Nonparametric Bayesian Models of Lexical Acquisition

by

Sharon J. Goldwater

Sc. B., Brown University, 1998

Sc. M., Brown University, 2005

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy

in the Department of Cognitive and Linguistic Sciences at Brown University.

Providence, Rhode Island

May 2007

This dissertation by Sharon J. Goldwater is accepted in its present form by the Department of Cognitive and Linguistic Sciences as satisfying the dissertation requirement for the degree of Doctor of Philosophy.

Date _____ _____

Mark Johnson, Director

Recommended to the Graduate Council

Date _____ _____

Thomas Griffiths, Reader

Date _____ _____

Katherine Demuth, Reader

Date _____ _____

John Goldsmith, Reader

Approved by the Graduate Council

Date _____ _____

Sheila Bonde
Dean of the Graduate School

Acknowledgements

I would like to begin by acknowledging the invaluable support of my advisor, Mark Johnson, whose level of knowledge and expertise I can only hope to achieve someday. Special thanks are also due to my other committee members: to Tom Griffiths, for arriving at Brown in the nick of time (among many other feats), and to Katherine Demuth and John Goldsmith, for providing valuable advice and additional perspectives. Eugene Charniak deserves recognition for his long-ago formative role in my research career, and for continued advice and discussion since then.

My time at Brown has been enriched by the presence of many wonderful student colleagues. I am grateful to the many past and present members of the Brown Laboratory for Linguistic Information Processing (especially Joe Austerweil, Don Blaheta, Will Headden, Matt Lease, David McClosky, Brendan Shean, and Jenine Turner), who provided friendship, stimulating discussion, and feedback on various drafts of this thesis. I would also like to thank Mr. McFrench, for his inspiring professionalism in all things; Frank Wood, for late-afternoon math cram sessions; Katherine White and Naomi Feldman, for helpful feedback on the many presentations of this work that they witnessed; and Julian Wong, for keeping me on my toes.

A number of other friends have also helped me through this effort. Eric Breck, Jeremy

Fisher, Laurel Glaser, Jamileh Jemison, Robin Meisner, David Pober, and Eric Uhrhane: thanks for your perspectives on school, research, and life.

Finally, there are innumerable reasons to thank my parents, Eugene and Eva Goldwater, without whom none of this would have been possible. My father especially deserves credit for choosing just the right birthday gift to inspire a whole career, and my mother for letting me explain my research to her frequently and in great detail. I'm the only kid I know whose mom can still help with my homework.

Contents

Chapter 1

Introduction

1.1 Goals and Approach

The child learning language is faced with a daunting task: to learn to extract meaning from an apparently meaningless stream of sound. In order to achieve this goal, a number of problems must be solved. These range from segmenting individual words out of the acoustic stream to understanding the relationship between syntax and meaning. The work presented here rests on the assumption that these problems cannot be completely solved in isolation, and that it is the interaction between different grammar components that leads to fully successful acquisition of the whole system. Interaction is crucial because partial knowledge of one part of the grammar can place constraints on the kinds of generalizations that can be made in another part of the grammar. Additional constraints may be placed on the system by the presence of a learning bias, the nature of which I investigate here.

Behavioral research is the usual method employed to study language acquisition. However, due to the complex interdependence of different components of language, it can be difficult to determine exactly how each component contributes to the overall learning process. Moreover, behavioral research is necessarily indirect – there is no way to directly examine or manipulate the linguistic representations in a child's mind. The approach taken here, in which computer programs are used to model certain aspects of language acquisition, contrasts with the behavioral approach in these respects. Computer programs can be designed to study individual components of the acquisition process, and can be manipulated to determine the effects of incorporating different sources of information and representational assumptions.

Some researchers question the validity of computational modeling as a methodology for investigating language acquisition in humans. Certainly, current computer processors are

fundamentally different from the human brain: far more powerful in some ways (memorization of large data sets, precise mathematical computation), impoverished in others (sensory input, world knowledge). Computers are nevertheless a useful tool for investigating certain kinds of proposals regarding human learning, particularly those based on the statistical properties of the input. The models I develop here are grounded in Bayesian statistics, and can be thought of as "ideal observer" models, making optimal use of the information presented to them. Examining the ways in which these models succeed or fail can tell us, for example, what kinds of representations are learned by attending to certain types of information. In turn, this may yield insight into whether such information is necessary or sufficient for acquisition.

In this thesis, I focus on developing models for two particular tasks that the child must master: word segmentation (identifying individual words from the speech stream) and morphological analysis (identifying the smaller units of meaning within words). I also explore the nature of any innate biases the child might bring to the acquisition process. Each of these components of lexical acquisition has been studied before, both computationally and behaviorally. My contribution is in developing a flexible unifying framework in which these components can be studied both in isolation (as presented in this document) and together (as pursued in ongoing work using methods sketched in the conclusion to this document). This framework is based on existing statistical models and techniques, but represents the first attempt to apply these methods to the acquisition of linguistic structure. The use of this flexible framework places me in a unique position to examine a number of different questions about the nature of lexical acquisition, including

- What kinds of structures are considered by the learning mechanism?

- How much and what sort of evidence is necessary to produce generalizations?
- Are there innate constraints that are specific to language acquisition, or can language be acquired successfully using only general learning biases?
- What kinds of interactions between linguistic components aid in learning?

The models developed here begin to address the first three of these questions, while the kind of combined model proposed in the final chapter (which builds on the models used here) will eventually allow me to investigate the fourth.

In developing the modeling framework to examine these questions, it was necessary to decide which aspects of human language acquisition were most important to capture. Perhaps the most striking feature of language acquisition is the fact that it takes place without explicit tutoring or access to "correct" analyses. I therefore chose to focus on **unsupervised** learning, where the input consists only of unannotated linguistic data. In particular, I am interested in learning from corpora of **naturalistic data** consisting, when possible, of utterances spoken by parents to their very young children. Ideally, my system would be designed to learn from acoustic data. Due to the constraints of time and technology, most of my input data instead takes the form of phonemically transcribed child-directed speech. For one experiment, I use data from a corpus of newspaper text.

The final aspect of my research that sets it apart from most previous work is that each learner I develop is based on an **explicit probabilistic model**. In particular, my learners are grounded in Bayesian statistics, where models consist of two parts. The *generative model* defines the probability of generating (or observing) a particular linguistic form given a hypothesis about the nature of the grammar. The *prior* distribution over grammars (which can be thought of as a description of the learning bias) defines the probability that

any particular grammar is correct, regardless of the observed data. By manipulating these two parts of the model, I can explore the effects of using different sources of information and prior assumptions for lexical acquisition. I describe this approach in more detail, and discuss some of the specific questions I address, in the following chapter outline.

1.2 Thesis Outline

Chapter 2: Background In this chapter, I discuss the problem of language acquisition as induction, or generalization from incomplete data. I review the two most commonly held views on how humans solve the language induction problem, nativism and connectionism. I then present the Bayesian approach taken in this thesis and argue that it can be more powerful and more informative than previous approaches. Following this theoretical introduction, I discuss some of the mathematical and computational background related to Bayesian learning that is useful for understanding my own work. In particular, I review the use of maximum-likelihood estimation and maximum *a posteriori* estimation for unsupervised learning. Both of these techniques focus on finding a single optimal hypothesis to explain the observed data; I argue in favor of a different approach, where the goal is to estimate a distribution over hypotheses. I explain the mathematics of this approach, and discuss algorithms that can be used to sample from such a distribution.

Chapter 3: The Two-Stage Modeling Framework This chapter introduces the modeling framework that will be used to develop the models in the remainder of the thesis. In this framework, models are specified using a *generator*, which generates lexical items, and an *adaptor*, which generates frequencies for those lexical items. I show that, with appropriate choices for the generator and adaptor, two-stage models are *nonparametric* (allowing the

number of inferred parameters to grow with the size of the data) and produce a power-law distribution on word frequencies. I describe one particular adaptor with these properties, the Chinese restaurant process, and explain its connection to the Dirichlet process, a model used in nonparametric Bayesian statistics. I also discuss the Pitman-Yor process, a generalization of the Chinese restaurant process that can also be used as a two-stage adaptor.

Chapter 4: Morphology In this chapter, I apply the two-stage framework to morphological acquisition and address the question of whether corpus statistics or lexicon statistics are more useful for this task. I review the literature relevant to this question as well as previous computational work on morphological acquisition. I then explain my own novel approach, which uses a two-stage model with a morpheme-based lexicon generator and a Pitman-Yor adaptor. Depending on the parameters chosen for the Pitman-Yor process, the model (whose input is a text corpus) infers morphology from statistical patterns found in the corpus, the set of lexical items in the corpus, or some in-between point. The results of my experiments indicate that morphological information is learned better from types than from tokens. The use of the generator-adaptor framework provides a mathematically principled way to learn from types when the input consists of tokens, and to combine type-based and token-based learning (as outlined in Chapter 6).

Chapter 5: Word Segmentation This chapter describes the application of the two-stage language modeling framework to the problem of word segmentation and investigates the importance of context for this task. Previous model-based computational work and stimuli used in behavioral experiments have typically assumed that word probabilities are independent of context. After reviewing this work, I develop a two-stage model for word

segmentation that also incorporates this assumption. I show that this model under-segments the data, and argue that previous results to the contrary were due to constraints imposed by the search algorithms used rather than to the underlying models. I then describe how to extend the two-stage model to account for sequential dependencies between words, and provide results showing that the extended model outperforms the original model as well as all previously published results on the corpus in question.

Chapter 6: Conclusion In the conclusion to this document, I return to the goals set out in the introduction and review how the modeling framework presented here is useful in addressing them. I summarize the specific results achieved in the previous chapters using separate models of morphology and word segmentation. I then discuss how those individual models can be combined in future work to create an integrated model for simultaneous acquisition of morphology and word segmentation. I present several additional avenues for future research and conclude.

Chapter 2

Background

2.1 The problem of language acquisition

At its heart, language acquisition is a problem of *induction* – the creation of an internal representation of language that allows the learner to generalize beyond the observed linguistic input, interpreting and producing novel linguistic forms. Starting in the first few months of life, children are already beginning to group together certain acoustically distinct waveforms into the categories linguists refer to as phonemes (Jusczyk, 1997). From there, they go on to tasks such as recognizing particular sequences of sounds as words, identifying morphological structure within known words, and applying this knowledge to novel words. Although they may initially make some incorrect generalizations (e.g. *I bringed my lunch*), eventually they master all the complexity of adult language, from phonology to pragmatics. It may take many years for a child to achieve adult-like proficiency with language, but the bulk of the work seems to be accomplished by the age of 7 or 8.

How do children accomplish this feat? In this section, I review two standard theoretical explanations, nativism and empiricism. I then discuss the Bayesian learning approach I have adopted in this work, and argue that it is both more flexible theoretically and more informative experimentally than the others.

2.1.1 Nativism

The apparent difficulty of the problem of language induction is central to nativist theories of acquisition. These theories assume that general learning mechanisms simply are not powerful enough to allow children to make sense of the linguistic input they receive. The reasoning behind this point of view (now referred to as the "argument from the poverty of the stimulus") was originally put forth by Chomsky (Chomsky, 1965):

> It seems clear that many children acquire first or second languages quite successfully even though no special care is taken to teach them and no special attention is given to their progress. It also seems apparent that much of the actual speech observed consists of fragments and deviant expressions of a variety of sorts. Thus it seems that a child must have the ability to "invent" a generative grammar that defines well-formedness and assigns interpretations to sentences even though the primary linguistic data that he uses as a basis for this act of theory construction may, from the point of view of the theory he constructs, be deficient in various respects.

In other words, Chomsky believes that because children are not explicitly taught language, and because their linguistic input is noisy, they must not be learning language entirely from the input, but rather "inventing" it in some sense. Many nativists have bolstered this claim by citing Gold's Theorem (Gold, 1967) as evidence that language is unlearnable even in the absence of noise[1]. Obviously, each child does not invent his or her own unique language; rather, Chomsky proposes the existence of an innate endowment (known as Universal Grammar) that places strong constraints on the kinds of grammars that children can conceivably acquire. Together with these constraints comes a special-purpose "Language Acquisition Device" (LAD) that enables children to move beyond their impoverished input in acquiring adult language (Chomsky, 1965; Crain, 1991).

In principle, the idea of nativism is agnostic regarding the nature of Universal Grammar, the LAD, and the final state of the grammar. In practice, nativists generally assume

[1] Gold's Theorem states that, after observing n strings from a language L belonging to some infinite class of languages C, it is impossible for any learner to identify with certainty which language in C is being presented (regardless of the number n). Critics have pointed out many weaknesses in using this theorem to make claims about human language acquisition; for a discussion see Johnson (2004).

that linguistic representations are highly structured, consisting of categories, rules, and the like (Chomsky, 1957; Pinker, 1988). In Chomsky's Principles and Parameters theory (Chomsky, 1981), Universal Grammar consists of a set of principles that reference these categories and rules, and delimit the set of all possible human languages. Languages differ only in the setting of particular parameters, such as whether syntactic heads come before their arguments or after. Optimality Theory (Prince and Smolensky, 1993), another major nativist theory of language, views Universal Grammar as a set of constraints operating over linguistic objects. Cross-linguistic variation results from the fact that not all constraints can be satisfied simultaneously, and different languages rank the importance of satisfying the various constraints differently.

According to the nativist view, the restriction to a highly structured, highly constrained space of grammars makes the problem of language induction much simpler. For example, certain features in the input might be specified by Universal Grammar to serve as cues, allowing children to set the value of some particular parameter (Dresher and Kaye, 1990; Dresher, 1999). Fixing a parameter eliminates a subset of the universally permissible grammars, thus reducing the space of grammars the child must choose between. Over time, enough cues will be observed to eliminate all but the correct grammar.

2.1.2 Empiricism

Psychologists and linguists who disagree with the nativist approach often align themselves instead with the *empiricist* view of language acquisition. Under this view, language acquisition is based on statistical properties of the input, and is an associative process essentially similar to other associative learning (Elman et al., 1996). Where nativists see children's input as noisy and lacking the full complexity of adult language, empiricists focus on the

rich statistical patterns that are available even in simple child-directed speech. A strict empiricist view rules out the possibility of innate learning mechanisms specific to language; whatever general-purpose learning mechanisms allow children to draw complex inferences in other domains are also sufficient for language.[2]

Just as empiricists minimize the importance of language-specific learning mechanisms, they are also skeptical of language-specific mental representations. In the connectionist theories that constitute much of the empiricist literature, mental representations of language are considered to be probabilistic, distributed, and unstructured, without hard categorical boundaries or explicit rules (Elman, 2004; Seidenberg and Gonnerman, 2000). Computer models have shown that many of the apparently rule-governed phenomena of language can be simulated using these kinds of networks (Christiansen and Curtin, 1999; Elman, 2003; Rumelhart and McClelland, 1986), and connectionists take this as evidence that linguistic rules and categories have no cognitive reality.

Overall, many of the differences between empiricism and nativism can be framed as differences in the *inductive bias* of the learner – the strength and nature of constraints on learning. Nativist theories assume that learning is highly constrained by the nature of linguistic representations and the ways those representations can be combined. Empiricism assumes that constraints are relatively weak, so that learning is guided primarily by the nature of the input. To the extent that constraints are discussed explicitly in the empiricist literature, they are usually viewed as architectural rather than informational (Elman et al., 1996). That is, the actual hardware used to implement a learning algorithm (whether it is an artificial neural network or a human brain) is considered to be an important source

[2]Note, however, that some researchers in the empiricist tradition do concede the possibility of special-purpose learning mechanisms, but downplay their importance (Elman et al., 1996).

of inductive bias, because it can impose constraints on the communication and storage of information. In contrast, nativist theories abstract away from architecture, focusing instead on the kinds of information and representations that can and cannot be learned.

2.1.3 Structured probabilistic approaches

The typical features of nativist learning theories (strong domain-specific constraints, structured deterministic representations, deterministic learning procedures) have often been seen as inextricably linked, as have the corresponding features of empiricism/connectionism (weak general-purpose constraints, distributed representations, statistical learning procedures). However, beginning in the 1990s, many researchers in artificial intelligence (and later, linguistics and cognitive science as well) began to investigate the problem of language acquisition using structured probabilistic approaches (Charniak, 1993; Manning and Schütze, 1999; Boersma and Levelt, 1999; Albright and Hayes, 2003). These approaches contain explicit linguistic representations, such as lexical items and rules governing how those items can combine. However, those representations are associated with probabilities indicating (roughly) how frequently each item occurs in practice. The process of learning is also crucially based on word frequencies and other statistical properties of the input.

Among structured probabilistic approaches, it is important to draw a distinction between two types of systems. This distinction can probably best be understood in terms of Marr's levels of information processing (Marr, 1982). In Marr's terminology, an information processing system can be studied at the *computational* level (focusing on the goal of the computation and strategies for carrying it out), the *algorithmic* level (focusing on input/output representations and procedures for implementing the computation), or the *hardware implementation* level (focusing on the physical realization of the algorithm). Some

structured statistical approaches are concerned primarily with the computational level of processing, others with the algorithmic level. (Note that connectionism, with its concern for architecture, is the only common approach to modeling language acquisition that considers the hardware implementation level as well as the algorithmic level.)

Examples of algorithmic-level models based on statistics and explicit structure date back at least as far as the 1950s, when Harris proposed a procedure for determining morpheme boundaries based on the statistical properties of words (Harris, 1954). More recent algorithmic-level models include the Gradual Learning Algorithm (Boersma and Hayes, 2001), which defines a procedure to follow for learning stochastic Optimality Theory grammars, and Albright and Hayes' stochastic rule-based proposal for learning morphology (2003).

Although these procedural proposals can be quite successful, the kinds of questions I am interested in are more suited to a computational-level approach. I therefore develop systems based on explicit probabilistic models. A probabilistic model is a computational-level proposal about learning – that is, it describes the goal of learning, the different kinds of information that should be considered, and how those kinds of information should interact. Of course, in order to implement a complete learning system, it is necessary to pair a probabilistic model with a compatible learning algorithm that implements a procedure for reaching the stated goal. However, the choice of algorithm is important only in the sense that some algorithms may be insufficient to achieve the goals of the model.[3] Two quite

[3]The widespread use of the term *model* to describe both a proposal about the nature of learning or its implementation (as in "connectionist model") and a specific mathematical statement regarding the process generating a set of data (as in "probabilistic model", "generative model") is unfortunately somewhat confusing. I will attempt to be as clear as possible about which sort of model I am referring to, and will generally use the term "system" to describe fully implemented (probabilistic-)model-based learners when the distinction between the probabilistic model and the model plus algorithm is important.

different algorithms could be used with the same probabilistic model; the two resulting systems would be equivalent at the computational level (and produce equivalent results) provided the algorithms were both able to achieve the model's objective. This contrasts with algorithmic-level proposals, where changing any details of the procedure used for learning would change the results.

The particular framework I will be using to model lexical acquisition is known as *Bayesian learning*. Like other structured probabilistic models, the kind of Bayesian models I use employ explicit linguistic representations, which is helpful for understanding the kinds of information the model has access to, and the ways that information can be used.[4] However, Bayesian models include an additional component that is particularly useful for studying language acquisition. This component, known as the prior, places constraints on the kinds of generalizations that can be made by the model. Unlike the hard constraints of Universal Grammar, these biases are statistical in nature, meaning that they guide the learner in certain directions, but can be overridden by enough contrary evidence. Because these biases are made explicit, they can be directly manipulated in order to examine the effects on linguistic generalization. Modeling constraints in this way is quite different from other approaches, where constraints are imposed implicitly by algorithms or architectures. Changing the number of nodes or connections in a neural network, for example, may lead to different results, but it will be difficult to determine why, or what the implications are.

A common argument against the use of Bayesian models (and other probabilistic models as well) in cognitive science is that the mathematical descriptions of these models, and the

[4]There is nothing inherent in the Bayesian approach that requires structured linguistic representations; Bayesian models can be defined over distributed representations similar to the ones typically used in neural networks. However, most Bayesian models of language acquisition have employed structured representations.

algorithms needed to perform inference, can be quite complex. Since a Bayesian model is intended to describe the computational level of processing, whether or not an associated inference algorithm is cognitively plausible is not of primary interest. Rather, it should only matter whether *some* inference algorithm for the model could be implemented in the human brain. The fact that we do not yet know what this algorithm might be should not be fatal to the theory. On the other hand, it would be a real problem if Bayesian learning is simply too complex for any cognitive process to implement. However, there is growing evidence from a variety of domains (including natural language) that humans do behave in a way consistent with Bayesian inference (Tenenbaum and Xu, 2000; Gerken, 2006; Griffiths and Tenenbaum, In press). This does not mean that humans literally have probabilistic equations in their minds, but it does indicate that these equations provide a useful way of describing human behavior. Moreover, if human learning is Bayesian, then exploring the sources of information available in language and how they are exploited by a Bayesian learner can tell us something about human language acquisition. We may discover that optimal Bayesian inference does not fully account for human acquisition; in that case, it would still be useful to examine how humans might differ from the "ideal" Bayesian learner, and why.

Prior to presenting my own work on Bayesian modeling for language acquisition, I review some useful computational and mathematical background material. In the remainder of this chapter, I first describe some previous model-based approaches to language acquisition. I then discuss how my own approach differs from these, and present in general terms the family of inference algorithms used in my research.

2.2 Computational preliminaries

Within the domain of structured probabilistic models, nearly all language learning systems have been based on Bayes' rule, which defines the probability of hypothesis h (i.e. a grammar or other linguistic analysis) given the observed data d:

$$\begin{aligned} P(h|d) &= \frac{P(d|h)P(h)}{P(d)} \\ &\propto P(d|h)P(h) \end{aligned}$$

That is, the *posterior* probability $P(h|d)$ is proportional to the product of $P(d|h)$ (the *likelihood*, or probability of the data under hypothesis h) and $P(h)$ (the *prior* probability of h). The likelihood evaluates how well h explains the observed data, and the prior evaluates how well h conforms to expectations about what a good hypothesis should be like, regardless of the observed data. A hypothesis with a high prior probability requires less evidence in its favor in order to be accepted. From a cognitive perspective, the prior distribution results from a combination of innate learning biases and previous experience, and serves as a constraint on learning.

In the remainder of this section, I review the two most common approaches that have been used in model-based computational language learning systems: maximum-likelihood estimation, which considers only the likelihood term when evaluating hypotheses, and the minimum description length principle, which is a particular way of encoding a prior over hypotheses. I then discuss in general terms how my approach differs from either of these.

2.2.1 Maximum-likelihood estimation

Maximum-likelihood estimation, as the name implies, assumes that the goal of learning is to select the hypothesis $\hat{h}$ with the highest likelihood:

$$\hat{h} \quad = \quad \underset{h}{\operatorname{argmax}}\, P(d|h)$$

For a fixed parameterization, this is equivalent to assuming that all hypotheses are equally probable *a priori*, and then choosing the single hypothesis with the highest posterior probability.

For many applications, maximum-likelihood estimation can be performed in a straightforward way using the algorithmic framework known as Expectation Maximization (EM) (Dempster et al., 1977; Neal and Hinton, 1998). EM is particularly useful for models with latent variables (whose values cannot be observed directly from the data), so it is popular for unsupervised learning. Examples of EM for linguistic tasks include the forward-backward algorithm for learning hidden Markov models and the inside-outside algorithm for learning context-free grammars. EM is an iterative procedure with the attractive property that the likelihood is guaranteed to increase (or remain the same) at each iteration.

There are at least two disadvantages to EM, however. First is the fact that it is guaranteed to converge only to a local maximum of the likelihood function, not the global maximum. Complex models such as those often found in linguistic applications generally have many local maxima, and these can be quite distant from the global maximum. This can lead to poor results that are highly dependent on parameter initialization (Carroll and Charniak, 1992).

Another limitation of EM (and maximum-likelihood estimation in general) is that it is only useful for finding optimal parameter values in models where the number of parameters

Figure 2.1: Regression using a simple function (left) and a complex function (right).

is known. Consider a regression problem, where we must find the function that best fits a set of data points. If we limit our hypothesis space to the set of linear functions (each of which has two parameters: slope and intercept), we can use maximum-likelihood estimation to find the best-fitting line. However, if we do not restrict the hypothesis space in this way, a more complex function (i.e. with more parameters) can be found that will perfectly fit the data points, yielding a much higher likelihood (see Figure 2.1). Judging purely by the likelihood, this complex function is a better solution, even though intuitively it is less likely to generalize to future observations. This phenomenon is known as *overfitting* the data.

Within the framework of maximum-likelihood estimation, there are methods (such as likelihood ratio tests) that can be used to determine whether the increase in likelihood achieved by a more complex model is sufficient to outweigh the extra parameters involved. However, these methods require enumerating all hypotheses under consideration, which can make it difficult to search through a complex hypothesis space. In addition, language learning often involves choosing between hypotheses that differ not only in the number of parameters, but also in the structure of those parameters. For example, two context-free grammars may have the same total number of rules and categories, but the rule structures may be completely different. This sort of learning problem is far more difficult than simply identifying the optimal parameter values in a space of similar hypotheses.

Because of the problem of structure identification, maximum-likelihood estimation techniques are often inappropriate for language learning. A number of researchers have therefore turned to methods that incorporate a non-uniform prior distribution over hypotheses. The prior can be used to favor hypotheses with fewer parameters or particular kinds of structures, and serves to counterbalance the importance of the likelihood in model inference. The remainder of this chapter is devoted to these kinds of methods.

2.2.2 Minimum description length

Rather than using maximum-likelihood estimation, many successful unsupervised learning systems have been based on the assumption that simpler hypotheses are *a priori* more probable, and have been designed to find the *Maximum a Posteriori* (MAP) solution under this assumption:

$$\hat{h} \quad = \quad \operatorname*{argmax}_{h} P(d \,|\, h) P(h) \tag{2.1}$$

This approach has the advantage that it provides a principled way to compare hypotheses with different numbers of parameters, but it raises the question of exactly how to evaluate the notion of "simplicity" in a hypothesis. I have mentioned one obvious metric, the number of parameters required to describe the hypothesis. However, in the domain of language, linguistic plausibility or naturalness is also a desirable criterion for many researchers. A common way of approaching the problem of designing a linguistically motivated prior is to use the minimum description length (MDL) principle (Rissanen, 1989).

Informally, an MDL prior favors hypotheses that can be described succinctly. Imagine that we need to encode a corpus of words as efficiently (i.e. with as few characters) as possible. Encoding each word as itself would be simple, but inefficient. Instead, we could

design a codebook where each word was represented by a unique string. Frequent words could be assigned short strings, and infrequent words would be assigned longer strings. With a well-designed codebook, the total number of characters required to encode the corpus (the length of the codebook plus the length of the encoded corpus) would be less than in the original corpus. We could go further and encode smaller units such as morphemes or phonemes, which would require fewer codewords and thus a shorter codebook. However, in general, as we use fewer codewords in the codebook, it will become more difficult to encode the corpus efficiently with them, leading to a trade-off between the length of the codebook and the length of the encoded corpus. The MDL principle states that we should choose the codebook that leads to the shortest total combined length.

The relationship between MDL and Bayesian inference becomes clear when we consider results from information theory. In particular, information theory tells us that, under an optimal encoding, the length (in bits) of an encoded corpus will be exactly $-\log P(d \mid h)$, where d is the corpus and h is the codebook used to encode d. Therefore the optimal codebook $\hat{h}$ will be the one that satisfies

$$\begin{aligned} \hat{h} &= \operatorname*{argmin}_h(\text{len}(\text{encoding}_h(d)) + \text{len}(h)) \\ &= \operatorname*{argmin}_h(-\log_2 P(d \mid h) + \text{len}(h)) \\ &= \operatorname*{argmax}_h(P(d \mid h) \cdot 2^{(-\text{len}(h))}) \end{aligned} \tag{2.2}$$

In other words, MDL is simply MAP Bayesian inference with the assumption that the prior probability of a particular hypothesized grammar (the codebook) decreases exponentially with its description length. The usual argument for MDL is that linguistically principled grammars should be able to describe a corpus well with relatively little descriptive overhead, so searching for the lowest combined description length will tend to find such grammars.

MDL has been used successfully for unsupervised learning in a variety of linguistic domains, including word segmentation (de Marcken, 1995; Brent and Cartwright, 1996), phonology (Ellison, 1994), morphology (Goldsmith, 2001b; Creutz and Lagus, 2002), and syntax (Dowman, 2000). It provides a principled approach to the structure-learning problem, which is an improvement on maximum likelihood estimation, but (at least with current technology) it is far from an ideal solution. One problem is that the results of an MDL-based system can be quite sensitive to seemingly minor changes in the encoding scheme, in ways that are not linguistically transparent (Goldwater and Johnson, 2004). Also, the flexibility of the MDL framework in allowing the researcher to choose an arbitrary encoding scheme means that there is no standard way to search the space of possible hypotheses for the optimal solution. Instead, researchers must design special-purpose algorithms using heuristics or stochastic search. Either way, there is no guarantee that the algorithm will find (or even approximate) the optimal solution.

There have been some attempts to develop Bayesian language learning systems using priors defined by mathematical equations rather than by encoding methods (Brent, 1999; Snover and Brent, 2003; Creutz, 2003). This approach can make the assumptions inherent in the prior more explicit, and is in fact the approach I will take here. The difference between my work and these previous systems lies in the added modeling flexibility allowed by the two-stage framework I propose, and in the inference procedures I use, which are instances of general-purpose algorithms with convergence guarantees. Previous Bayesian systems for language acquisition, MDL-based and otherwise, have relied on special-purpose algorithms with no such guarantees.

2.2.3 Bayesian statistics

The above review of previous language acquisition systems shows that most either use maximum likelihood estimation with EM (or some variant), or a non-uniform prior with some specially designed inference procedure. This raises the question of whether it is possible to combine EM or other standard algorithms with models that include non-uniform priors. In fact, using standard techniques from Bayesian statistics, it is. In this section, I review some of those techniques, to prepare the reader for my presentation of the two-stage modeling framework in the following chapter.

