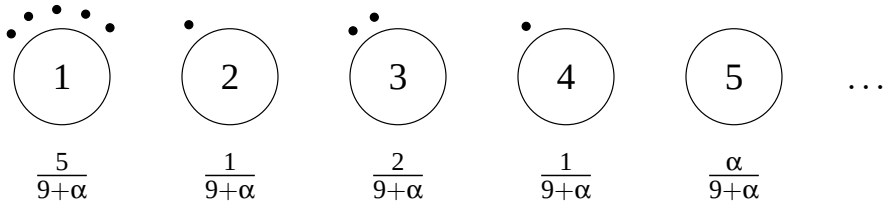


Figure 2: An illustration of the Chinese restaurant process, reproduced from Goldwater et al. (2009). Black dots indicate the number of customers sitting at each table for the example case $\mathbf{z}_{-10} = (1,2,1,1,3,1,1,4,3)$. Below each table is $P(z_{10} = k \,|\, \mathbf{z}_{-10})$. Note that the number of customers at each table—and thus $P(z_{10} = k \,|\, \mathbf{z}_{-10})$, the probability distribution over the next customer—would remain the same for any ordering of the integers in $\mathbf{z}_{-10}$. This is the property of exchangeability.

where $\mathbf{z}_{-i}$ is the seating arrangement of the previous $i-1$ customers, $n_k^{(\mathbf{z}_{-i})}$ is the number of customers already assigned to table k by $\mathbf{z}_{-i}$, $K(\mathbf{z}_{-i})$ is the total number of occupied tables in $\mathbf{z}_{-i}$, and $\alpha \geq 0$ is a parameter of the process determining how "spread out" the customers become. Higher values of α mean that more new tables will be occupied relative to the number of customers, leading to a more uniform distribution of customers across tables. The first customer by definition sits at the first table, so this distribution is well-defined even when $\alpha = 0$. See Figure 2 for an illustration.

Under this model, the probability of a particular sequence of table assignments for n customers is given by

$$\begin{aligned} P(\mathbf{z}\,|\,\alpha) &= 1 \cdot \prod_{i=2}^{n} P(z_i \,|\, \mathbf{z}_{-i}, \alpha) \\ &= \left(\prod_{i=2}^{n} \frac{1}{i-1+\alpha} \right) \left(\alpha^{K(\mathbf{z})-1} \right) \left(\prod_{k=1}^{K(\mathbf{z})} (n_k^{(\mathbf{z})} - 1)! \right) \\ &= \frac{\Gamma(1+\alpha)}{\Gamma(n+\alpha)} \cdot \alpha^{K(\mathbf{z})-1} \cdot \prod_{k=1}^{K(\mathbf{z})} (n_k^{(\mathbf{z})} - 1)! \end{aligned} \tag{1}$$

where the Gamma function is defined as $\Gamma(x) = \int_0^\infty u^{x-1} e^{-u} du$ for $x > 0$, and is a generalized factorial function: $\Gamma(x) = (x-1)!$ for positive integer x, and $\Gamma(x) = (x-1)\Gamma(x-1)$ for any $x > 0$.[2]

It is easy to see that any reordering of the table assignments in $\mathbf{z}$ will result in the same factors in Equation 1, so the CRP is exchangeable.[3] As the number of customers becomes large, the CRP produces a power-law distribution over the number of customers seated at each table, where the power-law exponent g is equal to 1 (Arratia et al., 1992).

2. It is more standard to see the joint distribution of table assignments in the CRP given as $P(\mathbf{z}) = \frac{\Gamma(\alpha)}{\Gamma(n+\alpha)} \cdot \alpha^{K(\mathbf{z})} \cdot \prod_{k=1}^{K(\mathbf{z})} (n_k^{(\mathbf{z})} - 1)!$. This distribution is derived from the Dirichlet process (see Section 5.2), which is defined only for $\alpha > 0$, and is equivalent to Equation 1 in that case. We use the distribution in Equation 1 because it is defined also for $\alpha = 0$, which is a possible (if uninteresting) parameter value in the CRP.

3. When considering exchangeability, the table assignments should be viewed as partitioning the integers $1, \ldots, i$ into equivalence classes. The requirement that $z_i \in 1, \ldots, \max(\mathbf{z}_{i-1}) + 1$ ensures there is a 1-to-1 mapping between equivalence classes and the set of integers in $\mathbf{z}$.

The preceding paragraphs indicate how the CRP can be used to create a power-law distribution over integers, but to create a distribution over words we need to combine it with a lexicon generator to make a full two-stage model. For expository purposes, we continue to use the generic lexicon generator P_φ, a distribution parameterized by φ, so the full model is TwoStage(CRP(α),P_φ). This model can be viewed as a restaurant in which each table is labeled with a lexical item produced by P_φ. Each customer represents a word token, so that the number of customers at a table corresponds to the frequency of the lexical item labeling that table. A new word token is generated by seating a new customer, producing either a new token of an existing lexical item (if the customer sits at an existing table: in this case the new token will have the same word type as the lexical item labeling that table) or the first token of a new lexical item (if the customer sits at a new table: in this case a new label is generated using P_φ, and all later customers at this table will be additional tokens of the same word type).

Under this model, the probability that the ith token in a sequence takes on the value w, given the previous labels and table assignments, can be found by summing over all the existing tables labeled with w, plus a possible new table labeled with w:

$$\begin{aligned}
&P(w_i = w \,|\, \mathbf{z}_{-i}, \boldsymbol{\ell}(\mathbf{z}_{-i}), \alpha, \varphi) \\
&\quad = \sum_{k=1}^{K(\mathbf{z}_{-i})} P(w_i = w \,|\, z_i = k, \ell_k) P(z_i = k \,|\, \mathbf{z}_{-i}, \alpha) \\
&\qquad + P(w_i = w \,|\, z_i = K(\mathbf{z}_{-i}) + 1, \varphi) P(z_i = K(\mathbf{z}_{-i}) + 1 \,|\, \mathbf{z}_{-i}, \alpha) \\
&\quad = \sum_{k=1}^{K(\mathbf{z}_{-i})} I(\ell_k = w) \frac{n_k^{(\mathbf{z}_{-i})}}{i - 1 + \alpha} + P_\varphi(w) \frac{\alpha}{i - 1 + \alpha} \\
&\quad = \frac{n_w^{(\mathbf{w}_{-i})} + \alpha P_\varphi(w)}{i - 1 + \alpha} \qquad (2)
\end{aligned}$$

where $\boldsymbol{\ell}(\mathbf{z}_{-i})$ are the labels of all the tables in $\mathbf{z}_{-i}$, $I(.)$ is an indicator function taking on the value 1 when its argument is true and 0 otherwise, and $n_w^{(\mathbf{w}_{-i})}$ is the number of previous occurrences of the word type w in $\mathbf{w}_{-i}$ (that is, the number of customers that $\mathbf{z}_{-i}$ assigns to tables labeled with w). This distribution is illustrated in Figure 3.

The probability of an entire sequence of words $P(\mathbf{w} \,|\, \alpha, \varphi)$ can be found by marginalizing out $\boldsymbol{\ell}$ and $\mathbf{z}$ from the joint distribution $P(\mathbf{w}, \mathbf{z}, \boldsymbol{\ell} \,|\, \alpha, \varphi)$. Note that unless $\ell_{z_i} = w_i$ for all i, $P(\mathbf{w}, \mathbf{z}, \boldsymbol{\ell} \,|\, \alpha, \varphi) = 0$, so we need only sum over cases where $\ell_{z_i} = w_i$ for all i. In this situation, $P(\mathbf{w}, \mathbf{z}, \boldsymbol{\ell} \,|\, \alpha, \varphi) = P(\mathbf{z}, \boldsymbol{\ell} \,|\, \alpha, \varphi) = P(\mathbf{z} \,|\, \alpha) P_\varphi(\boldsymbol{\ell})$,[4] so we can compute the desired distribution as

$$\begin{aligned}
P(\mathbf{w} \,|\, \alpha, \varphi) &= \sum_{\mathbf{z}, \boldsymbol{\ell}} P(\mathbf{z} \,|\, \alpha) P_\varphi(\boldsymbol{\ell}) \\
&= \sum_{\mathbf{z}, \boldsymbol{\ell}} \frac{\Gamma(1 + \alpha)}{\Gamma(n + \alpha)} \alpha^{K(\mathbf{z}) - 1} \prod_{k=1}^{K(\mathbf{z})} \left(P_\varphi(\ell_k) (n_k^{(\mathbf{z})} - 1)! \right) \qquad (3)
\end{aligned}$$

where the sums range only over those $\boldsymbol{\ell}$ and $\mathbf{z}$ such that $\ell_{z_i} = w_i$ for all i.

4. We use $P_\varphi(\boldsymbol{\ell})$ rather than the equivalent $P(\boldsymbol{\ell} \,|\, \varphi)$ for consistency with our notation for the generator probability of an individual lexical item $P_\varphi(\ell)$; both $P_\varphi(\boldsymbol{\ell})$ and $P(\boldsymbol{\ell} \,|\, \varphi)$ represent the probability of producing lexical items $\boldsymbol{\ell}$ using the generator parameterized by φ. Note that in contrast, $P(\mathbf{w} \,|\, \varphi) \neq P_\varphi(\mathbf{w})$, as the latter requires that all tokens in $\mathbf{w}$ are produced by the generator, whereas the former does not.

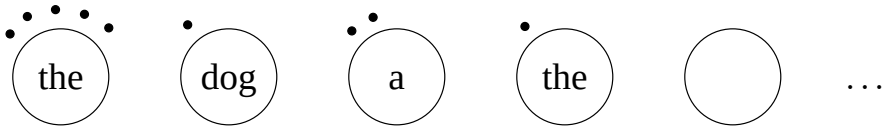


Figure 3: An illustration of the two-stage restaurant, adapted from Goldwater et al. (2009). In this example, $(\ell_1, \ldots, \ell_4)$ = (the, dog, a, the) and $\mathbf{z}_{-10} = (1,2,1,1,3,1,1,4,3)$. Each label ℓ_k is shown on table k. Black dots indicate the number of occurrences of each label in $\mathbf{w}_{-10}$ = (the, dog, the, the, a, the, the, the, a). Under this seating arrangement, $P(w_{10} = \mathsf{the}) = \frac{6+\alpha P_\varphi(\mathsf{the})}{9+\alpha}$, $P(w_{10} = \mathsf{dog}) = \frac{1+\alpha P_\varphi(\mathsf{dog})}{9+\alpha}$, $P(w_{10} = \mathsf{a}) = \frac{2+\alpha P_\varphi(\mathsf{a})}{9+\alpha}$, and for any other word w, $P(w_{10} = w) = \frac{\alpha P_\varphi(w)}{9+\alpha}$.

Notice that the distribution over words given in Equation 2 leads to an alternative way of viewing the TwoStage(CRP(α), P_φ) model, as a cache model. Under this view, each word is generated in one of two ways: from a cache of previously occurring lexical items (with probability $\frac{n}{n+\alpha}$ if we use the CRP adaptor) or as a novel lexical item (with probability $\frac{\alpha}{n+\alpha}$). Items from the cache are chosen with probability proportional to the number of times they have occurred before in $\mathbf{w}$. Novel items are chosen according to the probability distribution of the lexicon generator (which means that, strictly speaking, they are not always "novel"—that is, novel word types—since the generator may produce duplicates). This interpretation clarifies the significance of the parameters α and P_φ. Prior expectations regarding the probability of encountering a novel lexical item are reflected in the value of α, so lower values of α will lead to an expectation of fewer lexical items (and word types) during inference. Prior expectations about the relative probabilities of different novel items are reflected in P_φ, so the choice of generator determines the kinds of lexical items that are likely to be inferred from the data. If the generator is a distribution over an infinite number of items, the cache model makes it clear that the number of different word types that will be observed in a finite corpus is not fixed in advance. Rather, new word types can be generated "on the fly" from an infinite supply. In general, the number of different word types observed in a corpus will slowly grow as the size of the corpus grows.

4.2 The Pitman-Yor Generalization

For much of this paper, we will be focusing on an adaptor based on the Pitman-Yor process. This adaptor is a generalization of the CRP, defined as

$$P(z_i = k \,|\, \mathbf{z}_{-i}, a, b) = \begin{cases} \frac{n_k^{(\mathbf{z}_{-i})} - a}{i-1+b} & 1 \le k \le K(\mathbf{z}_{-i}) \\ \frac{K(\mathbf{z}_{-i})a+b}{i-1+b} & k = K(\mathbf{z}_{-i})+1 \end{cases} \tag{4}$$

where $0 \le a < 1$ and $b \ge 0$ are parameters of the process. As in the CRP, $z_1 = 1$ by definition. When $a = 0$ and $b = \alpha$, this process reduces to the CRP, so we refer to it as the Pitman-Yor Chinese restaurant process (PYCRP). Like the CRP, the PYCRP is exchangeable and produces a power-law distribution on the number of customers seated at each table. In this case, the power-law exponent g

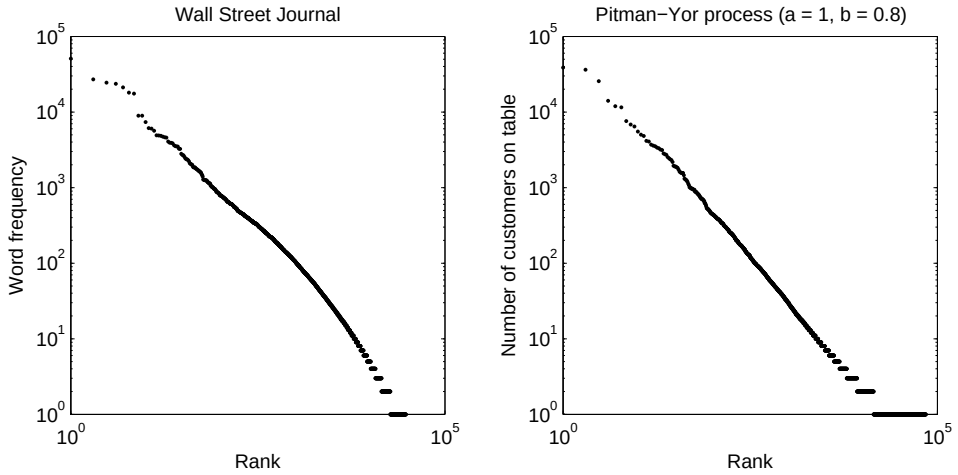


Figure 4: Simulating power laws in natural language, illustrated using Zipf plots. The Zipf plot displays the log frequency of a word as a function of the log of the rank of that frequency (i.e., the number of words with frequency greater than or equal to that word). A power-law distribution in word frequency, with the probability of a frequency of n_w proportional to n_w^{-g}, results in a straight line on the plot with slope $1/(g-1)$. Here, the left-hand plot shows the distribution of word frequencies in sections 0-20 from the Penn Wall Street Journal treebank, while the right-hand plot shows the distribution of the number of customers at each table produced by 500,000 draws from the PYCRP with parameters $a = 0.8$ and $b = 1$. Both plots have a slope of roughly -1.25, corresponding to a power-law distribution with exponent $\gamma = 1.8$.

is equal to $1+a$ (Pitman, 2006), which includes the $g \approx 1.8$ seen for natural languages (see Figure 4). We defer further discussion of the significance of the parameters a and b to Section 6.2.

Under the PYCRP, the probability of a particular seating arrangement $\mathbf{z}$ is

$$\begin{aligned} P(\mathbf{z}\,|\,a,b) &= 1 \cdot \prod_{i=2}^{n} P(z_i\,|\,\mathbf{z}_{-i},a,b) \\ &= \left(\prod_{i=2}^{n} \frac{1}{i-1+b}\right)\left(\prod_{k=1}^{K(\mathbf{z})-1}(ka+b)\right)\left(\prod_{k=1}^{K(\mathbf{z})}\prod_{i=1}^{n_k^{(\mathbf{z})}-1}(i-a)\right) \\ &= \frac{\Gamma(1+b)}{\Gamma(n+b)}\left(\prod_{k=1}^{K(\mathbf{z})-1}(ka+b)\right)\left(\prod_{k=1}^{K(\mathbf{z})}\frac{\Gamma(n_k^{(\mathbf{z})}-a)}{\Gamma(1-a)}\right). \end{aligned}$$

As with the CRP, we can define a generic two-stage model with a PYCRP adaptor by assuming a generator P_φ parameterized by φ. Under this TwoStage(PYCRP$(a,b),P_\varphi$) model, the probability of

generating word w given the seating arrangement and label assignments of the previous words is

$$\begin{aligned}
&P(w_i = w \,|\, \mathbf{z}_{-i}, \boldsymbol{\ell}(\mathbf{z}_{-i}), a, b, \varphi) \\
&\quad = \sum_{k=1}^{K(\mathbf{z}_{-i})} P(w_i = w \,|\, z_i = k, \ell_k) P(z_i = k \,|\, \mathbf{z}_{-i}, a, b) \\
&\qquad + P(w_i = w \,|\, z_i = K(\mathbf{z}_{-i}) + 1, \varphi) P(z_i = K(\mathbf{z}_{-i}) + 1 \,|\, \mathbf{z}_{-i}, a, b) \\
&\quad = \sum_{k=1}^{K(\mathbf{z}_{-i})} I(\ell_k = w) \frac{n_k^{(\mathbf{z}_{-i})} - a}{i - 1 + b} + P_\varphi(w) \frac{K(\mathbf{z}_{-i})a + b}{i - 1 + b} \\
&\quad = \frac{n_w^{(\mathbf{w}_{-i})} - K_w(\mathbf{z}_{-i})a + (K(\mathbf{z}_{-i})a + b) P_\varphi(w)}{i - 1 + b}
\end{aligned} \tag{5}$$

where $K_w(\mathbf{z}_{-i})$ is the number of tables labeled with w in $\mathbf{z}_{-i}$. The joint distribution of a sequence of words $\mathbf{w}$ is given by

$$\begin{aligned}
P(\mathbf{w} \,|\, a, b, \varphi) &= \sum_{\mathbf{z}, \boldsymbol{\ell}} P(\mathbf{z} \,|\, a, b) P_\varphi(\boldsymbol{\ell}) \\
&= \sum_{\mathbf{z}, \boldsymbol{\ell}} \frac{\Gamma(1+b)}{\Gamma(n+b)} \left(\prod_{k=1}^{K(\mathbf{z})-1} (ka + b) \right) \left(\prod_{k=1}^{K(\mathbf{z})} P_\varphi(\ell_k) \frac{\Gamma(n_k^{(\mathbf{z})} - a)}{\Gamma(1-a)} \right)
\end{aligned} \tag{6}$$

where, as in Equation 3, the sums are over only those $\boldsymbol{\ell}$ and $\mathbf{z}$ such that $\ell_{z_i} = w_i$ for all i.

5. Relationship to Other Models

The two-stage framework outlined in the previous sections has three special cases that correspond to models that have previously been used in computational linguistics and statistics: the Dirichlet-multinomial model, the Dirichlet process, and the two-parameter Poisson-Dirichlet process. In each of the following subsections, we first present the relevant equivalency, and then show that it holds.

5.1 The Dirichlet-multinomial Model

Proposition 1 *A TwoStage*$(CRP(\alpha), Multinomial(\varphi))$ *model is equivalent to a* $Dirichlet(\alpha\varphi)$*-multinomial model.*

As mentioned in Section 1, several researchers have proposed Bayesian language models based on the Dirichlet-multinomial model (MacKay and Peto, 1994; Madsen et al., 2005), also known as the Dirichlet compound multinomial model (Elkan, 2006). In this model, words are drawn from a multinomial distribution:

$$w_i \,|\, \theta \;\sim\; \text{Multinomial}(\theta)$$

where $\theta = (\theta_1, \ldots, \theta_K)$. That is, for a corpus $\mathbf{w} = (w_1, \ldots, w_n)$ made up of a finite lexicon of words $(\ell_1, \ldots, \ell_K)$, $P(w_i = \ell_k \,|\, \theta) = \theta_k$ and $P(\mathbf{w} \,|\, \theta) = \prod_{k=1}^{K} \theta_k^{n_k}$, where n_k is the number of occurrences of ℓ_k in $\mathbf{w}$. In addition, the parameters θ are themselves drawn from a Dirichlet distribution with hyperparameters $\beta = (\beta_1, \ldots, \beta_K)$:

$$\theta \,|\, \beta \;\sim\; \text{Dirichlet}(\beta).$$

The Dirichlet distribution is defined as

$$\begin{aligned} P(\theta\,|\,\beta) &= c\prod_{k=1}^{K}\theta_k^{\beta_k-1} \\ \text{with } c &= \frac{\Gamma(\sum_{k=1}^{K}\beta_k)}{\prod_{k=1}^{K}\Gamma(\beta_k)} \end{aligned}$$

where $\beta_k > 0$. It is *conjugate* to the multinomial, meaning that the posterior distribution over the parameters θ given a corpus $\mathbf{w}$ takes on the same parametric form as the prior—specifically, a Dirichlet distribution with parameters $n_k + \beta_k$, where n_k is the number of occurrences of outcome k in $\mathbf{w}$:

$$\begin{aligned} P(\theta\,|\,\mathbf{w},\beta) &\propto P(\mathbf{w}\,|\,\theta)P(\theta\,|\,\beta) \\ &\propto \prod_{k=1}^{K}\theta_k^{n_k}\prod_{k=1}^{K}\theta_k^{\beta_k-1} \\ &= \prod_{k=1}^{K}\theta_k^{n_k+\beta_k-1}. \end{aligned}$$

Due to the conjugacy of the Dirichlet and multinomial distributions, it is easy to compute the predictive distribution of w_i conditioned on the values of the previously observed words $\mathbf{w}_{-i}$ and the hyperparameters β:

$$\begin{aligned} P(w_i = j\,|\,\mathbf{w}_{-i},\beta) &= \int_\Delta P(w_i = j\,|\,\theta)P(\theta\,|\,\mathbf{w}_{-i},\beta)\,d\theta \\ &= \frac{\Gamma(n+\sum_{k=1}^{K}\beta_k)}{\prod_{k=1}^{K}\Gamma(n_k+\beta_k)}\int_\Delta \theta_j^{n_j+\beta_j}\prod_{k\neq j}\theta_k^{n_k+\beta_k-1}\,d\theta \\ &= \frac{\Gamma(n+\sum_{k=1}^{K}\beta_k)}{\prod_{k=1}^{K}\Gamma(n_k+\beta_k)}\cdot\frac{\Gamma(n_j+\beta_j+1)\prod_{j\neq k}\Gamma(n_k+\beta_k)}{\Gamma(n+\sum_{k=1}^{K}\beta_k+1)} \\ &= \frac{n_j+\beta_j}{n+\sum_{k=1}^{K}\beta_k} \end{aligned} \tag{7}$$

where all counts are with respect to $\mathbf{w}_{-i}$, and Δ indicates the probability simplex: the set of values for $\theta >= 0$ such that $\sum_k \theta_k = 1$. The third line can be derived using elementary calculus and the definition of the Gamma function, but can also be seen to hold by noting that the Dirichlet distribution must sum to 1, and therefore

$$\int_\Delta \prod_{k=1}^{K}\theta_k^{\beta_k-1}\,d\theta = \frac{\prod_{k=1}^{K}\Gamma(\beta_k)}{\Gamma(\sum_{k=1}^{K}\beta_k)}$$

holds for any positive values of β_k. Comparing Equation 7 to Equation 2 reveals that the Dirichlet-multinomial model is a special case of our two-stage framework, with a CRP adaptor and a finite generator distribution. In particular, a TwoStage(CRP(α), Multinomial(φ)) model is equivalent to a Dirichlet-multinomial model with $\beta = \alpha\varphi$.

5.2 The Dirichlet Process

Proposition 2 *A TwoStage*$(CRP(\alpha), P_\varphi)$ *model (where* P_φ *has infinite support) is equivalent to a* $DP(\alpha, P_\varphi)$ *model.*

The Dirichlet process (DP; Ferguson, 1973), used in nonparametric Bayesian statistics, can be seen as an infinite-dimensional analogue of the symmetric Dirichlet distribution (a Dirichlet distribution where all β_i are equal).[5] Whereas each sample from a Dirichlet distribution returns a distribution θ over a finite set of outcomes, each sample from a Dirichlet process returns a distribution G over a countably infinite set of outcomes. The Dirichlet process has two parameters. The *base distribution*, G_0, (which may be discrete or continuous) determines the probability that any particular outcome will be in the support of G. The *concentration parameter*, α, determines the variance in the probabilities of those outcomes under G.

Typically, the Dirichlet process is used as a prior in infinite mixture models (Lo, 1984; Escobar and West, 1995; Neal, 2000; Rasmussen, 2000), where the concentration parameter determines the relative size of each mixture component, and the base distribution determines the probable parameters for the component distributions. Instead, we can use the Dirichlet process to define a simple language model as follows:

$$\begin{aligned} G \,|\, \alpha, P_\varphi &\sim \mathrm{DP}(\alpha, P_\varphi), \\ w_i \,|\, G &\sim G \end{aligned}$$

where $\mathrm{DP}(\alpha, P_\varphi)$ refers to a Dirichlet process with concentration parameter α and base distribution $G_0 = P_\varphi$. The corresponding graphical model can be seen in Figure 5. Just as we integrated out the θ parameters of the Dirichlet-multinomial model, we can integrate out the distribution G to obtain the following predictive distribution over words (Blackwell and MacQueen, 1973):

$$w_i \,|\, \mathbf{w}_{-i}, \alpha, P_\varphi \;\sim\; \frac{1}{i-1+\alpha} \sum_{j=1}^{i-1} \delta(w_j) + \frac{\alpha}{i-1+\alpha} P_\varphi$$

where $\delta(w_j)$ is a point mass at w_j. Rewriting the predictive distribution as a probability mass function reveals that the $\mathrm{DP}(\alpha, P_\varphi)$ model is equivalent to a $\mathrm{TwoStage}(\mathrm{CRP}(\alpha), P_\varphi)$ model:

$$P(w_i = w \,|\, \mathbf{w}_{-i}, \alpha, P_\varphi) \;=\; \frac{n_w^{(\mathbf{w}_{-i})} + \alpha P_\varphi(w)}{i-1+\alpha}.$$

Note that although G assigns probability to a countably infinite set of outcomes, the predictive distribution can be computed using only the frequencies of previous items and the base distribution P_φ.

It is worth pointing out that this DP language model can still technically be viewed as a mixture model, although a degenerate one. Each lexical item corresponds to a separate mixture component parameterized by its label ℓ_k and with a 0/1 likelihood function: $P(w_i \,|\, \ell_{z_i}) = I(w_i = \ell_{z_i})$. Thus, every data point in a single mixture component is identical. As a result, the potential applications of two-stage models and infinite mixture models are somewhat different. Infinite mixture models are

5. Specifically, as described by Neal (2000), the predictive distribution of a Dirichlet process mixture model can be obtained by taking the limit as k goes to infinity of a k-component finite mixture model with a symmetric Dirichlet prior.

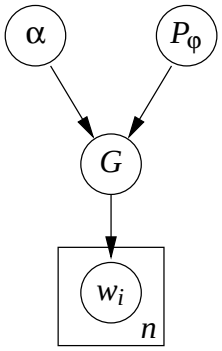


Figure 5: A graphical representation of the Dirichlet process language model.

more appropriate when the base distribution is a simple parameterized distribution (e.g., a Gaussian) and the clusters are expected to have some variability, whereas two-stage models are intended for cases where the base distribution may be more complex (e.g., a PCFG) but there is no variability between data points in a single cluster. An interesting area for future work lies in combining these two features to create models with complex base distributions as well as variability in the output of each cluster.

5.3 The Pitman-Yor Process

Proposition 3 *A TwoStage*$(PYCRP(a,b),P_\varphi)$ *model (where* P_φ *has infinite support) is equivalent to a* $PYP(a,b,P_\varphi)$ *model.*

Above, we described the Dirichlet process as the infinite dimensional analogue of the Dirichlet distribution. Another way of defining the Dirichlet process, which leads to the Pitman-Yor process as a generalization, is through the "stick-breaking" construction (Sethuraman, 1994). Recall that the distribution G produced by the Dirichlet process has two parts: a countably infinite set of possible outcomes drawn from the base distribution G_0, and weights assigned to those outcomes. The stick-breaking construction describes the distribution of these weights. Under this construction, we define a sequence of random variables $(V_1, V_2, \ldots)$, each following a Beta$(1,\alpha)$ distribution. The distribution of the weights from the Dirichlet process is the same as the distribution of the set of random variables in which the kth variable is defined to be $\prod_{j=1}^{k-1}(1-V_j)V_k$. Intuitively, this is the distribution we obtain over portions of a stick of length 1 when we break that stick into two pieces with sizes proportional to $(V_1,\ 1-V_1)$, then break the remainder into proportions $(V_2,\ 1-V_2)$, and so forth.

The stick-breaking construction for the Dirichlet process has just one parameter, α, but can be generalized through the introduction of a second parameter to define a new distribution, the Pitman-Yor process (PYP; Pitman, 1995; Pitman and Yor, 1997; Ishwaran and James, 2003). The stick-breaking construction for this two-parameter distribution is similar to that given above, except V_j is drawn from a Beta$(1-a, ja+b)$ distribution. Integrating over the weights in the two-parameter stick-breaking construction gives a predictive distribution that is similar to that of the Dirichlet process. More specifically, if we use $\mathbf{z} = (z_1, \ldots, z_n)$ to index the possible outcomes, we obtain the predictive distribution given in Equation 4, that is, the PYCRP. The relationship

between the PYCRP and the Pitman-Yor process is thus analogous to that between the CRP and the Dirichlet process: the PYCRP is the discrete distribution on partitions obtained by integrating over a distribution (the Pitman-Yor process) with weights generated from the two-parameter stick-breaking process. Therefore, just as TwoStage(CRP(α),P_φ) is equivalent to DP(α,P_φ), we have that TwoStage(PYCRP(a,b),P_φ) is equivalent to PYP(a,b,P_φ).

6. Effects of the Adaptor on Frequencies

We have now defined the two-stage modeling framework, shown that several Bayesian language models proposed elsewhere can be viewed as special cases of this framework, and presented two adaptors that generate power-law distributions over words. In this section, we consider how using these adaptors affects estimates of the parameters of the generator—the process that produces the underlying lexicon. In doing this, we return to our second motivating concern: the issue of how we might explain the damping of word frequencies, with the extreme case being reconciliation of models based on unique word *types* with those based on the observed frequencies of word *tokens*. We first discuss the general implications of using the CRP and PYCRP for estimating the parameters of the generator. We then explain how, in a TwoStage(PYCRP(a,b),P_φ) language model, the parameters of the PYCRP determine whether the parameters of the generator will be inferred based on word types, tokens, or some interpolation between the two. Finally, we show that this Pitman-Yor language model provides a principled explanation for the combination of token counts and type counts found in Kneser-Ney smoothing (Ney et al., 1994; Kneser and Ney, 1995).

6.1 Impact of the Adaptor on Frequencies used for Estimation

By introducing an adaptor into our model, we provide a route by which word tokens can appear in a corpus without having been directly produced by the generator. As a consequence, any estimate of the parameters of the generator will be based only on those tokens for which the generator is considered responsible, which will be a subset of the tokens in the corpus. The adaptor will thus have the effect of damping the frequencies from which the parameters of the generator are estimated, with the nature of this damping depending on the properties of the adaptor. In particular, we will show that using the CRP or PYCRP as adaptors is approximately equivalent to estimating the generator parameters from log transformed or inverse-power transformed token counts, respectively.

We can see how the choice of adaptor affects the frequencies used for estimating the parameters φ of the generator by considering how to estimate φ from the observed corpus $\mathbf{w}$.[6] In general, the parameters of generators can be estimated using Markov chain Monte Carlo methods, as we demonstrate in Section 7. Here, we will present some general results characterizing how the frequencies used in estimation are damped by using the CRP or PYCRP as an adaptor.

For either maximum-likelihood or Bayesian estimation, the relationship between φ and the corpus $\mathbf{w}$ is characterized by the likelihood $P(\mathbf{w}\,|\,\varphi)$ (where we suppress the conditioning on the adaptor parameters α or (a,b) here and in the remainder of this section). As noted in Section 4, the likelihood can be expressed as

$$P(\mathbf{w}|\varphi) = \sum_{\mathbf{z},\boldsymbol{\ell}} P(\mathbf{z}) P_\varphi(\boldsymbol{\ell}) \tag{8}$$

6. Under the interpretation of this model as a Pitman-Yor process mixture model, this is analogous to estimating the base measure G_0 in a Dirichlet process mixture model (e.g., Neal, 2000).

where the sum ranges over those $\mathbf{z},\boldsymbol{\ell}$ pairs that generate $\mathbf{w}$.