Within Bayesian statistics, certain kinds of priors have convenient mathematical properties that lead to their widespread use. To illustrate some of these properties, I will use the example of the Dirichlet distribution, a prior over multinomials. Suppose we have a multinomial with possible outcomes $\{1 \dots K\}$ and parameters $\boldsymbol{\theta} = \{\theta_1 \dots \theta_K\}$, i.e. the probability of outcome $k \in \{1 \dots K\}$ is θ_k. From this multinomial we sample a set of outcomes $\mathbf{x} = \{x_1 \dots x_n\}$, so that $P(x_i = k) = \theta_k$. This can also be written

$$x_i \,|\, \boldsymbol{\theta} \;\sim\; \text{Multinomial}(\boldsymbol{\theta}) \tag{2.3}$$

which can be read as "x_i is distributed according to a multinomial with parameters $\boldsymbol{\theta}$." The Dirichlet prior is a distribution over multinomials, i.e. each sample from a Dirichlet distribution is a set of parameter values $\boldsymbol{\theta}$. Using a Dirichlet prior over a multinomial distribution therefore gives us

$$x_i \,|\, \boldsymbol{\theta} \;\sim\; \text{Multinomial}(\boldsymbol{\theta}) \tag{2.4}$$

$$\boldsymbol{\theta} \,|\, \boldsymbol{\beta} \;\sim\; \text{Dirichlet}(\boldsymbol{\beta}) \tag{2.5}$$

where $\boldsymbol{\beta} = \{\beta_1 \dots \beta_K\}$ are the parameters (also known as *hyperparameters*) of the Dirichlet

distribution. I will discuss the effects of different choices for the hyperparameters in a moment. First, consider the definition of the Dirichlet distribution:

$$\begin{aligned} P(\boldsymbol{\theta} \,|\, \boldsymbol{\beta}) &= c \prod_{k=1}^{K} \theta_k^{\beta_k - 1} \\ \text{with } \; c &= \frac{\Gamma(\sum_{k=1}^{K} \beta_k)}{\prod_{k=1}^{K} \Gamma(\beta_k)} \end{aligned} \tag{2.6}$$

where $\beta_k > 0$. The Gamma function, which appears in the normalizing constant c, 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 $x > 0$. To determine the effect of a Dirichlet($\boldsymbol{\beta}$) prior on the multinomial parameters that are inferred from observing $\mathbf{x}$, we use Bayes' rule:

$$\begin{aligned} P(\boldsymbol{\theta} \,|\, \mathbf{x}, \boldsymbol{\beta}) &\propto P(\mathbf{x} \,|\, \boldsymbol{\theta}) P(\boldsymbol{\theta} \,|\, \boldsymbol{\beta}) \\ &\propto \prod_{i=1}^{n} P_\theta(x_i) \prod_{k=1}^{K} \theta_k^{\beta_k - 1} \\ &= \prod_{k=1}^{K} \theta_k^{n_k} \prod_{k=1}^{K} \theta_k^{\beta_k - 1} \\ &= \prod_{k=1}^{K} \theta_k^{n_k + \beta_k - 1} \end{aligned} \tag{2.7}$$

where n_k is the number of occurrences in $\mathbf{x}$ of outcome k. Notice that the posterior distribution $P(\boldsymbol{\theta} \,|\, \mathbf{x})$ takes the form of another Dirichlet, with parameters $n_k + \beta_k$. A prior is said to be *conjugate* to a distribution if the posterior has the same form as the prior. Thus, the Dirichlet is conjugate to the multinomial.

Once we know the expression for $P(\boldsymbol{\theta} \,|\, \mathbf{x}, \boldsymbol{\beta})$, how can we use this information? The approaches I have discussed so far search for the MAP estimate of a model, under the assumption that this estimate will yield good predictions for future observations. In the

case of the Dirichlet-multinomial, the MAP estimate $\text{argmax}_{\boldsymbol{\theta}} P(\boldsymbol{\theta} \mid \mathbf{x}, \boldsymbol{\beta})$ results in $\theta_k = \frac{n_k + \beta_k - 1}{n + \sum_{k=1}^{K} (\beta_k - 1)}$. Thus, the MAP estimate of $\boldsymbol{\theta}$ using a Dirichlet($\boldsymbol{\beta}$) prior and observed counts $\{n_1, \ldots, n_K\}$ is equivalent to the maximum likelihood estimate of $\boldsymbol{\theta}$ with observed counts $\{n_1 + \beta_1 - 1, \ldots, n_K + \beta_K - 1\}$. This means that standard maximum likelihood estimation techniques such as EM can be used to find the MAP estimate of a multinomial with a Dirichlet prior by simply adding "pseudocounts" of size $\beta_k - 1$ to each of the real expected counts n_k at each iteration. This method works if $\beta_k \geq 1$ for all k, in which case the Dirichlet prior has a smoothing effect: as β_k grows larger, it will tend to increase the size of θ_k relative to the other multinomial parameters. When all the hyperparameters have the same value γ (a *symmetric* Dirichlet prior), increasing γ will lead to more uniform multinomials.

There is a problem with using MAP estimation when any β_k is less than one, however. As Figure 2.2 shows, hyperparameters less than one lead to a probability density function with mass concentrated at values where one or more parameters are equal to zero. As a result, the posterior probability of a set of parameters may be maximized by setting some parameters equal to zero, even though doing so causes the likelihood to become zero and the data to be un-analyzable. As a toy example, suppose we are using EM to learn syntactic rule probabilities for parsing using the standard parameterization. The data d contains only two strings, a and b, and the grammar consists of the following rules (with probabilities shown to the left of each rule):

$$\begin{array}{rlrl} \theta_x & S \rightarrow X & 1 & X \rightarrow a \\ \theta_y & S \rightarrow Y & 1 & Y \rightarrow a \\ 1 - \theta_x - \theta_y & S \rightarrow B & 1 & B \rightarrow b \end{array}$$

We initialize the rules to have uniform probability, so $\theta_x = \theta_y = \frac{1}{3}$, and we use the same

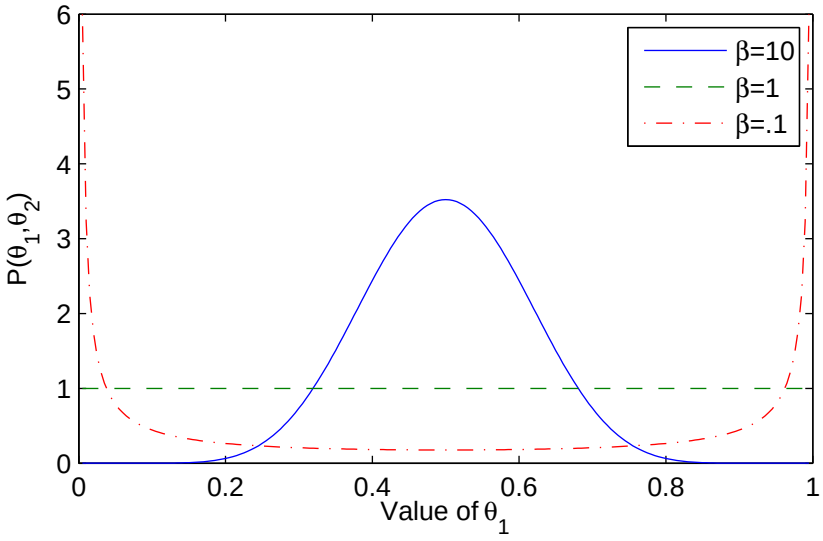

Figure 2.2: Probability density functions of the symmetric Beta(β, β) distribution (the two-dimensional Dirichlet), with different values of β. When $\beta < 1$, mass is concentrated where either θ_1 or $\theta_2 = 1 - \theta_1$ is close to zero. When $\beta > 1$, mass is concentrated where $\theta_1 \approx \theta_2$.

Dirichlet hyperparameter $\beta = .2$ for all S productions. Under these conditions, the expected counts n_x and n_y of rules $S \rightarrow X$ and $S \rightarrow Y$ are both .5, and the expected count n_b of rule $S \rightarrow B$ is 1. From equation 2.7, we have

$$\begin{aligned} P(\boldsymbol{\theta} \mid d, \beta) &\propto \theta_x^{n_x+\beta-1}\theta_y^{n_y+\beta-1}(1-\theta_x-\theta_y)^{n_b+\beta-1} \\ &= \theta_x^{-.3}\theta_y^{-.3}(1-\theta_x-\theta_y)^{.2} \end{aligned} \tag{2.8}$$

It is easy to see that this function is maximized when $\theta_x = \theta_y = 0$, yet using this MAP estimate for the parameter values makes the string a unparseable.

This example illustrates how MAP estimation may fail when $\beta_k < 1$. Yet these hyperparameter values are ideal for unsupervised learning, because multinomials where many of the θ_k are close to zero (often called *sparse solutions*) will have high probability, and the data will be explained using fewer parameters when possible. In other words, using a Dirichlet prior with hyperparameters between 0 and 1 is a way to introduce a bias against

model complexity and in favor of generalization.

Although using EM with $\beta_k < 1$ is problematic, there are other ways to approach the problem of Bayesian inference. In the models I will be discussing, there is no attempt to directly represent $\boldsymbol{\theta}$ at all. Rather than estimating a particular set of values for $\boldsymbol{\theta}$ and using these to predict future observations, the conditional distribution of a new observation is derived by integrating over all possible values of $\boldsymbol{\theta}$:

$$\begin{aligned} P(x_{n+1} = j \,|\, \mathbf{x}, \boldsymbol{\beta}) &= \int_\Delta P(x_{n+1} = j \,|\, \boldsymbol{\theta}) P(\boldsymbol{\theta} \,|\, \mathbf{x}, \boldsymbol{\beta})\, d\boldsymbol{\theta} \\ &= \int_\Delta \theta_j \frac{\Gamma(n + \sum_{k=1}^K \beta_k)}{\prod_{k=1}^K \Gamma(n_k + \beta_k)} \prod_{k=1}^K \theta_k^{n_k + \beta_k - 1}\, d\boldsymbol{\theta} \\ &= \frac{\Gamma(n + \sum_{k=1}^K \beta_k)}{\prod_{k=1}^K \Gamma(n_k + \beta_k)} \int_\Delta \theta_j^{n_j + \beta_j} \prod_{k \neq j} \theta_k^{n_k + \beta_k - 1}\, d\boldsymbol{\theta} \\ &= \frac{\Gamma(n + \sum_{k=1}^K \beta_k)}{\prod_{k=1}^K \Gamma(n_k + \beta_k)} \cdot \frac{\Gamma(n_j + \beta_j + 1) \prod_{j \neq k} \Gamma(n_k + \beta_k)}{\Gamma(n + \sum_{k=1}^K \beta_k + 1)} \\ &= \frac{n_j + \beta_j}{n + \sum_{k=1}^K \beta_k} \end{aligned} \tag{2.9}$$

where Δ indicates the probability simplex, i.e. the set of values for $\boldsymbol{\theta}$ such that $\sum_k \theta_k = 1$. The integral in the third line can be solved analytically, but the fourth line can also be derived by using the fact that the Dirichlet distribution must sum to 1, and therefore

$$\int_\Delta \prod_{k=1}^K \theta_k^{\beta_k - 1}\, d\boldsymbol{\theta} = \frac{\prod_{k=1}^K \Gamma(\beta_k)}{\Gamma(\sum_{k=1}^K \beta_k)} \tag{2.10}$$

for any positive values of β_k.

Integrating out the model parameters in this way is advantageous in several respects. First, it permits the use of Dirichlet hyperparameters arbitrarily close to zero, because these cannot cause negative estimates for $P(x_{n+1} = j \,|\, \mathbf{x}, \boldsymbol{\beta})$. This approach can therefore be used to infer sparse solutions in Dirichlet-multinomial models (and others, as we shall see in the next chapter). Additionally, although the MAP estimate can yield good predictions

when the distribution $P(\boldsymbol{\theta} \mid \mathbf{x}, \boldsymbol{\beta})$ has a single narrow peak, it may be less accurate if the distribution is relatively flat or has multiple modes. Considering all possible parameter values may lead to better predictions in these cases.

A final advantage of this approach is that, because the model parameters are not represented explicitly, it becomes possible to work with the kinds of model families I will describe in the next chapter, where the number of parameters is unbounded. Although individual models are infinite, averaging over models by integrating out the model parameters actually leads to a finite representation that can be used for inference.

2.2.4 Inference via sampling

Standard algorithms for MAP estimation of model parameters, such as EM, are inappropriate if we want to work with the full posterior distribution over models. These algorithms provide a point estimate of the MAP solution, not an estimate of the full posterior distribution. Moreover, estimating the posterior analytically is generally intractable. Fortunately, there is a class of algorithms we can use to obtain *samples* from the posterior distribution of interest. These samples can be used to estimate the posterior distribution of the parameters $\boldsymbol{\theta}$, or any function of the posterior, including the MAP or expected value.

These sampling algorithms, known as Markov chain Monte Carlo (MCMC) methods (Besag, 2000; Neal, 1993; Gilks et al., 1996), are based on the concept of a Markov chain — a stochastic process with random states $Y^1 \ldots Y^T$ where $P(Y^t = y \mid Y^1 \ldots Y^{t-1}) = P(Y^t = y \mid Y^{t-1})$. The *transition probability matrix* of the chain is a matrix $\mathbf{P}$ that defines $P(y, y')$ for all pairs of states (y, y'): the probability of transitioning to state y' given that the current state is y. The (i, j)th entry of $\mathbf{P}$ corresponds to the probability of a transition from the ith state to the jth state. Under certain conditions (detailed below), the chain will converge

to a unique *stationary distribution*, defined as the distribution π over states (specified as a row vector) such that

$$\pi \mathbf{P} = \pi \tag{2.11}$$

In other words, after converging to the stationary distribution, the probability of being in each state is the same at every time step.

In MCMC, each possible state of the Markov chain consists of an assignment of values to all the variables in the model we wish to sample from. This means that the state space of the Markov chain is equal to the hypothesis space of models. By defining $\mathbf{P}$ appropriately, we can guarantee that the stationary distribution of the chain will be the posterior distribution over the model variables. Once the chain has converged, each Y^t will be a sample from the posterior distribution.

Several conditions must hold in order for the Markov chain to converge to the stationary distribution we want. First, there must be a finite path with non-zero probability between any two states (the chain is *irreducible*). Second, no state may occur only after a multiple of $c \geq 2$ time steps (the chain is *aperiodic*). This second condition is fulfilled for any irreducible chain with $P(y, y) > 0$ for some y. Taken together, these two conditions specify an *ergodic* chain.

The final necessary property of the Markov chain is that the transition probability matrix satisfies $\pi \mathbf{P} = \pi$ (*general balance*) when π is the posterior distribution we wish to sample from. This property may be satisfied by constructing $\mathbf{P}$ as the composition of several simpler component matrices $P_1 \dots P_K$ where each of the components satisfies the more restrictive condition of *detailed balance*:

$$\pi(y) P_k(y, y') = \pi(y') P_k(y', y) \tag{2.12}$$

where $P_k(y, y')$ is the probability in P_k of transitioning from y to y'. Applying each of the P_k in turn (or choosing P_k at random) produces a $\mathbf{P}$ that satisfies general balance.

There are several kinds of sampling algorithms that are designed to satisfy detailed balance. The simplest of these is Gibbs sampling, which I use in my experiments, and will describe next. For completeness, I then discuss a more general class of algorithms (of which Gibbs samplers are a special case) called Metropolis-Hastings samplers. These algorithms will likely become necessary for the extensions to my work discussed in Chapter 6.

2.2.4.1 Gibbs sampling

In the above discussion, I have treated the Markov chain state variable Y as a unitary object. When discussing sampling algorithms, it is useful to consider the subcomponents $Y_1 \ldots Y_K$ of the state, where Y_k corresponds to a single variable in the model we are sampling. In the basic Gibbs sampling algorithm (Geman and Geman, 1984), each iteration of the sampler consists of K steps. In the kth step, a new value for Y_k is sampled from its conditional distribution given the current values of all the other variables. That is, we let

$$P_k(y, y') = P(y'_k \mid \{y_j : j \neq k\}) \prod_{j \neq k} I(y_j = y'_j) \tag{2.13}$$

where y_k is the value of the kth model variable and $I(.)$ is an indicator function taking on the value 1 when its argument is true, and 0 otherwise. Thus, the kth component of the transition matrix allows only Y_k to change values. The general proof of detailed balance for Metropolis-Hastings samplers (given below) applies to Gibbs samplers as a special case, but it is also easy to see intuitively that each P_k satisfies general balance when the distribution over states is the posterior distribution of interest. For $j \neq k$, y_j does not change under P_k, so the distribution over these components is clearly maintained. The distribution over y_k

is maintained by construction, so general balance is satisfied.

Before applying this version of Gibbs sampling, then, we need only verify that the chain constructed from the P_k is ergodic. If all the conditional probabilities used in the definition of every P_k are non-zero, this will be the case. However, it may be that certain variables are tied in such a way that changing only one results in an inconsistent (zero-probability) state. For example, when segmenting words into stems and suffixes, changing the suffix of a word requires changing its stem as well. A Markov chain where only a single variable may change at once will be non-ergodic.

To circumvent this problem, Gibbs samplers may be *blocked*: each P_k is designed to resample a block of variables at once. For example, both a stem and suffix may be resampled simultaneously. This procedure ensures that the chain is ergodic, while still maintaining detailed balance. Blocked Gibbs samplers can be used to avoid zero-probability transitions, but are also useful when certain variables are highly correlated, because they increase the *mobility* of the chain – the rate at which it moves from one state to another very different state. This can decrease the time to convergence and allow more efficient sampling.

2.2.4.2 Metropolis-Hastings sampling

While conceptually simple, Gibbs sampling requires that the exact conditional distribution of each variable (or set of variables) be computed at each step. In practice, these computations may be difficult or inefficient to perform. Metropolis-Hastings samplers (Metropolis et al., 1953; Hastings, 1970) provide a way to construct a transition probability matrix satisfying detailed balance without having to compute the exact conditionals. In particular, they define

$$P_k(y, y') = R_k(y, y')A_k(y, y') \tag{2.14}$$

where R_k is some *proposal distribution* over transitions from y to y', and A_k is the *acceptance probability*, defined as $A_k(y, y') = 0$ if $R_k(y, y') = 0$ and

$$A_k(y, y') = \min\left\{1, \frac{\pi(y')R_k(y', y)}{\pi(y)R_k(y, y')}\right\} \tag{2.15}$$

otherwise. In other words, a Metropolis-Hastings sampler samples a new state from R_k and transitions to it with probability A_k, otherwise remaining in the same state. Notice that Gibbs sampling is a special case of Metropolis-Hastings sampling where $R_k = P_k$, so that the acceptance probability is always 1.

Detailed balance can easily be verified for the Metropolis-Hastings sampler:

$$\begin{aligned}
\pi(y)R_k(y, y')A_k(y, y') &= \pi(y)R_k(y, y')\min\left\{1, \frac{\pi(y')R_k(y', y)}{\pi(y)R_k(y, y')}\right\} \\
&= \min\left\{\pi(y)R_k(y, y'), \pi(y')R_k(y', y)\right\} \\
&= \pi(y')R_k(y', y)\min\left\{\frac{\pi(y)R_k(y, y')}{\pi(y')R_k(y', y)}, 1\right\} \\
&= \pi(y')R_k(y', y)A_k(y', y)
\end{aligned} \tag{2.16}$$

Like Gibbs sampling, Metropolis-Hastings sampling can be applied to individual variables or blocks of variables.

2.2.4.3 Cognitive plausibility

As discussed in Section 2.1.3, the focus of the Bayesian approach to cognitive modeling is on the probabilistic model itself, rather than on the specifics of the inference procedure. Nevertheless, it is worth considering the cognitive plausibility of the kinds of sampling algorithms just described. Two different factors are often mentioned in discussions of the cognitive plausibility of an algorithm. One is whether the algorithm processes information online or in batch mode. Online algorithms receive a single presentation of each data point,

which they process before observing the next data point. This mode of operation seems to correspond roughly to the way humans process information. Batch algorithms, on the other hand, receive all their input data at once, and may iterate through it multiple times. They are generally considered implausible as models of human information processing.

The sampling algorithms that I have described clearly operate in batch processing mode, iterating over the entire data set at each sampling step. However, there are closely related sampling algorithms that can approximate the posterior distribution over hypotheses when presented a set of data online. These algorithms use a technique known as particle filtering (Doucet et al., 2000) to track a number of samples from the posterior at every time step. New samples are calculated from the old samples after each data point is observed, so the estimate of the posterior changes over time.

Another property of algorithms that cognitive scientists often consider is the amount of memory required for processing. On this metric, sampling algorithms do well compared to many other statistical procedures. At any given time, the state of an MCMC sampler consists only of the values currently assigned to each variable in the model. In the kinds of models I will be presenting, this corresponds to the number of times each word (or morpheme, bigram, etc.) occurs in the currently hypothesized analysis of the data. Crucially, items with zero counts need not be represented. In contrast, algorithms like EM must represent the values of every parameter in the model, including parameters corresponding to items with very low probability that never occur in the highest probability analysis of the data. As an example, suppose we have an HMM language model for part-of-speech tagging. In order to learn the model parameters using EM, the probabilities of every possible transition and emission must be stored at all times – a total of $T^2 + TW$ parameters if there are

T tags and W words. Using a sampling algorithm, each sample consists of an assignment of part-of-speech tags to the words in the data. At each iteration, the sampler need only keep track of the number of times each tag-tag and tag-word pair occurs *in the current sample*. In theory, the number of such pairs with non-zero counts could be as high as $T^2 + TW$, but in practice it is likely to be much smaller, especially for the kind of sparse solutions we are interested in.

2.3 Conclusion

In this chapter, I have presented the theoretical and computational underpinnings of my work. I have discussed how the Bayesian approach provides a way to explicitly specify the sources of information available to a learner and the constraints on learning. Bayesian models are therefore particularly useful for addressing questions at the computational level of processing. The use of a prior contrasts with maximum-likelihood estimation, which considers only the observed data and is therefore prone to overfitting. There are different ways to define the goal of a Bayesian learner; most previous researchers have assumed that the goal is to identify the most probable solution under the posterior distribution. I have argued that estimating the full posterior distribution can be more informative, and also provides a way to perform inference in cases where it would be difficult or impossible otherwise. Finally, I have described in general terms some algorithms that can be used to estimate the posterior by producing samples from it. In the following chapter, I show how the Bayesian approach described here can be used to develop models of lexical acquisition.

Chapter 3

The Two-Stage Modeling Framework

3.1 Introduction

This chapter introduces the two-stage modeling framework I will be applying in the remainder of this thesis, and illustrates its advantages over previous computational approaches to the problem of unsupervised language acquisition. The two-stage framework is based on the Bayesian statistical techniques discussed in the previous chapter, so it can be used in conjunction with standard sampling algorithms to infer sparse solutions – those that can be described with relatively few parameters. It is a *nonparametric* framework, which means that the number of parameters in a two-stage model need not be specified in advance. Instead, the number of parameters will tend to grow naturally with the size of the data. Finally, this framework is highly flexible, because models are specified as two separate components: a *lexicon generator* (or simply *generator*), which models the kind of items that are likely to be found in the lexicon, and an *adaptor*, which assigns frequencies to those lexical items.

3.2 Intuition

The basic generative process underlying any model within the two-stage framework creates a sequence of words $\mathbf{w} = w_1 \dots w_n$ as follows:

1. Generate a sequence of lexical items $\boldsymbol{\ell} = \ell_1 \dots \ell_K$ from some probability distribution P_ω, the *lexicon generator* (or simply *generator*).

2. Generate a sequence of integers $\mathbf{z} = z_1 \dots z_n$ with $1 \leq z_i \leq K$, where $z_i = k$ indicates that $w_i = \ell_k$ (i.e. z_i is the index of the lexical item corresponding to w_i). These integers are assumed to be generated by some stochastic process P_γ with one or more

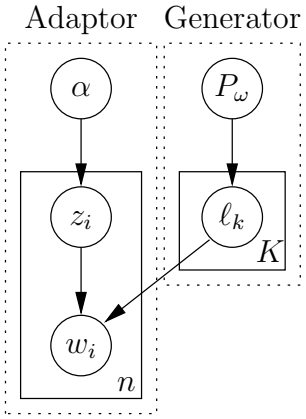

Figure 3.1: A graphical model representation of the two-stage language modeling framework. Arrows indicate dependencies between variables, and solid-line boxes indicate replicated portions of the model, with the number of replicants shown in the lower right hand corner. Variables associated with the 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.

parameters $\boldsymbol{\gamma}$. This process is the *adaptor*.

I will use the notation TwoStage(P_γ, P_ω) to refer to a two-stage model with adaptor P_γ and generator P_ω. A graphical model illustrating the dependencies between the variables in this framework is shown in Figure 3.1.

In principle, any distribution over integers could be used in Step 2 of this framework. However, the models I discuss all assume that P_γ is chosen so that the frequencies with which different integer outcomes are produced follow a power-law distribution — a few outcomes have very high probability and most outcomes occur with low probability. This is the same kind of distribution that word frequencies have been found to follow in natural language.

An important point to note is that the use of the term "lexical item" here is non-standard. Usually, a lexical item is considered to be a unique object. Here, P_ω is a discrete distribution and the lexical items are generated independently, so $\boldsymbol{\ell}$ may contain duplicate

items[1]. Usually, P_ω will be a distribution with infinite support (i.e. over an infinite number of items). In this case, for the choices of P_ω presented in this thesis, using an adaptor that produces a power-law distribution over integers will result in a power-law distribution on 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.

Given the assumption of an adaptor producing power laws, different choices for the generator model will allow different kinds of linguistic structure to be learned. For example, in Chapter 4, I show that morphological structure can be learned using a generator that produces words by choosing a stem and suffix and concatenating them together. In Chapter 5, I use a different generator to discover word boundaries in unsegmented text. Due to the task-specific nature of the generator, this chapter focuses primarily on the adaptor. I first review power-law distributions and their relationship to natural language, and then present the Chinese restaurant process, a simple statistical process that can be used as a power-law adaptor. I discuss the resulting two-stage model (using the generic lexicon generator P_ω) and show that it is equivalent to a standard Bayesian statistical model known as the Dirichlet process. I then present another possible power-law adaptor, the Pitman-Yor process. Finally, I explain the sampling methods that can be used to perform inference on the models I have discussed.

[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).

3.3 Power laws

One of the most striking statistical properties of natural language is the fact that word frequencies follow a power-law distribution, i.e.

$$P(n_w = x) \propto x^{-g}$$

where n_w is the number of times the word type w is observed in a corpus and g is some constant. This observation is often attributed to Zipf (1932), although according to Mitzenmacher (2003), there is an earlier reference in Estoup (1916). "Zipf's law", though equivalent to the above definition, is actually stated in terms of the frequency ranking of words: the probability of the kth most frequent word in a corpus is approximately proportional to k^{-c}, with $c = g - 1$ (Griffiths, 2005). Regardless of the chosen definition, the empirical probabilities of the words in a corpus appear approximately linear on a log-log plot (see Figure 3.2), a behavior that is characteristic of a power-law distribution[2].

In general, power-law distributions are produced by stochastic processes (non-independent sequences of events) in which frequent outcomes attract probability mass over time — "preferential attachment" or "rich-get-richer" processes. For example, the number of links pointing to a given web page follows 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, 2003). One widely used preferential attachment process is due to Simon (1955):

$$P(z_i = k \,|\, \mathbf{z}_{-i}) = a\frac{1}{Z} + (1-a)\frac{n_k^{(\mathbf{z}_{-i})}}{i-1} \tag{3.1}$$

where z_i is the ith outcome, Z is the number of possible values for that outcome, $\mathbf{z}_{-i} =$

[2] Although both the standard and Zipf plots of word frequencies appear linear on a log-log scale, the Zipf plot will appear non-linear for power-law distributions where $g = 1$ (Griffiths, 2005). In natural language, g is typically closer to 2 (i.e. c, the constant in Zipf's law, is close to 1).

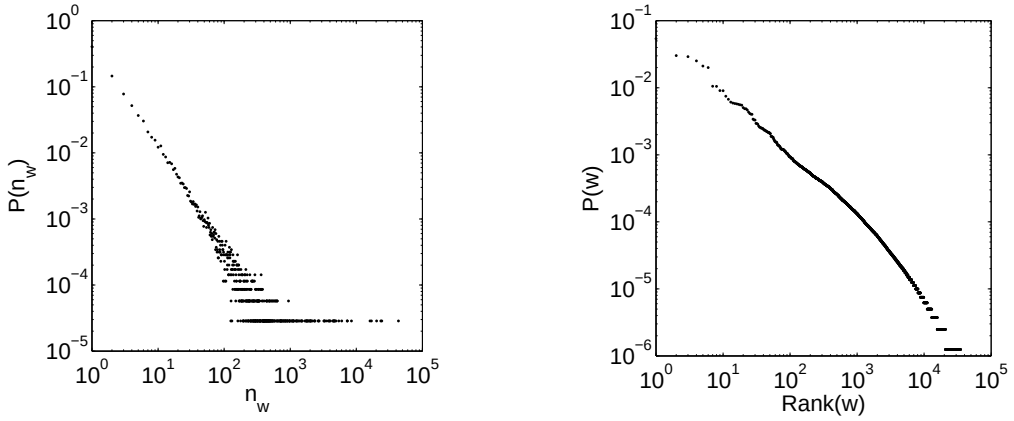

Figure 3.2: Standard plot (left) and Zipf-style plot (right) illustrating the power-law distribution of words in the Penn Wall Street Journal corpus. n_w is the number of occurrences of w and Rank(w) is the rank order of the frequency of w. The noise in the data is more apparent in the standard plot than the Zipf plot, but both are approximately linear on the log-log axes. The slope of each line corresponds to the exponent g or c.

$\{z_1 \dots z_{i-1}\}$, $n_k^{(\mathbf{z}_{-i})}$ is the number of times k occurs in $\mathbf{z}_{-i}$, and a is a parameter controlling the exponent of the power law (in particular, $g = 1/(1-a)$).

The difficulty with using this sort of process for the adaptor in a two-stage model is that different permutations of the outcomes $\mathbf{z}$ have different probabilities. For example, $P(1,1,2) = (\frac{a}{Z})(\frac{a}{Z}+1-a)(\frac{a}{Z})$ while $P(1,2,1) = (\frac{a}{Z})(\frac{a}{Z})(\frac{a}{Z}+\frac{1-a}{2})$. Of course, different orderings of the words in a text *do* have different probabilities, but those probabilities do not necessarily differ in the way that Equation 3.1 predicts. In the absence of additional information, it is more reasonable to assume that outcomes are *exchangeable*, i.e. all permutations of the set of outcomes have the same probability. In Chapter 5, I will show how to introduce more linguistically natural ordering dependencies into the two-stage framework; until then, it is simpler and less restrictive to assume that the probability of a set of words does not depend on their ordering. Note the difference from the usual language modeling

assumption of independent, identically distributed outcomes. Here, outcomes are not independent: generating a word once increases the probability of generating it again in the future.

The property of exchangeability also has practical benefits, in that it permits us to perform inference using the sampling algorithms discussed in Section 2.2.4. Recall, for example, that the Gibbs sampler requires us to compute the distribution of the ith variable conditioned on the values of the other n variables in the model. When outcomes are exchangeable, we can easily do so by treating the ith variable as if it were the last observation, and computing its distribution conditioned on the "previous" n observations.