Equation 8 makes it clear that the likelihood is affected not only by φ but also by $P(\mathbf{z})$. Nevertheless, we can still make some basic statements about the relationship between $\mathbf{w}$ and φ by considering the properties of the model as a whole. First, notice that the total frequency n_w of each word type w, as obtained by summing the counts on all tables labeled with that type, will equal the frequency of w in the corpus $\mathbf{w}$. Second, all that matters for the estimation of φ_w (the parameter(s) associated with word type w) is the number of tables labeled with w, since this value is equal to the number of times we have drawn w from the generator—all other instances of w are produced by the adaptor. Thus, we can gain insight into how estimates of φ are likely to be affected by the choice of adaptor by considering how the adaptor affects the relationship between the frequency of a word type and the number of tables labeled with that type.

The analysis given in the previous paragraph suggests that we want to compute the expected number of tables labeled with a given word type under different adaptors. This expectation can be computed from the posterior distribution on $\mathbf{z}$ and $\boldsymbol{\ell}$ given $\mathbf{w}$, which can be decomposed as $P(\mathbf{z},\boldsymbol{\ell}\,|\,\mathbf{w}) = P(\boldsymbol{\ell}\,|\,\mathbf{z},\mathbf{w})P(\mathbf{z}\,|\,\mathbf{w})$. Note that $P(\boldsymbol{\ell}\,|\,\mathbf{z},\mathbf{w})$ is equal to one if $\mathbf{z}$ and $\mathbf{w}$ are consistent with $\boldsymbol{\ell}$, and zero otherwise, so we can compute $P(\mathbf{z},\boldsymbol{\ell}\,|\,\mathbf{w})$ by computing $P(\mathbf{z}\,|\,\mathbf{w})$ subject to this consistency constraint, that is, such that for each word type w, the appropriate n_w tokens of $\mathbf{w}$ are of type w. In order to simplify the mathematics, in the rest of this section we assume that each lexical item ℓ_j produced by the generator is independent and identically distributed (i.i.d.) given φ. That is, if $\boldsymbol{\ell} = (\ell_1,\ldots,\ell_K)$, then

$$P_\varphi(\boldsymbol{\ell}) \;=\; \prod_{\ell=1}^{K} P_\varphi(\ell_j).$$

First, we consider the CRP adaptor. In this case, we can obtain a good approximation to the expectation of the number of tables over the posterior distribution. The posterior distribution is exchangeable, so we can calculate the distribution over the number of lexical entries for a given word type w by imagining that the n_w instances of w are the first n_w tokens in our corpus. The posterior probability distribution for the seating assignment of the ith token is

$$P(z_i = k\,|\,w_i = w,\mathbf{z}_{-i},\boldsymbol{\ell}(\mathbf{z}_{-i}),\varphi) \;=\; \frac{P(z_i = k, w_i = w\,|\,\mathbf{z}_{-i},\boldsymbol{\ell}(\mathbf{z}_{-i}),\varphi)}{P(w_i = w\,|\,\mathbf{z}_{-i},\boldsymbol{\ell}(\mathbf{z}_{-i}),\varphi)}$$

where

$$P(z_i = k, w_i = w\,|\,\mathbf{z}_{-i},\boldsymbol{\ell}(\mathbf{z}_{-i}),\varphi) \;=\; \begin{cases} \frac{n_k^{(\mathbf{z}_{-i})}}{i-1+\alpha}\cdot I(\ell_k = w) & 1 \le k \le K(\mathbf{z}_{-i}) \\ \frac{\alpha}{i-1+\alpha}\cdot P_\varphi(w) & k = K(\mathbf{z}_{-i})+1 \end{cases}$$

and the denominator is given by Equation 2. Dividing through yields

$$P(z_i = k\,|\,w_i = w,\mathbf{z}_{-i},\boldsymbol{\ell}(\mathbf{z}_{-i}),\varphi) \;=\; \begin{cases} \frac{n_k^{(\mathbf{z}_{-i})}}{n_w^{(\mathbf{w}_{-i})}+\alpha P_\varphi(w)}\cdot I(\ell_k = w) & 1 \le k \le K(\mathbf{z}_{-i}) \\ \frac{\alpha}{n_w^{(\mathbf{w}_{-i})}+\alpha P_\varphi(w)}\cdot P_\varphi(w) & k = K(\mathbf{z}_{-i})+1 \end{cases} \tag{9}$$

which we can now use to calculate the expected number of occupied tables (i.e., lexical entries) for a word type that occurs n_w times. Taking $w_i = w$ for $i = 1,\ldots,n_w$ means that $P_\varphi(w)$ is fixed

for all n_w decisions, and $\alpha P_\varphi(w)$ simply becomes a constant. Inspection of Equation 9 reveals that the posterior distribution on seating assignments for all tokens of type w is given by the CRP with parameter $\alpha P_\varphi(w)$ and a total of n_w customers. As Antoniak (1974) showed, the expected number of occupied tables in this case is $\alpha P_\varphi(w) \sum_{i=1}^{n_w} 1/(\alpha P_\varphi(w) + i - 1)$, or approximately $\alpha P_\varphi(w) \log \frac{n_w + \alpha P_\varphi(w)}{\alpha P_\varphi(w)} = O(\log(n_w))$.

Unfortunately, we cannot apply a similar analysis for use of the PYCRP adaptor. While the CRP treats each word type independently (that is, ignoring dependencies in the generator, in a CRP the number of tables associated with a word type is independent of the number of tables associated with other word types), this is not true for the PYCRP. As with the CRP, the probabilities defined by the generator multiply with the terms of the PYCRP when we generate a new table, so that

$$P(z_i = k \,|\, w_i = w, \mathbf{z}_{-i}, \boldsymbol{\ell}(\mathbf{z}_{-i}), \varphi) \propto \begin{cases} (n_k^{(\mathbf{z}_{-i})} - a) \cdot I(\ell_k = w) & 1 \le k \le K(\mathbf{z}_{-i}) \\ (K(\mathbf{z}_{-i})a + b) \cdot P_\varphi(w) & k = K(\mathbf{z}_{-i}) + 1. \end{cases} \tag{10}$$

However, this distribution does not take the form of another PYCRP. We can only say that the probability of choosing a new table under this distribution is bounded above by the probability of choosing a new table under a PYCRP with parameters a and $bP_\varphi(w)$. Ignoring the effect of the number of tables associated with other word types, we expect the number of tables to be less than the number produced by simply running a PYCRP$(a, bP_\varphi(w))$ over the n_w tokens of w. The expectation of the number of tables occupied after seating n_w customers increases as $O(n_w^a)$ for the PYCRP (Teh, 2006a), providing an upper bound on the number of tables we should expect a word with frequency n_w to produce when the PYCRP is used as an adaptor.[7]

These results provide a rough heuristic for understanding how using the CRP and the PYCRP as adaptors damps the frequencies from which the parameters of the generator are estimated: using the CRP and PYCRP as adaptors will be approximately equivalent to estimation from log and inverse-power transformed frequencies respectively. To evaluate the accuracy of these approximations, we conducted an experiment using a corpus derived from sections 0-20 from the Penn Wall Street Journal treebank (Marcus et al., 1993). The corpus consisted of 30,114 unique word types, with a total of 831,190 tokens. We then examined the parameter estimates produced by several two-stage models, varying both the generator and the adaptor.

In all models, the generator was taken to be a multinomial distribution over the full vocabulary, with a symmetric Dirichlet(β) prior. This generator was used because it is relatively generic, since any distribution over a discrete set ultimately grounds out in a multinomial, and because it allows us to parametrically explore the consequences of varying the strength of the prior. We used three different kinds of prior, corresponding to different settings of the hyperparameters: $\beta = 0.001, \beta = 1$, and $\beta \to \infty$. With $\beta = 0.001$, the prior prefers sparse multinomial distributions, which means that the number of tables assigned to w has a strong effect on the resulting estimate of φ_w: word types with many tables will tend to have high φ_w, while the sparse prior will push the estimated parameters for the remaining word types closer to zero. With $\beta = 1$, the prior is uniform over multinomials, which provides some regularization of the resulting estimates towards the uniform distribution. With $\beta \to \infty$, the prior forces the estimated parameters to be the uniform distribution over all word types, so the number of tables assigned to any given word type has no effect on the estimates. Note that the i.i.d. generator assumption made above only holds when $\beta \to \infty$.

7. We recently became aware of work by Buntine and Hutter (2010), in which the expected number of occupied tables in the PYCRP is derived. In future work, we hope to include this result in our analysis.

We combined these three generators with a total of fifteen different adaptors. For each generator, five models used a CRP adaptor with $\alpha = \{1, 10, 100, 1000, 10000\}$ and ten others used a PYCRP adaptor with $a = \{0.1, 0.2, \ldots, 1.0\}$ and $b = 1$. For each combination of generator and adaptor, a Markov chain Monte Carlo (MCMC) algorithm was used to calculate the expected number of occupied tables (from which the corresponding multinomial parameters were estimated) for each word in the corpus. Details of this algorithm are provided in Appendix A. Figure 6 displays the results: the expected number of occupied tables is shown, plotted as black dots, as a function of n_w for all combinations of generators and adaptors. To produce the figure, words were binned by frequency using bins that were uniform on a log scale, and the posterior mean of the number of occupied tables per word was averaged within bins.

Figure 6 also shows as gray lines the number of tables predicted by the heuristic approximations described above. The predictions for the CRP (left column) assume that the number of tables is equal to $\alpha P_\varphi(w) \sum_{i=1}^{n_w} 1/(\alpha P_\varphi(w) + i - 1)$, using the appropriate value of α but taking $P_\varphi(w)$ to be uniform over all words. The result is accurate when $P_\varphi(w)$ is constrained to be uniform (row (c); $\beta \to \infty$), but underestimates the number of tables for high frequency words when $P_\varphi(w)$ is itself more sensitive to the number of tables (rows (a) and (b); $\beta = 0.001$ or 1). The predictions for the PYCRP (right column) assume that the number of tables is equal to n_w^a, and provide a good approximate upper bound on the number of tables, with the actual numbers being closer to this upper bound when $P_\varphi(w)$ is free to become higher for high-frequency words (row (a)). In general, the heuristic of the number of tables increasing as $O(n_w^a)$ seems more accurate when n_w is small.

The influence of the prior on the number of tables per word under the two-stage model with PYCRP adaptor can be understood in terms of how the prior affects the difference between the posterior distribution on the number of tables and the simpler PYCRP we use to approximate it. The approximation PYCRP always assigns a higher probability to new tables than the posterior distribution, but the difference between the two for a word w will depend on the value of $P_\varphi(w)$, since the approximation assumes the probability of a new table is proportional to $K(\mathbf{z}_{-i})a + bP_\varphi(w)$, while the true probability is proportional to $K(\mathbf{z}_{-i})aP_\varphi(w) + bP_\varphi(w)$. With a prior that allows the number of tables to have a strong influence on $P_\varphi(w)$ (row (a)), the most frequent words will tend to have much larger values of $P_\varphi(w)$ than the less frequent words, so the difference between the approximation and the true distribution for the most frequent words will not be very great. However, when $P_\varphi(w)$ is constrained to be more uniform (rows (b) and (c)), the difference between the approximation and the true distribution for frequent words is much larger, so the approximation is bad.

A surprising feature of the PYCRP models is the nonmonotonic relationship between n_w and the true number of tables occupied by w, which is noticeable with higher values of a (except $a = 1$) in the bottom two plots on the right. This behavior is due to a confluence of factors, which include both the high value of a and the very large number of tables required to account for all the words in the corpus (a result of the large number of word types). Under these circumstances, when the total number of tokens of w is small, it is not possible to have a table with enough tokens of w so that the probability of placing another token of w on that table is much higher than placing the token on a new table.[8] Thus, the posterior distribution over the number of tables for w will be

8. Empirically, the total number of tables K inferred by our sampler is around 65,000, so the posterior probability of assigning a token of w to a new table with $a = .9$, $b = 1$, and uniform $P_\varphi(w)$ is roughly proportional to $((65,000)(0.9) + 1)\frac{1}{30,114} \approx 2$, wheras the probability of assigning w to an old table is proportional to $n_k^{(\mathbf{z}_{-i})} - 0.9$, which is actually less than two unless there are already more than two tokens on the old table. Even with five tokens already on the old table, the probability of using the old table is only about twice that of using the new table.

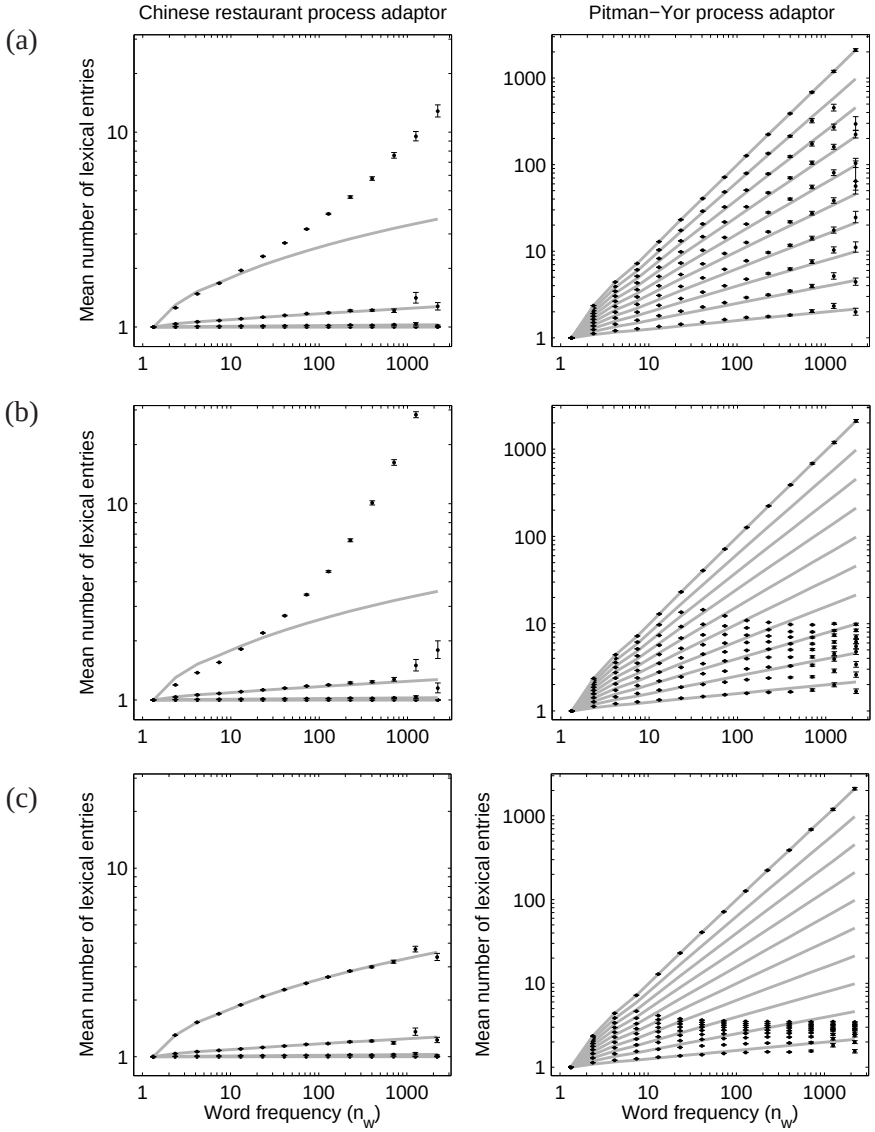


Figure 6: Mean number of occupied tables as a function of word frequency (n_w) under models of the text of sections 0-20 of the Penn Wall Street Journal treebank. The three rows of panels correspond to multinomial generators with Dirichlet(β) priors and (a) $\beta = 0.001$, (b) $\beta = 1$, and (c) $\beta \to \infty$. Each row shows the results of using the CRP (left) and PYCRP (right) as adaptors. All axes are on a log scale. Black dots and error bars show the empirical means and standard errors computed using MCMC; gray lines indicate approximations described in the text. The left-hand column shows results for the CRP with parameter $\alpha = \{1, 10, 100, 1000, 10000\}$ (from bottom to top; results for the first three are nearly identical and lie on top of each other in the graphs). The right-hand column shows results for the PYCRP with $b = 1$ and $a = \{0.1, 0.2, \ldots, 1.0\}$ (from bottom to top).

relatively uniform, and the average number of inferred tables will be large relative to the size of n_w. However, as n_w increases, it becomes possible to cluster the tokens so as to place a larger number on each table. There is a bigger win in probability for inferring a configuration with fewer tables when n_w is large, because this situation implies that more of the tokens were generated from old tables with many existing tokens, which have much higher probability than tables with zero or even a handful of existing tokens. Note that when P_φ is uniform (i.e., low for all words, as in row (c)), the probability of a new table is quickly outweighed by the probability of an existing table even with low counts. However, when β is small so that P_φ is estimated to be higher for more frequent words, the nonmonotonicity is not observed until n_w becomes much larger.

Overall, the theoretical and empirical results presented in this section suggest that our two-stage approach can provide a way to justify the use of logarithmic and inverse-power frequency damping in text processing applications. More significantly, this justification explains why adopting these schemes improves performance: it compensates for the kind of "rich-get-richer" processes that produce power-law distributions in natural language.

6.2 Types and Tokens

The most extreme kind of frequency damping is throwing away all but a single instance of each word type, and only keeping track of the unique word types that appear in the corpus. Just as we can explain other forms of frequency damping in terms of our two-stage framework, we can show that the TwoStage(PYCRP$(a,b),P_\varphi$) model provides a justification for the role of word types in formal analyses of natural language. We will now show that estimation schemes based upon type and token frequencies are special cases of the Pitman-Yor language model, corresponding to the extreme values of the parameter a. Values of a between these extremes identify estimation methods that interpolate between types and tokens.

Recall the joint distribution over words defined by the TwoStage(PYCRP$(a,b),P_\varphi$) model (from Equation 6):

$$P(\mathbf{w}\,|\,\varphi) \;=\; \sum_{\mathbf{z},\boldsymbol{\ell}} \frac{\Gamma(1+b)}{\Gamma(n+b)} \left(\prod_{k=1}^{K(\mathbf{z})-1} (ka+b) \right) \left(\prod_{k=1}^{K(\mathbf{z})} P_\varphi(\ell_k) \frac{\Gamma(n_k^{(\mathbf{z})}-a)}{\Gamma(1-a)} \right)$$

where the sum ranges over those $\mathbf{z}$ and $\boldsymbol{\ell}$ that generate $\mathbf{w}$. When $b=0$, this equation reduces to

$$\begin{aligned} P(\mathbf{w}\,|\,\varphi) &= \sum_{\mathbf{z},\boldsymbol{\ell}} \frac{\Gamma(1)}{\Gamma(n)} \cdot a^{K(\mathbf{z})-1}(K(\mathbf{z})-1)! \cdot \prod_{k=1}^{K(\mathbf{z})} P_\varphi(\ell_k) \frac{\Gamma(n_k^{(\mathbf{z})}-a)}{\Gamma(1-a)} \\ &= \sum_{\mathbf{z},\boldsymbol{\ell}} \frac{(K(\mathbf{z})-1)!}{(n-1)!} \cdot a^{K(\mathbf{z})-1} \cdot \prod_{k=1}^{K(\mathbf{z})} P_\varphi(\ell_k) \frac{\Gamma(n_k^{(\mathbf{z})}-a)}{\Gamma(1-a)}. \end{aligned} \tag{11}$$

The distribution $P(\mathbf{w}\,|\,\varphi)$ determines how the data $\mathbf{w}$ influence estimates of φ, so we will consider how $P(\mathbf{w}\,|\,\varphi)$ changes under different limits of a.

When $a \to 0$, the $a^{K(\mathbf{z})-1}$ term in Equation 11 causes the sum over $(\mathbf{z},\boldsymbol{\ell})$ to be dominated by the partition of customers with the smallest value of $K(\mathbf{z})$, that is, the fewest number of tables. Since seating arrangements are restricted so that $\ell_{z_i} = w_i$, the dominant arrangement contains exactly one table, and one occurrence of $P_\varphi(w)$, per word type w. Therefore estimates of φ will be based on word types.

When $a \to 1$, $a^{K(\mathbf{z})-1} \to 1$. If $n_k = 1$ then $\frac{\Gamma(n_k^{(\mathbf{z})}-a)}{\Gamma(1-a)} = 1$, but otherwise this term approaches 0. Therefore all terms in the sum approach 0 except for those where there is only a single token assigned to each table. In this case, $K(\mathbf{z}) = n$ and $\ell_k = w_k$, which means that P_φ is responsible for generating all the word tokens in the data. Estimates of φ will consequently be based on word tokens.

The extreme values of the a parameter in the PYCRP thus correspond to type-based inference ($a = 0$) or token-based inference ($a = 1$), while choosing other values of a between 0 and 1 provides a systematic way of smoothly interpolating between the type-based and token-based extremes.

6.3 Pitman-Yor Processes and Kneser-Ney Smoothing

In addition to justifying the role of types in formal analyses of language in general, using the PYCRP as an adaptor to create a Pitman-Yor language model can provide an explanation of the assumptions behind a specific scheme for combining token and type frequencies: Kneser-Ney smoothing. In this section, we outline the relationship between Kneser-Ney smoothing and the PYCRP, showing that the predictive distribution of the Kneser-Ney smoother can be viewed as an approximation to that of the Pitman-Yor language model. This relationship was first pointed out in a conference paper presenting preliminary versions of some of the results in this paper (Goldwater et al., 2006a), and then independently identified by Teh (2006a,b), who expanded on this observation and presented the first empirical comparisons of the two methods. We return to the results of empirical comparisons briefly below.

The Kneser-Ney smoother estimates the probability that a word token will belong to a particular type by combining type and token frequencies, and has proven particularly effective for *n*-gram models (Ney et al., 1994; Kneser and Ney, 1995; Chen and Goodman, 1998). To use an *n*-gram language model, we need to estimate the probability distribution over word types given a particular *history*, that is, the $n-1$ preceding tokens. Assume we are given a multiset $\mathbf{w}$ of N tokens that all share a common history, and we want to predict the next token, w_{N+1}, that will occur with that history. For example, the history might be *in the*, with $\mathbf{w}$ = (*house book way school house* ...). (We use a multiset rather than a vector because we care only about the counts of the word types in $\mathbf{w}$, not their ordering.) Assume that we also have H other multisets $\mathbf{w}^{(1)}, \ldots, \mathbf{w}^{(H)}$, each associated with one of H other histories. The interpolated Kneser-Ney (IKN) smoother (Chen and Goodman, 1998) makes the prediction

$$P(w_{N+1} = w \,|\, \mathbf{w}) = \frac{n_w^{(\mathbf{w})} - I(n_w^{(\mathbf{w})} > D)D}{N} + \frac{\sum_{w'} I(n_{w'}^{(\mathbf{w})} > D)D}{N} \frac{\sum_h I(w \in \mathbf{w}^{(h)})}{\sum_{w'} \sum_h I(w' \in \mathbf{w}^{(h)})} \qquad (12)$$

where D is a "discount factor" specified as a parameter of the model, the sum over h includes $\mathbf{w}$, and we have suppressed the dependence on $\mathbf{w}^{(1)}, \ldots, \mathbf{w}^{(H)}$.

We can define a two-stage model that approximates the Kneser-Ney smoother by assuming that each $\mathbf{w}^{(h)}$ is produced by a two-stage restaurant with a PYCRP adaptor (i.e., a separate restaurant for each history), where all the restaurants share the same generator, parameterized by φ. We assume φ is a multinomial distribution, which we estimate using maximum-likelihood estimation. Under this model, the probability that token w_{N+1} takes on the value w given $\mathbf{w}$ and φ is

$$P(w_{N+1} = w \,|\, \mathbf{w}, \varphi) = \sum_{\mathbf{z}} P(w_{N+1} = w | \mathbf{w}, \mathbf{z}, \varphi) P(\mathbf{z} | \mathbf{w}, \varphi)$$

where $\mathbf{z}$ is the seating assignment for $\mathbf{w}$, and $P(w_{N+1} = w|\mathbf{w}, \mathbf{z}, \varphi)$ is equivalent to $P(w_{N+1} = w|\boldsymbol{\ell}(\mathbf{z}), \mathbf{z}, \varphi)$, given by Equation 5. Substituting in Equation 5 and assuming $b = 0$, this becomes

$$\begin{aligned}
&P(w_{N+1} = w \,|\, \mathbf{w}, \varphi) \\
&= \sum_{\mathbf{z}} \frac{n_w^{\mathbf{w}} - K_w(\mathbf{z})a + K(\mathbf{z})aP_\varphi(w)}{N} P(\mathbf{z}|\mathbf{w}, \varphi) \\
&= \sum_{\mathbf{z}} \frac{n_w^{\mathbf{w}} P(\mathbf{z}|\mathbf{w}, \varphi)}{N} - \sum_{\mathbf{z}} \frac{K_w(\mathbf{z})aP(\mathbf{z}|\mathbf{w}, \varphi)}{N} + \sum_{\mathbf{z}} \frac{K(\mathbf{z})aP_\varphi(w)P(\mathbf{z}|\mathbf{w}, \varphi)}{N} \\
&= \frac{n_w^{\mathbf{w}}}{N} - \sum_{\mathbf{z}} \frac{K_w(\mathbf{z})aP(\mathbf{z}|\mathbf{w}, \varphi)}{N} + \sum_{\mathbf{z}} \frac{\sum_{w'} K_{w'}(\mathbf{z})aP_\varphi(w)P(\mathbf{z}|\mathbf{w}, \varphi)}{N} \\
&= \frac{n_w^{\mathbf{w}} - E_{\mathbf{z}}[K_w(\mathbf{z})]\, a}{N} + \frac{\sum_{w'} E_{\mathbf{z}}[K_{w'}(\mathbf{z})]\, a}{N} P_\varphi(w) \qquad (13)
\end{aligned}$$

where $E_{\mathbf{z}}[K_w(\mathbf{z})] = \sum_{\mathbf{z}} K_w(\mathbf{z})P(\mathbf{z}|\mathbf{w}, \varphi)$, and $K_w(\mathbf{z})$ is the number of tables with label w under the seating assignment $\mathbf{z}$. The other histories enter into this expression via φ. Since all the $\mathbf{w}^{(h)}$ are assumed to be produced from a single set of parameters φ, the maximum-likelihood estimate of $P_\varphi(w)$ will approach

$$P_\varphi(w) = \frac{\sum_h I(w \in \mathbf{w}^{(h)})}{\sum_{w'} \sum_h I(w' \in \mathbf{w}^{(h)})}$$

as a approaches 0, since only a single instance of each word type in each context will contribute to the estimate of φ. Substituting this value of $P_\varphi(w)$ into Equation 13 reveals the correspondence to the Kneser-Ney smoother (Equation 12). The only difference is that the constant discount factor D is replaced by $aE_{\mathbf{z}}[K_w(\mathbf{z})]$, which will increase slowly as n_w increases.

Note that the formulation given above is very general in that we do not specify a particular generator model P_φ. However, to complete the correspondence with IKN n-gram smoothing, we can assume that the generator for the model that computes the distribution over word types conditioned on a history of size n is another two-stage PYCRP model that computes probabilities conditioned on histories of size $n-1$. The recursion bottoms out with a uniform distribution over the W word types in the vocabulary, $P_0(w) = 1/W$. This *hierarchical* Pitman-Yor language model (Teh, 2006b) is analogous to the hierarchical Dirichlet process introduced by Teh (2006a). Intuitively, we can imagine a separate restaurant for each history of size n, where the counts in that restaurant correspond to the distribution of word tokens given that history. If a customer sits at a new table in one of these restaurants, the label on that table is distributed according to the counts in a "backoff" restaurant with history size $n-1$. All restaurants with the same final $n-1$ history words will share the same backoff restaurant.

As noted above, there are slight differences between the predictions of this Pitman-Yor language model and IKN smoothing due to the replacement of the constant discount factor D in IKN with an expression that increases as a function of n_w. Interestingly, *modified Kneser-Ney* (MKN) smoothing (Chen and Goodman, 1998) also replaces the single constant D in IKN with a small set of D values that increase as a function of n_w (Chen and Goodman 1998 use three values, for $n_w = 1$, 2, and 3 or more). MKN was introduced by Chen and Goodman (1998) as an alternative to IKN that was shown to work better in practice. So it has been known for a number of years that increasing D with n_w seems to provide better predictions, and initial experiments with the Pitman-Yor language model (Teh, 2006a,b) did not show improvements over MKN (although they did show improvements over

IKN). However, these experiments were performed on a relatively small corpus of text (16 million words of newswire). More recently, Huang and Renals (2010) developed a parallel approximate training algorithm for the Pitman-Yor language model and performed a more thorough set of experiments comparing IKN, MKN, and the Pitman-Yor language model within a speech recognition system. The models were trained on a large corpus of conversational speech (200 million words) and evaluated on perplexity and word error rate. The Pitman-Yor model achieved the best results on both measures, and gains over the other two models became larger as corpus size increased. So although empirical investigation was sufficient to develop a very close approximation to the Pitman-Yor language model, discovery of the true model has nevertheless led to better language models in practice.

7. Types and Tokens in Modeling Morphology

Our attempt to develop statistical models of language that generically produce power-law distributions was motivated by the possibility that models that account for this statistical regularity might be able to learn linguistic information better than those that do not. Our two-stage language modeling framework allows us to create exactly these sorts of models, with the generator producing individual lexical items, and the adaptor producing the power-law distribution over words. In this section, we show that adding a PYCRP adaptor to a simple generative model for morphology can vastly improve unsupervised learning of the morphological structure of English, and we explore the effects of varying the PYCRP parameters in this task. Morphology provides a particularly interesting case for testing our model, as it is one context in which formal linguists focus on accounting for the appearance of word types (e.g., Pierrehumbert, 2003), while computational linguists have typically developed supervised models based on the token frequencies of those words (e.g., Hakkani-Tür et al., 2002). Interestingly, previous work on *unsupervised* learning of morphology often ignores token frequencies, instead using word types as input (Goldsmith, 2001, 2006; Snover and Brent, 2003; Monson et al., 2004).[9] This fact suggests that the additional information provided by token frequencies may actually be harmful for learning morphology using standard models. Indeed, the results we report below support this hypothesis; we provide some possible explanations in the Section 8.1.2, where we discuss the results of our first set of experiments.

Previous morphology learning models have sidestepped the problems presented by token frequencies by simply ignoring them and using only a list of unique word types as input instead. It is worth reiterating here that our own two-stage model can be made to behave equivalently: with appropriate values of the PYCRP parameters (specifically, $a = b = 0$), our two-stage model assigns every token of the same word type to the same table, so that the parameters of the generator model (here, the morphology model) are inferred based on a list of unique word types. The result is equivalent to that of a model consisting only of the generator, where the input is a list of word types, as in the systems mentioned above. However, our full two-stage model is more flexible than these other systems. First, by choosing different adaptor parameters, different damping regimes can be achieved. Although these too could be simulated through different preprocessing schemes (e.g., taking logs of token frequencies rather than removing frequencies entirely), our model is more

9. Descriptions of Goldsmith's Linguistica system (Goldsmith, 2001, 2006) do not mention that frequencies are discarded before analysis. However, the version of the program we downloaded from `http://humanities.uchicago.edu/faculty/goldsmith` produced the same results when run on a full corpus as when run on a list of the unique word types in the corpus.

promising precisely because it can achieve the effects of damping while leaving the actual input frequencies unchanged. Thus, unlike previous models, ours can be used to learn directly from a corpus without preprocessing. This makes it possible to extend the model to incorporate additional information available from the corpus but not from a word list, such as contextual information. The experiments presented here are intended only to explore the effects of different parameter values, and do not take immediate advantage of this difference between our model and previous unsupervised systems. However, recent work using adaptor grammars has suggested some ways in which context can be incorporated into models based on the two-stage framework, for example by learning collocations between words at the same time as sub-word units (Johnson, 2008a; Johnson and Goldwater, 2009). Another example of using contextual information might be a hidden Markov model for part-of-speech tagging, where the standard multinomial emission distributions could be replaced with our morphology model, so that the learned part-of-speech classes would be informed both by corpus context and morphological structure. It is difficult to see how this kind of joint learning could take place in a probabilistic model requiring one instance of each word type as input.