3.4 The Chinese restaurant process

A simple stochastic process that is exchangeable and produces power laws is the Chinese restaurant process (CRP) (Aldous, 1985; Pitman, 1995; Griffiths, 2006). Imagine a restaurant containing an infinite number of tables, each with infinite seating capacity. Customers enter the restaurant and seat themselves. Each customer sits at an occupied table with probability proportional to the number of people already seated there, and at an unoccupied table with probability proportional to some constant α. That is, if z_i is the number of the table chosen by the ith customer, then

$$P(z_i = k \,|\, \mathbf{z}_{-i}) = \begin{cases} \frac{n_k^{(\mathbf{z}_{-i})}}{i-1+\alpha} & 1 \leq k \leq K(\mathbf{z}_{-i}) \\ \\ \frac{\alpha}{i-1+\alpha} & k = K(\mathbf{z}_{-i}) + 1 \end{cases} \tag{3.2}$$

where $n_k^{(\mathbf{z}_{-i})}$ is the number of customers already sitting at table k, $K(\mathbf{z}_{-i})$ 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

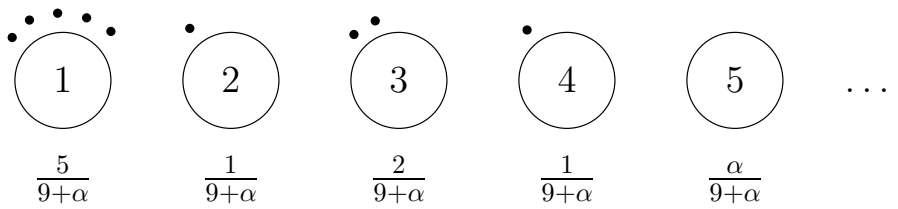

Figure 3.3: The Chinese restaurant process. Black dots indicate the seating arrangement of the first nine customers. Below each table is $P(z_{10} = k \mid \mathbf{z}_{-10})$.

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 3.3 for an illustration.

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

$$\begin{aligned} P(\mathbf{z}) &= \prod_{i=1}^{n} P(z_i \mid \mathbf{z}_{-i}) \\ &= 1 \cdot \prod_{i=2}^{n} P(z_i \mid \mathbf{z}_{-i}) \\ &= \left(\prod_{j=1}^{n-1} \frac{1}{j+\alpha} \right) \left(\alpha^{K(\mathbf{z})-1} \right) \left(\prod_{k=1}^{K(\mathbf{z})} (n_k^{(\mathbf{z})} - 1)! \right) \\ &= \frac{\Gamma(1+\alpha)}{\Gamma(n+\alpha)} \cdot \alpha^{K(\mathbf{z})-1} \cdot \prod_{k=1}^{K(\mathbf{z})} (n_k^{(\mathbf{z})} - 1)! \end{aligned} \tag{3.3}$$

where the first factor in line 3 accounts for the product of all the denominators from the factors in line 2, the second factor accounts for the numerators in the cases where a new table was introduced, and the third factor accounts for the remaining numerators[3]. It is easy to see that any reordering of the table assignments in $\mathbf{z}$ will result in the same factors in Equation 3.3, so the CRP is exchangeable. In other words, all partitions of the n customers

[3]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 3.6), which is defined only for $\alpha > 0$, and is equivalent to Equation 3.3 in that case. I use the distribution in Equation 3.3 because it is defined also for $\alpha = 0$, which is a possible parameter value in the CRP.

into groups with sizes $\{n_1^{(\mathbf{z})}, \ldots, n_{K(\mathbf{z})}^{(\mathbf{z})}\}$ are equivalent.

3.5 Generating words

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, I will continue to use the generic lexicon generator P_ω, a distribution parameterized by ω, so the full model is TwoStage(CRP(α), P_ω). This model can be viewed as a restaurant in which each table is labeled with a word produced by P_ω. 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 token may only be assigned to a table whose label matches the token. The probability of the ith word in a sequence, given the previous labels and table assignments, can be found by summing over all the tables labeled with that word:

$$\begin{aligned}
P(w_i = w \,|\, \mathbf{z}_{-i}, \boldsymbol{\ell}(\mathbf{z}_{-i}), \omega) &= \sum_{k=1}^{K(\mathbf{z}_{-i})+1} P(w_i = w \,|\, z_i = k, \boldsymbol{\ell}(\mathbf{z}_{-i}))P(z_i = k \,|\, \mathbf{z}_{-i}) \\
&= \sum_{k=1}^{K(\mathbf{z}_{-i})} P(w_i = w \,|\, z_i = k, \ell_k)P(z_i = k \,|\, \mathbf{z}_{-i}) \\
&\quad + P(w_i = w \,|\, z_i = K(\mathbf{z}_{-i}) + 1)P(z_i = K(\mathbf{z}_{-i}) + 1 \,|\, \mathbf{z}_{-i}) \\
&= \sum_{k=1}^{K(\mathbf{z}_{-i})} I(\ell_k = w)\frac{n_k^{(\mathbf{z}_{-i})}}{i - 1 + \alpha} + P_\omega(w)\frac{\alpha}{i - 1 + \alpha} \\
&= \frac{n_w^{(\mathbf{w}_{-i})} + \alpha P_\omega(w)}{i - 1 + \alpha} \qquad (3.4)
\end{aligned}$$

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 w in $\mathbf{w}_{-i}$ (i.e. the number of assignments in $\mathbf{z}_{-i}$ to tables labeled with w).

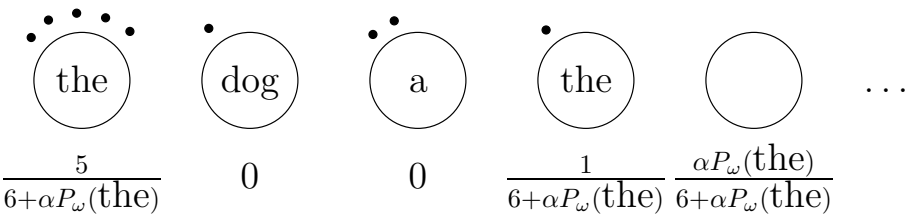

Figure 3.4: The two-stage restaurant. Each label ℓ_k is shown on table k. Black dots indicate the number of occurrences of each label in $\mathbf{w}_{-10}$. Below each table is $P(z_{10} = k \,|\, w_{10} = \text{`the'}, \mathbf{z}_{-10}, \boldsymbol{\ell}(\mathbf{z}_{-10}), \omega)$. Under this seating arrangement, $P(w_{10} = \text{`the'}) = \frac{6+\alpha P_\omega(\text{the})}{9+\alpha}$, $P(w_{10} = \text{`dog'}) = \frac{1+\alpha P_\omega(\text{dog})}{9+\alpha}$, $P(w_{10} = \text{`a'}) = \frac{2+\alpha P_\omega(\text{a})}{9+\alpha}$, and for any other word w, $P(w_{10} = w) = \frac{\alpha P_\omega(w)}{9+\alpha}$.

The probability of an entire sequence of words is

$$\begin{aligned} P(\mathbf{w} \,|\, \omega) &= \sum_{\mathbf{z},\boldsymbol{\ell}} P(\mathbf{w}, \mathbf{z}, \boldsymbol{\ell} \,|\, \omega) \\ &= \sum_{\mathbf{z},\boldsymbol{\ell}} \left(\frac{\Gamma(1+\alpha)}{\Gamma(n+\alpha)} \alpha^{K(\mathbf{z})-1} \prod_{k=1}^{K(\mathbf{z})} \left(P_\omega(\ell_k) \cdot (n_k^{(\mathbf{z})} - 1)! \right) \right) \end{aligned} \tag{3.5}$$

where $\boldsymbol{\ell}$ and $\mathbf{z}$ are constrained in the sum in the second line such that $\ell_{z_i} = w_i$ for all i.

In later chapters, it will sometimes be necessary to use the conditional distribution over table assignments for sampling. This distribution, which is illustrated in Figure 3.4, can be determined using Equation 3.4. By definition,

$$P(z_i = k \,|\, w_i = w, \mathbf{z}_{-i}, \boldsymbol{\ell}(\mathbf{z}_{-i}), \omega) = \frac{P(z_i = k, w_i = w \,|\, \mathbf{z}_{-i}, \boldsymbol{\ell}(\mathbf{z}_{-i}), \omega)}{P(w_i = w \,|\, \mathbf{z}_{-i}, \boldsymbol{\ell}(\mathbf{z}_{-i}), \omega)} \tag{3.6}$$

where

$$P(z_i = k, w_i = w \,|\, \mathbf{z}_{-i}, \boldsymbol{\ell}(\mathbf{z}_{-i}), \omega) = \begin{cases} \frac{n_k^{(\mathbf{z}_{-i})}}{i-1+\alpha} \cdot I(\ell_k = w) & 1 \le k \le K(\mathbf{z}_{-i}) \\ \frac{\alpha}{i-1+\alpha} \cdot P_\omega(w) & k = K(\mathbf{z}_{-i}) + 1 \end{cases} \tag{3.7}$$

Dividing through by Equation 3.4 yields

$$P(z_i = k \,|\, w_i = w, \mathbf{z}_{-i}, \boldsymbol{\ell}(\mathbf{z}_{-i}), \omega) = \begin{cases} \frac{n_k^{(\mathbf{z}_{-i})}}{n_w^{(\mathbf{w}_{-i})} + \alpha P_\omega(w)} \cdot I(\ell_k = w) & 1 \le k \le K(\mathbf{z}_{-i}) \\ \frac{\alpha}{n_w^{(\mathbf{w}_{-i})} + \alpha P_\omega(w)} \cdot P_\omega(w) & k = K(\mathbf{z}_{-i}) + 1 \end{cases} \tag{3.8}$$

The conditional distribution over words given in Equation 3.4 leads to an alternative way of viewing the two-stage framework, as a cache model. Under this view, each word is generated in one of two ways: from a cache of previously occurring words (with probability $\frac{1-\alpha}{n+\alpha}$ if we use the CRP adaptor) or as a novel word (with probability $\frac{\alpha}{n+\alpha}$). Words from the cache are chosen with probability proportional to the number of times they have occurred in the cache. Novel words are chosen according to the probability distribution of the lexicon generator (which means that, strictly speaking, they are not always "novel", since the generator may produce duplicates). This interpretation clarifies the significance of the parameters α and P_ω. Prior expectations regarding the probability of encountering a novel word are reflected in the value of α, so lower values of α lead to sparser solutions. Prior expectations about the relative probabilities of different novel words are reflected in P_ω, so the choice of generator determines the kinds of words that are likely to be inferred from the data.

In the case where the generator is a K-dimensional multinomial distribution with parameters $\boldsymbol{\omega}$, $P_\omega(w_i = k) = \omega_k$, so Equation 3.4 becomes equivalent to Equation 2.9, indicating that the cache model reduces to a Dirichlet($\alpha\boldsymbol{\omega}$)-multinomial model. However, 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 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.

Models whose complexity grows with the size of the data are referred to in the statistical literature as *nonparametric*. In the following section I show that the two-stage model with

generator P_ω and CRP adaptor is equivalent to a standard nonparametric statistical model known as the Dirichlet process.

3.6 The Dirichlet process

In Chapter 2, I discussed the Dirichlet distribution, which can be used as a prior over multinomials. Each draw from a K-dimensional Dirichlet distribution returns a set of parameters $\boldsymbol{\theta}$ for a K-dimensional multinomial – a distribution over a finite set of outcomes. In a symmetric Dirichlet distribution, the hyperparameter α determines the variance in the values of $\boldsymbol{\theta}$. The Dirichlet process is like an infinite-dimensional symmetric Dirichlet distribution: each draw returns a distribution G over a countably infinite set of outcomes. The Dirichlet process has two parameters: G_0, a distribution which determines the probability that any particular outcome will be in the support of G, and α, which determines the variance in the probabilities of those outcomes under G. In general, G_0 may have uncountably infinite support, but in the models presented here, G_0 is a distribution over words, a countably infinite set.

A Dirichlet process (DP) unigram language model can be characterized as follows:

$$\begin{aligned} w_i \,|\, G &\sim G \\ G \,|\, \alpha, P_\omega &\sim \mathrm{DP}(\alpha, P_\omega) \end{aligned} \tag{3.9}$$

The corresponding graphical model can be seen in Figure 3.5. This formulation makes the distribution G sampled from the DP explicit. However, it is possible to integrate over G to obtain the conditional distribution $P(w_i \,|\, \mathbf{w}_{-i}, \alpha, P_\omega)$ just as we integrated over $\boldsymbol{\theta}$ in the Dirichlet-multinomial model (Equation 2.9). This integration results in the following

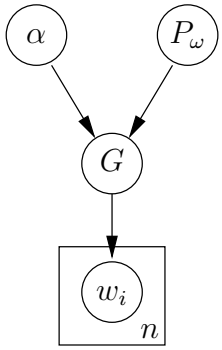

Figure 3.5: A graphical representation of the Dirichlet process language model.

conditional distribution (Blackwell and MacQueen, 1973):

$$w_i \,|\, \mathbf{w}_{-i}, \alpha, P_\omega \;\sim\; \frac{1}{i-1+\alpha}\sum_{j=1}^{i-1}\delta(w_j) + \frac{\alpha}{i-1+\alpha}P_\omega \tag{3.10}$$

where $\delta(w_j)$ is a distribution with all its mass at w_j. Rewriting this distribution as a probability mass function makes clear the equivalence between TwoStage(CRP(α), P_ω) and DP(α, P_ω):

$$\begin{aligned} P(w_i = w \,|\, \mathbf{w}_{-i}, \alpha, P_\omega) &= \frac{1}{i-1+\alpha}\sum_{j=1}^{i-1} I(w_j = w) + \frac{\alpha}{i-1+\alpha}P_\omega(w) \\ &= \frac{n_w^{(\mathbf{w}_{-i})} + \alpha P_\omega(w)}{i-1+\alpha} \end{aligned} \tag{3.11}$$

The sort of language model I have just described is an unusual application of the Dirichlet process. The Dirichlet process is more typically used as a prior in infinite mixture models (Lo, 1984; Escobar and West, 1995; Neal, 2000), where each table represents a mixture component, and the data points at each table are assumed to be generated from some parameterized distribution. Technically, the DP language model can be viewed as a mixture model where each table is parameterized by its label ℓ_k, and $P(w_i \,|\, \ell_{z_i}) = I(w_i = \ell_{z_i})$, so every data point in a single mixture component is identical. Previous applications of DP mixture models use more complex distributions to permit variation in the data within

components. Usually Gaussians are used for continuous data (Rasmussen, 2000; Wood et al., 2006) and multinomials for discrete data (Blei et al., 2002; Navarro et al., 2006). DPs and their extensions have been used for language-related tasks (Blei et al., 2004), but this work has focused on modeling the overall semantic content of text rather than fine-grained linguistic structure. Models that are similar to the Dirichlet process model used here have been described before (MacKay and Peto, 1994; Elkan, 2006; Madsen et al., 2005), but these assume a finite number of parameters (i.e. they are based on the Dirichlet-multinomial distribution and can be described as TwoStage(CRP(α), Multinomial(ω))). To my knowledge, the language models described in this document are the first to apply DPs to the problem of low-level language acquisition.

3.7 The Pitman-Yor process

The models I develop in Chapter 5 of this thesis use the CRP adaptor. In Chapter 4, however, I describe a model that uses a different adaptor, the Pitman-Yor process (Pitman, 1995; Pitman and Yor, 1997; Ishwaran and James, 2003). The Pitman-Yor process is a generalization of the CRP, defined as

$$P(z_i = k \,|\, \mathbf{z}_{-i}) = \begin{cases} \frac{n_k^{(\mathbf{z}_{-i})} - a}{i - 1 + b} & 1 \leq k \leq K(\mathbf{z}_{-i}) \\ \frac{K(\mathbf{z}_{-i})a + b}{i - 1 + b} & k = K(\mathbf{z}_{-i}) + 1 \end{cases} \tag{3.12}$$

where $0 \leq a < 1$ and $b \geq 0$ 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. Like the CRP, the Pitman-Yor process 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 is equal to $1 + a$ (Griffiths, 2006).

Under the Pitman-Yor process, the probability of a particular seating arrangement $\mathbf{z}$ is

$$\begin{aligned} P(\mathbf{z}) &= \prod_{i=1}^{n} P(z_i \,|\, \mathbf{z}_{-i}) \\ &= 1 \cdot \prod_{i=2}^{n} P(z_i \,|\, \mathbf{z}_{-i}) \\ &= \left(\prod_{j=1}^{n-1} \frac{1}{j+b} \right) \left(\prod_{k=1}^{K(\mathbf{z})-1} (ka+b) \right) \left(\prod_{k=1}^{K(\mathbf{z})} \prod_{j=1}^{n_k^{(\mathbf{z})}-1} (j-a) \right) \\ &= \frac{\Gamma(1+b)}{\Gamma(n+b)} \prod_{k=1}^{K(\mathbf{z})-1} (ka+b) \prod_{k=1}^{K(\mathbf{z})} \frac{\Gamma(n_k^{(\mathbf{z})}-a)}{\Gamma(1-a)} \end{aligned} \tag{3.13}$$

The derivation is similar to the derivation for the CRP. Further discussion of the Pitman-Yor process and the significance of the parameters a and b will be deferred to Chapter 4.

3.8 Conclusion

In this chapter, I have presented a two-stage framework for language modeling that is based on nonparametric Bayesian statistical techniques. Models in this framework consist of a lexicon generator, which can be modified to suit the needs of specific tasks, and an adaptor, which produces a power-law distribution on the frequencies of items produced by the generator. I have discussed two different adaptors, the Chinese restaurant process and the Pitman-Yor process. Both of these adaptors include parameters that control the sparseness of the solutions that will be found, and both are exchangeable, so they can be used with the sampling procedures described in Section 2.2.4.

The two-stage framework I have described is both general and flexible. I have shown

that it subsumes at least two widely used Bayesian model families: the Dirichlet($\alpha\omega$)-multinomial can be viewed as TwoStage(CRP(α), Multinomial(ω)), and DP(α, P_ω) is equivalent to TwoStage(CRP(α), P_ω). The following chapters illustrate how the modular nature of the framework allows models to be developed and modified more easily than in previous approaches. This flexibility will permit me to address a broad range of questions regarding the kinds of information that are available and useful for linguistic acquisition, and how these kinds of information can be integrated to improve learning.

Chapter 4

Morphology

4.1 Introduction

In the previous chapters, I have laid out an argument in favor of taking a statistical (in particular, Bayesian) approach to modeling language acquisition, and have presented a general framework for doing so. In this chapter, I apply that framework to the particular task of learning simple morphological structure (stems and suffixes) in order to address the question of what kinds of statistics are most useful for this task. Most proponents of statistical learning implicitly assume that the statistics that are relevant for learning are those of the input data; i.e. what matters is the frequency with which different lingustic forms (or sets of forms) are observed in natural speech (Saffran et al., 1996b; Roark and Demuth, 2000; Boersma and Hayes, 2001; Mintz et al., 2002). This view contrasts with the approach of most nativist learning theories, where learning is usually assumed to be triggered by individual linguistic instances, and has little (if anything) to do with observed frequencies (Randall, 1992; Gibson and Wexler, 1994; Dresher and Kaye, 1990; Tesar and Smolensky, 2000). There is also a third possibility, which has received less attention than the first two. In many cases, there are statistical patterns that can be found amongst the set of lingustic types that are found in the input. That is, if we consider the set of unique forms at some level of structure (syllables, words, phrases), the statistics of this set may be informative about the language in question. Notice that the kinds of generalizations that may be drawn by considering statistical regularities amongst lexical items (counting each unique item only once) may be different from the generalizations drawn by standard corpus-based models (which consider also the frequency with which each lexical item appears in the corpus.) Although some previous work has made use of statistics gathered from the lexicon, there has not been much attention devoted to determining whether or when these

kinds of statistics might be helpful, or how they might interact with corpus frequencies during learning.

The remainder of this chapter is structured as follows. I first review the literature on learning from types (unique lexical items) and tokens (instances of those items in the input), which suggests that sublexical generalizations (including morphology) are informed by type frequencies, while token frequencies are useful for learning many other kinds of linguistic structure. I discuss previous computational approaches to morphological analysis and then describe my own morphological learner, which is based on the two-stage language modeling framework from Chapter 3. I explain how the parameters of the two-stage model determine whether it makes morphological generalizations based on corpus frequencies, damped corpus frequencies, or word types. I present experiments explicitly comparing these possibilities, and provide evidence that morphological structure is learned better by attending to the statistics of the lexicon rather than of the input corpus. In light of these results, I argue that the two-stage framework provides a natural way to simultaneously learn from types (as seems to be necessary for morphological acquisition) and tokens (as in, e.g., word segmentation).

4.2 Previous work

4.2.1 Types and tokens in learning

4.2.1.1 Learning from types without statistics

Traditionally, non-statistical approaches to learning have assumed that learning is based on types. That is, to the extent that the learner's input influences the final state of the grammar, it does so by the presence or absence of certain lingustic forms, rather than by

their frequency. There are two basic mechanisms that have been proposed to explain how observing particular constructions could affect the learning process. Under a view in which language acquisition consists of parameter setting (Chomsky, 1981), certain constructions may provide evidence as to the proper setting of particular parameters (Randall, 1992; Dresher and Kaye, 1990; Dresher, 1999). When one of these constructions (or "cues", in the terminology of Dresher and Kaye (1990)) is observed, it causes the learner to set the value of the associated parameter appropriately. According to the theory of cue-based learning articulated by Dresher (1999), once a parameter value has been set in this way, no further evidence can modify its value. The types of constructions that serve as cues and the order in which those cues may be used for parameter setting is specified by Universal Grammar. Notice that cue-based learning does not assume that the learner is attempting to match the input in any way; in fact, it is possible that the correct setting of a parameter could temporarily cause the learner to make more overt mistakes.

In contrast to cue-based learning, other non-statistical theories of language acquisition do assume that the learning process is driven by mismatches between the learner's grammar and the observed input (Gibson and Wexler, 1994; Tesar and Smolensky, 2000). Probably the most well-known and well-studied learning algorithms are various incarnations of the Constraint Demotion Algorithm (CDA) (Tesar and Smolensky, 1996; Tesar, 1998; Tesar and Smolensky, 2000) for learning in Optimality Theory (Prince and Smolensky, 1993). In essence, the CDA works by noting cases where an observed form is considered ungrammatical by the learner's current grammar (constraint ranking). When such a case occurs, the learner reranks some of the constraints so as to make the observed form optimal.

Although the CDA, cue-based learning, and other non-statistical methods have been

shown to learn correctly on constructed examples, they all share the common drawback of being highly sensitive to noisy or variable input data. Data with multiple surface realizations of the same lexical item can prevent the CDA from converging (Boersma and Hayes, 2001; Boersma, 2003); noisy input could cause an irrecoverable parameter setting error in a cue-based learner. Since my purpose is to develop a model of learning that uses naturalistic input, the kind of type-based learning exemplified by these models does not seem appropriate, since it is not clear how to apply it successfully to potentially noisy input.

4.2.1.2 Learning from tokens

Not surprisingly, most work on statistical learning has focused on the most obvious source of statistics – the raw linguistic input received by the learner. For early acquisition tasks such as word segmentation, this is the only source of statistics that could plausibly be of use, since lexical representations have not yet been built. Much of the behavioral research on statistical learning has indeed focused on word segmentation tasks, showing that humans are able to extract word-like units from continuous speech by attending to differences in the transitional probabilities between syllables or phonemes[1] (Saffran et al., 1996b; Saffran et al., 1996a; Newport and Aslin, 2004). In these experiments, artificial languages must be used in order to create stimuli with appropriate statistical properties that are simple enough to learn in the laboratory. There have also been some experiments suggesting that token frequencies play a role in the kinds of representations that are stored in the mental lexicon. For example, the frequency of a morphologically complex form relative to its base form seems to affect how morphologically transparent ("decomposable") the derived form

[1] The term *transitional probability* is typically used in this community rather than the standard *conditional probability* to refer to $P(x|y)$.

is judged to be (Hay, 2001).

Computational modeling makes it possible to explore statistical learning of a wider variety of linguistic information from more complex input. Many different connectionist simulations have been performed using full corpora or examples extracted from corpora (with the corresponding frequencies) to train neural networks. Tasks learned successfully have ranged from word segmentation (Christiansen et al., 1998) to prepositional phrase attachment and verb subcategorization (Elman, 2003). The Gradual Learning Algorithm (Boersma, 1997; Boersma, 1998), an algorithm for learning statistical Optimality Theory grammars, has been shown to correctly learn various phonological grammars using input example frequencies drawn from corpora (Boersma and Hayes, 2001), and to accurately predict the acquisition order of syllable shapes in Dutch (Boersma and Levelt, 1999).

Natural language processing software is another place where the use of token frequencies is widespread for a large range of tasks such as tagging (Merialdo, 1994; Ratnaparkhi, 1996), parsing (Charniak, 2000; Collins, 1999), machine translation (Brown et al., 1993; Och, 2005), and speech recognition (Jelinek, 1997). Supervised systems for morphological analysis are regularly trained on large corpora (Hajič and Hladká, 1998; Hakkani-Tür et al., 2000), and token-based learning has been used for semi-supervised and unsupervised morphological learning as well (Yarowsky and Wicentowski, 2000; Wicentowski, 2004; Creutz and Lagus, 2005; Baroni, 2003). The few tasks for which raw token frequencies are generally not used are semantic in nature – information retrieval, where token frequency is downweighted to emphasize infrequent but important content words, is a prime example (Salton and Buckley, 1988; Baeza-Yates and Ribeiro-Neto, 1999).

Overall, the success of token-based approaches to statistical language learning is strong

evidence that there is valuable information to be found in raw token frequencies. However, it does not rule out the possibility that statistical properties of the lexicon might also be useful for learning certain kinds of linguistic information, perhaps even more useful than the statistical properties of the input.

4.2.1.3 Learning from the statistics of types

Compared to the body of work studying the effects of token statistics on learning, there has been relatively little attention devoted to the question of statistics in the lexicon. Most cognitive work in this area has focused on phonological generalizations (Pierrehumbert, 2001; Pierrehumbert, 2003) and their effect on word recognition (Luce and Pisoni, 1998). There is strong evidence that the time it takes to recognize a word is affected not only by that word's frequency, but also by the number of phonologically similar words in the lexicon, and by the frequencies of those words (Luce and Pisoni, 1998; Vitevitch and Luce, 1999; Luce et al., 2000). Judgments of phonotactic well-formedness also depend on both lexical and token statistics (Bailey and Hahn, 2001). One of the few researchers to discuss morphological generalization is Bybee (2001), who provides some evidence that morphological productivity is correlated with type frequency, and may even be inversely correlated with token frequency.

In contrast to these behavioral researchers, most researchers developing computational models have either ignored the possibility of learning from lexical statistics, or have implicitly assumed this sort of learning without actually justifying or explaining their decision. For example, many connectionist simulations of morphological or word learning use lists of inflected lexical items as input to the learner, but never discuss the implications of this decision[2] (Plaut and Gonnerman, 2000; Regier et al., 2001). In particular, there is no attempt

[2]It is interesting to note that in the groundbreaking work of Rumelhart and McClelland (1986), a neural

to explain whether or how lexical statistics interact with the statistics of the input. This disregard of the potential importance of input frequencies seems strange and problematic, especially considering the connectionists' usual focus on input statistics. In addition, it is not clear how to reconcile training from lexical types with the standard connectionist view that lexical types do not even exist in the traditional sense – each input token of the same "type" will have a slightly different internal representation, so types only exist in the sense that they can be induced by clustering similar representations (Elman, 2004).

The use of word lists for training morphological analyzers is not limited to connectionist models. Structured statistical approaches have also been trained on lists of lexical items, with good results (Snover and Brent, 2003; Monson et al., 2004). Unfortunately, the researchers implementing these systems have made no more effort than the connectionists to explain why (or even whether) types are the appropriate form of input for learning morphological information.

The only work I am aware of on computational morphology that explicitly addresses the question of learning from types or tokens is that of Albright and Hayes (2003). In this work, the authors develop and test a model of regular and irregular morphology based on statistical rules. They mention briefly that the model was tested using both types and tokens as input, and that the model trained on types more closely matches human performance. No results were provided for the model trained on tokens, however, which makes it difficult to evaluate how different those results were, and in what way.

network was trained on lists of verbs, but the most frequent verbs were presented first in an initial round of training, and then again along with the rest of the verbs. Although this regime does not accurately match input frequencies, it does suggest that the authors believed that token frequencies were a relevant (if not the relevant) source of information in morphological learning.

4.2.1.4 Summary

To summarize, nearly all the work on language learning using statistical methods has assumed that generalizations should be based on token frequencies in the input data. A few researchers have suggested that generalizations about sublexical structure (morphology and phonology) are based on the statistics of lexical types. However, except in the case of modeling adult word recognition, almost no attempt has been made to assess the relative importance of lexical versus token statistics, to determine the qualitative or quantitative differences in learning that might result from using one or the other, or to propose a unified approach that can make use of both sorts of statistics to learn different kinds of information. These are exactly the issues I plan to address in this chapter, using a model developed within the framework presented in Chapter 3. Before presenting that model, however, I briefly review previous computational work in the area of morphology.

4.2.2 Approaches to computational morphology

4.2.2.1 Hand-built and supervised approaches

There are several reasons why researchers have tried to build morphological analysis systems. In some cases, the focus is on engineering: for highly inflected languages such as Arabic, Czech, and Turkish, it can be helpful or even necessary to perform some morphological analysis before applying standard natural language processing tools, simply to avoid problems with data sparseness (Larkey et al., 2002; Hajič and Hladká, 1998; Hakkani-Tür et al., 2000). The most basic tools for morphological analysis are hand-built "stemmers" for information retrieval, which essentially just strip off inflectional and some derivational morphology, leaving the more semantically contentful stem form (Porter, 1980; Popovič

and Willett, 1992; Kraaij and Pohlmann, 1996). Other hand-built systems use finite-state techniques to produce complete linguistic analyses, transforming each surface form into a sequence of morphemes (Koskenniemi, 1983; Karttunen et al., 1992; Beesley and Karttunen, 2003). These systems can be useful tools for linguists, but because they frequently produce several possible analyses for each word, they are often not sufficient for morphological pre-processing in more complex systems. A number of researchers have therefore developed statistical systems trained on hand-annotated corpora that can be combined with hand-built systems and/or lexical resources to disambiguate between possible morphological analyses (Hajič and Hladká, 1998; Hajič, 2000; Hakkani-Tür et al., 2000; Lee et al., 2003).

In contrast to these tools for language processing or lingustic analysis are systems built to investigate questions of language learning in humans and machines. Many of these systems, like those above, are trained in a supervised manner; the difference is usually in an emphasis on generating correct inflected forms rather than analyzing the input. Rumelhart and McClelland (1986) created an early system intended to show that rule-like behavior could be produced by a neural network. Albright and Hayes' later work with a statistical rule-based model (2002; 2003) suggests that their approach may provide a better explanation of human behavioral results.

Although these kinds of supervised systems can provide some insight into the possible mechanisms involved in morphological acquisition, it is probably more informative to examine the existing work on unsupervised and minimally supervised morphological learning, as I do in the following sections.