7.1 A Lexicon Generator for Morphology

Many languages contain words built up of smaller units of meaning, or *morphemes*. These units can contain lexical information (as stems) or grammatical information (as affixes). For example, the English word *walked* can be parsed into the stem *walk* and the past-tense suffix *-ed*. Knowledge of morphological structure enables language learners to understand and produce novel wordforms, and is important for many natural language processing tasks in morphologically rich languages (Collins et al., 1999; Larkey et al., 2002; Cowan and Collins, 2005; Koehn and Hoang, 2007).

As a basic model of morphology, we assume that each word consists of a single stem and (possibly empty) suffix, and belongs to some inflectional class. Each class is associated with a stem distribution and a suffix distribution. We assume that stems and suffixes are independent given the class, so the joint probability of generating a particular class c, stem t, and suffix f is defined as

$$P(c,t,f) = P(c)P(t\,|\,c)P(f\,|\,c)$$

where the distributions on the right hand side are all assumed to be multinomial, generated from symmetric Dirichlet priors with hyperparameters κ, τ, and ϕ respectively. So far, we have been assuming that the generator in a two-stage model is a distribution over lexical items that are strings. However, in this morphology model, the generator produces analyses of strings (class, stem, suffix), rather than the strings themselves. We will therefore distinguish between the label ℓ_k on each table, which we continue to assume is a string, and the analysis of that label $A(\ell_k)$, which is an object produced by the generator. We can, if we wish, compute the probability of a label regardless of its analysis as

$$P(\ell) = \sum_{(c,t,f)} I(\ell = t.f)P(c)P(t\,|\,c)P(f\,|\,c)$$

where $t.f$ is the concatenation of t and f, and $I(.)$ is an indicator function taking on the value 1 when its argument is true, and 0 otherwise.

Our generator model for morphology is inspired by the model described by Goldsmith (2001), and is intended to encode two basic linguistic intuitions. The first is that different morphological classes contain different sets of stems and suffixes. Also, although stems and suffixes are not truly independent even within a morphological class, morphological boundaries do tend to coincide with

points of low predictability in a string of phonemes or characters (Harris, 1955). That is, there is greater independence between stems and suffixes than between other possible substrings. Another way of looking at this is that, if we know that, for example, past and present tense verbs are each relatively common, then if we see a particular verb very frequently in the past tense, we would expect to see it very frequently in the present tense as well (Yarowsky and Wicentowski, 2000).

We also note two important differences between our model and that of Goldsmith. First, Goldsmith's model is recursive (i.e., a word stem can be further split into a smaller stem plus suffix), which makes it better able to deal with complex morphology than the model presented here. However, the simplifying assumption of a single stem and suffix per word is often sufficient for English inflectional morphology. We emphasize that our primary goal here is to illustrate the effects of the generator-adaptor framework rather than to develop a state-of-the-art morphology learning system.

The second difference between Goldsmith's model and our own is that Goldsmith's model assumes that all occurrences of each word type have the same analysis. The model here allows different tokens with the same observed form to have different analyses when $a > 0$ or $b > 0$. This feature could be important for representing homonymous words with different morphological analyses.

7.2 Gibbs Sampler

Our goal in defining this morphology model is to be able to automatically infer the morphological structure of a language. Since our model is exchangeable, this can be done using Gibbs sampling, a standard Markov chain Monte Carlo method (Gilks et al., 1996). In Markov chain Monte Carlo, variables in the model are repeatedly sampled, with each sample conditioned on the current values of all other variables in the model. This process defines a Markov chain whose stationary distribution is the posterior distribution over model variables given the input data.

Rather than sampling all the variables in our two-stage model simultaneously, our Gibbs sampler alternates between sampling the variables in the generator and those in the adaptor (here, a PYCRP). Our algorithm iterates over the following two steps, as illustrated in Figure 7:

1. Fix the assignment $\mathbf{z}$ of words to tables, and sample a new morphological analysis $A(\ell_k)$ for the label on each table.

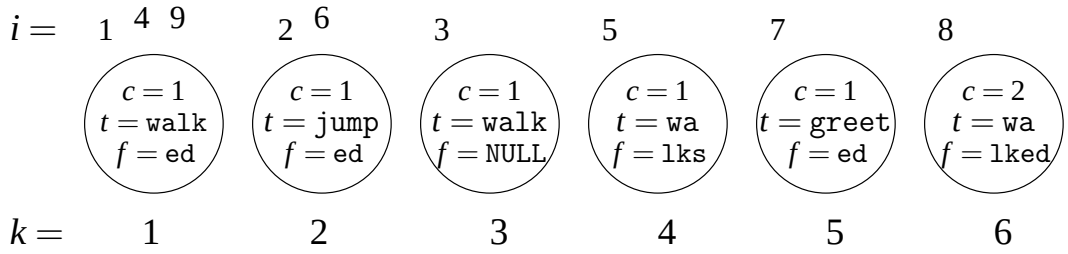
2. Fix the morphological analyses $A(\boldsymbol{\ell})$ of the labels, and sample a new table assignment z_i for each word token w_i.

In Step 1, we compute the probability distribution over analyses of the current label $A(\ell_k)$ conditioned on the analyses of all other labels $A(\boldsymbol{\ell}_{-k})$:

$$\begin{aligned} P(A(\ell_k) = (c,t,f) \,|\, A(\boldsymbol{\ell}_{-k}), \kappa, \tau, \phi) & \\ \propto \;& I(\ell_k = t.f) \cdot P(c,t,f \,|\, A(\boldsymbol{\ell}_{-k}), \kappa, \tau, \phi) \\ = \;& I(\ell_k = t.f) \cdot P(c \,|\, \mathbf{c}_{-i}, \mathbf{z}, \kappa) \cdot P(t \,|\, \mathbf{t}_{-i}, \mathbf{c}, \mathbf{z}, \tau) \cdot P(f \,|\, \mathbf{f}_{-i}, \mathbf{c}, \mathbf{z}, \phi) \\ = \;& I(\ell_k = t.f) \cdot \frac{m_c + \kappa}{m + \kappa C} \cdot \frac{m_{t,c} + \tau}{m_c + \tau T} \cdot \frac{m_{f,c} + \phi}{m_c + \phi F} \end{aligned} \tag{14}$$

where the notation $\mathbf{x}_{-i}$ is now used to indicate $(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$ (by exchangeability, we can nevertheless treat x_i as though it is the last of the n variables when computing probabilities); C, T, and F are the total possible number of classes, stems, and suffixes; and m_x is the number of tables in

(a)



(b)

$$
\begin{aligned}
A(\ell_1) = (1, \texttt{NULL}, \texttt{walked}) \,|\, A(\boldsymbol{\ell}_{-1}), \kappa, \tau, \phi) &\propto \frac{5+\kappa}{6+2\kappa} \cdot \frac{\tau}{5+21\tau} \cdot \frac{\phi}{5+25\phi} \\
A(\ell_1) = (1, \texttt{w}, \texttt{alked}) \,|\, A(\boldsymbol{\ell}_{-1}), \kappa, \tau, \phi) &\propto \frac{5+\kappa}{6+2\kappa} \cdot \frac{\tau}{5+21\tau} \cdot \frac{\phi}{5+25\phi} \\
A(\ell_1) = (1, \texttt{wa}, \texttt{lked}) \,|\, A(\boldsymbol{\ell}_{-1}), \kappa, \tau, \phi) &\propto \frac{5+\kappa}{6+2\kappa} \cdot \frac{1+\tau}{5+21\tau} \cdot \frac{\phi}{5+25\phi} \\
&\cdots \\
A(\ell_1) = (2, \texttt{wa}, \texttt{lked}) \,|\, A(\boldsymbol{\ell}_{-1}), \kappa, \tau, \phi) &\propto \frac{1+\kappa}{6+2\kappa} \cdot \frac{1+\tau}{1+21\tau} \cdot \frac{1+\phi}{1+25\phi} \\
&\cdots \\
A(\ell_1) = (2, \texttt{walked}, \texttt{NULL}) \,|\, A(\boldsymbol{\ell}_{-1}), \kappa, \tau, \phi) &\propto \frac{1+\kappa}{6+2\kappa} \cdot \frac{\tau}{1+21\tau} \cdot \frac{\phi}{1+25\phi}
\end{aligned}
$$

(c)

$$
\begin{aligned}
P(z_1 = 1 \,|\, w_i = w, \mathbf{z}_{-i}, \boldsymbol{\ell}(\mathbf{z}_{-i}), \varphi, a, b) &\propto 2 - a \\
P(z_1 = 6 \,|\, w_i = w, \mathbf{z}_{-i}, \boldsymbol{\ell}(\mathbf{z}_{-i}), \varphi, a, b) &\propto 1 - a \\
P(z_1 = 7 \,|\, w_i = w, \mathbf{z}_{-i}, \boldsymbol{\ell}(\mathbf{z}_{-i}), \varphi, a, b) &\propto (6a + b) P_\varphi(\texttt{walked})
\end{aligned}
$$

Figure 7: An example illustrating our Gibbs sampler. In this example, the corpus $\mathbf{w} =$ (walked, jumped, walk, walked, walks, jumped, greeted, walked, walked), and initially $\mathbf{z} =$ (1, 2, 3, 1, 4, 2, 5, 6, 1). (a) illustrates the current seating arrangement, with numbers above each table indicating the indices i of customers seated there and the number below each table indicating the index k of the table. The morphological analysis associated with each table is also shown. T and F for this corpus (the total number of possible stems and suffixes) are 21 and 25, and we let $C = 2$. To complete a full Gibbs iteration, we first resample the analyses, and then the table assignments. In this case, we start by removing the current analysis of walked on table 1 (and its associated counts), and computing the probability of each of the 14 possible new analyses, as shown in (b). We sample from this distribution, replace the new analysis on table 1 (incrementing the associated counts), and repeat for the remaining five tables. Then, we sample new values for $z_1 \ldots z_9$ in a similar fashion. (c) shows the computations for z_1, which is restricted to taking on the values 1, 6, or 7 (a new table) because only these tables may be labeled with walked.

$A(\boldsymbol{\ell}_{-z})$ whose label includes x. (We use m to distinguish these counts over labels from the n counts over tokens.) The last line is obtained by integrating over the multinomial parameters for the classes, stems, and suffixes as in Equation 7; for example, $P(c\,|\,\mathbf{c}_{-i},\mathbf{z},\kappa) = \int P(c\,|\,\theta_c)P(\theta_c\,|\,\mathbf{c}_{-i},\mathbf{z},\kappa)d\theta_c$ where θ_c are the parameters of the multinomial distribution over classes.

In the experiments presented here, C is fixed empirically and T and F are determined for each set of input data by computing the number of possible segmentations of the words in the data into stems and suffixes (i.e., determining all the prefix and suffix strings for those words; the empty string is considered as a possible stem as well as a possible suffix).

In Step 2 of our sampler, we compute the distribution over table assignments z_i for the ith word token using Equation 10, repeated below with the conditioning adaptor parameters included:

$$P(z_i = k\,|\,w_i = w,\mathbf{z}_{-i},\boldsymbol{\ell}(\mathbf{z}_{-i}),a,b,\varphi) \propto \begin{cases} (n_k^{(\mathbf{z}_{-i})} - a)\cdot I(\ell_k = w) & 1 \le k \le K(\mathbf{z}_{-i}) \\ (K(\mathbf{z}_{-i})a + b)\cdot P_\varphi(w) & k = K(\mathbf{z}_{-i})+1 \end{cases}$$

where $P_\varphi(w)$ is found using Equation 14 by summing over all possible analyses.

Note that in Step 2, tables may appear or disappear, which will cause the label counts to change. When a table is removed, the class, stem, and suffix counts of its label are decremented. When a new table is added, a morphological analysis is chosen at random according to Equation 14, and the appropriate counts are incremented.

8. Experiments

In this section, we use the simple morphology model defined above as an example to demonstrate that applying an appropriate adaptor can significantly improve the learning of linguistic structure. We also examine how the choice of parameters in the PYCRP affects learning behavior. We perform two experiments, one using verbs in standard written form from a corpus of newspaper text, and the other using all words from a corpus of phonemically transcribed child-directed speech. In each experiment, evaluations were performed on a single sample taken after 1000 iterations of our Gibbs sampler, with $C = 6$ classes, $\kappa = .5$ and $\tau = \phi = .001$.[10] For the PYCRP parameters, we fixed $b = 0$ and experimented with values of a between 0 and 1.[11]

8.1 Experiment 1: Verbs

We begin by describing the data and evaluation method used in this experiment, followed by the experimental results.

8.1.1 DATA AND EVALUATION

We prepared a data set consisting of English verbs in written form from the Penn Wall Street Journal treebank (Marcus et al., 1993), a corpus of hand-tagged and parsed text from the Wall Street Journal. Using the part-of-speech tags, we extracted all the verbs from sections 0-21 of the corpus, which yielded 137,997 tokens belonging to 7,761 types. This list of verbs served as the input to the

10. Although we fixed the values for the hyperparameters in our experiments, all of our models can be extended to include prior distributions over the hyperparameters. In that case the hyperparameter values can be inferred by sampling. (West, 1992).

11. Technically, setting $a = 0$ and $b = 0$ leads to undefined results, but algorithmically one can simulate $\lim_{a\to 0}$ by using exactly one table for each word type, which is what we did.

morphological segmentation system. In this data set, the total number of unique prefix strings T is 22,396, and the total number of unique suffix strings F is 21,544.

To create a gold standard for evaluation, we automatically segmented each verb in the input corpus using heuristics based on its part-of-speech tag and spelling. For example, verbs tagged as VBD (past tense) or VBN (past participle) and ending in *-ed* were assigned a morpheme boundary before the *-ed*, while most verbs tagged as VBZ (third person present singular) and ending in *-s* were assigned a boundary before the *-s*. (The VBZ forms *does* and *goes*, as well as forms ending in *-xes* or *-ches*, such as *mixes*, were assigned a boundary before *-es* instead.) Potentially irregular forms such as past participles ending in *-n* were examined by hand to ensure correct segmentation.

It is important to note that any choice of segmentation will lead to some inconsistencies due to spelling rules that insert or delete characters before certain endings. The segmentation we used prefers consistency among suffixes rather than stems when there is a conflict. That is, suffixes will be the same across words such as *jump.ed* and *stat.ed*, or *jump.s* and *state.s*, but the stems in *stat.ed* and *state.s* will be different.

Given the gold standard analysis for each word and a sample analysis from our algorithm, segmentation accuracy was computed in two different ways. First, for each word type, the most frequent suffix for that type (in the sampled hypothesis) was determined and counted once to evaluate the proportion of types with each suffix. Second, since different tokens of the same type may be assigned different analyses, the proportion of word tokens with each suffix is also displayed. This analysis gives more weight to the results of frequent words, and also takes into account any uncertainty in the model (although in fact less than 1.5% of types have multiple analyses for any value of a).

8.1.2 RESULTS

As a model for learning morphology, our generator by itself is not very effective. Only 55.4% of word types and 62.2% of word tokens are segmented correctly. For comparison, baseline accuracy for a system that always leaves words unsegmented is 30.7% for types and 57.1% for tokens. It turns out that for most words, the segmentation identified by the generator model is actually the same as the unsegmented baseline, as illustrated in Figure 8. In other words, the model simply memorizes full words rather than splitting off (non-empty) suffixes. This is particularly true of frequent words, which is why token accuracy is so similar for the baseline and the generator model.

One might expect that the sparse Dirichlet priors used in our generator, which encourage fewer total stems and suffixes overall, would push the system towards a more parsimonious solution (i.e., fewer complete memorized wordforms). We know that the priors do have some effect, because the maximum-likelihood solution for this model is the baseline described above, with each word left unsegmented. However, even with much stronger Dirichlet priors than the ones reported here, the performance of the generator model alone is underwhelming. The reason is twofold. First, our generator model assumes complete independence between stem and suffix probabilities given the class of the word. In reality, stem and suffix probabilities are not completely independent (e.g., *announce* tends to occur more often with *-ed* than does *head*). As the amount of data for a particular verb accumulates, any deviation from independence becomes more apparent, and the model resolves this by memorizing entire words rather than segmenting them. This tendency is compounded by a second factor, which is that the most frequent words in the data are almost all irregular (e.g., *rise/rose*). Since our model deals only with segmentation, irregular words must be analyzed as

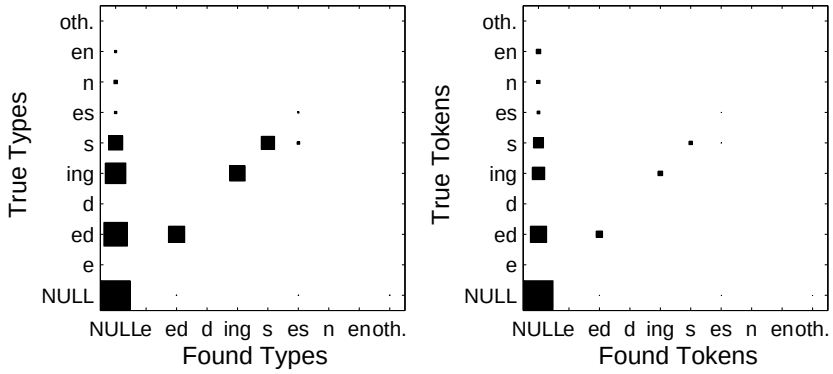


Figure 8: Confusion matrices for the morphological generator model alone (equivalent to the two-stage morphology model with $a = 1$) on the verb data set. The area of a square at location (i, j) is proportional to the number of word types (left) or tokens (right) with true suffix i and found suffix j.

having empty suffixes. This raises the overall probability of empty suffixes, making the model less likely to propose non-empty suffixes even when these are appropriate.

These issues may seem particular to our very simple model, or to the problem of morphological learning in English. However, we would argue that they are far more general. While it is true that English verbal morphology is notorious for its large number of irregular verbs, irregularity is found to varying degrees across all languages and types of linguistic structure. For example, in English, idiomatic expressions such as *X has got it made* or *X is fit to be tied*[12] can be viewed as syntactically irregular forms, in the sense that they both use the passive construction but have no corresponding active version. And, like other idioms, they also have irregular (that is, non-compositional) semantics. Importantly, the relationship between frequency and regularity observed in the current experiment (i.e., that irregular forms tend to be the most frequent) seems to be a very general property of language (Greenberg, 1966; Bybee, 1985). Together with the power-law distribution of linguistic forms, this fact implies that irregular forms will often dominate the input to statistical learning systems, which in turn may cause significant problems for an unsupervised model that does not take these facts into account.

One solution to these problems would be to simply change the input by removing repeated tokens of each type, that is, to present the system with only a list of unique word types. As discussed in the introduction to this section, many previous morphology learning systems have taken this approach. Instead, we address the problem by applying our two-stage framework, adding a PYCRP adaptor to our generator model. With this approach, we find that for a wide range of a, from 0 up to about 0.6 or 0.7, results are stable and considerably better than when using the generator model alone (or, equivalently, the 2-stage model with $a = 1$). Accuracy scores are shown in Figure 9, and confusion matrices for the model with $a = 0.6$ are shown in Figure 10. Given our discussion above, it should be no surprise that the better performance is due to the system finding more non-empty

12. These examples are due to Jackendoff (2002).

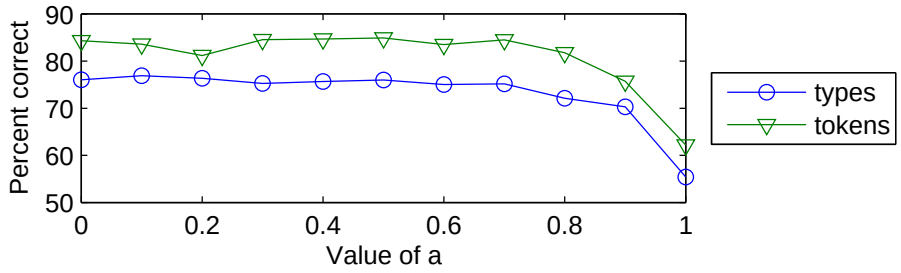


Figure 9: Percentage of verb types and tokens assigned the gold standard analysis.

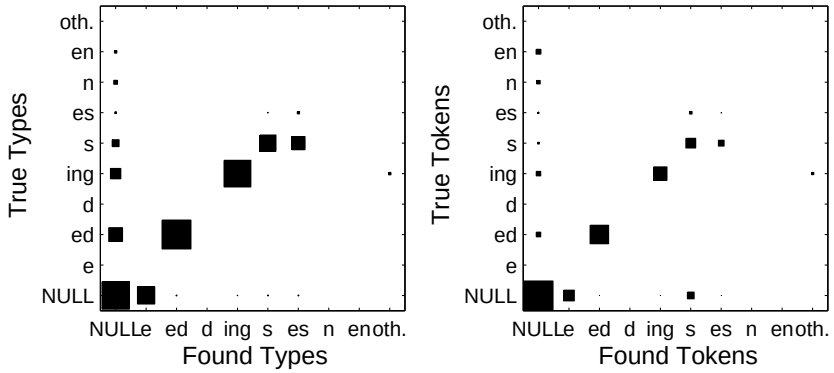


Figure 10: Confusion matrices for the 2-stage morphology model with $a = 0.6$.

suffixes overall. This is illustrated both in the confusion matrices and in Figure 11, which shows the true distribution of words with each suffix and the distribution found by the two-stage system for various values of a. Again, we see that the distribution is stable for $0 \leq a \leq 0.7$. For $a > 0.7$, empty suffixes begin to take over, causing performance to drop. Figure 12 indicates that the average number of tables per word type for $a \leq .7$ rises slowly from one to about four, whereas higher values of a cause a sharp increase in the average number of tables per type, up to almost 18. It is this increase that seems to be problematic for learning.

Finally, we provide a summary of the final sample in each of two runs of our sampler, with $a = 0.1$ and $a = 0.6$, in Table 1. An interesting feature seen in Table 1(b) is that the system has created a separate class for verbs with irregular past tense forms (second from the top). Also, in both runs, the system frequently hypothesizes analyses in which stem identity is kept constant across forms (as in *stat.e, stat.ing, stat.ed, stat.es*), whereas the gold standard maintains suffix identity (*state, stat.ing, stat.ed, state.s*). This leads the system to assume *-e* and *-es* suffixes where the gold standard has *NULL* and *-s*, and to place stems ending in *e* in separate classes from the other stems. This kind of problem is common to many morphological learning systems, and cannot be solved with a purely concatenative approach to morphology. It is also worth noting that, if the goal is to achieve a segmentation with the fewest total number of stems plus suffixes (minimizing storage cost) then the choice of segmentation taken by the system is actually better than the gold standard,

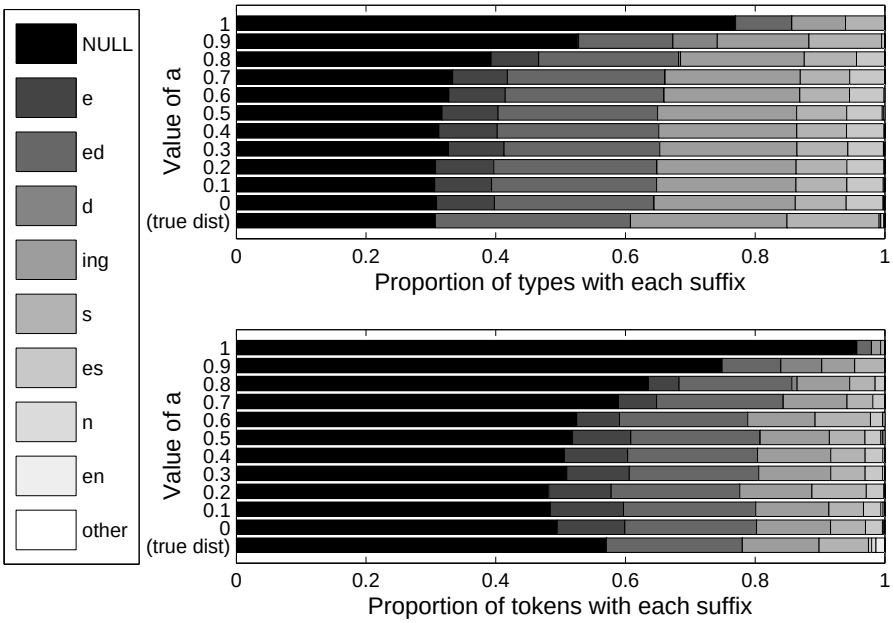


Figure 11: Results of the two-stage morphology learner for various values of *a* on the verb data set. The proportion of word types (top) and tokens (bottom) found with each suffix is shown, along with the distribution of suffixes in the gold standard.

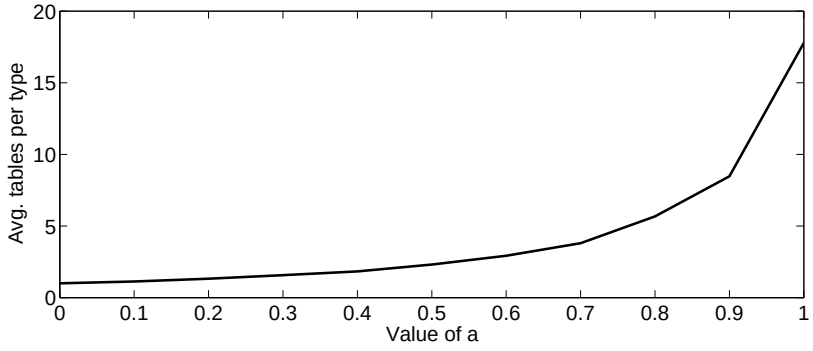


Figure 12: Average number of tables used per word type for each value of *a*.

since the total number of distinct stems plus suffixes is smaller. Only a few extra suffixes must be included to avoid near duplication of a large number of stems.

The primary remaining source of error that can be seen in the confusion matrices comes from wordforms analyzed as containing no suffix, where actually some non-empty suffix was present. In most cases, these were words where only a single inflected form was present in the data, so there was no reason for the system to postulate a complex analysis.

8.2 Experiment 2: Child-directed Speech

Experiment 1 used a corpus of verbs in orthographic form as input data, partly because learning English verbs is a standard task for computational models of morphology, and partly because this choice of corpus makes it possible to evaluate against a gold standard. However, using a single part of speech is a gross oversimplification of the learning problem. We therefore performed a second experiment using a corpus of phonemically transcribed child-directed speech, as described below.

8.2.1 DATA

The original source of the data used in this experiment was the Brown corpus (Brown, 1973) from the CHILDES database (MacWhinney and Snow, 1985), which contains transcribed parent-child interactions from long-term observational studies on three English-learning children. We extracted all the words spoken by caretakers, and converted the representations of these from standard written form to phonemic form using a phonemic dictionary.[13] Variations in pronunciation indicated in the original transcriptions (e.g., *going* vs. *goin'*) were preserved as much as possible in the phonemic forms (`go1N`, `go1n`),[14] and many non-words (e.g., *hm*) were also retained, making this corpus somewhat noisy. There are a total of 369,443 word tokens in the corpus belonging to 6,807 types. The total number of unique prefix strings T is 14,639, and the total number of unique suffix strings F is 16,313. Since there is no gold standard for this corpus, our evaluation is qualitative, based on examining the output of the algorithm.

8.2.2 RESULTS

Qualitatively, the results of varying the PYCRP parameter a are similar for this data set and the corpus of English verbs. Table 2 shows that as a increases, the number of different suffixes found decreases, and the proportion of word types analyzed with empty suffixes increases. As an indicator of the effect on other suffixes, the proportion of words found to contain the most common non-empty suffix `z` is also shown. As in the verb corpus, the highest values of a lead to analyses with almost no interesting morphological structure, while for lower values, many words are found to contain non-empty suffixes.

An interesting difference between the results from the two corpora is noticeable for the lowest values of a. In the verb corpus, results were very similar for values of $a \leq .7$. Here, there is a more graded effect, and for $a \leq .2$ the system actually produces too many different suffix types. Examining the output of the system with $a = 0$ (summarized in Table 3) illustrates the problem. Five of the classes are reasonable: three contain primarily nouns, with possible suffixes NULL and

13. We thank James Morgan and the Metcalf Infant Research Lab at Brown University for providing the phonemic dictionary for this corpus.
14. We use `typewriter` font to indicate phonemic symbols. The phonemic alphabet used in this data set is provided in Appendix B.

(a) $a = 0.1$

Tables	Stems		Suffixes	
1473	advis	9	ed	499
	rang	8	ing	371
	eliminat	8	e	255
	pass	8	NULL	177
	settl	8	es	171
	compar	8		
	...			
1936	remov	13	ed	615
	assum	10	e	539
	enabl	9	ing	480
	produc	9	es	296
	continu	9	en	6
	prov	8		
	...			
1333	represent	9	NULL	612
	back	9	ed	305
	contend	8	ing	250
	list	8	s	166
	maintain	8		
	walk	8		
	...			
1255	see	13	NULL	650
	adjust	12	ed	228
	yield	10	ing	217
	want	9	s	148
	limit	8	n	12
	fill	8		
	...			
1319	total	13	NULL	674
	work	10	ed	255
	respond	9	ing	244
	add	9	s	146
	equal	8		
	shift	8		
	...			
1531	open	11	NULL	715
	ask	9	ed	337
	fund	8	ing	285
	turn	8	s	194
	reflect	8		
	demand	8		
	...			

(b) $a = 0.6$

Tables	Stems		Suffixes	
2684	reach	44	NULL	1240
	discuss	42	ed	859
	push	42	ing	466
	match	38	es	70
	learn	37	s	49
	talk	35		
	...			
4127	say	138	NULL	3697
	think	96	s	267
	see	91	ing	132
	know	70	ting	15
	keep	63	n	13
	find	60	th	3
	...			
3672	includ	113	ed	1485
	increas	111	e	1003
	requir	73	ing	849
	involv	68	es	335
	reduc	66		
	indicat	64		
	...			
4351	us	182	ed	1712
	continu	110	e	1293
	mov	81	ing	933
	provid	68	es	413
	fac	67		
	receiv	63		
	...			
4268	offer	97	NULL	1851
	add	78	ed	1084
	report	73	ing	872
	boost	66	s	461
	start	56		
	follow	56		
	...			
3902	reflect	76	NULL	1601
	help	68	ed	1204
	develop	64	ing	721
	show	61	s	375
	consider	55	-sorting	1
	allow	52		
	...			

Table 1: Sample solutions for the WSJ verb corpus with (a) $a = .1$ and (b) $a = .6$, with boundaries initialized at random. The number of tables assigned to each class is shown in column 1, followed by the most frequent stems and suffixes in that class, and their table counts.

a	Suffix types	% NULL	% -z
0	78	58.0	10.2
.1	76	64.1	9.6
.2	40	73.8	8.8
.3	17	80.8	7.7
.4	17	84.9	6.6
.5	13	88.0	5.4
.6	12	90.5	4.8
.7	13	94.3	2.9
.8	10	99.6	2.2
.9	12	98.7	0.8
1	11	99.8	0.2

Table 2: Effects of varying the parameter a on the results from the Bernstein-Ratner-Morgan corpus. Columns show the total number of suffix types found, percentage of word types with empty suffixes, and percentage of word types with the suffix -z.

-z, and two contain large numbers of verbs with a variety of inflectional and derivational suffixes (including allomorphic and phonetic variants). The final class, however, contains a set of words that are phonologically rather than morphosyntactically similar. In particular, the words dominating this class are very short (mostly monosyllabic) and consist of common sequences of phonemes. Among these words, the hypothesized "stems" consist of the initial consonant(s) and vowel of a syllable, and the "suffixes" are the final consonant(s), or occasionally a second syllable. Rather than morphological structure, the system has discovered phonological structure.

Interestingly, as the value of a is increased, the system's tendency to split words into half-syllables decreases faster than its tendency to split words at morpheme boundaries. Moving from $a = 0$ to $a = .3$ reduces the number of hypothesized suffix types from 78 to 17 (those found in the noun and verb classes in Table 3, plus -n, -6n, -l, -&d, and -1nz) and reduces the percentage of words with non-empty suffixes by 54%, but only reduces the percentage of words with the -z suffix by 25%. All six classes in this condition correspond roughly to either nouns or verbs. We hypothesize that adding just a small amount of frequency information (with $a = .3$, the sampled solution contained 12,463 tables, versus 6,807 with $a = 0$) is enough for the system to realize that half-syllables do not have the same kind of near-independence between "stem" and "suffix" that true stem-suffix words do. Unfortunately, since there is no gold standard for this corpus, we don't know the true percentage of morphologically complex types, or types with the -z suffix. In future work, it would be useful to perform a more detailed analysis of a representative sample of the corpus to get a better sense of the accuracy of the system and the kinds of errors it makes.