4.2.2.2 Maximum-likelihood approaches

Despite the problems with maximum-likelihood estimation discussed in Chapter 2, a few researchers have still experimented with using it for morphological segmentation, i.e. splitting words into their constituent morphemes. Creutz and Lagus (2002), for example, present a segmentation procedure for Finnish that includes a maximum-likelihood step embedded in a larger algorithm. However, as I will show in Section 4.3, the true maximum-likelihood solution for the kind of model Creutz and Lagus propose actually contains no morpheme boundaries at all. Their algorithm works only because it limits the number of parameters in the model in an ad hoc way. In fact, the same argument holds true for a later system presented by the same researchers, which includes a more sophisticated generative model (Creutz and Lagus, 2004). The later model incorporates morphotactics (which are important in agglutinative languages such as Finnish) through the use of an HMM to model prefix, suffix, and stem classes.

4.2.2.3 MAP approaches

Most unsupervised morphological segmentation systems have used model-based approaches incorporating MDL or other priors favoring sparse solutions. Probably the most general of these is Linguistica (Goldsmith, 2001b; Goldsmith, 2001a), an MDL-based system that analyzes words into a stem plus one or more affixes (both prefixes and suffixes are considered). There are also extensions to the basic system for detecting allomorphic variation (Goldsmith, In press) and incorporating some syntactic context (Hu et al., 2005). Snover and Brent (2003) present a different approach, using an explicitly formulated prior to segment words into a single stem and suffix. The approach taken by Creutz and Lagus (2005)

is similar, except that their model generates words using the HMM from their earlier work (Creutz and Lagus, 2004).

Although none of these systems is general enough to yield good results on all languages, each of them performs well on at least some languages. Collectively, they suggest that probabilistic learning of morphology is possible using an appropriate model and prior, and that incorporating additional sources of information such as phonology or contextual information can improve performance. On the other hand, because all of these systems use heuristic search procedures to find an approximate MAP solution, it is difficult to know how close the published results are to the true MAP solution. We must therefore be wary of drawing too strong conclusions about learning from the results of these systems.

4.2.2.4 Other approaches

A final set of morphological learning systems uses statistics, but no explicit probabilistic models. These include one "minimally supervised" learning system (Yarowsky and Wicentowski, 2000; Wicentowski, 2004) and at least two unsupervised systems (Schone and Jurafsky, 2001; Baroni et al., 2002). All of these are intended to perform morphological alignment (i.e. identify related forms) rather than segmentation, and share the property of incorporating many more types of information than the model-based systems described above.[3] For example, Wicentowski (2004) assesses the probability that two forms are morphologically related using frequency similarity, contextual similarity, orthographic similarity, and the frequency of the particular orthographic transformation required to change one form to the other. The other systems use several of these forms of information as well, with Schone

[3] Neuvel and Fulop (2002) present a system that also produces morphological alignments using multiple sources of information, but in this case the information (the syntactic and morphological function of each word) must be provided as input, and the system uses only very simple count information rather than the more sophisticated statistics of the other systems discussed here.

and Jurafsky (2001) distinguishing between contextual similarity within a local (syntactic) window and broad (semantic) area.

Although the lack of explicit probabilistic models in these systems makes them inappropriate for detailed investigations of the kind I am interested in, they do suggest a number of sources of information that might be useful to incorporate into a probabilistic model. In particular, the use of frequency similarity between inflected forms as a potential cue to morphological relatedness suggests that perhaps token frequency is indeed important in learning morphology. Contextual similarity is also a token-based property, while orthographic similarity is a property of lexical items, and orthographic transformation probabilities could be computed based on either types or tokens. In short, there seemed to be useful cues available in both the type and token domains, which makes the question of how to unify these domains only more important.

4.2.2.5 Summary

There have been many different approaches to computational morphology. The most sophisticated hand-built and supervised systems are able to analyze even complex morphophonological interactions and non-concatenative phenomena, but they are usually specialized to process particular languages and require a great deal of human knowledge for development. Most unsupervised systems are only able to segment surface forms into a stem and one or (in some cases) more affixes, although some are designed to include the possibility of phonological/orthographic processes at morpheme boundaries or even within the stem. Successful model-based systems generally include a prior favoring sparse solutions and rely on heuristic search procedures. Evidence from these and other successful systems indicates that useful statistical cues to morphological structure can be found both in the lexicon and

by examining the full set of input data.

In the remainder of this chapter, I first show why, using the kind of generative models that are standard in this domain, maximum-likelihood estimation fails as a method for segmenting words into morphemes. I then present my own work on morphological segmentation, which uses the two-stage modeling framework developed in Chapter 3 to discover the stems and suffixes of English verbs. While the model's assumption of a single stem and suffix is more limited than some other approaches, it has two novel advantages. First, inference can be performed using a Gibbs sampler rather than heuristic methods. Second, the use of the two-stage framework provides a principled way to downweight the frequencies of input forms to any degree, yielding type-based learning in the limiting case. Using this property, I examine the differences in learning morphology from types and tokens, and conclude that, with the kinds of information included in my model, type-based learning is more successful.

4.3 Failure of maximum-likelihood estimation

In this section, I show why proper maximum-likelihood estimation with the kinds of models used in the papers described above does not lead to a successful solution to the problem of morphological segmentation. In fact, the maximum-likelihood solution for all these models is the solution with no boundaries at all. To see why, consider a simplified version of the model described in Creutz and Lagus (2002), where each word w consists of a single stem and (possibly empty) suffix. Let $x_1 \dots x_m$ be the phonemes or characters in w, and

$$P(A(w) = (x_{1,i-1}, x_{i,m})) = P_{stem}(x_{1,i-1}) P_{suffix}(x_{i,m}) \tag{4.1}$$

where $A(w)$ is a random variable representing the morphological analysis of w (a stem/suffix pair), $1 < i \leq m$, and $x_{i,j}$ is shorthand for $x_i \ldots x_j$[4]. If we assume every word has an empty suffix (i.e. $P_{suffix}(\varepsilon) = 1$) and that $P_{stem}(x_{1m})$ equals the empirical probability of the word x_{1m}, we obtain a model in which $P(w) = P_{stem}(w)$, and the distribution over words matches the empirical distribution in the corpus exactly. Since Equation 4.1 assumes that the probability of a suffix is independent of the stem (an approximation that will be incorrect for all but trivial corpora), any other analysis of the corpus will assign a distribution that differs from the empirical distribution. However, it is easy to show that choosing the maximum-likelihood parameter values for a model is equivalent to choosing the values yielding the smallest KL divergence[5] from the empirical distribution of the data[6]. The empty-suffix solution always has zero KL divergence from the empirical distribution, and is therefore a maximum-likelihood solution.

Although my analysis uses a particular very simple generative model, the same kind of argument holds for any model in which a) there exists a solution with no boundaries that matches the empirical distribution in the corpus exactly, and b) independence assumptions cause an imperfect match between the empirical distribution and any solution containing boundaries. This includes the morphological segmentation models in Creutz and Lagus (2002; 2004), as well as the word segmentation models described in Venkataraman (2001) (see Chapter 5). The only reason these systems work in practice is because they employ search procedures that do not explore the entire space described by their models, limiting

[4]The model in Equation 4.1 is also discussed in Goldsmith (2001b). The actual model used in Creutz and Lagus (2002) allows multiple morphemes per word.

[5]The Kullback-Leibler (KL) divergence of a discrete probability distribution Q from another discrete distribution P, is defined as $D(P||Q) = \sum_y P(y) \log \frac{P(y)}{Q(y)}$. It yields a measure of how different the two probability distributions are.

[6]See e.g. http//www.cs.berkeley.edu/∼jordan/courses/281A-fall04/lectures/lec-10-7.pdf

the number of parameters under consideration at any given time. While this can lead to successful results in practice, it is undesirable if we wish to examine the effects of changing the underlying model, as we do here, since those effects can be obscured by the search procedure.

At this point, it is natural to ask whether simply adding a Bayesian prior favoring sparse solutions would be sufficient to acquire morphological structure from a corpus. My own preliminary experiments with this approach yielded poor results, leading me to explore more fully the differences between type-based and token-based learning. Since the model I used for these experiments can simulate the token-based Bayesian model (given appropriate parameter settings), I will delay reporting these results until Section 4.6.1.3.

4.4 Two-stage model

To examine the question of whether word types or word tokens are more useful for learning morphology, I use a TwoStage(PY$(a, b), P_\mu$) model. The Pitman-Yor adaptor was described in Chapter 3; below I discuss its relevance to the question of types vs. tokens. I then present the morphological lexicon generator P_μ and the full two-stage model.

4.4.1 Types and tokens

Recall the definition of the Pitman-Yor process from Section 3.7, which gives the probability of seating the ith customer at table k given the previous seating arrangement:

$$P(z_i = k \,|\, \mathbf{z}_{-i}) = \begin{cases} \frac{n_k^{(\mathbf{z}_{-i})} - a}{i-1+b} & 1 \leq k \leq K(\mathbf{z}_{-i}) \\ \\ \frac{K(\mathbf{z}_{-i})a+b}{i-1+b} & k = K(\mathbf{z}_{-i}) + 1 \end{cases} \tag{4.2}$$

for $0 \leq a < 1$ and $b \geq 1$, with $P(z_1 = 1) = 1$ always. If we create a two-stage model using a Pitman-Yor adaptor and a generator P_μ with parameters μ, the joint distribution on a sequence of words $\mathbf{w}$ will be

$$\begin{aligned} P(\mathbf{w} \,|\, \mu) &= \sum_{\mathbf{z},\boldsymbol{\ell}} P(\mathbf{w}, \mathbf{z}, \boldsymbol{\ell} \,|\, \mu) \\ &= \sum_{\mathbf{z},\boldsymbol{\ell}} \left(\prod_{j=1}^{n-1} \frac{1}{j+b} \right) \left(\prod_{k=1}^{K(\mathbf{z})} P_\mu(\ell_k) \prod_{k=2}^{K(\mathbf{z})} ((k-1)a + b) \right) \left(\prod_{k=1}^{K(\mathbf{z})} \prod_{j=1}^{n_k^{(\mathbf{z})}-1} (j-a) \right) \\ &= \sum_{\mathbf{z},\boldsymbol{\ell}} \frac{\Gamma(1+b)}{\Gamma(n+b)} \prod_{k=1}^{K(\mathbf{z})} P_\mu(\ell_k) \prod_{k=1}^{K(\mathbf{z})-1} (ka+b) \frac{\Gamma(n_k^{(\mathbf{z})} - a)}{\Gamma(1-a)} \end{aligned} \tag{4.3}$$

where the sums in lines 2 and 3 are over only those $\boldsymbol{\ell}$ and $\mathbf{z}$ such that $\ell_{z_i} = w_i$ for all i. The three parenthesized products in the second line correspond respectively to the factors introduced by the Pitman-Yor denominators, by the numerators for new tables (with new labels), and by the numerators for additional tokens placed on old tables.

I will be using this model to learn the parameters of the generator, which (as we will see) will provide morphological analyses of the words in the input data. To see how the Pitman-Yor process can be used to address the question of learning from types or tokens, assume that $b = 0$ and consider how different values of a affect estimates of μ. When $b = 0$, the probability of $\mathbf{w}$ reduces to

$$\begin{aligned} P(\mathbf{w} \,|\, \mu) &= \sum_{\mathbf{z},\boldsymbol{\ell}} \frac{\Gamma(1)}{\Gamma(n)} \cdot a^{K(\mathbf{z})-1} (K(\mathbf{z}) - 1)! \cdot \prod_{k=1}^{K(\mathbf{z})} P_\mu(\ell_k) \frac{\Gamma(n_k^{(\mathbf{z})} - a)}{\Gamma(1-a)} \\ &= \sum_{\mathbf{z},\boldsymbol{\ell}} \frac{(K(\mathbf{z}) - 1)!}{(n-1)!} \cdot a^{K(\mathbf{z})-1} \cdot \prod_{k=1}^{K(\mathbf{z})} P_\mu(\ell_k) \frac{\Gamma(n_k^{(\mathbf{z})} - a)}{\Gamma(1-a)} \end{aligned} \tag{4.4}$$

In this case, the limiting cases of $a \to 0$ and $a \to 1$ lead to estimates of μ based on types and tokens respectively, as I will now show.

When $a \to 0$, the $a^{K(\mathbf{z})-1}$ term in Equation 4.4 causes the sum over $(\mathbf{z}, \boldsymbol{\ell})$ to be dominated by the partition of customers with the smallest value of $K(\mathbf{z})$, i.e. 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 $P_\mu(w_k)$, per word type w_k. Therefore estimates of μ 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_μ is responsible for generating all the word tokens in the data. Estimates of μ will consequently be based on word tokens.

4.4.2 A generator for morphology

The lexicon generator I use to model morphological structure assumes that each word consists of a single stem and (possibly empty) suffix and belongs to some morphological class. The joint probability of generating a particular class c, stem t, and suffix f is defined as

$$P_\mu(c, t, f) = P(c)P(t \mid c)P(f \mid c) \tag{4.5}$$

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. Up to now, I have been assuming that the generator in a two-stage model is a distribution over words, not analyses of words, as above. However, in this model, it is the analyses themselves that are produced by the generator. I will therefore distinguish between the label ℓ_k on each table, which I continue to assume is a string, and the analysis of that label $A(\ell_k)$, which is an object produced by the generator. Since it will be necessary to refer to the probability of a word or label itself (regardless of its analysis), I will abuse notation somewhat and use

$P_\mu(w)$ to refer to this probability as well. $P_\mu(w)$ is defined as

$$P_\mu(w) = \sum_{(c,t,f)} I(w = t.f)P(c)P(t \mid c)P(f \mid c) \tag{4.6}$$

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.

The generator model for morphology is intended to encode several 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 past and present tenses are each relatively common, then if we see a word very frequently in the past tense, we would expect to see it very frequently in the present tense as well (Yarowsky and Wicentowski, 2000).

I should also note that this model is similar to the one used by Goldsmith (2001b), with two important differences. 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, for English, the simpler model is probably sufficient. The second difference between the two models 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 $a > 0$. This could be important for representing ambiguity and homonymy.

4.5 Gibbs sampler

To implement a Gibbs sampler for the model described above, we need to determine the distribution of each variable (or block of variables) conditioned on the remaining variables in the model. Because the model is exchangeable, sampling the ith variable conditioned on the values of the other $n-1$ variables is equivalent to sampling the nth variable conditioned on the previous $n-1$. Grouping together all the variables associated with a particular analysis of w_i and exploiting exchangeability in this way, we have

$$\begin{aligned} & P(z_i = z, c_i = c, t_i = t, f_i = f \mid h^-, d) \\ & \quad = \begin{cases} I(A(\ell_z) = (c,t,f)) \cdot \frac{n_z^{(h^-)} - a}{n^{(h^-)} + b} & 1 \le z \le K(\mathbf{z}_{-i}) \\ \frac{K(\mathbf{z}_{-i})a + b}{n^{(h^-)} + b} \cdot P_\mu(c,t,f \mid h^-, d) & z = K(\mathbf{z}_{-i}) + 1 \end{cases} \end{aligned} \tag{4.7}$$

where z_i, c_i, t_i, and f_i are the table number, morphological class, stem, and suffix associated with the ith word, $h^- = (\mathbf{z}_{-i}, \mathbf{c}_{-i}, \mathbf{t}_{-i}, \mathbf{f}_{-i}, \ell_1, \ldots, \ell_{K(\mathbf{z}_{-i})})$, and d is the observed data (corpus of words). The meaning of the notation $\mathbf{x}_{-i}$ is slightly different than before: here, it is used to indicate $\{x_1 \ldots x_{i-1}, x_{i+1} \ldots x_n\}$. P_μ is the distribution over analyses specified in Equation 4.5, which yields the following conditional distribution:

$$\begin{aligned} & P_\mu(c,t,f \mid h^-, d) \\ & \quad = P(c \mid \mathbf{c}_{-i}, \mathbf{z}) \cdot P(t \mid \mathbf{t}_{-i}, \mathbf{c}, \mathbf{z}) \cdot P(f \mid \mathbf{f}_{-i}, \mathbf{c}, \mathbf{z}) \\ & \quad = \int_\Delta P(c|\theta_c) P(\theta_c|\mathbf{c}_{-i}, \mathbf{z})\, d\theta_c \cdot \int_\Delta P(t|\theta_t) P(\theta_t|\mathbf{t}_{-i}, \mathbf{c}, \mathbf{z})\, d\theta_t \cdot \int_\Delta P(f|\theta_f) P(\theta_f|\mathbf{f}_{-i}, \mathbf{c}, \mathbf{z})\, d\theta_f \\ & \quad = \frac{m_c^{(h^-)} + \kappa}{m^{(h^-)} + \kappa C} \cdot \frac{m_{t,c}^{(h^-)} + \tau}{m_c^{(h^-)} + \tau T} \cdot \frac{m_{f,c}^{(h^-)} + \phi}{m_c^{(h^-)} + \phi F} \end{aligned} \tag{4.8}$$

where θ_c, θ_t, and θ_f are the parameters of the multinomial distributions over classes, stems, and suffixes; C, T, and F are the total possible number of classes, stems, and suffixes; and $m_x^{(h^-)}$ is the number of tables in h^- whose label includes x. (I use m to distinguish these

counts over labels from the n counts over tokens.) In the experiments presented in this chapter, 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). This is a simple but not terribly realistic approach; in Chapter 6 I discuss how to generalize the morphological model so that the number of possible classes, stems, and suffixes need not be specified in advance.

Although it would be possible to sample all the variables in the two-stage morphology model simultaneously using Equation 4.7, it is simpler to alternate between sampling the variables in the generator and those in the adaptor. The algorithm works by iterating over the following two steps:

1. Fixing the assignment of words to tables, sample a new morphological analysis for each table according to

$$\begin{aligned} P(A(\ell_z) = (c,t,f) \,|\, A(\boldsymbol{\ell}_{-z})) \;&\propto\; I(\ell_z = t.f) \cdot P_\mu(c,t,f \,|\, A(\boldsymbol{\ell}_{-z})) \\ &=\; I(\ell_z = t.f) \cdot \frac{m_c + \kappa}{m + \kappa C} \cdot \frac{m_{t,c} + \tau}{m_c + \tau T} \cdot \frac{m_{f,c} + \phi}{m_c + \phi F} \end{aligned} \quad (4.9)$$

   where all counts are with respect to $A(\boldsymbol{\ell}_{-z})$, the analyses of the labels $\boldsymbol{\ell}_{-z}$.

2. Sample a new table assignment z_i for each word from

$$P(z_i = z \,|\, \mathbf{z}_{-i}, \mathbf{w}, A(\boldsymbol{\ell}_{\mathbf{z}_{-i}})) \propto \begin{cases} I(\ell_z = w_i)(n_z^{(\mathbf{z}_{-i})} - a) & 1 \le z \le K(\mathbf{z}_{-i}) \\ \\ P(\ell_z = w_i)(K(\mathbf{z}_{-i})a + b) & z = K(\mathbf{z}_{-i}) + 1 \end{cases} \quad (4.10)$$

   where $\boldsymbol{\ell}_{\mathbf{z}_{-i}} = \{\ell_1, \ldots, \ell_{K(\mathbf{z}_{-i})}\}$ and $P(\ell_z = w_i)$ is found using Equation 4.9 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 4.9, and the appropriate counts are incremented.

4.6 Experiments

4.6.1 Experiment 1: Verbs

4.6.1.1 Data

I prepared a data set consisting of English verbs in orthographic 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, I 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 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, I 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 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 I used gives preference to consistency among suffixes rather than stems whenever possible. 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.

4.6.1.2 Evaluation procedure

To evaluate the morphological segmenter, I ran the sampler described in Section 4.5 with $b = 0$ and values of a between 0 and 1[7]. I evaluated the results of each parameter setting on a single sample taken after 1000 iterations of the algorithm, with $C = 6$ classes, $\kappa = .5$ and $\tau = \phi = .001$[8]. I compared the suffixes found by my system to the true suffixes given in the gold standard. As a comparison point, I also evaluated the results of the Linguistica program (Goldsmith, 2001a) on the same data set. Note that the results of the Linguistica system were the same regardless of whether it was presented with the full corpus or a list of unique word types, indicating that the system ignores token frequencies and works from lexical statistics.

4.6.1.3 Results and discussion

Effects of the hyperparameter a

Figure 4.1 shows the true distribution of words with each suffix and the distribution found by the two-stage system for various values of a. The results are displayed in two different

[7] 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 I did.

[8] Although I fixed the values for the hyperparameters in all the experiments presented in this thesis, it is possible to extend all of my models to contain prior distributions over the hyperparameters as well. In that case, the hyperparameters can be inferred using sampling procedures similar to those used here (West, 1992).

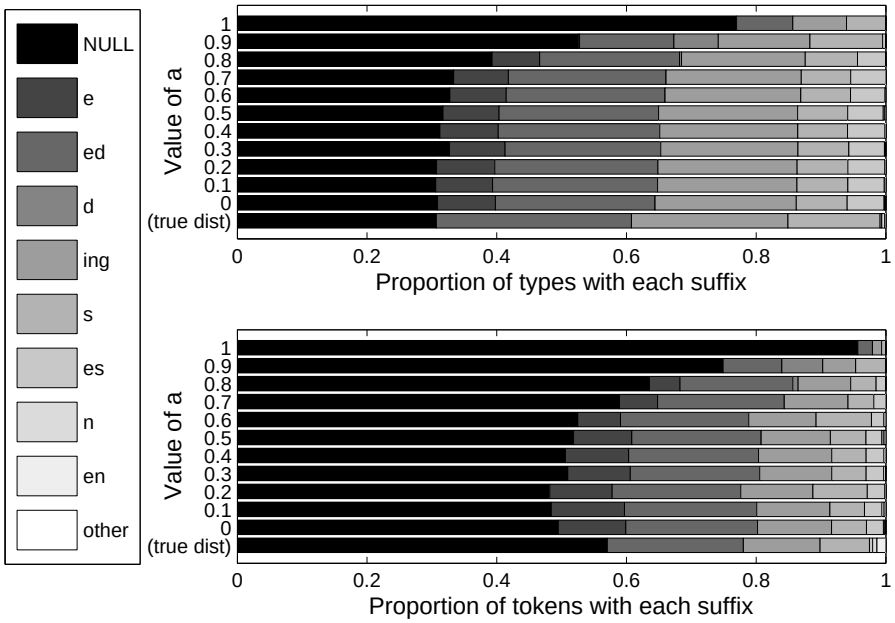

Figure 4.1: Results of the morphology learner for various values of a. The proportion of word types (top) and tokens (bottom) found with each suffix is shown, along with the distribution of suffixes in the gold standard.

ways. First, for each word type, the most probable 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).

From Figure 4.1, it is clear that values of a close to 1 cause the system to propose far too few suffixes overall, while results are more accurate and surprisingly consistent for values of a up to 0.7. This consistency is confirmed by Figure 4.2, which shows that the percentage of verbs whose analyses match the gold standard holds steady for $a \leq .7$, and drops off thereafter. Table 4.1 indicates that the average number of tables per word type for $a \leq .7$

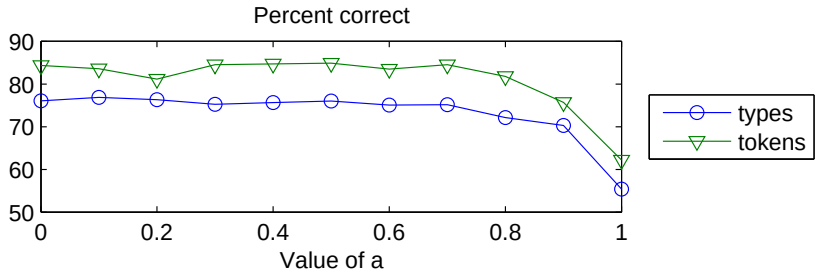

Figure 4.2: Percentage of verbs types and tokens assigned the gold standard analysis.

a	No. of Tables	Tables/Type
0	7761	1
.1	8758	1.13
.2	10272	1.32
.3	12172	1.57
.4	14178	1.83
.5	17957	2.31
.6	22676	2.92
.7	29478	3.80
.8	44020	5.67
.9	65751	8.47
1	137997	17.78

Table 4.1: Total number of tables in the sampled solution for each value of a, and average number of tables per word type.

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, since for $a \leq 0.7$, results match the distribution of suffixes in the gold standard fairly well, although certain suffixes (*-e* and *-es*) are found too frequently. I discuss this point further in the section on error analysis below.

Convergence

To ensure that the results presented above reflect the properties of the underlying model, it is important to examine the convergence of the Gibbs sampler. I did so by running the

sampler for 20,000 iterations and tracking several statistics at each iteration, including the number of tables in the current solution and its posterior probability. In addition to the standard initialization (assigning the morpheme boundary at random within each word), I also performed experiments initializing boundaries at the beginning of each word (i.e. with empty stems) or at the end of each word (i.e. with empty suffixes). The results of these experiments are shown in Figure 4.3. As these trace plots show, both the number of tables and the posterior probability level out by 1000 iterations, and remain steady (aside from the expected random fluctuations) thereafter. Although the numbers are different for the two values of a shown, the trends are the same. Iterations 10,000 through 20,000 are not shown, since there is essentially no change in the plots.

One interesting fact that these plots illustrate is that there seem to be two different local maxima in the search space. The solutions found by initializing randomly or with boundaries at the ends of words are essentially similar, but initializing with word-initial boundaries leads to different kinds of solutions, which I will describe in a moment. Figure 4.3 shows that these solutions have lower probability (primarily due to the morphological component), but are very stable nonetheless. Given enough time, of course, all of the samplers would produce samples from both of these local maxima, but there is no way to know how long this would take. If mobility within the search space were more of a concern, one solution would be to implement an annealing procedure similar to those described in Chapter 5. Since random initialization produces good results in practice, I did not implement annealing.

Example output

To give a better sense of the kind of output produced by the morphology learning system, Table 4.2 summarizes the final samples generated by two of the samplers plotted in Figure

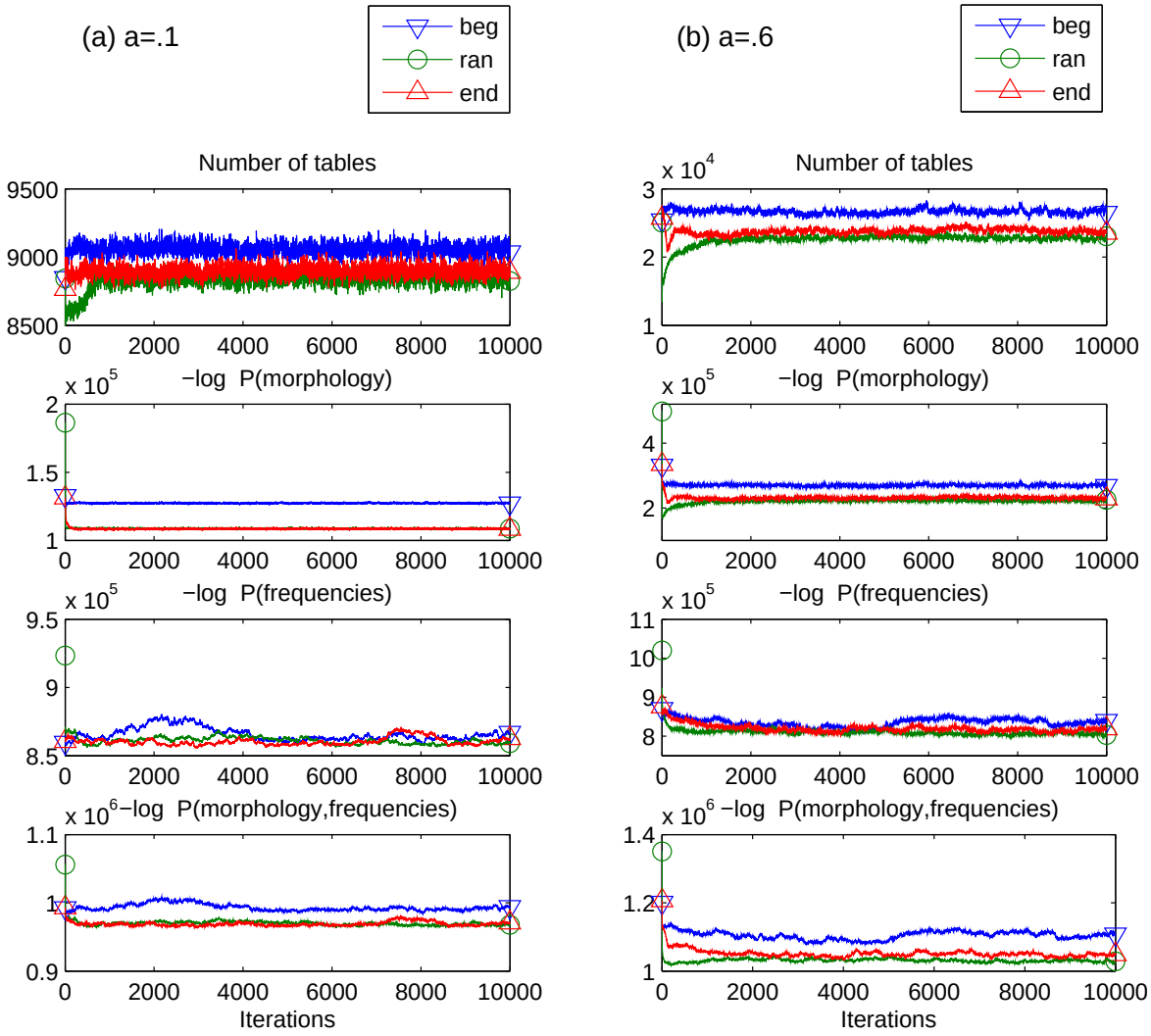

Figure 4.3: Trace plots of the morphology sampler over 10,000 iterations on the WSJ verb corpus with Pitman-Yor parameters $b = 0$ and (a) $a = .1$ or (b) $a = .6$. In three different conditions, boundaries were initially assigned at the beginning of each word ('beg'), at random ('ran'), or at the end of each word ('end'). From top to bottom, plots indicate at each iteration: the number of occupied tables; the probability of the morphological component of the model (i.e. the analyses $A(\boldsymbol{\ell})$); the probability of the frequency component of the model (i.e. the table assignments $\mathbf{z}$); and the total probability of the sample $P(A(\boldsymbol{\ell}), \mathbf{z} \,|\, d)$. The initial and final positions of the plots are marked on the axes.

4.3. The solution produced by the sampler initialized with word-initial boundaries, while not the expected analysis of the data, is nevertheless quite interesting: instead of discovering suffixes, the sampler has analyzed the data using prefixes. This result suggests that an extended model designed to handle multiple affixes per word might succeed in finding both prefixes and suffixes from this data set. Note that in the current system, the prefix solution can be avoided by disallowing empty stems.