8.3 Discussion

Our two experiments demonstrate how the PYCRP adaptor can be used within our two-stage framework to interpolate between type and token frequencies in a model for learning non-trivial linguistic structure. Our results suggest that, for induction of regular morphology, statistics derived from the

Tables	Stems		Suffixes		Tables	Stems		Suffixes	
915	gArti	2	NULL	777	1212	jAmp	6	NULL	736
	barbar6	2	z	138		fOl	6	z	153
	kIC1n	2				spIl	6	1N	83
	kro	2				slip	6	s	64
	k&m6l	2				kUk	6	d	49
	TIN	2				yEl	5	1n	38
	Cer	2				f9t	5	i	32
	skQt	2				r9d	5	6r	25
	pIkC6r	2				sp&Nk	5	t	16
	nobadi	2				pIk	5	6l	16
	bAt6rfl9	2				tep	5		
	b&nded	2				tArn	5		
	...					...			
867	EvribAdi	2	NULL	761	1437	ple	9	NULL	687
	notbUk	2	z	106		muv	8	1N	170
	lEp6rd	2				kQnt	7	1n	98
	fAn6l	2				slIp	7	z	97
	pl&n	2				klin	7	6r	79
	wUd	2				tiC	6	d	65
	brAD6r	2				wOk	6	s	59
	r&mbl6r	2				mark	6	t	57
	duti	2				rol	6	i	53
	kartun	2				dr9v	6	6z	45
	f9rm6n	2				rAb	6	6rz	27
	dorbEl	2				k&ri	6		
	...					...			
862	kUS6n	2	NULL	735	1514	NULL	22	NULL	255
	p6tuny6	2	z	127		p&	19	t	89
	meri6n	2				&	19	n	84
	DEm	2				bi	18	z	73
	pEns1l	2				hI	16	d	72
	pep6r	2				e	16	l	65
	bAlb	2				pE	15	r	52
	fom	2				ste	15	k	44
	stAf1n	2				t9	15	p	41
	b9s1k6l	2				dI	15	s	40
	hEv6n	2				w9	14	ni	38
	tEl6fon	2				bE	14	nz	36
	...					...		...	

Table 3: Sample solution for the Brown-Morgan corpus with $a = 0$. For each class, the number of tables assigned to that class is shown in column 1, followed by the most frequent stems and suffixes in that class, with their table counts. Note that since $a = 0$, table counts in this case are equal to type counts.

lexicon are more useful than statistics derived from corpus frequencies. This result agrees with the previous computational work of Albright and Hayes (2003), and supports the conclusions of Bybee (2001). It also justifies the use of word lists in many previous morphological learning systems (Plaut and Gonnerman, 2000; Regier et al., 2001; Snover and Brent, 2003). Interestingly, our experiments also suggest that partially damping corpus frequencies may be as effective, or perhaps even more effective, than fully damping frequencies (i.e., using only lexical statistics).

Of course, the experiments described here are limited in scope. The evidence against token-based learning of morphology would be stronger if additional experiments were performed with a larger variety of data from multiple languages, and if more detailed analysis were undertaken on the output from the Brown-Morgan corpus of child-directed speech. It would also be desirable to extend our model to account for more complex morphology, since the limitation to a single stem and suffix is inadequate to account for the morphology of most languages (including English, if derivational as well as inflectional morphology is considered). However, we emphasize that our focus here was not to develop a state-of-the-art morphological induction system, but rather to explore the consequences of using the PYCRP adaptor and its different parameter settings. We found that, with appropriate parameter settings, our model was sufficient to identify common suffixes in both corpora, and distinguish roughly between noun stems and verb stems in the Brown-Morgan corpus.

We have proposed that there are two main reasons that using the PYCRP adaptor to damp corpus frequencies yields better morphological segmentations than learning directly from corpus frequencies. First, the generator model assumes that stems and suffixes are independent given the morphological class, but this assumption is only approximately correct. Damping corpus frequencies brings the assumptions of the model and the data more in line, whereas using full corpus frequencies provides more evidence that stems and suffixes are not truly independent and therefore should not be split. Second, the most frequent words in any language tend to be irregular, and due to the power-law distribution of word frequencies, these words strongly dominate the corpus statistics. The effect of these suffix-less words is so strong that, despite a prior preference for solutions with fewer stems and suffixes, the system learns that most words should have no suffix. This causes many regular forms to go unsegmented.

Finally, we note that there are other important connections between our two-stage model and psycholinguistic theories of morphological processing. One question of concern to many psycholinguists is the extent to which morphologically complex words are stored and processed as single lexical units, as opposed to being decomposed into individual morphemes (Alegre and Gordon, 1999; Hay, 2001; Hay and Baayen, 2005). Our model provides an answer to this question, predicting specific testable relationships between word frequency, statistical independence of stem and suffix, and the probability of decomposition. While a thorough examination of these predictions and a comparison to behavioral data is beyond the scope of this paper, we note that an extension of our model (described further in the following section) has produced promising preliminary results in this area (O'Donnell, in preparation).

9. Further Applications and Extensions

The morphological segmentation model considered in the preceding sections illustrates how different assumptions about word frequency can result in different conclusions about the latent structure expressed in linguistic data. However, the potential of the two-stage approach to modeling language lies in its generality, with any existing probabilistic model of language potentially acting as a gen-

erator that can be combined with different adaptors. In this section, we consider how the two-stage framework can be applied to some other popular probabilistic models, how it can be extended to work with other kinds of linguistic structure, and how the challenges of scaling to larger corpora that arise with these applications and extensions can be addressed.

9.1 Applying the Two-stage Framework to Other Models

While the tension between types and tokens has been most explicit in computational linguistics, similar issues arise in other areas of research involving the analysis of text. For example, information retrieval systems typically represent documents in one of two ways: as a binary vector indicating which words appear in the document, or as a vector of word frequency counts (Baeza-Yates and Ribeiro-Neto, 1999). These two kinds of representations have different strengths and weaknesses, with the basic issue being that multiple occurrences of a word in a document do carry some information about the relevance of that document to a query, but not in a way that increases linearly with the number of instances. As a consequence, information retrieval systems typically make use of some kind of scheme for damping word frequencies.

Our two-stage framework provides a way to define an adaptive damping scheme for information retrieval models that have a probabilistic interpretation, such as the naïve Bayes classifier. In the standard naïve Bayes classifier, each class is assumed to be associated with a multinomial distribution over words, and the words that appear in each document are assumed to be drawn independently from that distribution. This model can be used as the generator for a two-stage model, with an adaptor such as the PYCRP being used to guarantee that the resulting word frequency distribution has statistical properties closer to natural language. This is essentially the model used in our analysis in Section 6.1, where we show that multinomial generators estimated using this model are similar to those that damp word frequencies. Evidence that this approach should lead to good empirical results comes from the work of Elkan (2006), who used a Dirichlet compound multinomial model (which is a special case of our framework, as noted above) to improve performance on several information retrieval tasks.

More complex machine learning models that have been applied to text also face a choice between representing documents in terms of types or tokens. For example, latent Dirichlet allocation (Blei et al., 2003) treats each document as a "bag of words", represented by a vector of word frequencies, as does its nonparametric analogue based on the hierarchical Dirichlet process (Teh et al., 2005). In contrast, a recent hierarchical nonparametric Bayesian model based on the beta process treats documents as binary vectors of word types (Thibaux and Jordan, 2007). It is straightforward to define a two-stage model in which LDA is used as a generator, which would provide a way to automatically interpolate between these two extremes. Probabilistic inference in this model is comparable in computational complexity to the Gibbs sampling scheme commonly used with LDA (Griffiths and Steyvers, 2004): to return to the restaurant metaphor used above, while a new random variable is introduced for each word indicating the table from which it is drawn, the number of random variables that need to be sampled in the LDA model scales with the total number of tables rather than the total number of words.

9.2 Extending the Framework to Other Linguistic Structures

We argued briefly above that the tendency of irregular forms to dominate corpus statistics is not specific to the problem addressed here, but can be expected to occur in many linguistic learning

tasks. Similarly, nearly all probabilistic models used for language learning (most notably, hidden Markov models and PCFGs) encode strong independence assumptions similar to those in our morphology generator model. Thus, we extrapolate from the results of our experiments to suggest that using the PYCRP or other power-law adaptors in combination with more standard models as generators may be able to improve unsupervised learning in many areas of language. Indeed, in other recent work we have developed several two-stage models for learning linguistic structure, achieving results comparable to, and in some cases better than, the best existing systems. For example, adaptor grammars (Johnson et al., 2007) combine a PYCRP adaptor with a PCFG generator to create a model for learning linguistic tree structures without the strong independence assumptions made by a standard PCFG. The adaptor effectively caches entire subtrees so that frequent structures can be reused, and will be assigned probabilities that are higher than the product of the PCFG rules that would be needed to create them anew. Although PCFGs are typically associated with syntactic constituency structure, they can also be used to express other types of linguistic relationships, and adaptor grammars have been used to learn word segmentation, syllable structure, morphology, dependency parses, and named-entity clusters (Johnson et al., 2007; Johnson, 2008a,b; Johnson and Goldwater, 2009; Cohen et al., 2010; Elsner et al., 2009). In fact, it is even possible to express the standard LDA model using the adaptor grammar framework (Johnson, 2010).

In addition to adaptor grammars, the two-stage framework provides the basis of another recent model for learning trees, independently introduced by Cohn et al. (2009), Post and Gildea (2009), and O'Donnell et al. (2009).[15] This model can be viewed as a generalization of the adaptor grammar. In an adaptor grammar, all trees produced by the generator are complete, with terminal symbols at all leaf nodes. In contrast, the model presented by the authors above allows the generator to produce incomplete tree fragments or *elementary trees*, with either terminal or non-terminal symbols as leaves. It therefore instantiates a nonparametric Bayesian model of tree-substitution grammar (Joshi, 2003). So far, the model has been used in NLP research to induce tree-substitution grammars from parsed sentences (Cohn et al., 2009; Post and Gildea, 2009) and to induce dependency structure from strings (Cohn et al., 2010). It has also shown promise as a model of human language processing, with applications to children's acquisition of syntax (O'Donnell et al., 2009) and adult morphological processing (O'Donnell, in preparation).

9.3 Strategies for Scaling to Larger Corpora

Using the two-stage framework with adaptors based on the CRP introduces a potentially challenging problem of probabilistic inference. In these models, each word is associated with a random variable indicating its source (or the table from which it was generated, under the restaurant analogy). The number of random variables in the model thus grows linearly with the number of words. While this is not unusual for probabilistic models of language that involve latent variables (for example, LDA has the same property), it means that alternatives to the simple Gibbs sampling algorithm we used in our morphological segmentation example will need to be developed in order to apply these models to large corpora of the kind used in modern machine learning and computational linguistics. There are three strategies for dealing with this scaling problem: using the two-stage framework to justify heuristic approximations but not explicitly performing inference, exploring parallelization schemes,

15. There are actually very slight differences in formulation between the model introduced by O'Donnell et al. (2009) and the other two, but they are conceptually similar.

and applying approximate inference techniques such as variational inference. We consider these options in turn.

Part of the motivation for our detailed treatment of the relationship between our two-stage framework and existing smoothing methods was to point out that these highly successful methods can be viewed as heuristic approximations to a model that makes reasonable assumptions about the structure of natural language. Kneser-Ney smoothing approximates a simple application of our two-stage framework, suggesting that it might be possible to derive similar heuristic approximations for more complex models. Some very simple approximations are the *minimal* and *maximal* schemes discussed by Cowans (2006) and Wallach (2008) in relation to other Bayesian language models. These make the respective assumptions that only one token of each type is drawn from the base distribution, or that all tokens of each type are drawn from the base distribution. However, the prospect of developing better approximations to more complex models seems promising, especially given recent results on the approximate and asymptotic properties of discrete models based on the Pitman-Yor process (e.g., Teh, 2006a; Buntine and Hutter, 2010). One strategy for applying two-stage models to large corpora may thus be to avoid performing inference explicitly, and instead derive approximations based on these results.

A second strategy is parallelization. As noted above, the property that makes probabilistic inference potentially problematic in two-stage models—the number of latent variables increasing linearly with the number of words in a corpus—is shared with other probabilistic models such as LDA. Parallelization has proven to be an effective strategy for applying models such as LDA to very large corpora (e.g., Newman et al., 2009). Recent work has already examined how parallelization can be used to increase the scale of the corpora on which language models based on the Pitman-Yor process can be applied, making it possible to use these models on a corpus containing 200 million words (Huang and Renals, 2010).

Finally, variational inference presents a third avenue for developing two-stage models that can be applied to large corpora, trading the stochastic approximation produced by Gibbs sampling for a deterministic approximation to the posterior distribution over the latent variables in the model. Recent work has focused on applying this strategy with adaptor grammars, which can be used to express many two-stage models as noted above. This work suggests that variational inference may yield a different pattern of scaling in the computational cost of using these models, making it more plausible that they can be applied to large corpora (Cohen et al., 2010).

10. Conclusion

In this paper we have introduced a framework for developing statistical models of language that breaks those models into two stages: one stage in which a basic set of lexical items is generated, and one stage in which the frequencies of those items are adapted to match the statistical structure of natural language. This two-stage framework solves two basic problems for statistical models of language: defining models that can generically exhibit power-law frequency distributions, and understanding how the observed frequencies of words should be damped when estimating parameters. Surprisingly, our work shows that these two problems are directly related, with damping of frequencies falling naturally out of our framework when we take into account the possibility that a secondary "rich-get-richer" process might be responsible for the power-law distribution in word frequencies.

More generally, the framework we have introduced in this paper illustrates how ideas from nonparametric Bayesian statistics can be valuable in the context of computational linguistics. The key innovation in nonparametric Bayesian statistics is the idea of defining models with potentially infinite complexity, allowing the structures recovered by those models to grow as more data are observed. In many ways, computational linguistics is the ideal application of this idea, since larger corpora always bring with them new vocabulary items, new constituents, and new constructions to be incorporated into a model. Recent work provides many other examples suggesting that nonparametric Bayesian statistics and natural language may be well suited to one another (Beal et al., 2002; Liang et al., 2007; Goldwater et al., 2006a,b; Teh et al., 2005; Teh, 2006a,b; Cohn et al., 2010) and we anticipate that this relationship will continue to be fruitful.

Acknowledgments

This paper expands on work that was presented at the Neural Information Processing Systems conference (Goldwater et al., 2006a). TLG was supported by National Science Foundation grant number BCS-0631518 and the DARPA CALO project. MJ was supported by NSF awards 0544127 and 0631667.

Appendix A. Details of Table Count Approximation Experiments

The generator used in this model was assumed to be a multinomial distribution over 30,114 word types, with φ being the probabilities assigned to these types. Estimation of φ was performed using Markov chain Monte Carlo. Taking a symmetric Dirichlet(β) prior over φ, the posterior distribution over φ given $\mathbf{w}$ and a particular value of $\mathbf{z}$ and $\boldsymbol{\ell}$ is Dirichlet with hyperparameters $m_w + \beta$, where m_w is the number of lexical items corresponding to the word type w (ie. the number of tables on which w appears). The mean probability of w under this distribution is proportional to $m_w + \beta$. Consequently, we can compute the posterior mean of φ by drawing samples of $\mathbf{z}$ and $\boldsymbol{\ell}$ from $P(\mathbf{z}, \boldsymbol{\ell} | \mathbf{w})$, computing the mean probability of each word type w given each of these samples, and then averaging the results across samples.

To draw samples from $P(\mathbf{z}, \boldsymbol{\ell} | \mathbf{w})$ we used a Gibbs sampling procedure very similar to that used with the morphology model in the main text. Since the lexical items had no internal analyses, it was only necessary to sample the table assignment z_i for each word token in the corpus in each sweep of sampling. This was done by drawing a value from the distribution

$$P(z_i = z | \mathbf{z}_{-i}, \mathbf{w}, \boldsymbol{\ell}(\mathbf{z}_{-i})) \propto \begin{cases} I(\ell_z = w_i)(n_z^{(\mathbf{z}_{-i})} - a) & 1 \le z \le K(\mathbf{z}_{-i}) \\ P(\ell_z = w_i)(K(\mathbf{z}_{-i})a + b) & z = K(\mathbf{z}_{-i}) + 1 \end{cases}$$

where $\mathbf{z}_{-i}$ is all $\mathbf{z}$ but z_i, $n_z^{(\mathbf{z}_{-i})}$ is the number of times z occurs in $\mathbf{z}_{-i}$, $K(\mathbf{z}_{-i})$ is the number of unique values in $\mathbf{z}_{-i}$, and a and b are the parameters of the PYCRP adaptor (the CRP adaptor was simulated by taking $a = 0$, in which case b plays the same role as α). $P(\ell_z = w_i)$ was obtained by integrating over the posterior distribution on φ given $\mathbf{z}_{-i}$ and $\boldsymbol{\ell}(\mathbf{z}_{-i})$, namely $(m_{w_i} + \beta) / \sum_w (m_w + \beta)$.