Examining the output of the system also provides insight into some of the differences between the model developed here and the model used by Linguistica. In Linguistica, each class is associated with a unique set of suffixes that have been observed to combine with every stem in that class. Here, different classes may be associated with the same set of suffixes, and not every stem-suffix combination in each class is necessarily observed in the data. This property allows the model to generalize more freely than Linguistica, by placing stems together if they share several suffixes, even if some suffixes are different. In theory, this should be useful for generating novel (i.e. unobserved) inflected forms. In practice, more than one paradigm is sometimes contained within a single class (as in the first two classes in Table 4.2), which leads to overgeneralization.

Error analysis

Since the accuracy of the morphological learner is similar for $0 \leq a \leq 0.7$, I focused my error analysis on the case where $a = 0$. Figure 4.4 gives a better sense of the sorts of errors that are made by the system when $a = 0$. In particular, 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 is designed to maintain suffix identity (*state, stat.ing, stat.ed, state.s*). This leads the system to assume *-e* and *-es* suffixes where

(a)

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)

Tables	Stems		Suffixes	
1492	NULL	1492	followed	6
			recorded	6
			controlled	4
			thought	4
			gives	3
			dashed	3
			...	
1415	NULL	1415	improving	4
			downgraded	4
			proposing	4
			posted	4
			pulled	4
			charging	4
			...	
1631	NULL	1476	turn	8
	re	103	sold	8
	over	27	valued	7
	under	25	doing	6
			stated	6
			saw	5
			...	
1678	NULL	1479	reported	11
	re	156	continuing	7
	dis	43	place	6
			placed	6
			sought	5
			acquired	5
			...	
1359	NULL	1359	play	5
			praised	4
			bother	4
			guarantees	4
			tends	4
			blocking	4
			...	
1443	NULL	1409	thinks	5
	s	34	has	4
			elected	4
			wore	4
			earned	4
			log	4
			...	

Table 4.2: Sample solutions for the WSJ verb corpus with $a = .1$, with boundaries initialized (a) at random or (b) or word-initially. 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. Solution (a) has higher probability than solution (b).

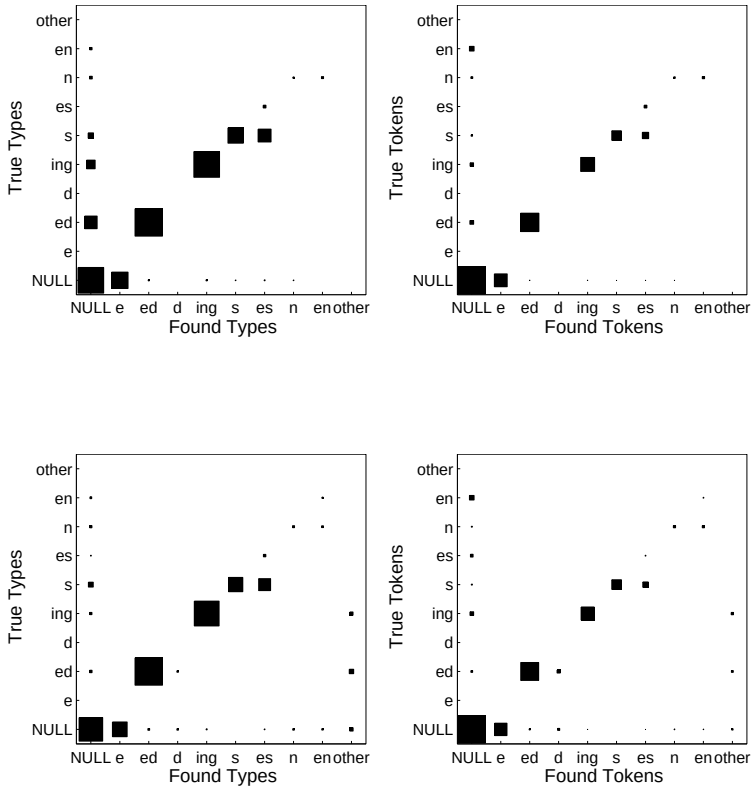

Figure 4.4: Confusion matrices for the 2-stage morphology model with $a = 0$ (top), and for Linguistica (bottom). The area of a square at location (i, j) is proportional to the number of word types (left) or tokens (right) with true suffix i and found suffix j.

the gold standard has *NULL* and *-s*. This kind of problem is common to other morphological learning systems, including Linguistica (see Figure 4.4), and cannot be solved without the addition of phonological or orthographic capabilities. It is also worth noting that, if stems and suffixes are given equal weight in terms of economy, the choice of segmentation taken by the system is actually better than the gold standard, 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

	Missing	Extra	Wrong	Correct
Linguistica, types	1.65%	9.21%	6.84%	82.31%
2-stage, types	9.28%	9.11%	5.57%	76.05%
Linguistica, tokens	2.91%	10.04%	3.27%	83.78%
2-stage, tokens	2.89%	10.46%	2.32%	84.33%

Table 4.3: Percentage of types and tokens analyzed by the two systems with missing suffixes (gold standard contains a suffix, but none was found), extra suffixes (gold standard contains no suffix, but one was found), wrong suffixes (when both gold standard and analysis contained non-empty suffixes), and correct suffixes. Results for the two-stage system are with $a = 0$.

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. These errors are far less prevalent in Linguistica, where encoding a longer stem is more costly than encoding a shorter stem, so words that can be analyzed using an existing suffix tend to be segmented. Overall, this is the main difference in performance between the two systems, as can be seen in Table 4.3. Because these errors occur on the most infrequent words, the difference in error percentages between the two systems is much more pronounced when scored on types rather than tokens.

The confusion matrices highlight one other difference in the behavior of the two systems, which is that Linguistica proposes more suffix types than does the 2-stage model, including the following ten in the "other" class: {*ting, ied, y, ted, led, ate, ying, ized, ped, ize*}. Between 3 and 34 word types are analyzed with one of these suffixes. Although analyses using these suffixes are counted as errors relative to the gold standard, in some cases they are arguably correct.

4.6.2 Experiment 2: Child-directed speech

4.6.2.1 Data

Experiment 1 used a corpus of verbs in orthographic form as input data, partly because learning English verbs has become a standard task for computational models of morphology, and partly because this choice of corpus makes evaluation against a gold standard possible. However, the verb corpus does not accurately reflect the input to the child learning English. I therefore performed a second experiment using a corpus of phonemically transcribed child-directed speech. The source of this data 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. Utterances not made by caretakers were filtered out and the remaining transcriptions were converted from orthographic to phonemic representations using a phonemic dictionary[9]. Variations in pronunciation indicated in the orthography (e.g. *going* vs. *goin'*) were preserved as much as possible in the phonemic forms[10] (`go1N`, `go1n`), and many non-words (e.g. *hm*) were also retained. This Brown-Morgan corpus is therefore noisier than the Bernstein-Ratner-Brent corpus used in Chapter 5. 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.

4.6.2.2 Evaluation procedure

For this experiment, the sampler was run using the same parameter values as in Experiment 1: $b = 0$, $C = 6$, $\kappa = .5$, and $\tau = \phi = .001$. The value of a was varied between 0 and

[9]Thanks to James Morgan and the Metcalf Infant Research Lab for the phonemic version of this corpus.

[10]The phonemic alphabet used for this data set is provided in Appendix A.

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 4.4: 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.

1, and a single sample was taken after 1000 iterations. Since there is no gold standard for this corpus, my evaluation is qualitative, based on examining the output of the algorithm. Note also that because Linguistica automatically removes capitalization from the input, and capitalization is used to distinguish between different phonemes in the phonemic alphabet used in this corpus, it was not possible to run Linguistica on the input.

4.6.2.3 Results and discussion

Qualitatively, the results of varying the Pitman-Yor parameter a are similar for this data set and the corpus of English verbs. Table 4.4 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 contained 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 4.5) illustrates the problem. Three of the classes found contain primarily nouns, with possible suffixes NULL and `z`. Two of the remaining classes contain large numbers of verbs with a variety of verbal inflections (including allomorphic and phonetic variants) and derivational endings. 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" are onset+nucleus sequences and the "suffixes" are codas (or sometimes 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 syllable types from 78 to 17 (those found in the noun and verb classes in Table 4.5, plus `n`, `6n`, `l`, `&d`, and `1nz`) and 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. Unfortunately, since there is no gold standard for this corpus, it is difficult to say what the actual percentage of morphologically complex types, or types with the `z` suffix, is. 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. This analysis might also shed light on the question of

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 4.5: 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.

why morphological inference from types leads to a class of monosyllables, while adding only a very small amount of frequency information[11] eliminates this phenomenon. Does this indicate some problem with discarding all frequency information, or does it simply reflect weaknesses in the current model? In natural language, for example, stems tend to be longer than affixes, yet in the model used here, all strings are equally likely as stems or suffixes. Is it necessary to have prior knowledge that stems tend to be longer? In most of the classes found by the system here, this fact simply falls out; perhaps the use of additional sources of information, such as context, would be sufficient to eliminate the spurious analyses of the remaining words.

4.6.3 General Discussion

The experiments presented here provide evidence that, for induction of regular morphology, statistics derived from the 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 provides a justification for the use of word lists in many previous morphological learning systems (Plaut and Gonnerman, 2000; Regier et al., 2001; Snover and Brent, 2003). Interestingly, my experiments also suggest that damping corpus frequencies may be as effective, or perhaps even more effective, than using lexical statistics. Further investigation into this possibility, and its implications for theories of human learning, is called for.

Given the success of corpus-based learning for many other linguistic tasks, it is natural to ask why using corpus frequencies would be detrimental to morphological induction. There are two likely reasons for this. First, the most frequent words in any language tend to be

[11] With $a = .3$, the sampled solution contained 12,463 tables, versus 6,807 with $a = 0$.

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.

The other reason that token frequencies may be problematic for learning regular morphology is that the modeling assumption of independence between stems and suffixes is only approximately correct. This assumption is intended to capture the fact that stems and suffixes are less tightly correlated with each other than are arbitrary subsequences of words. The system uses this information to determine where suffix boundaries are likely to be. However, different stems do pattern somewhat differently, with some stems appearing more often in the past tense and others in the infinitive, for example. By partially or completely damping the corpus statistics, the differences in behavior between stems are reduced, while the relative lack of predictability of suffixes is retained.

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 greater variety of data from multiple languages, and if more detailed analysis were undertaken on the output from the Brown-Morgan corpus. Most of my analysis was performed on orthographic verb forms, simply because the WSJ verb data set is large and provides an easy way to produce a mostly-accurate gold standard. It is unfortunately the case that gold standards for morphological segmentation are difficult to produce and open to disagreement (Goldsmith, In press). Nevertheless, even a basic qualitative analysis on additional data sets would be useful.

Another possible objection to my experiments lie in the simplistic nature of the morphological model. The limitation to a single stem and suffix is certainly not adequate to account for the morphology of many languages, even those with fairly basic morphology, such as Spanish[12]. However, for regular English verb inflections and the very simple morphology found in child-directed speech, it is mostly sufficient. The system is able to identify common suffixes in both corpora, and distinguish roughly between nouns stems and verb stems in the Brown-Morgan corpus. The classes found are fairly noisy, but in combination with other cues such as local context, they might be used effectively as a bootstrapping mechanism for discovering more accurate syntactic information.

One modeling deficiency that does affect the results obtained from the English data sets is the fact that all possible stems and suffixes are assumed to be equally probable *a priori*. It was suggested that this assumption might be partially responsible for the "short word" class found in Experiment 2. In Experiment 1, another problem with this assumption was highlighted: if only a single inflected form of a particular stem is observed (e.g. *acquiesced*, with no other form of *acquiesce*), the system will introduce a new stem to account for this word. Any new stem has the same probability, so the overall probability of the word depends on the suffix that is chosen. The system will therefore (with high probability) choose whichever possible suffix appears most frequently on other wordforms. In the English data, the empty suffix is most frequent, so the word will not be segmented. This is not completely unreasonable as a conservative learning strategy, but does not seem to match the way that humans are able to generalize from single instances.

One way the model could be extended in order to segment words when no direct evidence

[12]Pilot experiments on Spanish data do in fact reveal underanalysis of suffix sequences. For example, the suffix *ito* is not found in *animalitos* (although both *animal* and *ito* are found in other words) because the model only permits a single suffix per word. *Animalitos* is analyzed as *animalito.s*.

of the stem is available is by changing the assumption of a uniform prior distribution over stems and suffixes. I have discussed how Linguistica segments these kinds of words because of its preference for shorter morphemes as well as fewer morphemes. Introducing a preference for shorter morphemes would not be difficult in the framework I have been working with. It could be done by replacing the Dirichlet priors over $P(t|c)$ and $P(f|c)$ with an n-gram phoneme model, for example. Since longer strings tend to have lower probability in an n-gram model, this prior would favor shorter stems and suffixes.

Favoring shorter stems is probably the simplest way to improve performance on unique items, but there are other possibilities as well. An obvious source of information that is not exploited by the current model is syntactic context: words with similar inflection tend to appear in similar syntactic contexts, and vice versa. Certain semantic properties are also correlated with morphology. While these sources of information are surely useful to human learners, the results of my experiments indicate that distributional information in the lexicon, along with a prior favoring sparse solutions, is sufficient to induce a fairly accurate morphological analysis of English verbs.

4.7 Conclusion

In this chapter, I have presented a two-stage model with a morpheme-based lexicon generator and a Pitman-Yor adaptor. By varying the parameters of the adaptor, I have shown how morphological induction changes as the frequencies observed by the morphological component of the model vary smoothly from those in the lexicon to those in the corpus. For two very different English corpora, I found that, when inference was based on corpus frequencies, the model was unable to discover much morphological structure at all. On the

other hand, type-based inference led to successful discovery of many common suffixes, and allowed the system to create separate noun and verb classes. In one corpus, I found that type-based inference led to over-analysis of monosyllabic words, but that this problem was eliminated with the addition of a small amount of frequency information. These results support the hypothesis that lexical statistics are more useful for learning morphology than corpus statistics, although damped corpus frequencies might be better still.

Although the learning system presented here incorporates a very simple model of morphology, it nevertheless has several advantages over many other systems. The explicit nature of the model, including assumptions about the prior, makes it easy to analyze how these assumptions affect the results of learning. For example, the two-stage model behaves differently from Linguistica for words with unique stems; Linguistica includes a prior assumption about stem length that the two-stage model does not; therefore, the difference in behavior is probably due to the difference in the prior. The flexibility of the two-stage framework means that this hypothesis will be easy to test in future work simply by modifying the prior in the two-stage model.

A final (to my knowledge, unique) advantage of the current system is its ability to represent both corpus frequencies and lexicon information within a single unified framework. Other systems based on lexicon statistics assume that the corpus is preprocessed to extract a lexicon, but this assumption requires that the word tokens in the corpus are known. There is evidence that children are sensitive to morphological information at a stage when word segmentation is far from perfect (Santelmann and Jusczyk, 1998), which suggests that token identification and lexicon building are taking place at the same time as morphological acquisition. This raises the question of how changes in the structure of the lexicon might flow

back to affect corpus processing. Representing both token frequencies and type frequencies within the same model makes it possible to address such a question by learning linguistic structure at both levels simultaneously. Although a complete system of this sort is beyond the scope of this thesis, in Chapter 6 I do provide a rough outline of an integrated model for discovering both word boundaries (based on corpus statistics) and morphological analyses (based on lexicon statistics). Before doing so, however, I review the word segmentation task and explain how the two-stage framework can be applied to word segmentation.

Chapter 5

Word Segmentation

5.1 Introduction

One of the first tasks infants must solve as they are acquiring language is the problem of word segmentation: identifying word boundaries in continuous speech. About 9% of utterances directed at English-learning infants consist of isolated words (Brent and Siskind, 2001), but there is no obvious way for children to know from the outset which utterances these are. Since multi-word utterances generally have no apparent pauses between words, children must be using other cues to identify word boundaries. In fact, there is evidence that infants use a wide range of weak cues for word segmentation. These cues include phonotactics (Mattys et al., 1999), allophonic variation (Jusczyk et al., 1999a), metrical (stress) patterns (Morgan et al., 1995; Jusczyk et al., 1999b), effects of coarticulation (Johnson and Jusczyk, 2001), and statistical regularities in the sequences of syllables found in speech (Saffran et al., 1996a). This last source of information, which I will refer to as *distributional cues*[1], seems to be used by infants earlier than most other cues, by the age of 7 months (Thiessen and Saffran, 2003). In addition, word segmentation strategies based on distributional cues can be language-independent, which many other proposed strategies are not. These facts have caused some researchers to propose that the use of distributional cues is a crucial first step in bootstrapping word segmentation (Thiessen and Saffran, 2003), and have provoked a great deal of interest in distributional segmentation among cognitive scientists.

Computational models of word segmentation have proven to be a useful complement

[1]The use of the term *distributional cues* to refer specifically to statistical regularities in syllable or phonemes sequences can be a bit misleading, since other cues can also be viewed as "distributional". For example, metrical strategies rely on the distribution of stress patterns in lexical items, and allophonic and phonotactic cues depend on the distribution of different allophones or phoneme sequences within syllables, words, or phrases. In addition, there is potential for confusion with the related, but distinct, concept of a probability distribution. Nevertheless, I will continue to use this term as just defined, following Saffran et al. (1996b) and others.

to behavioral studies of distributional cues. These models provide a way to examine distributional learning from much larger and more realistic data sets than is usually possible behaviorally, and can be used to implement and test possible theories of distributional learning. Most of the models that have been developed take an algorithmic approach, asking *how* statistical information can be used procedurally to identify word boundaries. However, there have been few attempts to approach the problem at the computational level, asking *what* statistics are actually needed in order to solve the problem successfully. In this chapter, I discuss my own word segmentation modeling experiments, which are designed to address the latter question. In particular, I focus on the importance of context for segmentation. Most previous researchers have developed models based on the assumption that words are generated independently, i.e. the probability of generating a particular word is the same regardless of the words that came before it. I show that this *unigram* assumption leads to relatively poor performance. When context is taken into account using a *bigram* model (where the probability of a word depends on the previous word), segmentation improves markedly. I argue that previous results to the contrary (Venkataraman, 2001) were due to the use of approximate search techniques.

The remainder of this chapter is organized as follows. In Section 5.2, I review some of the previous work on distributional word segmentation, including approaches based on local association statistics, neural networks, maximum likelihood, and Bayesian modeling. I discuss systems intended as cognitive models, as well as some designed for practical application in the segmentation of Asian texts (where word boundaries are not made explicit in the orthography). In Section 5.3, I present a unigram model for word segmentation based on the two-stage language modeling framework in Chapter 3, and provide experimental

results illustrating the failures of this model. I consider some improvements to the unigram model in Section 5.4 and show that, because they do not address the underlying unigram assumption, results remain relatively poor. In Section 5.5, I describe a different model that incorporates contextual dependencies, and show that segmentation performance improves dramatically, outscoring previous distributional approaches. In Section 5.6, I discuss the implications of these results and conclude.

5.2 Previous work

5.2.1 Supervised and lexicon-based approaches

In the natural language processing community, most approaches to word segmentation involve language-specific knowledge in the form of a lexicon, hand-segmented training data, or both. The simplest lexicon-based methods match the longest possible substrings to lexicon entries, possibly using heuristics to resolve ambiguities (Nie et al., 1994). Training data can be used to evaluate the probabilities of many different possible segmentations and choose the most probable one (Chang and Chen, 1993). These methods, and hybrid approaches, include techniques of varying sophistication to deal with the problem of novel words. Examples include modeling different classes of unknown words with a weighted finite state transducer (Sproat et al., 1996) and iteratively growing the dictionary using local association statistics similar to those described below (Chang and Su, 1997). These methods are useful in practice, but have little to say about infant word segmentation, where lexical knowledge is minimal or absent. Most cognitive models of adult word recognition are similarly tied to pre-existing lexical knowledge (Marslen-Wilson, 1987; McClelland and Elman, 1986; Norris, 1994; Luce and Pisoni, 1998) and are therefore unsuitable as models of

the earliest stages of word segmentation. I therefore turn to work focusing on unsupervised methods with no initial lexicon.

5.2.2 Approaches based on local association statistics

Probably the most popular method for unsupervised word segmentation is based on the observation that transitions between linguistic units (characters, phonemes, or syllables) within words are generally more predictable than transitions across word boundaries. Originally, Harris (1954) suggested that boundaries between morphemes within words could be found by counting the number of phonemes that can extend each prefix string of a word to form another legal prefix in the language. Morpheme boundaries should be assigned at peaks of this statistic, *successor frequency*. For example, the strings *gover* and *governm* each have a successor frequency of 1, while *govern* has a successor frequency of at least 5 (*govern***m***ent, govern***e***d, govern***o***r, govern***i***ng, govern***a***nce*), suggesting a morpheme boundary after this prefix. More recently, cognitive scientists have proposed that children use the *transitional probabilities* between syllables (i.e. the conditional probability of syllable s_i given the previous syllable s_{i-1}) as cues to word boundaries (Saffran et al., 1996a). Other statistics measuring the degree of association between adjacent units or groups of units have been also been proposed in both the cognitive science and the natural language processing literature. These include mutual information (Sun et al., 1998; Brent, 1999; Swingley, 2005), n-gram frequency (Ando and Lee, 2000; Swingley, 2005), "accessor variety" (Feng et al., 2004), and boundary entropy (Cohen and Adams, 2001).

Using any of these statistics, the most straightforward segmentation algorithm simply calculates the chosen statistic at each possible boundary point and inserts a boundary at every local minimum (or maximum, depending on the statistic). However, this procedure

yields poor results on realistic corpora when using transitional probabilities (Brent, 1999; Gambell and Yang, In submission), mutual information (Brent, 1999), and presumably other statistics as well. Gambell and Yang (In submission) suggest that the addition of language-universal constraints on the placement of primary stress can improve performance, and that a non-statistical stress-based strategy works best of all. These proposals are problematic, however, since stress can be manifested in a variety of ways cross-linguistically (Hayes, 1995) and it isn't clear that the primary stress of each word can be identified solely on the basis of acoustic cues, without knowing the word boundaries in the first place. Other researchers have demonstrated good performance using local association statistics by combining multiple statistics or thresholding based on the magnitude of the statistic or the difference between statistics (Sproat and Shih, 1990; Sun et al., 1998; Swingley, 2005; Ando and Lee, 2000; Chang and Su, 1997; Cohen and Adams, 2001). Unfortunately, the lack of explicit probabilistic models in these systems makes it difficult to evaluate whether differences in performance are due to the different statistics used, differences in the way those statistics are combined, or differences in other aspects of the systems.

5.2.3 Neural networks

Several researchers have used neural networks to segment representations of speech using distributional cues. Some researchers have used artificial corpora as input (Elman, 1990; Allen and Christiansen, 1996), while others have trained their networks from phonological transcriptions of natural speech (Cairns and Shillcock, 1997; Christiansen et al., 1998). These simulations have typically used the simple recurrent network (SRN) architecture (Elman, 1990), which combines a standard feed-forward network with a set of "copy-back" units, so that the input at each time step is a combination of the current corpus input and

the previous output of the network. By including recurrence, these kinds of networks allow predictions based on context, where the context is the state of the network at the previous time step. This use of context is a very different approach from most of the algorithms described above, and may provide advantages. Unfortunately, it is very difficult to determine exactly what part of the contextual information is actually useful for prediction, and there is no way to explicitly manipulate the use of context. The results of most of these systems are also difficult to compare to others, since they have been based on different input corpora, and in many cases different input representations as well.

The work that is probably most comparable to other cognitive models is that of Christiansen, Allen, and Seidenberg (1998). They trained an SRN using a single pass through a corpus of phonologically transcribed utterances of child-directed speech. As in the cognitive models described below as well as my own work, word boundaries were removed from the input, but utterance boundaries remained. The corpus was similar in size to the corpus used by these other systems. Each phoneme was converted to a set of phonological features for input to the system, and stress markings and utterance boundaries were input as well. The task of the network was to predict, at each time step, the next set of inputs. Since the network had no explicit notion of a word boundary, the authors counted a word boundary whenever the network predicted an utterance boundary with higher-than-average activation. Despite the additional cue of stress, the results reported by Christiansen, Allen, and Seidenberg are worse than those of other systems using similar input (Brent, 1999; Venkataraman, 2001; Batchelder, 2002), and they degrade in simulations where the stress cue was not included. Moreover, the decision to equate utterance boundaries with word boundaries means that if the input contains no utterance boundaries, no word boundaries

will be predicted. Natural language does contain utterance boundaries, of course, but the experiments of Saffran et al. (Saffran et al., 1996a) suggest that infants are able to detect words even when no utterance boundaries are available.

5.2.4 Maximum-likelihood approaches

In contrast to the systems described above are those based on explicit probabilistic models. Several of these use maximum-likelihood estimation as a primary component, despite the fact that, unless some constraint is placed on the set of allowable words, the maximum-likelihood segmentation of a corpus contains no word boundaries (aside from any utterance boundaries known at the outset). The reason is the same as in the case of morphology: the solution where each utterance is postulated as a "word", with probability proportional to its empirical frequency, captures the empirical distribution of the data perfectly. Any other solution is sub-optimal.

Given this fact, it follows that systems based on maximum-likelihood estimation all place constraints on the set of possible words. In the Chinese word segmentation systems of Peng and Schuurmans (2001) and Ge et al. (1999), only words with four or fewer characters are considered, and additional heuristics are used to split up agglomerations of multiple words. These heuristics improve performance, but may drive the system to a solution with lower likelihood. The cognitive model described by Batchelder (2002) uses an online search procedure that segments each utterance in turn using maximum-likelihood estimation, then adds the newly segmented words to a progressively built lexicon. The initial lexicon contains only single characters, and after each utterance is segmented, the frequencies of the segmented words are incremented and each pair of adjacent words is concatenated and added to the lexicon (with a frequency of 1) as a potential word for future segmentations. The

probability of each word is assumed to be proportional to its relative frequency, and the probability of an utterance is the product of the word probabilities. Batchelder notes that "as longer and longer words become eligible for parsing, they tend to be selected over shorter words" so that "a typical unconstrained run tends to underdivide the text". To correct for this effect, Batchelder combines the maximum-likelihood evaluation of a segmentation with an evaluation based on an "optimum-length parameter", which downgrades the scores of words that are longer than the true average for the corpus. A more principled approach would replace this parameter with a Bayesian prior on word length. Note also that making the average word length of the corpus available to the algorithm gives it a big advantage over truly unsupervised systems. Nevertheless, performance is similar to other model-based systems.

A final system based on maximum-likelihood estimation is that of Venkataraman (2001). This system is similar to Batchelder's in that it uses an online search procedure to segment each utterance in turn, adding the new words to its lexicon before moving on to the next utterance. The difference is that Venkataraman's search procedure starts with each utterance unsegmented, and proceeds by splitting utterances and adding shorter words to the lexicon. The probability of a possible segmentation of an utterance u into words $w_1 \dots w_n$ is

$$P(u) = P(w_1)P(w_2 \mid w_1) \dots P(w_n \mid w_1 \dots w_{n-1}) \tag{5.1}$$

Venkataraman experiments with three different ways of approximating $P(w_i \mid w_1 \dots w_{i-1})$:

$$P(w_i \mid w_1 \dots w_{i-1}) \approx P(w_i) \tag{5.2}$$

$$P(w_i \mid w_1 \dots w_{i-1}) \approx P(w_i \mid w_{i-1}) \tag{5.3}$$

$$P(w_i \mid w_1 \dots w_{i-1}) \approx P(w_i \mid w_{i-2}, w_{i-1}) \tag{5.4}$$

i.e. he uses unigram, bigram, and trigram models. Since the empirical probability of a novel n-gram is 0, Venkataraman uses a backoff procedure to estimate probabilities using lower-order models when needed. The word unigram model backs off to a phoneme unigram model.

In Venkataraman's system, the probability of a string under the phoneme model will generally be less than its probability calculated as the product of (previously observed) word probabilities, so utterances tend to be split into known words when possible. Essentially, the online search and backoff scheme together introduce a constraint that can be roughly stated as follows: "add a new n-gram to the lexicon only when the utterance cannot be parsed using the current lexicon[2]." This constraint prevents the algorithm from introducing an arbitrary number of new lexical items, so it does not memorize every utterance whole. However, limiting the number of parameters in this way is not a principled solution to the over-fitting problem of maximum likelihood. In practice, Venkataraman's results are very similar to those of Batchelder (2002) and the Bayesian system developed by Brent (1999) (see below).

Venkataraman's work is most notable because he presents the only model-based system I am aware of that goes beyond a unigram model. All of the other maximum-likelihood-based systems described above, as well as the Bayesian systems described below, assume that words are generated independently. This assumption also underlies the experimental stimuli used in many word segmentation experiments with artificial languages[3] (Saffran et al., 1996a; Saffran et al., 1996b; Newport and Aslin, 2004; Thiessen and Saffran, 2003; Johnson

[2]The real situation is fuzzier than this statement suggests. An item could be added to the lexicon if the rest of the items in the parse were sufficiently probable to outweigh a parse consisting entirely of known, but improbable, items.

[3]Technically, the stimuli in these experiments do not quite follow a unigram model, since the same word never appears twice in a row. Aside from this constraint, words in the stimuli are independent.

and Jusczyk, 2001). Unlike these experimental stimuli, natural language displays many complex dependencies, so one would expect that incorporating some notion of sequential context into a word segmentation model should improve results. Venkataraman found instead that his system performed almost the same regardless of which n-gram model was used. However, the results of my own work (detailed below) suggest that this lack of effect is misleading, caused by a search procedure whose effects overwhelm any differences in the underlying model.

This problem with Venkataraman's system is symptomatic of a more general problem with all the maximum-likelihood systems described here: by using unprincipled methods of constraining the number of parameters (words) in the model, they lose the benefit that is normally conferred by using a probabilistic model. The results of these systems are due to some unknown combination of model and search procedure. A better way to apply constraints to a model-based learning system is by using a Bayesian prior. Constraints are explicitly incorporated into the model itself, so that results are independent of the particular search procedure used (assuming that the procedure is sufficient to achieve the objective). Changing the assumptions in the model should then yield insights about the effects these assumptions have on learning. I review the previous work on Bayesian modeling for word segmentation in the following section, and then present my own research in this area starting in Section 5.3.

5.2.5 Bayesian models

Bayesian models for word segmentation fall into two categories: those based on the MDL principle, and those based on some other Bayesian formulation. One of the earliest MDL-based systems is described by de Marcken (de Marcken, 1995; de Marcken, 1996). This

system builds a representation of the input corpus as a tree-structured hierarchy of chunks, where the smallest chunks are single characters, and the largest chunks tend to correspond to phrases, words, or morphemes. De Marcken argues that the hierarchical nature of the representation captures the fact that, for example, the meaning of *National Football League* is partially explained by the meaning of *national*, which in turn is partially explained by the meaning of *nation*. The model assumes that complex structures contain both compositional and non-compositional aspects of meaning and behavior. Although these properties of the model are attractive in some respects, they also make it difficult to evaluate the performance of the system alone or with reference to other systems, since there is no distinguished word level representation. Later work seems to have been influenced more by the idea of using MDL for word segmentation than by the specifics of de Marcken's approach.

Another early MDL system for word segmentation is described by Brent and Cartwright (1996). This system is intended as a model of early lexical acquisition in humans, and its input consists of phonological transcriptions of child-directed speech. The results of Brent and Cartwright's experiments suggest that phonotactic constraints (derived by observing the sequences of consonants that occur at utterance boundaries) can be very helpful for a distributional learner. However, the search algorithm used in this system was sufficiently compute-intensive that Brent and Cartwright were only able to segment a very small corpus (about 170 utterances).

In later research, Brent presents another Bayesian model for word segmentation with a more efficient search algorithm (Brent, 1999). This system, which Brent calls Model-Based Dynamic Programming-1 (MBDP-1), is not based on MDL. Instead, it uses a model that generates the entire corpus at once in a sequence of five steps:

1. Generate L, the number of types in the lexicon.

2. Generate the phonemic representation of each type (except for the single distinguished "utterance boundary" type, $).

3. Generate a token frequency for each lexical type.

4. Generate an ordering for the set of tokens.

5. Concatenate all the tokens to create an unsegmented corpus.

The first four steps of this model define a probability distribution over all possible segmented corpora. The last step is deterministic, which means that certain segmented corpora will produce the observed data with probability 1, and all others will produce it with probability 0. Therefore the posterior probability of a segmentation given the data is proportional to its prior probability under the model, and the best segmentation will be the one with the highest prior probability.

Rather than exhaustively evaluating all possible segmentations of the corpus, Brent instead implements an efficient dynamic programming algorithm to search for the best segmentation. He defines the *relative probability* of a word to be the ratio of the probability of the segmented corpus up to and including that word to the probability of the segmented corpus prior to that word. This permits him to use an algorithm that segments one utterance at a time, assuming the segmentations of all previous utterances to be fixed.

There are two important points to note about the MBDP-1 model. First, the distribution over L assigns higher probability to models with fewer lexical items. I have argued that this is necessary to avoid memorization, and indeed the unsegmented corpus is not the optimal solution under this model, as I will show in Section 5.3. Second, the factorization into four

separate steps makes it theoretically possible to modify each step independently in order to investigate the effects of the various modeling assumptions. However, the mathematical statement of the model and the approximations necessary for the search procedure make it unclear how to modify the model in any interesting way. In particular, the fourth step uses a uniform distribution over orderings, which creates a unigram constraint that cannot easily be changed. In the following sections, I show how the two-stage modeling framework described in Chapter 3 can be used to develop both unigram and bigram models for word segmentation, while also incorporating a preference for sparse solutions. This flexibility will allow me to investigate the effects of context on word segmentation.

5.3 Unigram word segmentation

5.3.1 The Dirichlet process model

The two-stage model I use for unigram word segmentation is TwoStage(CRP(α_0), P_0), where the generator P_0 is a unigram phoneme model:

$$P_0(w) = p_\#(1-p_\#)^{n-1}\prod_{i=1}^{n} P(m_i) \tag{5.5}$$

where word w consists of the phonemes $m_1 \ldots m_n$, and $p_\#$ is the probability of the word boundary #. For simplicity I used a uniform distribution over phonemes, and experimented with different fixed values of $p_\#$. (I discuss a different distribution over phonemes in Section 5.4. It is also possible to infer both the distribution over phonemes and $p_\#$, although I did not do so here.)

As discussed in Chapter 3, this two-stage model is equivalent to the following Dirichlet

process language model:

$$w_i \,|\, G \qquad \sim G$$

$$G \,|\, \alpha_0, P_0 \sim \mathrm{DP}(\alpha_0, P_0)$$

where w_i is the ith word in the corpus. From Equation 3.11, the probability of w_i given previously observed words $\mathbf{w}_{-i}$ is

$$P(w_i = w \,|\, \mathbf{w}_{-i}) = \frac{n_w^{(\mathbf{w}_{-i})} + \alpha_0 P_0(w)}{i - 1 + \alpha_0} \tag{5.6}$$

where $n_w^{(\mathbf{w}_{-i})}$ is the number of instances of w observed in $\mathbf{w}_{-i}$. The actual CRP table assignments for each word $\mathbf{z}_{-i}$ only become important later, in the bigram model.

So far, the model accounts for the number of times each word appears in the data. The input corpus used in my experiments also includes utterance boundaries, so these must be accounted for in the model as well. This is done by assuming that utterances are generated as follows:

1. Decide whether the next word will end the utterance or not.

2. Choose the identity of the that word.

3. If more words are to be generated, return to step 1.

This process can be described by a probabilistic grammar with context-free production rules, where each rule is assigned a probability that depends on the number of times that rule has already been observed:

$$P(r_i = r_{branch} \,|\, \mathbf{r}_{-i}) \quad U \rightarrow W\ U$$

$$P(r_i = r_{end} \,|\, \mathbf{r}_{-i}) \quad U \rightarrow W$$

$$P(w_i = w \,|\, \mathbf{w}_{-i}) \quad W \rightarrow w \qquad \forall w_i \in \Sigma^*$$

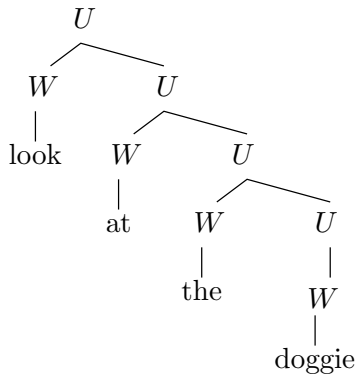

Figure 5.1: A hypothetical utterance, as parsed by the unigram DP word model.

where $r_{branch} = U \rightarrow W\ U$, $r_{end} = U \rightarrow W$, and $r_i \in \{r_{branch}, r_{end}\}$ is the ith U production generated. I assume a symmetric Beta($\frac{\rho}{2}$) prior[4] over the probability of the U productions, so

$$P(r_i = r_{branch} \,|\, \mathbf{r}_{-i}) = \frac{n_{r_{branch}} + \frac{\rho}{2}}{i + 1 + \rho} \tag{5.7}$$

$P(r_i = r_{end}) = 1 - P(r_i = r_{branch})$ and $P(w_i = w)$ is given by Equation 5.6. Notice that this grammar has the form of a standard probabilistic context-free grammar, but the probabilities of the rules change as more data is observed. Figure 5.1 provides an example parse.

5.3.2 Gibbs sampler

With the generative model described above, Gibbs sampling can be used to sample from the posterior distribution of segmentations given an unsegmented input corpus. The Gibbs sampler I implemented considers a single possible boundary point at a time, so each sample is from a set of two hypotheses, h_1 and h_2. These hypotheses contain all the same boundaries for the entire corpus except at the one position under consideration, where h_2 has a boundary

[4]The Beta distribution is a Dirichlet distribution over two outcomes.

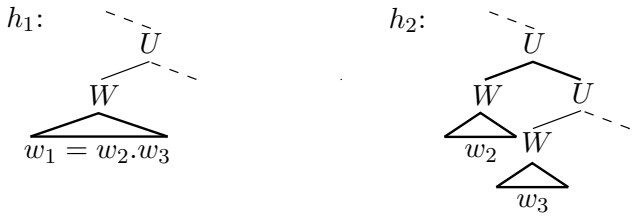

Figure 5.2: The two hypotheses considered by the unigram sampler, where the possible boundary location is between w_2 and w_3. Dashed lines indicate possible additional structure. Rules in bold differ between h_1 and h_2; all other rules are part of h^-.

and h_1 does not (i.e. h_1 contains a single word w_1 spanning the same characters as words w_2 and w_3 in h_2). The structures are shown in Figure 5.2. In order to sample a hypothesis, we need only calculate the relative probabilities of h_1 and h_2. Since h_1 and h_2 are the same except for a few rules, this is straightforward. Let h^- be all of the structure shared by the two hypotheses, including n^- words, and let d be the observed data. Then

$$\begin{aligned} P(h_1 \,|\, h^-, d) &= P(w_1 \,|\, h^-, d) \\ &= \frac{n_{w_1}^{(h^-)} + \alpha_0 P_0(w_1)}{n^- + \alpha_0} \end{aligned} \tag{5.8}$$

where the second line follows from Equation 5.6 and the exchangeability of the CRP. Also,

$$\begin{aligned} & P(h_2 \,|\, h^-, d) \\ &= P(r_{branch}, w_2, w_3 \,|\, h^-, d) \\ &= P(r_{branch} \,|\, h^-, d) P(w_2 \,|\, h^-, d) P(w_3 \,|\, w_2, h^-, d) \\ &= \frac{n_{r_{branch}}^{(h^-)} + \frac{\rho}{2}}{n^- + 1 + \rho} \cdot \frac{n_{w_2}^{(h^-)} + \alpha_0 P_0(w_2)}{n^- + \alpha_0} \cdot \frac{n_{w_3}^{(h^-)} + I(w_2 = w_3) + \alpha_0 P_0(w_3)}{n^- + 1 + \alpha_0} \end{aligned} \tag{5.9}$$

where the denominator of the $n_{r_{branch}}^{(h^-)}$ term is derived by noting that the number of U productions in h^- is $n^- + 1$ (one production for each of the n^- words in h^-, plus one additional production dominating either w_1 or w_3).

After initializing word boundaries at random (or non-randomly; see experiments below),

the Gibbs sampler iterates over the entire data set multiple times. On each iteration, every potential boundary point is sampled once using the equations above. When the sampler converges, these samples will be drawn from the posterior distribution $P(h|d)$.

Although relatively straightforward to implement, this Gibbs sampler has the practical disadvantage that transitions are only possible between states with high similarity. This leads to low mobility through the state space, because movement from one state to another very different state may require transitions through many low-probability intermediate states. Since the initial state is unlikely to be near the high-probability part of the solution space, it may take a very long time for the algorithm to converge to that part of the space.

To alleviate this problem and improve convergence time, I modified the Gibbs sampler to use simulated annealing (Aarts and Korst, 1989). Annealing the sampler causes it to choose low-probability transitions more frequently early in search, which allows it to explore a larger area of the search space more rapidly than otherwise. This is achieved by using a *temperature* parameter γ that starts high and is gradually reduced to 1. Annealing with a temperature of γ corresponds to raising the probabilities in the distribution under consideration (in this case, h_1 and h_2) to the power of $\frac{1}{\gamma}$ prior to sampling. Thus, when $\gamma > 1$, the sampled distribution becomes more uniform, with low-probability transitions becoming more probable. As the temperature is reduced, samples become more and more concentrated in the high-probability areas of the search space. Notice also that if the temperature is reduced below 1, the sampled distribution becomes even more peaked, so that in the limit as $\gamma \to 0$, all probability mass will be concentrated on the mode of the distribution. This means that, by reducing the temperature to almost zero, we can obtain

an approximation to the MAP solution. I use this method in some of the experiments below.

5.3.3 Experiments

5.3.3.1 Data

For all the experiments described in this chapter, I report results on the same corpus used by Brent (1999) and Venkataraman (2001), which allows me to compare my results directly to theirs. The data is derived from the Bernstein-Ratner corpus (Bernstein-Ratner, 1987) of the CHILDES database (MacWhinney and Snow, 1985), which contains orthographic transcriptions of utterances directed at 13- to 23-month-olds. The data was post-processed by Brent, who removed disfluencies and non-words, discarded parental utterances not directed at the children, and converted the rest of the words into a phonemic representation using a phonemic dictionary (i.e. each orthographic form was always given the same phonemic form). The resulting corpus contains 9790 utterances, with 33399 word tokens and 1321 unique types. The average number of words per utterance is 3.41 and the average word length (in phonemes) is 2.87. The word boundaries in the corpus are used as the gold standard for evaluation, but are not provided in the input to the system (except for word boundaries that are also utterance boundaries).

The process used to create this corpus means that it is missing many of the complexities of real child-directed speech. Not the least of these is the acoustic variability with which different tokens of the same word are produced, a factor which presumably makes word segmentation more difficult. On the other hand, the corpus is also missing many cues which could aid in segmentation, such as coarticulation information, stress, and duration. While

this idealization of child-directed speech is somewhat unrealistic, the corpus does provide a way to investigate the use of purely distributional cues for segmentation, and permits direct comparison to other word segmentation systems.

5.3.3.2 Evaluation procedure

For quantitative evaluation, I use the metrics of precision (number of correct items found out of all items found), recall (number of correct items found out of all correct items), and F-score ($= \frac{2*\text{precision}*\text{recall}}{\text{precision}+\text{recall}}$). I report the following scores for each model I propose:

- P, R, F: precision, recall, and F-score on words: both boundaries must be correctly identified to count as correct.
- LP, LR, LF: precision, recall, and F-score on the lexicon, i.e. word types.
- BP, BR, BF: precision, recall, and F-score on potentially ambiguous boundaries (i.e. utterance boundaries are not included in the counts).

For comparison, I report scores as well for Brent's MBDP-1 system (Brent, 1999) and Venkataraman's n-gram segmentation systems (Venkataraman, 2001), which I will refer to as NGS-u and NGS-b (for the unigram and bigram models). Both Brent and Venkataraman use online search procedures, so in their papers they calculate precision and recall separately on each 500-utterance block of the corpus and graph the results to show how scores change as more data is processed. They do not report lexicon recall or boundary precision and recall. Their results are rather noisy, but performance seems to stabilize rapidly, after about 1500 utterances. To facilitate comparison with my own results, I calculated scores for MBDP-1 and NGS over the whole corpus, using Venkataraman's implementations of

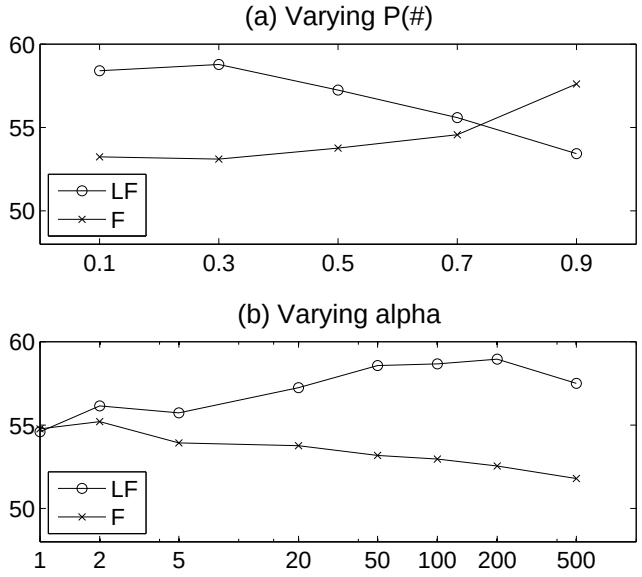

Figure 5.3: Word (F) and lexicon (LF) F-score in the DP model (a) as a function of $p_{\#}$, with $\alpha_0 = 20$ and (b) as a function of α_0, with $p_{\#} = .5$.

these algorithms[5].

Since my algorithm produces random segmentations sampled from the posterior distribution rather than a single optimal solution, there are several possible ways to evaluate its performance. For initial comparison to previous work, I evaluated a single sample taken after 20,000 iterations. The sampler was annealed in 10 increments of 2000 iterations each, with $\frac{1}{\gamma} = \{.1, .2, \ldots, .9, 1\}$. Unless otherwise noted, scores and examples are based on this evaluation procedure. Additional experiments were performed to estimate the variation between different samples, and to evaluate performance using annealing to approximate the MAP solution. These additional evaluation methods are described in more detail below.

	P	R	F	BP	BR	BF	LP	LR	LF
NGS-u	**67.7**	**70.2**	**68.9**	80.6	**84.8**	**82.6**	52.9	51.3	52.0
MBDP-1	67.0	69.4	68.2	80.3	84.3	82.3	53.6	51.3	52.4
DP	61.9	47.6	53.8	**92.4**	62.2	74.3	**57.0**	**57.5**	**57.2**

Table 5.1: Accuracy of the various systems, with best scores in bold. DP results are with $p_{\#} = .5$ and $\alpha_0 = 20$.

5.3.3.3 Results and Discussion

Recall that the DP model has three parameters: ρ (the prior on r_{branch}), $p_{\#}$ (the prior probability of a word boundary), and α_0 (which affects the number of word types proposed). Given the large number of known utterance boundaries, the value of ρ should have little effect on the results, so I simply fixed $\rho = 2$ for all experiments. Figure 5.3 shows the effects of varying of $p_{\#}$ and α_0. Lower values of $p_{\#}$ result in more long words, which tends to improve recall (and thus F-score) in the lexicon. The accompanying decrease in token accuracy is due to an increasing tendency for the model to concatenate short words together, a phenomenon I discuss further below. Higher values of α_0 allow more novel words, which also improves lexicon recall, but begins to degrade precision after a point. Due to the negative correlation between token accuracy and lexicon accuracy, there is no single best value for either $p_{\#}$ or α_0. In the remainder of this section, I discuss results for $p_{\#} = .5, \alpha_0 = 20$ (though others are qualitatively similar).

In Table 5.1, I compare the results of my system to those of MBDP-1 and NGS-u. Although my system has higher lexicon accuracy than the others, its token accuracy is much worse. Performance does not vary a great deal between different samples, since calculating the score for a single sample already involves averaging over many random choices – the choices of whether to place a boundary at each location or not. Table 5.2

[5]The implementations are available at http://www.speech.sri.com/people/anand/.

	F		LF		$-\log P(\mathbf{w} \mid d)$	
	mean	std	mean	std	mean	std
Samples(10 runs)	53.9	.32	57.8	.60	200587	192
Samples(1 run)	53.5	.07	57.7	.43	200493	25
MAP Approx.	53.7	.26	58.7	.56	199853	228

Table 5.2: Comparison of the accuracy and posterior probability of DP solutions found by sampling (averaged over ten independent runs, or over ten samples from the same run) and by approximating the MAP solution (averaged over ten independent runs).

shows the mean and standard deviation in F-scores and posterior probabilities over samples taken from 10 independent runs of the algorithm with different random initializations. The same statistics are also provided for ten samples obtained from a single run of the sampler. Samples from a single run are not independent, so to reduce the amount of correlation between these samples they were taken at 100-iteration intervals (at iterations 19100, 19200, ... 20000). Nevertheless, they show less variability than the truly independent samples. In both cases, lexicon accuracy is more variable than token accuracy, probably because there are far fewer lexical items to average over within a single sample. Finally, Table 5.2 provides results for the approximate MAP evaluation procedure. This procedure is clearly imperfect, since if it were able to identify the true MAP solution, there would be no difference in results across multiple runs of the algorithm. In fact, compared to the standard sampling procedure, there is only slightly less variation in F-scores, and greater variation in probability[6]. Nevertheless, the MAP approximation does succeed in finding solutions with significantly higher probabilities. These solutions also have higher lexicon accuracy, although token accuracy remains low.

The reason that token accuracy is so low with the DP model is that it often mis-analyzes

[6]The large standard deviation in the probabilities of the approximate MAP solutions is due to a single outlier. The standard deviation among the remaining nine solutions is 160, well below the standard deviation in the sample solutions, where there are no outliers.

frequently occurring words. Many instances of these words occur in common collocations such as *what's that* and *do you*, which the system interprets as a single words. This pattern of errors is apparent in the boundary scores: boundary precision is very high, indicating that when the system proposes a boundary, it is almost always correct. Boundary recall is low, indicating undersegmentation.

I analyzed the behavior of the system more carefully by examining the segmented corpus and lexicon. A full 31% of the proposed lexicon and nearly 30% of tokens consist of undersegmentation (collocation) errors, while only 12% of types and 5% of tokens are other non-words. (Some additional token errors, under 4%, are caused by proposing a correct word in an incorrect location.) About 85% of collocations (both types and tokens) are composed of two words, nearly all the rest are three words. To illustrate the phenomenon, I provide the system's segmentation of the first 35 utterances in the corpus in Figure 5.4, and the 35 most frequently found lexical items in Figure 5.5. The 70 most frequent collocations identified as single words by the system are shown in the Figure 5.6.

There is some evidence that young children may treat phrases like *what's that* or social conventions as single units (Peters, 1983). However, many of the other collocations found by the system do not seem cognitively plausible. Other than the Det+N phrases, none of the remaining collocations constitute linguistic units, and there is no evidence that children view them as such. Why, then, are these units found by the DP model? The answer seems clear: groups of words that frequently co-occur violate the unigram assumption in the model, since they exhibit strong word-to-word dependencies. The only way the model can capture these dependencies is by assuming that these collocations are in fact words themselves.

yuwant tu si D6bUk	yu want tu si D6 bUk
lUk D*z 6b7 wIT hIz h&t	lUk D*z 6 b7 wIT hIz h&t
&nd 6dOgi	&nd 6 dOgi
yu wanttu lUk&tDIs	yu want tu lUk &t DIs
lUk&tDIs	lUk &t DIs
h&v6 drINk	h&v 6 drINk
oke nQ	oke nQ
WAtsDIs	WAts DIs
WAtsD&t	WAts D&t
WAtIzIt	WAt Iz It
lUk k&nyu tek ItQt	lUk k&n yu tek It Qt
tek ItQt	tek It Qt
yuwant It In	yu want It In
pUt D&t an	pUt D&t an
D&t	D&t
yEs	yEs
oke	oke
op~ ItAp	op~ It Ap
tek D6dOgi Qt	tek D6 dOgi Qt
9TINk It wIl kAmQt	9 TINk It wIl kAm Qt
lEtssi	lEts si
y&	y&
pUl ItQt	pUl It Qt
WAts It	WAts It
lUk	lUk
lUk	lUk
WAtsD&t	WAts D&t
gEt It	gEt It
gEtIt	gEt It
gEtIt	gEt It
IzD&t f% D6dOgi	Iz D&t f% D6 dOgi
k&nyu fid It tu D6dOgi	k&n yu fid It tu D6 dOgi
fid It	fid It
pUtIt In oke	pUt It In oke
oke	oke

Figure 5.4: The first 35 utterances in the corpus as segmented by the DP model (left) and the correct segmentation (right). The model undersegments the corpus. The stochastic nature of the Gibbs sampling procedure is apparent in the segmentation of the sequence `gEtIt`, which receives two different analyses. A key to the ASCII transcriptions used in the corpus is given in Appendix A.

+ 396 WAt	1704 yu
+ 383 yu	1291 D6
+ 360 oke	895 6
+ 351 tu	798 D&t
+ 340 &nd	783 WAt
+ 337 y&	653 Iz
+ 313 D6	632 It
+ 295 It	588 DIs
+ 293 lUk	569 WAts
- 275 z	528 tu
- 258 WAtsD&t	463 du
+ 248 D&t	429 lUk
+ 235 6	412 k&n
+ 217 no	399 D&ts
+ 196 D&ts	389 si
+ 193 D*	389 D*
+ 189 DIs	382 9
+ 188 si	378 &nd
- 184 k&nyu	375 In
+ 178 In	363 y)
+ 177 y)	362 #
+ 172 h(	360 oke
+ 160 WAts	337 y&
+ 147 Its	301 no
- 141 IN	268 l9k
- 138 IzD&t	266 Its
- 135 WAtsDIs	250 an
+ 123 Iz	246 h(
+ 117 du	246 wAn
+ 116 nQ	244 want
- 112 s	239 pUt
+ 104 f%	227 hi
- 102 D6dOgi	226 wan6
- 101 D&ts6	221 r9t
+ 101 bUk	217 bUk

Figure 5.5: The 35 most frequent items in the lexicon found by the DP model (left) and in the correct lexicon (right). The frequency of each lexical item is shown to its left. Items in the segmented lexicon are indicated as correct (+) or incorrect (-). Frequencies of correct items in the segmented lexicon are lower than in the true lexicon because many occurrences of these items are accounted for by collocations.

Full S or socialization:

```
258 WAts D&t
135 WAts DIs
 65 T&Nk yu
 61 D&ts r9t
 57 b9 b9
 53 WAt Iz It
 41 lUk &t DIs
 38 WAt # Doz
 38 WAt Els
 36 huz D&t
 33 D&t wAn
 31 n9t n9t
 30 lEt mi Qt
 30 sIt dQn
 30 kloz D6 d%
 29 gUd g3l
 28 lUk &t D&t
```

Det + N:

```
102 D6 dOgi
 64 D6 dr&g~
 56 DIs wAn
 47 D6 dOg
 43 D6 b7
 43 D6 bUk
 37 D6 bAni
 36 DIs bUk
 31 6 bUk
 31 D6 d%
 30 y) h&nd
 29 6nADR wAn
```

Aux + NP (+ V):

```
184 k&n yu
138 Iz D&t
 91 du yu
 56 du yu want
 53 wUd yu l9k
 50 Iz It
 48 dId yu
 39 du yu si
 30 # yu
 29 Iz hi
```

Pronoun + Aux/V:

```
 95 yu want
 85 yu k&n
 65 yu wan6
 55 9 TINk
 48 yu l9k
 34 y) g6n6
 33 yu dont
 31 9 si
 29 yu no WAt
 28 9 dont
```

Wh + X:

```
 44 W*z D6
 39 hQ mEni
 36 WAt kAlR
 34 WAt # yu
 30 WAt du yu
```

Other:

```
101 D&ts 6
 87 lUk &t
 78 Its 6
 69 In D*
 51 DIs Iz
 48 an D6
 42 Doz #
 41 Iz f%
 39 pUt It
 38 du It
 36 si D6
 36 In D6
 32 ple wIT
 30 pUt hIm
 28 k9nd 6v
 27 wan6 si
```

Figure 5.6: The 70 most frequently occurring items in the segmented lexicon that consist of multiple words from the true lexicon. These items are all identified as single words; the true word boundaries have been inserted for readability. The frequency of each item is shown to its left.

Seg:	True	None	MBDP-1	NGS-u	DP
NGS-u	204.5	**90.9**	210.7	210.8	183.0
MBDP-1	208.2	321.7	217.0	218.0	**189.8**
DP	222.4	393.6	231.2	231.6	**200.6**

Table 5.3: Negative log probabilities (x 1000) under each model of the true solution, the solution with no utterance-internal boundaries, and the solutions found by each algorithm. The most probable solution under each model is shown in bold.

This analysis raises the question of why MBDP-1 and NGS-u, which also use unigram models, don't exhibit the same problem with collocations. I have already shown that NGS's results are due to its search procedure rather than its model. The same turns out to be true for MBDP-1. I calculated the probability of various segmentations of the corpus under each model, as shown in Table 5.3. These figures indicate that the MBDP-1 model assigns higher probability to the solution found by my Gibbs sampler than to the solution found by Brent's own incremental search algorithm. In other words, the model underlying MBDP-1 *does* favor the lower-accuracy collocation solution, but Brent's approximate search algorithm finds a different solution that has higher accuracy but lower probability under the model.

I performed two experiments suggesting that my own inference procedure does not suffer from similar problems. First, I initialized the Gibbs sampler in three different ways: with no utterance-internal boundaries, with a boundary after every character, and with random boundaries. The results were virtually the same regardless of initialization. In fact, the effects of initialization are almost immediately washed out in the high-temperature initial stage of sampling, as shown in Figure 5.7 (top). Figure 5.7 (bottom) shows a detail of the final stage of sampling (with $\gamma = 1$), illustrating the large degree of overlap in the range of posterior probabilities produced by the three samplers.

For a second experiment testing the convergence of the Gibbs sampler, I created an

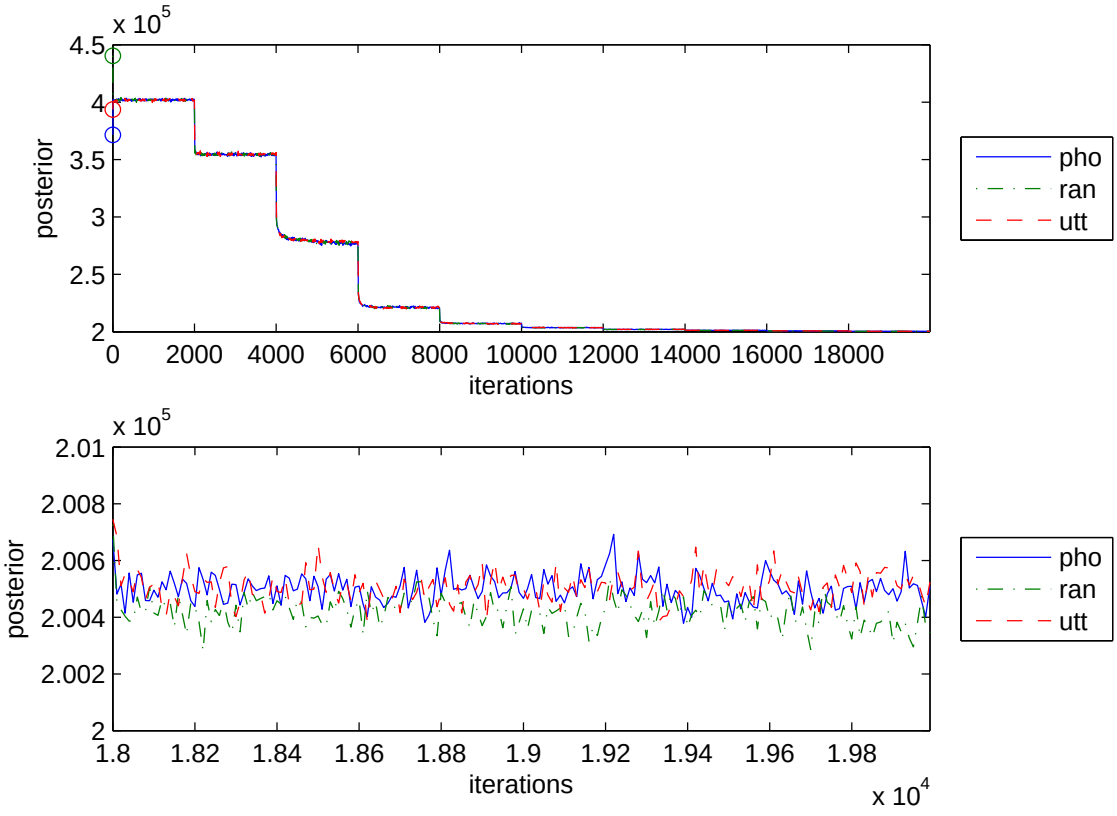

Figure 5.7: Trace plots of the negative log probabilities of samples from samplers for the DP model initialized with a boundary after every phoneme ('pho'), with random boundaries ('ran'), and with a boundary only at the end of each utterance ('utt'). Top: trace plot for the entire run of the algorithm, plotted every 10 iterations. The initial probabilities of each run (circles at $x = 0$) are very different, but within a few iterations the plots are barely distinguishable. Steep drops in the plots occur when the temperature is lowered. Bottom: detail of the final part of the plot.