A total of 1000 sweeps of sampling were conducted for each adaptor, and the posterior mean of φ was computed for each sweep, which involved finding the mean number of lexical entries for each word type w. These values where then averaged over the last 500 iterations, discarding the initial sweeps to allow convergence of the Markov chain. The results shown in Figure 6 are thus

the posterior mean of the number of lexical entries assigned to each word type given the corpus **w**, and provide an indication of how word frequency translates into the frequencies from which the generator is estimated in this model.

Appendix B. Phonemic Symbols

The following ASCII characters are used in the phonemic transcriptions in the Brown-Morgan corpus, which was used as input to the morphological learner in Section 8.2.

Consonants

ASCII	Example	ASCII	Example
D	THe	k	Cut
N	siNG	l	Lamp
S	SHip	m	Man
T	THin	n	Net
Z	aZure	p	Pipe
C	CHip	r	Run
b	Boy	s	Sit
d	Dog	t	Toy
f	Fox	v	View
g	Go	w	We
h	Hat	y	You
j	Jump	z	Zip

Vowels

ASCII	Example	ASCII	Example
&	thAt	e	bAY
1	hopelEss	i	bEE
6	About	o	bOAt
7	bOY	u	bOOt
9	flY		
A	bUt		
E	bEt		
I	bIt		
O	lAW		
Q	bOUt		
U	pUt		
a	hOt		

References

Adam Albright and Bruce Hayes. Rules vs. analogy in English past tenses: a computational/experimental study. *Cognition*, 90:118–161, 2003.

David Aldous. Exchangeability and related topics. In *École d'été de probabilités de Saint-Flour, XIII—1983*, pages 1–198. Springer, Berlin, 1985.

Maria Alegre and Peter Gordon. Frequency effects and the representational status of regular inflections. *Journal of Memory and Language*, 40(1):41–61, 1999.

Charles Antoniak. Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems. *The Annals of Statistics*, 2:1152–1174, 1974.

Richard Arratia, A. D. Barbour, and Simon Tavaré. Poisson process approximations for the Ewens sampling formula. *Annals of Applied Probability*, 2:519–535, 1992.

Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval*. ACM press, New York, 1999.

Matthew Beal, Zoubin Ghahramani, and Carl Rasmussen. The infinite hidden Markov model. In *Advances in Neural Information Processing Systems 14*, Cambridge, MA, 2002. MIT Press.

David Blackwell and James B. MacQueen. Ferguson distributions via Pólya urn schemes. *Annals of Statistics*, 1:353–355, 1973.

David Blei, Andrew Ng, and Michael Jordan. Latent Dirichlet allocation. In T. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, Cambridge, MA, 2002. MIT Press.

David Blei, Andrew Ng, and Michael Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

David Blei, Thomas L. Griffiths, Michael Jordan, and Joshua B. Tenenbaum. Hierarchical topic models and the nested Chinese restaurant process. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing 16*, Cambridge, MA, 2004. MIT Press.

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

Roger Brown. *A First Language: The Early Stages*. Harvard University Press, Cambridge, MA, 1973.

Wray L. Buntine and Marcus Hutter. A Bayesian review of the Poisson-Dirichlet process. http://arxiv.org/abs/1007.0296, Jul 2010.

Joan Bybee. *Morphology: a Study of Relation between Meaning and Form*. Benjamins, Amsterdam, 1985.

Joan Bybee. *Phonology and Language Use*. Cambridge University press, Cambridge, UK, 2001.

Stanley F. Chen and Joshua Goodman. An empirical study of smoothing techniques for language modeling. Technical Report TR-10-98, Center for Research in Computing Technology, Harvard University, 1998.

Shay B. Cohen, David M. Blei, and Noah A. Smith. Variational inference for adaptor grammars. In *Proceedings of Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 564–572, Los Angeles, California, 2010.

Trevor Cohn, Sharon Goldwater, and Phil Blunsom. Inducing compact but accurate tree-substitution grammars. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 548–556, Boulder, Colorado, 2009.

Trevor Cohn, Sharon Goldwater, and Phil Blunsom. Inducing tree substitution grammars. *Journal of Machine Learning Research*, 11:3053–3096, Nov. 2010.

Michael Collins, Lance Ramshaw, Jan Hajič, and Christoph Tillmann. A statistical parser for Czech. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 505–512, College Park, Maryland, USA, 1999.

Brooke Cowan and Michael Collins. Morphology and reranking for the statistical parsing of Spanish. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 795–802, Vancouver, 2005.

Philip Cowans. *Probabilistic Document Modelling*. PhD thesis, Cambridge University, 2006.

Mathias Creutz and Krista Lagus. Induction of a simple morphology for highly-inflecting languages. In *Proceedings of the Seventh Meeting of the ACL Special Interest Group in Computational Phonology (SIGPHON)*, pages 43–51, Barcelona, Spain, 2004.

Mathias Creutz and Krista Lagus. Inducing the morphological lexicon of a natural language from unannotated text. In *Proceedings of the International and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning*, pages 51–59, Espoo, Finland, 2005.

Charles Elkan. Clustering documents with an exponential-family approximation of the Dirichlet compound multinomial distribution. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 289–296, Pittsburgh, Pennsylvania, 2006.

Micha Elsner, Eugene Charniak, and Mark Johnson. Structured generative models for unsupervised named-entity clustering. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 164–172, Boulder, Colorado, 2009.

Michael D. Escobar and Mike West. Bayesian density estimation and inference using mixtures. *Journal of the American Statistical Association*, 90(430):577–588, 1995.

Thomas S. Ferguson. A Bayesian analysis of some nonparametric problems. *Annals of Statistics*, 1:209–230, 1973.

Jenny Finkel, Trond Grenager, and Christopher D. Manning. The infinite tree. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 272–279, Prague, Czech Republic, 2007.

Walter R. Gilks, Sylvia Richardson, and David J. Spiegelhalter, editors. *Markov Chain Monte Carlo in Practice*. Chapman and Hall, Suffolk, 1996.

John Goldsmith. Unsupervised learning of the morphology of a natural language. *Computational Linguistics*, 27:153–198, 2001.

John Goldsmith. An algorithm for the unsupervised learning of morphology. *Journal of Natural Language Engineering*, 12:353–371, 2006.

Sharon Goldwater, Thomas L. Griffiths, and Mark Johnson. Interpolating between types and tokens by estimating power-law generators. In *Advances in Neural Information Processing Systems 18*, Cambridge, MA, 2006a. MIT Press.

Sharon Goldwater, Thomas L. Griffiths, and Mark Johnson. Contextual dependencies in unsupervised word segmentation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, Syndney, Australia, 2006b.

Sharon Goldwater, Thomas L. Griffiths, and Mark Johnson. A Bayesian framework for word segmentation: Exploring the effects of context. *Cognition*, 112(1):21–54, 2009.

Joseph Greenberg. Language universals. In T. A. Sebok, editor, *Current Trends in Linguistics III*. Mouton, The Hague, 1966.

Thomas L. Griffiths. Power-law distributions and nonparametric Bayes. Unpublished manuscript, 2006.

Thomas L. Griffiths and Mark Steyvers. Finding scientific topics. *Proceedings of the National Academy of Science*, 101:5228–5235, 2004.

Dilek Z. Hakkani-Tür, Kemal Oflazer, and Gökhan Tür. Statistical morphological disambiguation for agglutinative languages. *Computers and the Humanities*, 36(4):381–410, 2002.

Zelig Harris. From phoneme to morpheme. *Language*, 31:190–222, 1955.

Jennifer Hay. Lexical frequency in morphology: Is everything relative? *Linguistics*, 39(6):1041–1070, 2001.

Jennifer Hay and R. Harald Baayen. Shifting paradigms: gradient structure in morphology. *Trends in Cognitive Sciences*, 9(7):342–348, 2005.

Songfang Huang and Steve Renals. Hierarchical Bayesian language models for conversational speech recognition. *IEEE Transactions on Audio, Speech and Language Processing*, 18(8):1941–1954, 2010.

Hemant Ishwaran and Lancelot James. Generalized weighted Chinese restaurant processes for species sampling mixture models. *Statistica Sinica*, 13:1211–1235, 2003.

Ray Jackendoff. *Foundations of Language: Brain, Meaning, Grammar, Evolution*. Oxford University Press, Oxford/New York, 2002.

Mark Johnson. Using adaptor grammars to identify synergies in the unsupervised acquisition of linguistic structure. In *Proceedings of ACL-08: HLT*, Columbus, Ohio, 2008a.

Mark Johnson. Unsupervised word segmentation for Sesotho using adaptor grammars. In *Proceedings of the Tenth Meeting of ACL Special Interest Group on Computational Morphology and Phonology (SIGMORPHON)*, Columbus, Ohio, 2008b.

Mark Johnson. PCFGs, topic models, adaptor grammars and learning topical collocations and the structure of proper names. In *Proceedings of the 48th Annual Meeting of the Association of Computational Linguistics*, pages 1148–1157, Uppsala, Sweden, 2010.

Mark Johnson and Sharon Goldwater. Improving nonparametric Bayesian inference: Experiments on unsupervised word segmentation with adaptor grammars. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, Boulder, Colorado, 2009.

Mark Johnson, Thomas L. Griffiths, and Sharon Goldwater. Adaptor grammars: a framework for specifying compositional nonparametric Bayesian models. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, Cambridge, MA, 2007. MIT Press.

Aravind Joshi. Tree adjoining grammars. In Ruslan Mikkov, editor, *The Oxford Handbook of Computational Linguistics*, pages 483–501. Oxford University Press, Oxford, England, 2003.

Reinhard Kneser and Hermann Ney. Improved backing-off for *n*-gram language modeling. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 181–184, Detroit, Michigan, 1995.

Philipp Koehn and Hieu Hoang. Factored translation models. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 868–876, Prague, Czech Republic, 2007.

Leah S. Larkey, Lisa Ballesteros, and Margaret Connell. Improving stemming for Arabic information retrieval: Light stemming and co-occurrence analysis. In *Proceedings of the 25th International Conference on Research and Development in Information Retrieval (SIGIR)*, pages 275–282, Tampere, Finland, 2002.

Percy Liang, Slav Petrov, Michael Jordan, and Dan Klein. The infinite PCFG using hierarchical Dirichlet processes. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 688–697, Prague, Czech Republic, 2007.

Albert Lo. On a class of Bayesian nonparametric estimates. *Annals of Statistics*, 12:351–357, 1984.

David MacKay and Linda Bauman Peto. A hierarchical Dirichlet language model. *Natural Language Engineering*, 1(1), 1994.

Brian MacWhinney and Catharine Snow. The child language data exchange system. *Journal of Child Language*, 12:271–296, 1985.

Rasmus Madsen, David Kauchak, and Charles Elkan. Modeling word burstiness using the Dirichlet distribution. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 545–552, Bonn, Germany, 2005.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of English: the Penn treebank. *Computational Linguistics*, 19(2):331–330, 1993.

Michael Mitzenmacher. A brief history of generative models for power law and lognormal distributions. *Internet Mathematics*, 1(2):226–251, 2004.

Christian Monson, Alon Lavie, Jaime Carbonell, and Lori Levin. Unsupervised induction of natural language morphology inflection classes. In *Proceedings of the Seventh Meeting of the ACL Special Interest Group in Computational Phonology (SIGPHON)*, pages 52–61, Barcelona, Spain, 2004.

Radford Neal. Markov chain sampling methods for Dirichlet process mixture models. *Journal of Computational and Graphical Statistics*, 9:249–265, 2000.

David Newman, Arthur Asuncion, Padhraic Smyth, and Max Welling. Distributed algorithms for topic models. *Journal of Machine Learning Research*, 10(Aug):1801–1828, 2009.

Hermann Ney, Ufe Essen, and Reinhard Kneser. On structuring probabilistic dependencies in stochastic language modeling. *Computer, Speech, and Language*, 8:1–38, 1994.

Timothy O'Donnell. *Computation and Reuse in Language*. PhD thesis, Harvard University, in preparation.

Timothy O'Donnell, Noah Goodman, and Joshua B. Tenenbaum. Fragment grammars: Exploring computation and reuse in language. Technical Report MIT-CSAIL-TR-2009-013, MIT, 2009.

Janet Pierrehumbert. Probabilistic phonology: Discrimination and robustness. In R. Bod, J. Hay, and S. Jannedy, editors, *Probabilistic Linguistics*. MIT Press, Cambridge, MA, 2003.

Jim Pitman. Exchangeable and partially exchangeable random partitions. *Probability Theory and Related Fields*, 102:145–158, 1995.

Jim Pitman. *Combinatorial Stochastic Processes*. Springer-Verlag, New York, 2006.

Jim Pitman and Marc Yor. The two-parameter Poisson-Dirichlet distribution derived from a stable subordinator. *Annals of Probability*, 25:855–900, 1997.

David Plaut and Laura Gonnerman. Are non-semantic morphological effects incompatible with a distributed connectionist approach to lexical processing? *Language and Cognitive Processes*, 15: 445–485, 2000.

Matt Post and Daniel Gildea. Bayesian learning of a tree substitution grammar. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 45–48, Suntec, Singapore, August 2009.

Carl Rasmussen. The infinite Gaussian mixture model. In *Advances in Neural Information Processing Systems 12*, Cambridge, MA, 2000. MIT Press.

Terry Regier, Bryce Corrigan, Rachael Cabasaan, Amanda Woodward, Michael Gasser, and Linda Smith. The emergence of words. In *Proceedings of the 23rd Annual Conference of the Cognitive Science Society*, pages 815–820, Mahwah, NJ, 2001. Erlbaum.

Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management: an International Journal*, 24(5):513–523, 1988.

Jayazam Sethuraman. A constructive definition of Dirichlet priors. *Statistica Sinica*, 4:639–650, 1994.

Herbert Simon. On a class of skew distribution functions. *Biometrika*, 42(3/4):425–440, 1955.

Matthew Snover and Michael Brent. A probabilistic model for learning concatenative morphology. In *Advances in Neural Information Processing Systems 15*, Cambridge, MA, 2003. MIT Press.

Benjamin Snyder and Regina Barzilay. Unsupervised multilingual learning for morphological segmentation. In *Proceedings of ACL-08: HLT*, pages 737–745, Columbus, Ohio, 2008.

Yee Whye Teh. A Bayesian interpretation of interpolated Kneser-Ney. Technical Report TRA2/06, National University of Singapore, School of Computing, 2006a.

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

Yee Whye Teh, Michael Jordan, Matthew Beal, and David Blei. Hierarchical Dirichlet processes. In *Advances in Neural Information Processing Systems 17*. MIT Press, Cambridge, MA, 2005.

Romain Thibaux and Michael I. Jordan. Hierarchical beta processes and the Indian buffet process. In *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, San Juan, Puerto Rico, 2007.

Hanna M. Wallach. *Structured Topic Models for Language*. PhD thesis, University of Cambridge, 2008.

Mike West. Hyperparameter estimation in Dirichlet process mixture models. Technical Report 92-A03, Institute of Statistics and Decision Sciences, Duke University, 1992.

Frank Wood and Yee Whye Teh. A hierarchical, hierarchical Pitman Yor process language model. In *Proceedings of the ICML/UAI Workshop on Nonparametric Bayes*, Helsinki, Finland, 2008.

Frank Wood and Yee Whye Teh. A hierarchical nonparametric Bayesian approach to statistical language model domain adaptation. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, Clearwater Beach, Florida, 2009.

David Yarowsky and Richard Wicentowski. Minimally supervised morphological analysis by multimodal alignment. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 207–216, Hong Kong, 2000.

George Zipf. *Selective Studies and the Principle of Relative Frequency in Language*. Harvard University Press, Cambridge, MA, 1932.