	P	R	F	BP	BR	BF	LP	LR	LF
NGS-u	76.6	85.8	81.0	83.5	97.6	90.0	60.0	52.4	55.9
MBDP-1	77.0	86.1	81.3	83.7	97.7	90.2	60.8	53.0	56.6
DP	**94.2**	**97.1**	**95.6**	**95.7**	**99.8**	**97.7**	**86.5**	**62.2**	**72.4**

Table 5.4: Accuracy of the various systems on the permuted corpus, with best scores in bold. DP results are with $p_{\#} = .5$ and $\alpha_0 = 20$.

artificial corpus by randomly permuting all the words in the true corpus and arranging them into utterances with the same number of words as in the true corpus. This manipulation creates a corpus where the unigram assumption is correct. If my inference procedure works properly, the unigram system should be able to correctly identify the words in the permuted corpus. This is exactly what I found, as shown in Table 5.4. The performance of the DP model jumps dramatically, and most errors occur on infrequent words (as evidenced by the fact that token accuracy is much higher than lexicon accuracy). In contrast, MBDP-1 and NGS-u receive a much smaller benefit from the permuted corpus, again indicating the influence of search.

These results imply that the DP model itself, rather than the Gibbs sampling procedure I used for inference, is responsible for the poor segmentation performance on the natural language corpus. In particular, the unigram assumption of the model seems to be at fault. In the following section I present some additional experiments designed to further test this hypothesis. In these experiments, I replace the (admittedly impoverished) lexicon generator with a better model of lexical items. If the poor lexical model is responsible for the DP model's undersegmentation of the corpus, then improving the generator should improve performance. However, if the problem is that the unigram assumption fails to account for sequential dependencies in the corpus, then a better lexicon generator will not make much difference.

5.4 The impact of the generator on word segmentation

One possible improvement to the lexicon generator is to replace the assumption of a uniform distribution over phonemes with the more realistic assumption that phonemes have different probabilities of occurrence. This assumption is more in line with the MBDP-1 and NGS models. In NGS, phoneme probabilities are estimated online according to their empirical distribution in the corpus. In MBDP-1, phoneme probabilities are also estimated online, but according to their empirical distribution in the current lexicon. For models like MBDP-1 and the DP model, where the phoneme distribution is used to generate lexicon items rather than word tokens, the latter approach makes more sense. It is relatively straightforward to extend the DP model to infer the phoneme distribution in the lexicon simultaneously with inferring the lexicon itself. Before implementing this extension, however, I tried simply fixing the phoneme distribution to the empirical distribution in the true lexicon. This procedure gives an upper bound on the performance that could be expected if the distribution were learned. I found that this change improved lexicon F-score somewhat (to 60.5, with $\alpha = 20$ and $p_{\#} = .5$), but made almost no difference on token F-score (53.6). Inference of the phoneme distribution was therefore not implemented.

Other changes could be made to the lexicon generator in order to create a better model of word shapes. For example, using a bigram or trigram phoneme model would allow the learner to acquire some notion of phonotactics. Basing the model on syllables rather than phonemes could incorporate constraints on the presence of vowels or syllable weight. Rather than testing all these different possibilities, I designed an experiment to determine an approximate upper bound on performance in the unigram DP model. In this experiment, I provided the model with information that no infant would actually have access to: the

	F-score		% Collocations	
ϵ	Tokens	Lexicon	Tokens	Lexicon
10^{-2}	61.4	82.1	26.9	21.1
10^{-3}	62.5	83.5	26.0	19.3
10^{-4}	63.8	84.6	25.1	17.3
10^{-5}	65.2	85.2	24.1	16.1
10^{-6}	67.7	86.0	22.0	13.9

Table 5.5: Results of the DP model using P_{true}. The percentage of tokens and lexicon entries consisting of collocations is shown for each value of ϵ.

set of word types that occur in the correctly segmented corpus. The lexicon generator is defined as follows:

$$P_{true}(w) = \begin{cases} (1-\epsilon)\frac{1}{|L|} + \epsilon P_0(w) & w \in L \\ \epsilon P_0(w) & w \notin L \end{cases}$$

where L is the true set of lexical items in the data, and ϵ is some small mixing constant. In other words, this model is a mixture between a uniform distribution over the true lexical items and the basic model P_0. If $\epsilon = 0$, the model is constrained so that segmentations may only contain words from the true lexicon. If $\epsilon > 0$, a small amount of noise is introduced so that new lexical items are possible, but have much lower probability than the true lexical items. If the model still postulates collocations when ϵ is very small, we have evidence that the unigram assumption, rather than any failure in the lexicon model, is responsible for the problem.

The results from this model are shown in Table 5.5. Not surprisingly, the lexicon F-scores in this model are very high, and there is a large improvement in token F-scores against previous models. However, considering the amount of information provided to the model, its scores are still rather low, and collocations remain a problem, especially for frequent items.

Considering the case where $\epsilon = 10^{-6}$ yields some insight into the performance of these

models with improved generators. The solution found, with a lexicon consisting of 13.9% collocations, has higher probability than the true solution. This is despite the fact that the most probable incorrect lexical items are about five orders of magnitude less probable than the true lexical items[7]. These incorrect lexical items are proposed despite their extremely low probability because only the first occurrence of each word is accounted for by the lexicon generator. Subsequent occurrences are accounted for by the unigram adaptor, so low-probability lexical items incur no additional probability cost after the first occurrence. This is why the collocations remaining in the DP model using P_{true} are the highest-frequency collocations: over many occurrences, the probability mass gained by modeling these collocations as single words outweighs the mass lost in generating the first occurrence.

The results of this experiment suggest that problems with the lexicon generator are not primarily responsible for the large number of collocations found by the unigram DP model. Regardless of how good a model the lexicon generator is, it will not be able to completely overcome the influence of the unigram adaptor when modeling the full corpus. In order to reduce the number of collocations, it is necessary to improve the adaptor by accounting for sequential dependencies between words. I show how to do so in the following section.

5.5 Bigram word segmentation

5.5.1 The hierarchical Dirichlet process model

The results of my unigram experiments suggest that word segmentation can be improved by taking into account dependencies between words. To test this hypothesis, I extended

[7]There are 1321 lexical items in the corpus, so the generator probability of each of these is approximately 10^{-3}. There are 50 phonemes and $p_{\#} = .5$, so a single-character word has probability .01 under P_0. Multiplying by the discount factor $\epsilon = 10^{-6}$ yields $P_{true} = 10^{-8}$ for one-character words not in the true lexicon. Longer incorrect words will have much lower probability.

the DP model to incorporate bigram dependencies using a *hierarchical Dirichlet process* (HDP) (Teh et al., 2005). This approach is similar to previously proposed n-gram models using hierarchical Pitman-Yor processes (Goldwater et al., 2006; Teh, 2006). The HDP is appropriate for situations in which there are multiple distributions over similar sets of outcomes, and the distributions are believed to be similar. For language modeling, we can define a bigram model by assuming each word has a different distribution over the words that follow it, but all these distributions are linked. The definition of the HDP bigram language model (disregarding utterance boundaries for the moment) is

$$\begin{aligned} w_i \,|\, w_{i-1} = w,\ H_w &\sim H_w & \forall w \\ H_w \,|\, \alpha_1, G &\sim \mathrm{DP}(\alpha_1, G) & \forall w \\ G \,|\, \alpha_0, P_0 &\sim \mathrm{DP}(\alpha_0, P_0) \end{aligned}$$

That is, $P(w_i \,|\, w_{i-1} = w)$ is distributed according to H_w, a DP specific to word w. H_w is linked to the DPs for all other words by the fact that they share a common base distribution G, which is generated from another DP. This model is $\mathrm{TwoStage}(\mathrm{CRP}(\alpha_1), \mathrm{DP}(\alpha_0, P_0))$.

As in the unigram model, H_w and G are never represented explicitly. By integrating over them, we get a distribution over bigram frequencies that can be understood in terms of the CRP, as illustrated in Figure 5.8. Each word type w is associated with its own restaurant, which represents the distribution over words that follow w. Different restaurants are not completely independent, however: the labels on the tables in the restaurants are all chosen from a common base distribution, which is represented by another CRP. A word w' that has high probability in the base distribution will tend to appear in many different bigram types (i.e. following many other word types). However, $P(w' \,|\, w)$ may be very different for different w, since each w has its own restaurant for bigram counts.

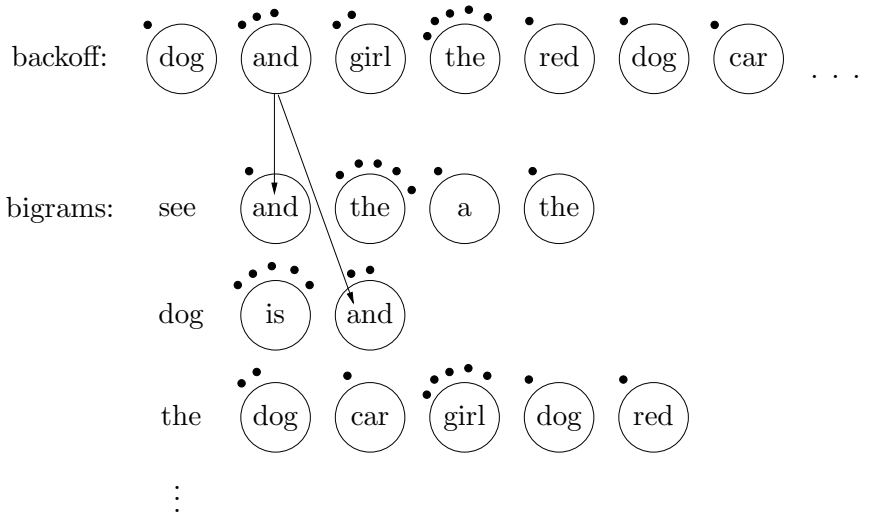

Figure 5.8: Bigrams are modeled using a hierarchical Chinese restaurant process. Each word type w has its own restaurant to represent the distribution of tokens following w in the data. The labels on the tables in these bigram restaurants are drawn from the distribution in the backoff or "master" restaurant (top). Each customer (black dot) in the bigram restaurants represents a bigram token; each customer in the backoff restaurant represents a label on some bigram table.

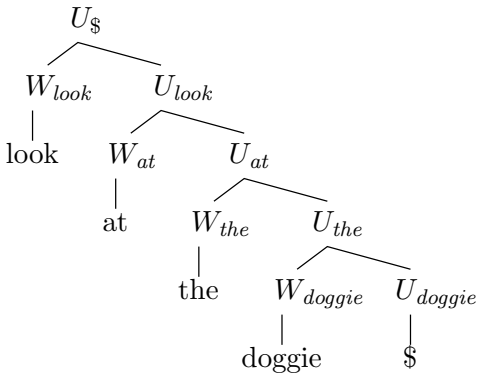

Figure 5.9: A hypothetical utterance as parsed by the bigram HDP grammar.

As in the unigram model, we can account for utterance boundaries using a grammar construction. In the bigram grammar, the utterance boundary marker \$ is considered a special word type, so that w_i ranges over $\Sigma^* \cup \{\$\}$. After observing $\mathbf{w}_{-i}$, the HDP grammar is

$$\begin{array}{rll} P_2(w_i \,|\, \mathbf{w}_{-i}, \mathbf{z}_{-i}) & U_{w_{i-1}} \rightarrow W_{w_i}\ U_{w_i} & \forall w_i \in \Sigma^*, w_{i-1} \in \Sigma^* \cup \{\$\} \\ P_2(\$ \,|\, \mathbf{w}_{-i}, \mathbf{z}_{-i}) & U_{w_{i-1}} \rightarrow \$ & \forall w_{i-1} \in \Sigma^* \\ 1 & W_{w_i} \rightarrow w_i & \forall w_i \in \Sigma^* \end{array}$$

See Figure 5.9 for an illustration. The distribution P_2, which gives the bigram posterior probability of w_i, is derived as follows:

$$\begin{aligned} P_2(w_i \,|\, \mathbf{w}_{-i}, \mathbf{z}_{-i}) &= \int P(w_i \,|\, H_{w_{i-1}}) P(H_{w_{i-1}} \,|\, \mathbf{w}_{-i}, \mathbf{z}_{-i})\ dH_{w_{i-1}} \\ &= \frac{n_{(w_{i-1},w_i)} + \alpha_1 P_1(w_i \,|\, \mathbf{w}_{-i}, \mathbf{z}_{-i})}{n_{w_{i-1}} + \alpha_1} \end{aligned} \tag{5.10}$$

where $n_{(w_{i-1},w_i)}$ is the number of occurrences of the bigram (w_{i-1}, w_i) and

$$\begin{aligned} & P_1(w_i \,|\, \mathbf{w}_{-i}, \mathbf{z}_{-i}) \\ &= \begin{cases} \int P(\$ \,|\, \theta) P(\theta \,|\, \mathbf{w}_{-i}, \mathbf{z}_{-i})\ d\theta & w_i = \$ \\ (1 - \int P(\$ \,|\, \theta) P(\theta \,|\, \mathbf{w}_{-i}, \mathbf{z}_{-i})\ d\theta) \cdot \int P(w_i \,|\, G) P(G \,|\, \mathbf{w}_{-i}, \mathbf{z}_{-i})\ dG & w_i \in \Sigma^* \end{cases} \\ &= \begin{cases} \frac{t_\$ + \frac{\rho}{2}}{t + \rho} & w_i = \$ \\ \frac{t_{\Sigma^*} + \frac{\rho}{2}}{t + \rho} \cdot \frac{t_{w_i} + \alpha_0 P_0(w_i)}{t_{\Sigma^*} + \alpha_0} & w_i \in \Sigma^* \end{cases} \end{aligned}$$

where θ is the parameter of a binomial distribution; $t_\$, t_{\Sigma^*}$, and t_{w_i} are the total number of bigram tables (across all words) labeled with \$, non-\$, and w_i, respectively; and $t = t_\$ + t_{\Sigma^*}$ is the total number of bigram tables. I have suppressed the superscript $(\mathbf{w}_{-i})$ notation in all cases. P_1 is the posterior estimate of the base distribution shared by all bigrams, and can be viewed as a unigram backoff. In P_1, utterance boundaries are generated from a binomial

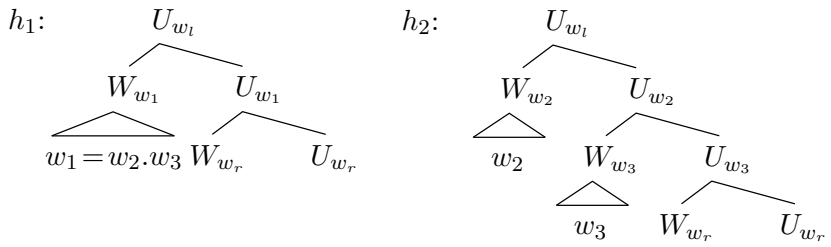

Figure 5.10: The portions of the derivation trees that differ between the two hypotheses considered by the Gibbs sampler for the HDP model.

distribution with parameter θ and other words are generated from the DP G. Since θ and G determine the probability that a word type appears on a bigram table, P_1 is estimated from the number of tables on which each type appears. In other words, when a particular bigram sequence (w_{i-1}, w_i) is never observed in $\mathbf{w}_{-i}$, the probability of w_i following w_{i-1} is estimated using the number of different word types that have been observed to precede w_i. If this number is high, then $P(w_i|w_{i-1})$ will be higher than if this number is low[8].

5.5.2 Gibbs sampler

Inference can be performed on the HDP bigram model using a Gibbs sampler similar to the unigram sampler. To sample from the posterior distribution over segmentations in the bigram model, I define h_1 and h_2 as I did in the unigram sampler so that for the corpus substring s, h_1 has a single word ($s = w_1$) where h_2 has two ($s = w_2.w_3$). Let w_l and w_r be the words (or \$) preceding and following s. The rules that differ between h_1 and h_2 are

[8]Many standard n-gram procedures use similar kinds of estimates based on both type and token counts. In fact, Kneser-Ney smoothing (Kneser and Ney, 1995), a particularly effective smoothing technique for n-gram models (Chen and Goodman, 1998), has been shown to fall out naturally as the posterior estimate in a hierarchical Bayesian language model similar to the one described here, with the DPs replaced by Pitman-Yor processes (Goldwater et al., 2006; Teh, 2006).

shown in Figure 5.10. The posterior probability of h_1 can be calculated as

$$\begin{aligned} & P(h_1 \,|\, h^-, d) \\ &= P(r_{(w_l,w_1)} \,|\, h^-, d) \cdot P(r_{(w_1,w_r)} \,|\, r_{(w_l,w_1)}, h^-, d) \\ &= \frac{n_{(w_l,w_1)} + \alpha_1 P_1(w_1 \,|\, h^-, d)}{n_{w_l} + \alpha_1} \cdot \frac{n_{(w_1,w_r)} + I(w_l = w_1 = w_r) + \alpha_1 P_1(w_r \,|\, h^-, d)}{n_{w_1} + 1 + \alpha_1} \end{aligned} \tag{5.11}$$

where $r_{(w_i,w_j)}$ is shorthand for the grammar rule $U_{w_i} \rightarrow W_{w_j}\ U_{w_j}$ and all counts are with respect to h^-. The terms for the W productions have been left out, since they all have probability 1. Similarly, the posterior probability of h_2 is

$$\begin{aligned} & P(h_2 \,|\, h^-, d) \\ &= P(r_{(w_l,w_2)} \,|\, h^-, d) \cdot P(r_{(w_2,w_3)} \,|\, r_{(w_l,w_2)}, h^-, d) \cdot P(r_{(w_3,w_r)} \,|\, r_{(w_l,w_2)}, r_{(w_2,w_3)}, h^-, d) \\ &= \frac{n_{(w_l,w_2)} + \alpha_1 P_1(w_2 \,|\, h^-, d)}{n_{w_l} + \alpha_1} \cdot \frac{n_{(w_2,w_3)} + I(w_l = w_2 = w_3) + \alpha_1 P_1(w_3 \,|\, h^-, d)}{n_{w_2} + 1 + \alpha_1} \\ &\quad \cdot \frac{n_{(w_3,w_r)} + I(w_l = w_3, w_2 = w_r) + I(w_2 = w_3 = w_r) + \alpha_1 P_1(w_r \,|\, h^-, d)}{n_{w_3} + 1 + I(w_2 = w_4) + \alpha_1} \end{aligned} \tag{5.12}$$

$P_1(.)$ can be calculated exactly using Equation 5.10, but this requires explicitly tracking and sampling the assignment of words to tables, which is computationally expensive. Instead, I used an approximation, replacing each table count t_{w_i} by its expected value $E[t_{w_i}]$. In a $\mathrm{DP}(\alpha, P)$, the expected number of CRP tables for an item occurring n times is $\alpha \log \frac{n+\alpha}{\alpha}$ (Antoniak, 1974), so

$$E[t_{w_i}] = \alpha_1 \sum_j \log \frac{n_{(w_j,w_i)} + \alpha_1}{\alpha_1}$$

This approximation requires only the bigram counts, which must be tracked anyway.

	P	R	F	BP	BR	BF	LP	LR	LF
NGS-b	68.1	68.6	68.3	81.7	82.5	82.1	54.5	57.0	55.7
HDP	**79.4**	**74.0**	**76.6**	**92.4**	**83.5**	**87.7**	**67.9**	**58.9**	**63.1**

Table 5.6: Bigram system accuracy, with best scores in bold. HDP results are with $p_{\#} = .5$, $\alpha_0 = 1000$, and $\alpha_1 = 10$.

5.5.3 Experiments

I used the same basic setup for my experiments with the HDP model as I used for the DP model. The model was initialized by treating each utterance as a single word[9]. I experimented with different values of α_0 and α_1, keeping $p_{\#} = .5$ throughout. Some results of these experiments are plotted in Figure 5.11. In the bigram model, there is now a positive correlation between type and token accuracy, and with appropriate parameter settings, both are higher than in the unigram model (dramatically so, for tokens). High-frequency words are segmented correctly far more often than in the unigram model. The best values of α_0 are much larger than in the unigram model, presumably because all unique word types must be generated via P_0, but in the bigram model there is an additional level of discounting (the unigram process) before reaching P_0. Smaller values of α_0 lead to fewer word types with fewer characters on average, which causes oversegmentation. The effect of α_1 is less pronounced. Larger values lead to more bigram types and a distribution that is more similar to the unigram distribution in the corpus. Smaller values create greater disparities between the bigram distributions for each word.

Table 5.6 compares the results of the HDP model using optimal parameter settings to the only previous model incorporating bigram dependencies, NGS-b. Due to search, the

[9]This initialization is different from the random initialization used for most of the DP model experiments. Based on the results of those experiments, and preliminary analysis of the HDP model results, it was assumed that initialization would not affect the results reported here. More complete later analysis (described below) supports this assumption.

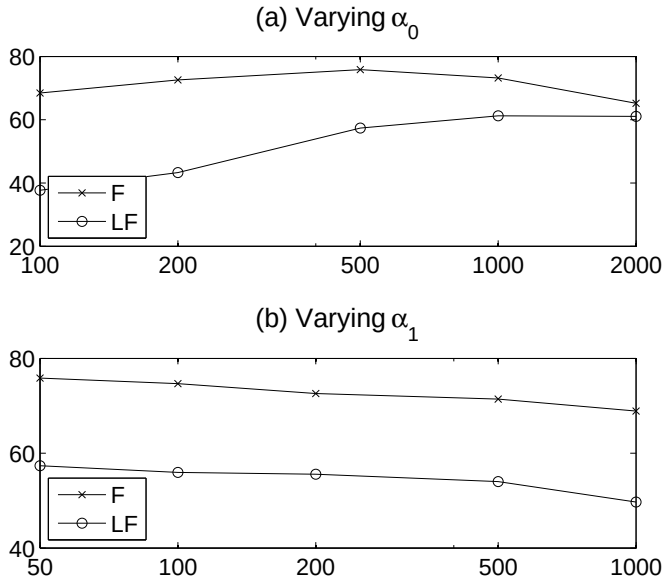

Figure 5.11: Word (F) and lexicon (LF) F-score in the HDP model (a) as a function of α_0, with $\alpha_1 = 10$ and (b) as a function of α_1, with $\alpha_0 = 1000$.

performance of the bigram NGS model is not much different from that of the unigram model. In contrast, the HDP model performs far better than the DP model, leading to the highest published accuracy for this corpus that I am aware of, on both tokens and lexical items. Figure 5.12 gives some example results.

Not only are the results of the bigram model much better than those of the basic unigram model, they are qualitatively different. In the unigram model, type accuracy is higher than token accuracy, indicating many errors on frequent words. In the bigram model, the opposite is true: frequent words are much more likely to be segmented correctly, so token accuracy is higher than type accuracy. As Table 5.7 shows, the bigram model does make some collocation errors, but they are far less common (both proportionally and absolutely) than in the unigram model, especially with frequent words. Other kinds of errors make up a larger proportion of the errors in the bigram model. A particularly interesting kind of

yu want tu si D6 bUk	+ 1065 D6
lUk D*z 6 b7 wIT hIz h&t	+ 922 6
&nd 6 dOgi	+ 875 yu
yu want tu lUk&t DIs	+ 720 D&t
lUk&t DIs	+ 599 WAt
h&v 6 drINk	+ 550 WAts
oke nQ	+ 517 DIs
WAts DIs	+ 485 It
WAts D&t	- 485 z
WAtIz It	+ 461 tu
lUk k&nyu tek It Qt	+ 399 D&ts
tek It Qt	+ 386 si
yuwan t It In	+ 379 In
pUt D&t an	+ 359 oke
D&t	+ 356 &nd
yEs	+ 352 D*
oke	+ 337 y&
op~ It Ap	+ 317 9
tek D6 dOgi Qt	+ 279 lUk
9 TINk It wIl kAmQt	+ 273 Its
lEts si	+ 269 no
y&	+ 265 y)
pUl It Qt	+ 264 wAn
WAts It	+ 253 l9k
lUk	- 251 duyu
lUk	+ 248 bUk
WAts D&t	- 229 k&nyu
gEt It	+ 225 h(
gEt It	+ 223 want
gEt It	+ 220 wan6
IzD&t f% D6 dOgi	+ 201 pUt
k&nyu fid It tu D6 dOgi	+ 200 Iz
fid It	+ 197 hIz
pUtIt In oke	+ 194 du
oke	+ 193 dOgi

Figure 5.12: Results for the HDP model with $p_{\#} = .5$, $\alpha_0 = 1000$, and $\alpha_1 = 10$: the first 35 segmented utterances (left) and the 35 most frequent lexical items (right). Fewer collocations appear than in the DP model, there are fewer errors on high-frequency words, and word frequencies match the true frequencies (Figure 5.5) more closely.

	Count		Collocations		Other non-words		Placement
	Tokens	Lexicon	Tokens	Lexicon	Tokens	Lexicon	Tokens
DP(α_0, P_0)	25677	1331	31.0	29.6	11.9	4.7	3.8
DP(α_0, P_{true})	27295	1347	13.9	22.0	1.0	1.1	1.5
HDP	31128	1146	16.9	10.6	15.2	4.9	5.1

Table 5.7: Error analysis for two unigram models and the bigram model. Figures give the number of proposed tokens and lexical items, and the percentage of those consisting of collocations, other items not in the true lexicon, and placement errors (words belonging to the true lexicon, but proposed in the wrong location). Parameters for the DP models were $p_\# = .5$, $\alpha_0 = 20$. The mixing constant in the DP(α_0, P_{true}) model was $\epsilon = 10^{-6}$. Parameters for the HDP model were $p_\# = .5$, $\alpha_0 = 1000$, and $\alpha_1 = 10$.

error is the segmentation of suffixes as individual words. The top 100 most frequent lexical items proposed by the bigram model include `z`, `s`, `IN`, `i`, and `t`, which correspond to plural, progressive, diminutive, and past tense endings. This effect suggests that incorporating a notion of morphology into the lexicon generator could improve results. I explore this possibility in the next chapter.

Comparison of the bigram model to the DP(α_0, P_{true}) model is particularly enlightening. Access to the true word types gives the unigram model much higher accuracy on lexical items, but frequent items are still analyzed as collocations at a much higher rate than in the bigram model. The net result is that the bigram model scores better on token accuracy, even though it is completely unsupervised. This difference between type accuracy and token accuracy is not surprising: the contextual dependencies built into the bigram model primarily encode information about the behavior of word tokens. With even a small amount of uncertainty in the contents of the lexicon, a model that doesn't take word usage into account will have difficulty segmenting natural language. On the other hand, incorporating contextual dependencies seems to be helpful not only for learning about likely sequences of words, but also for building an accurate lexicon. We see evidence in the improvement in

	F		LF		$-\log P(\mathbf{w} \mid d)$	
	mean	std	mean	std	mean	std
Samples(10 runs)	75.9	.91	61.5	1.03	183084	163
Samples(1 run)	76.5	.06	61.8	.17	182902	21
MAP Approx.	75.5	.86	61.6	.59	182970	126

Table 5.8: Comparison of the accuracy and posterior probability of HDP solutions found by sampling (averaged over ten independent runs, or over ten samples from the same run) and by approximating the MAP solution (averaged over ten independent runs).

lexicon accuracy between the unsupervised unigram model and the bigram model.

As in the unigram model, I performed additional experiments to examine the amount of variability in the results produced by a single sample of the bigram model. Average results over ten samples are shown in Table 5.8, and indicate that the bigram sampler yields more varied scores than the unigram sampler. On the other hand, variation in the posterior probabilities of samples from the bigram model is lower than in the unigram model. This suggests that the correlation between probability and performance is somewhat weaker in the bigram model than in the unigram model. The effect is not very large, and may be due to the approximation of table counts used in the bigram model, which was also used in calculating the bigram posterior.

Also shown in Table 5.8 are the results of the MAP approximation for the bigram model. The difference between the average MAP results and the average (10-run) sampled results is not statistically significant, and the variation in MAP results is almost as great as that in the sampled results. The trace plots shown in Figure 5.13 suggest a possible explanation. Three different initialization conditions are shown. At the beginning of sampling, the different forms of initialization are quickly washed out, and the three runs overlap in probability space. However, by the end of sampling the three runs no longer overlap. This suggests that the mobility of the algorithm is too low relative to the number of iterations considered,

and each sampler is only observed to explore a small part of the search space. Because of annealing, the samplers are all likely to be near the mode of the distribution as their mobility decreases, so they end up fairly close to each other in probability space (and in terms of performance). Annealing past $\gamma = 1$ for the MAP approximation has little effect, because the samplers are already relatively confined in their exploration of solutions. If this analysis is correct, it suggests that the sampled solutions are already very close to the MAP, and that annealing more slowly would probably bring them even closer and make them more similar. Achieving a more representative sample of the posterior on a reasonable timescale would probably require redesigning the sampling algorithm to increase its mobility.

5.6 General discussion

The experiments presented in this chapter strongly suggest that the unigram assumption made by most previous model-based approaches to word segmentation presents an obstacle to learning. I have argued that the results presented by previous researchers are misleading, because the results of their systems are overly influenced by search. The evidence from my own experiments indicates that the Gibbs samplers I implemented give a more accurate picture of the solutions preferred by different models. In particular, I found that models incorporating a unigram assumption tend to undersegment the data, concatenating common sequences of words. I showed that incorporating sequential dependencies into a model of word segmentation can greatly reduce this problem, improving accuracy on both lexical items and word tokens.

It is important to consider the implications of these results for the kinds of statistical learning experiments exemplified by Saffran et al. (1996a). As I mentioned earlier, most of

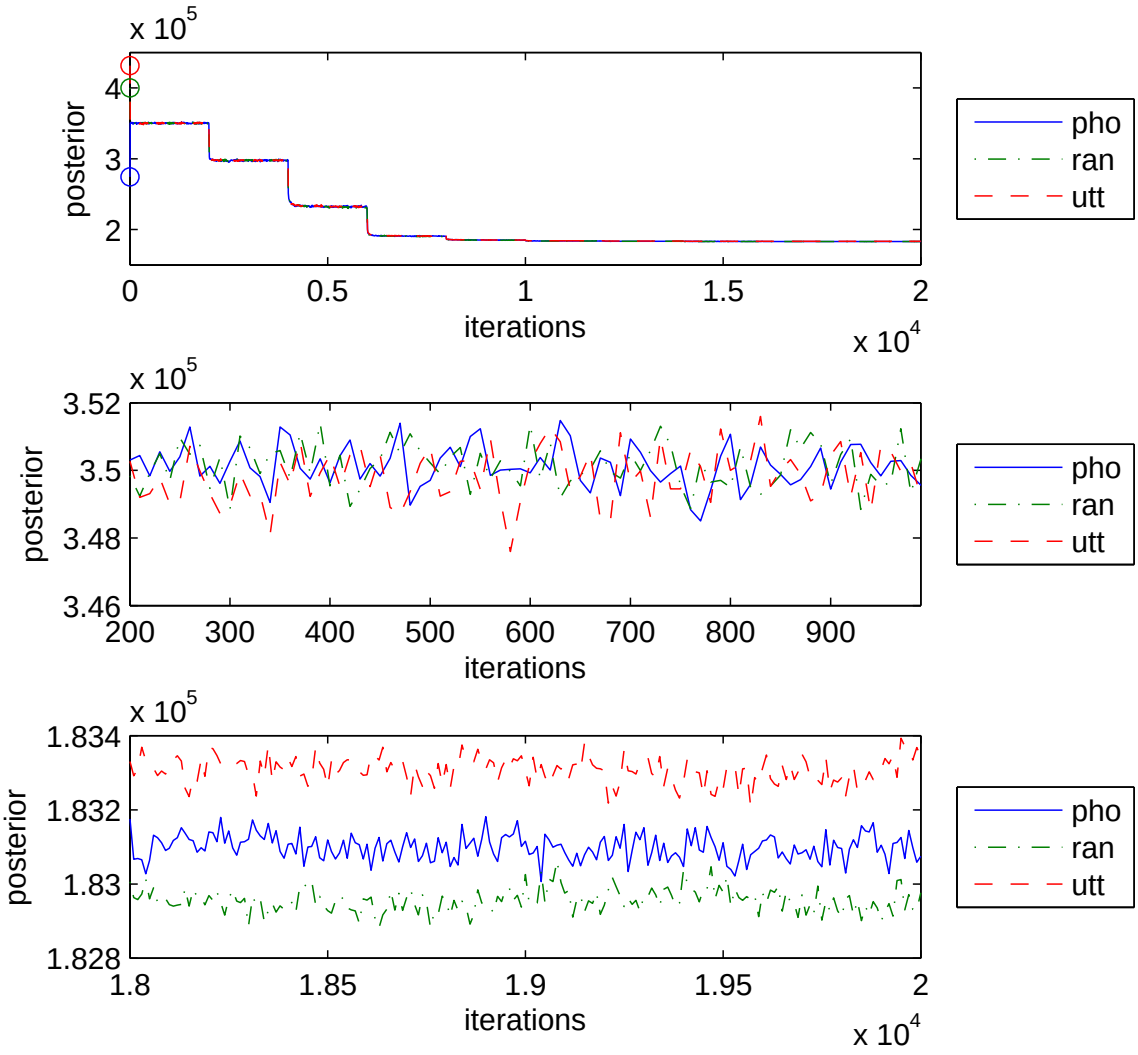

Figure 5.13: Trace plots of the negative log probabilities of samples from samplers for the HDP model initialized with a boundary after every phoneme ('pho'), with random boundaries ('ran'), and with a boundary only at the end of each utterance ('utt'). Top: trace plot for the entire run of the algorithm, plotted every 10 iterations, with the initial probabilities of each run circled at $x = 0$. Middle: detail from the beginning of the plot. Bottom: detail of the final part of the plot.

these experiments use stimuli that are constructed according to a unigram model. Thus, following a word-final syllable[10], there is a uniform probability of observing any word-initial syllable – transitions between syllables at word boundaries are completely random. Transitions between syllables within words are non-random, and therefore have higher probability. Implicit in this design is the idea that unpredictable transitions always occur at word boundaries, and predictable transitions always occur within words. My experiments suggest that this is an oversimplification, because words that frequently occur in sequence create predictable transitions across word boundaries. In other words, high transitional probabilities can occur in natural language *either* because there is no word boundary, *or* because there is a boundary between two words that frequently co-occur. Accounting for both of these possibilities seems to be important for distributional segmentation to succeed.

These subtleties of transitional probabilities point to a need to revisit some of the behavioral work used to establish the possibility of distributional word segmentation in humans. In particular, it would be valuable to know whether humans are able to track and use bigram statistics as well as unigram statistics for word segmentation. Unfortunately, moving beyond unigrams would considerably complicate the artificial language stimuli, which might make typical statistical learning paradigms infeasible. However, it might be possible to investigate this question less directly by probing adults' sensitivity to bigram frequencies, or by examining the kinds of errors made by young children.

In addition to these behavioral investigations, it is important to continue exploring

[10] Unlike the models presented here, where words are constructed from phoneme-sized units, the words in statistical learning experiments are typically constructed from syllabic units. Given the results of my experiments with modifying the unigram generator, this difference seems unlikely affect the conclusions drawn here.

distributional word segmentation via computational modeling. In my experiments, I followed the convention of many previous researchers in using a phonemically transcribed and normalized corpus, which allowed me to compare the performance of different systems. However, it is still an open question how much influence the input representation has on the performance of distributional word segmentation systems. It would be useful to investigate whether a more phonetically accurate representation leads to different kinds of results. In addition, prosodic cues are believed by many researchers to be very important in early word segmentation. In future work, I hope to explore how the incorporation of stress or other prosodic information into the model affects results. Based on the hypothesis of this dissertation, I would expect that combining multiple sources of information in this way would lead to improved performance.

Another question that is worth exploring in future is how my results are affected by processing different amounts of data. My analysis of the unigram models suggests that additional data would be likely to degrade performance, since it is the more frequent words that tend to be mis-analyzed by these systems. On the other hand, the performance of the bigram model is less easy to predict, making this a worthwhile experiment.

A final area for future work lies in examining possible differences between languages. The models I have presented incorporate almost no language-specific information, but the optimal settings of the sparseness parameters might vary between languages. In addition, I am interested in the possibility of incorporating general lingustic tendencies (language universals) into these models, probably as part of the lexicon generator. The models presented here contain only very general constraints (i.e. prefer shorter words, and prefer fewer words), but performance could very likely improve with the addition of some basic linguistic

intuition (e.g. words tend to contain vowels). If these kinds of constraints are added to the model, it would be very important to test on a variety of different languages.

5.7 Conclusion

In this chapter, I have shown how to develop unigram and bigram models of word segmentation based on the two-stage framework. Using these models, I have presented evidence that, contrary to previous results, unigram modeling is not sufficient for successful word segmentation based on distributional cues. My experiments illustrate the advantages of the two-stage approach: I was able to develop several different models combining different modular components, and thus examine the effects of each of these components individually. I have argued that my results are more informative than previous work, because other researchers have (intentionally or unintentionally) relied on algorithms that constrained the solutions found by their systems. The effects of these algorithms are such that, on a corpus that conforms to the modeling assumptions underlying these systems, results are still far from ideal. In contrast, the Gibbs sampler used in my own system finds a solution that is close to correct in this circumstance. This and other evidence strongly suggests that the benefit I found in modeling contextual dependencies for word segmentation is a real effect of the model, rather than an artifact of the particular inference procedure used.

To my knowledge, the bigram model I have presented here achieves higher performance on this corpus than any previously published unsupervised system. However, it is not error-free. One particularly common type of error is in proposing bound morphemes (inflectional and derivational suffixes) as separate words. In the following chapter, I discuss how the two-stage modeling framework could be used to develop a model incorporating both word

segmentation and morphological analysis capabilities, potentially solving this problem.

Chapter 6

Conclusion

6.1 Introduction

I began this thesis with the claim that successful language acquisition requires interaction between different components of the grammar and integration of different sources of information. In Chapters 4 and 5, I discussed experiments designed to address questions relevant to the acquisition of morphology and word segmentation. The models used to perform these experiments were developed within the two-stage modeling framework described in Chapter 3, whose flexibility allowed me to investigate the effects of various sources of information, including context and learning biases. However, the question of how different components of the learner's grammar interact during learning has yet to be addressed. The implementation of a complete system combining word segmentation and morphology is beyond the scope of this thesis, but it is important to emphasize that such a system is completely feasible using models and algorithms very similar to those presented here. The probabilistic model-based approach embodied by the two-stage framework makes it relatively straightforward to layer multiple components, with Gibbs or Metropolis-Hastings sampling for inference. In the remainder of this chapter, I sketch out how a combined word segmentation-morphology model could be built, while also reviewing the models and results obtained for the individual components. I then discuss some additional directions for future work and conclude.

6.2 Combining word segmentation and morphology

In this thesis, I have developed several models within the two-stage framework in order to address questions regarding the kinds of information that are useful for morphological acquisition and word segmentation. In particular, my experiments in Chapter 4 used a

Pitman-Yor morphology model to investigate the use of corpus frequencies in morphological acquisition. My results on a large set of English verbs and on transcribed child-directed speech suggest that partially or fully discounting corpus frequencies leads to better morphological generalization. Previous computational systems for acquiring morphology have often used lists of lexical items as input, which is consistent with my results, but leads to the question of how improved morphological knowledge feeds back into other areas of processing, including word segmentation. One would expect, for example, that a child who recognizes that *-ing* is a common verbal ending in English would be more likely to correctly segment an utterance containing a novel progressive verb form than a child without this morphological knowledge. Correct segmentation could in turn lead to the addition of a new word to the lexicon, the analysis of a new stem, and the generation of other inflected forms from that stem.

The advantage of the modeling framework presented here is that it provides a way for multiple grammar components to influence each other in precisely the way I have just described. Rather than preprocessing a tokenized corpus to extract a lexicon for input, the generator-adaptor framework allows the morphology model to accept the full corpus as input, using the adaptor to downweight corpus frequencies within the model itself. This raises the possibility of using an input corpus in which word boundaries have not yet been identified, and learning both word and morpheme boundaries simultaneously. In this section, I discuss how the models presented in the previous chapters can be extended and combined to do exactly that[1].

[1]It is worth pointing out that de Marcken (1996) also developed a model capable of discovering both word and morpheme (as well as phrasal) boundaries. The model suggested here differs from de Marcken's in that words and morphemes are conceptually different, and modeled differently: words come in unbounded sequences, while morphemes come in pairs and are associated with classes. In de Marcken's model, lexical units (sequences of characters) are hierarchical but conceptually undifferentiated, so that any size unit may or may not contain smaller subunits.

6.2.1 Extending the morphology model

Recall the model of morphological acquisition presented in Section 4.4, which was defined as TwoStage(PY$(a, b), P_\mu$), with $P_\mu(c, t, f) = P(c)P(t \mid c)P(f \mid c)$. The probability distributions over classes, stems, and suffixes were taken to be multinomial, with symmetric Dirichlet priors. As a result, the generator in this model is a distribution over a finite set of (class, stem, suffix) combinations, each of which has the same prior probability. As I have already discussed, the assumption of a uniform prior over a finite set is not very realistic, but it was sufficient for my experiments because the word types in the corpus were known, and there were relatively few of them. Therefore, only stems and suffixes that were prefix or suffix strings of the words in the corpus had to be considered. The number of prefix and suffix strings could be easily pre-computed, and most of these strings are plausible as stems or suffixes (i.e. they are all relatively short and do not violate English phonotactics).

In moving to a corpus where word boundaries are not known, the assumptions made in Chapter 4 become problematic. Any substring of the corpus could in theory constitute a stem or suffix, which causes a blowup in the number of possible stems and suffixes, while also making most of them linguistically implausible. It therefore makes sense to adopt a different generator model that places a distribution over an infinite number of possible stems and suffixes, with some far more likely than others. To see how this can be done, consider

first the finite generator model:

$$\begin{aligned}
c_k \,|\, \theta_c &\sim \text{Multinomial}(\theta_c) \\
\theta_c \,|\, \kappa &\sim \text{Dirichlet}\,(\kappa) \\
t_k \,|\, c_k = c,\ \theta_{t|c} &\sim \text{Multinomial}(\theta_{t|c}) \quad \forall c \\
\theta_{t|c} \,|\, \tau &\sim \text{Dirichlet}\,(\tau) \quad \forall c \\
f_k \,|\, c_k = c,\ \theta_{f|c} &\sim \text{Multinomial}(\theta_{f|c}) \quad \forall c \\
\theta_{f|c} \,|\, \phi &\sim \text{Dirichlet}\,(\phi) \quad \forall c
\end{aligned} \tag{6.1}$$

where c_k, t_k, and f_k are the class, stem, and suffix associated with table k, and $\theta_{t|c}$ and $\theta_{f|c}$ indicate the parameters of the multinomials from which the stems and suffixes in class c are drawn. A straightforward way to extend this model to the infinite case is by replacing the multinomial-Dirichlet distributions over stems and suffixes by Dirichlet processes:

$$\begin{aligned}
c_k \,|\, \theta_c &\sim \text{Multinomial}(\theta_c) \\
\theta_c \,|\, \kappa &\sim \text{Dirichlet}\,(\kappa) \\
t_k \,|\, c_k = c,\ T_c &\sim T_c \quad \forall c \\
T_c \,|\, \alpha_\mu, P_0 &\sim \text{DP}(\alpha_t, P_0) \quad \forall c \\
f_k \,|\, c_k = c,\ T_c &\sim F_c \quad \forall c \\
F_c \,|\, \alpha_\mu, P_0 &\sim \text{DP}(\alpha_f, P_0) \quad \forall c
\end{aligned} \tag{6.2}$$

where α_t and α_f are concentration parameters for the stem and suffix DPs, and P_0 is the unigram phoneme model introduced in Equation 5.5 as the lexicon generator for word segmentation. In other words, stems and suffixes are now generated by the unigram phoneme model P_0, which is a distribution over all possible strings and gives more weight to shorter strings, discouraging very long stems and suffixes.

The particular model given in (6.2) is, of course, only one possible way of extending (6.1) to the infinite case. Other possibilities include using a Dirichlet process for classes as well, or using different lexicon generators for the stems and suffixes, perhaps with different values of $p_\#$, the boundary probability. Setting $p_\#$ to be larger for suffixes than for stems would encode the fact that stems are usually longer than suffixes. It would be particularly interesting to experiment with inferring the parameters of P_0, to see whether it is possible to learn the phonotactic properties of typical stems and suffixes.

6.2.2 Extending the word segmentation models

In Chapter 5, I discussed two different kinds of word segmentation models incorporating unigram and bigram dependencies respectively. The simplest unigram model was defined as

$$\begin{aligned} w_i \,|\, G &\sim G \\ G \,|\, \alpha_0, P_0 &\sim \mathrm{DP}(\alpha_0, P_0) \end{aligned} \tag{6.3}$$

where w_i is the ith word in the corpus, P_0 is the unigram phoneme model, and α_0 is the concentration parameter of the DP. This model can very easily be extended to account for morphology by replacing P_0 with P_{μ_0}, the infinite morphological generator defined in (6.2). The result is a TwoStage(CRP(α_0), P_{μ_0}) model. Replacing the CRP adaptor with a Pitman-Yor adaptor would make this model more similar to the morphological model used in Chapter 4, but based on the results of that chapter, it is not clear whether the extra flexibility provided by the more complex Pitman-Yor adaptor is really critical. My results showed that, for low values of the Pitman-Yor parameter a, varying a led to minimal differences in morphological learning. This suggests that morphological generalization can

occur as long as the differences in frequency between words is fairly low. The Dirichlet process prior favors solutions where the number of tables is logarithmic in the number of tokens, so morphological induction (which works from table labels) will be based on log frequencies. This is probably a sufficient damping of the corpus frequencies for successful generalization to occur.

Regardless of the adaptor used, however, it is unlikely that a unigram word model with morphology would succeed in finding either words or morphemes correctly. My experiments in Chapter 5 showed that, even knowing the correct lexicon with high probability, word segmentation using a unigram model leads to poor results. Most of the errors are due to undersegmentation of frequently co-occurring words, not to oversegmentation at morpheme boundaries, so adding a morphological component is probably not to a good way to improve results.

Using a hierarchical Dirichlet process to encode bigram dependencies between words, I found not only that word segmentation improved markedly, but also that many of the remaining errors consisted of suffixes that were identified as separate words. Therefore, incorporating morphological information into the bigram model seems likely to improve word segmentation. The bigram word model with morphology differs from the model in Section 5.5.1 only in that the lexicon generator for the unigram backoff has been replaced by P_{μ_0}:

$$\begin{aligned} w_i \,|\, w_{i-1} = w,\ H_w &\sim H_w && \forall w \\ H_w \,|\, \alpha_1, G &\sim \mathrm{DP}(\alpha_1, G) && \forall w \\ G \,|\, \alpha_0, P_0 &\sim \mathrm{DP}(\alpha_0, P_{\mu_0}) \end{aligned} \tag{6.4}$$

Note that, because P_{μ_0} is itself defined in terms of Dirichlet processes, an additional level

has been added to the hierarchical model: labels on the bigram tables are generated by the unigram backoff DP, labels on the unigram tables are generated by the stem and suffix DPs, and labels on the stem and suffix tables are generated by P_0. The extension to account for utterance boundaries is similar to the one given in Section 5.5.1.

6.2.3 Inference

While the specification of the combined word segmentation-morphology model is simple, performing inference on this model is somewhat more complicated. First, note that all tables for a single word type must be explicitly tracked in this model, because they may contain different morphological analyses. This differs from the algorithms used in Chapter 5, where only the total number (or expected number) of tables with each word type was tracked. Second, if we assume an algorithm that considers a single potential word boundary location at each sampling step, the number of different hypotheses that must be considered at each step is much larger than in the basic word segmentation models. For each of the three possible words, all possible assignments to tables labeled with that word must be considered, as well as all possible new tables (i.e. all possible assignments of class, stem, and suffix). If we use a Gibbs sampler, each of these hypotheses must be enumerated and its probability computed exactly. For the case where two words are proposed, this involves $O((T_1 + CM)(T_2 + CN))$ computations, where T_1 and T_2 are the number of tables occupied by other instances of the two words, C is the number of classes, and M and N are the number of possible morpheme boundary locations in the two words. Note that we cannot simply sample an analysis of the first word, and then an analysis of the second, because of the dependencies induced by integrating out the model parameters. The choice of a particular class or stem or suffix in each word affects the probability that it will appear in

the other.

Instead of using Gibbs sampling, then, it would be more efficient to perform inference using a Metropolis-Hastings sampler. I will only provide a brief sketch of a possible sampler here; it is essentially the same as the one described in Johnson et al. (In preparation). Recall from Section 2.2.4.2 that in Metropolis-Hastings sampling, a *proposal* probability for each hypothesis must be computed, but the proposal probability need not be the actual probability of that hypothesis given the remaining data. Instead, it could be an approximation of the true probability, obtained by ignoring the dependencies between the analyses of the two words. A sample from the proposal distribution can therefore be found by choosing an analysis for the first word, followed by an analysis for the second word. This procedure requires only $O((T_1 + CM + T_2 + CN)$ computations. The sampler then requires one additional step to determine whether to accept the proposal as the next state or reject it (remaining in the same state). The acceptance probability is computed as

$$\min\left\{1, \frac{\pi(y')R(y)}{\pi(y)R(y')}\right\}$$

where y and y' are the old and new states, R is the proposal distribution, and π is the true state transition probability distribution (accounting for all dependencies).

6.3 Other extensions

In addition to combining the word segmentation and morphology models as described above, there are a number of other possible extensions to the work described in this thesis. One weakness of the current work is its use of normalized transcriptions – orthographic forms and dictionary-based phonemic representations with little or no phonetic variation. The assumption that each word has a unique orthographic or phonemic representation is built

into the models I have presented, in the sense that every token assigned to a table must have the same form as that table's label. However, this assumption could be relaxed by allowing the label on each table to represent a sort of prototype or cluster center, with a distribution over actual pronunciations. The models in this thesis can be viewed as a special case where each label is associated with a point distribution placing all its mass on the pronunciation corresponding to the label itself. To allow multiple phonetic realizations, the distribution could be defined so that phonetic forms similar to the prototype would have high probability, while more phonetically distant forms would have lower probability. This would represent a novel type of Dirichlet process mixture model[2], and could provide insight into the interactions in word recognition between word frequency, neighborhood size, and acoustic realization.

Another deficiency of the current models is in their treatment (or rather, lack of treatment) of syntax. It seems clear from the experiments in Chapter 5 that some notion of context is needed for accurate word segmentation; an obvious question is how abstracting away from specific context words to classes of words would affect the results of the model. In the morphology experiments in Chapter 4, the learner tended to place nouns and verbs in separate classes based on differing morphological behavior, yet these classes were noisy, and words with only one or two inflected forms were often misclassified. Since the model classes are intended to represent morphosyntactic categories, it would make sense to explicitly incorporate syntactic context into the model, perhaps by treating the morphosyntactic

[2]The mixture model is novel in the kind of distribution that would be defined over tokens at each table. Previous Dirichlet process mixture models have typically used multinomials (Blei et al., 2002; Navarro et al., 2006) or Gaussians (Rasmussen, 2000; Wood et al., 2006). Multidimensional Gaussian distributions might be appropriate if the input representation is even less abstract, as in continuous acoustic data rather than discrete phonetic transcriptions.

categories as states in an HMM[3]. Ideally, the morphological alternations and syntactic behavior observed in the data would complement each other to produce a more linguistically realistic assignment of words to classes.

A final area that would be worth pursuing further in the future is the question of how to define more linguistically motivated priors in the kinds of Bayesian models I have discussed. The priors I have used have been fairly basic, favoring solutions with fewer and shorter lexical items. It would be useful to explore whether other kinds of cross-linguistic tendencies (e.g. words usually contain vowels, and more complex consonant clusters are often permitted at word edges than word-medially) can be encoded into the prior, and what effects this has on learning. In more complex models including phonetic, phonological, or syntactic information, other kinds of priors could be investigated. Of course, the more information is included in the prior, the more important it is to make sure that this information isn't accidentally tuned to a particular language or set of languages. Therefore, cross-linguistic experiments (which are called for even to support my current results) would be absolutely necessary for this line of research.

6.4 Conclusion

The preceding discussion makes it clear that there are many questions left open by the work presented in this thesis. Nevertheless, this research represents an important contribution to the computational study of language acquisition. I have presented a generic, flexible framework for language modeling based on techniques from nonparametric Bayesian statistics. I

[3]Nonparametric Bayesian methods have been applied to create "infinite hidden Markov models" (Beal et al., 2002), where the complexity of the model, including the number of states, transitions between states, and emissions from states, grows with the size of the data.

have shown how this framework can be used to develop models of morphological acquisition and word segmentation, and how standard sampling methods can be used for inference with these models. This work represents the first application of nonparametric Bayesian methods to the acquisition of linguistic structure, and demonstrates that these methods can be used successfully for unsupervised learning from natural language data. Unlike maximum-likelihood estimation, where model selection must be approached as a separate problem, nonparametric Bayesian learning provides a model-internal way to limit the complexity of any proposed hypothesis, while also allowing hypotheses to grow in complexity as more evidence is accumulated.

In addition to these methodological contributions, my experiments provide a valuable addition to the literature on statistical language learning. In particular, I have provided evidence to support the hypothesis that morphological generalizations are based on statistical patterns found among word types, rather than word tokens. In the area of word segmentation, I have shown that the results of previous model-based approaches were misleading due to the effects of the search procedures used, and that failure to account for context leads to undersegmentation of the data. This suggests that current explanations of word segmentation based on transitional probabilities may be oversimplified, since they generally assume independence between words. On the other hand, my experiments also show that accounting for bigram dependencies between words leads to better segmentation than any previously published computational system on a comparable data set. This result indicates that the idea of statistical word segmentation itself is sound, and that with more subtle use of information comes more successful learning.

If there is one point in particular I hope to make with this research, it is precisely that

the ability to use statistical information from different sources in a variety of ways is what allows language learners to succeed. The simplest model using only the most obvious source of information often fails, yet examining the ways in which it fails is often instructive. I have argued that the nonparametric Bayesian framework presented in this thesis is useful for developing both simple models and more complex ones that integrate multiple sources of information. Here, I have used this framework to investigate the differences between learning from types or tokens, and learning with or without access to context. The results I have achieved so far bode well for future research, in which I plan to develop models that are able to learn from more realistic input using a wider variety of information. These more sophisticated models will no doubt lead to more successful unsupervised language learning in machines, as well as to important insights into the process of language acquisition in humans.

References

E. Aarts and J. Korst. 1989. *Simulated Annealing and Boltzmann Machines: A Stochastic Approach to Combinatorial Optimization and Neural Computing*. Wiley, New York.

A. Albright and B. Hayes. 2002. Modeling English past tense intuitions with minimal generalization. In *Proceedings of the Workshop on Morphological and Phonological Learning at ACL '02*, pages 58–69.

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

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

J. Allen and M. Christiansen. 1996. Integrating multiple cues in word segmentation: a connectionist model using hints. In *Proceedings of the 18th Annual Conference of the Cognitive Science Society*, pages 370–375, Mahwah, NJ. Lawrence Erlbaum.

R. Ando and L. Lee. 2000. Mostly-unsupervised statistical segmentation of Japanese: Application to kanji. In *Proceedings of ANLP-NAACL*.

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

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

T. M. Bailey and U. Hahn. 2001. Determinants of wordlikeness: Phonotactics or lexical neighborhoods? *Journal of Memory and Language*, 44:568–591.

M. Baroni, J. Matiasek, and H. Trost. 2002. Unsupervised discovery of morphologically related words based on orthographic and semantic similarity. In *Proceedings of the Workshop on Morphological and Phonological Learning at ACL '02*, pages 11–20.

M. Baroni. 2003. Distribution-driven morpheme discovery: A computational/experimental study. In G. Booij and J. van Marle, editors, *Yearbook of Morphology*, pages 213–248. Kluwer Academic Publishers.

E. Batchelder. 2002. Bootstrapping the lexicon: A computational model of infant speech segmentation. *Cognition*, 83:167–206.

M. Beal, Z. Ghahramani, and C. Rasmussen. 2002. The infinite hidden Markov model. In *Advances in Neural Information Processing Systems 14*.

K. Beesley and L. Karttunen. 2003. *Finite State Morphology*. CSLI Publications, Stanford University.

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

J. Besag. 2000. Markov chain Monte Carlo for statistical inference. Technical report, University of Washington, Center for Statistics and the Social Sciences. Working Paper no. 9.

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

D. Blei, A. Ng, and M. Jordan. 2002. Latent Dirichlet allocation. In *Advances in Neural Information Processing Systems 14*.

D. Blei, T. Griffiths, M. Jordan, and J. Tenenbaum. 2004. Hierarchical topic models and the nested Chinese restaurant process. In *Advances in Neural Information Processing Systems 16*.

P. Boersma and B. Hayes. 2001. Empirical tests of the Gradual Learning Algorithm. *Linguistic Inquiry*, 32(1):45–86.

P. Boersma and C. Levelt. 1999. Gradual constraint-ranking learning algorithm predicts acquisition order. In *Proceedings of the 30th Child Language Research Forum*.

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

P. Boersma. 1998. *Functional Phonology: Formalizing the interactions between articulatory and perceptual drives*. Ph.D. thesis, University of Amsterdam, The Hague.

P. Boersma. 2003. Review of *Learnability in Optimality Theory*. *Phonology*, 20:436–446.

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

M. Brent and J. Siskind. 2001. The role of exposure to isolated words in early vocabulary development. *Cognition*, 81(2):B33–44.

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

P. Brown, S. Della Pietra, V. Della Pietra, and R. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

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

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

P. Cairns and R. Shillcock. 1997. Bootstrapping word boundaries: A bottom-up corpus-based approach to speech segmentation. *Cognitive Psychology*, 33:111–153.

G. Carroll and E. Charniak. 1992. Two experiments on learning probabilistic dependency grammars from corpora. In *Proceedings of the AAAI Workshop on Statistically-Based Natural Language Processing Techniques*, San Jose, CA.

C.-H. Chang and C.-D. Chen. 1993. A study on integrating Chinese word segmentation and part-of-speech tagging. *Communications of the Chinese and Oriental Languages Information Processing Society*, 3(2):69–77.

J.-S. Chang and K.-Y. Su. 1997. An unsupervised iterative method for Chinese new lexicon extraction. *International Journal of Computational Linguistics and Chinese Language Processing*.

E. Charniak. 1993. *Statistical Language Learning*. MIT Press, Cambridge, MA.

E. Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of NAACL*.

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

N. Chomsky. 1957. *Syntactic Structures*. Mouton, The Hague.

N. Chomsky. 1965. *Aspects of the Theory of Syntax*. MIT Press, Cambridge.

N. Chomsky. 1981. *Lectures on Government and Binding*. Foris, Cinnaminson, NJ.

M. Christiansen and S. Curtin. 1999. The power of statistical learning: No need for algebraic rules. In *Proceedings of the 21st Annual Conference of the Cognitive Science Society*, Mahwah, NJ.

M. Christiansen, J. Allen, and M. Seidenberg. 1998. Learning to segment speech using multiple cues: A connectionist model. *Language and Cognitive Processes*, 13:221–268.

P. Cohen and N. Adams. 2001. An algorithm for segmenting categorical timeseries into meaningful episodes. In *Proceedings of the Fourth Symposium on Intelligent Data Analysis*.

M. Collins. 1999. *Head-Driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania.

S. Crain. 1991. Language acquisition in the absence of experience. *Behavioral and Brain Sciences*, 14:597–650.

M. Creutz and K. Lagus. 2002. Unsupervised discovery of morphemes. In *Proceedings of the Workshop on Morphological and Phonological Learning at ACL '02*.

M. Creutz and K. Lagus. 2004. 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 '04)*.

M. Creutz and K. Lagus. 2005. 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*.

M. Creutz. 2003. Unsupervised segmentation of words using prior distributions of morph length and frequency. In *Proceedings of the 41st Meeting of the Association for Competition Linguistics*.

C. de Marcken. 1995. The unsupervised acquisition of a lexicon from continuous speech. Technical report, Massachusetts Institute of Technology. A.I. Memo No. 1558.

C. de Marcken. 1996. *Unsupervised Language Acquisition*. Ph.D. thesis, Massachusetts Institute of Technology.

A. Dempster, N. Laird, and D. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm (with discussion). *Journal of the Royal Statistical Society B*, 39:1–38.

A. Doucet, C. Andrieu, and S. Godsill. 2000. On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and Computing*, 10(3):197–208.

M. Dowman. 2000. Addressing the learnability of verb subcategorizations with Bayesian inference. In L. Gleitman and A. Joshi, editors, *Proceedings of the Twenty-Second Annual Conference of the Cognitive Science Society*, Mahwah, NJ. Lawrence Erlbaum Associates.

B. E. Dresher and J. Kaye. 1990. A computational learning model for metrical phonology. *cognition*, 34:137–195.

B. E. Dresher. 1999. Charting the learning path: Cues to parameter setting. *Linguistic Inquiry*, 30(1):27–67.

C. Elkan. 2006. Clustering documents with an exponential-family approximation of the Dirichlet compound multinomial distribution. In *Proceedings of the 23rd International Conference on Machine Learning*.

T. M. Ellison. 1994. The iterative learning of phonological constraints. *Computational Linguistics*, 20(3).

J. Elman, E. Bates, M. H. Johnson, A. Karmiloff-Smith, D. Parisi, and K. Plunkett. 1996. *Rethinking Innateness: A Connectionist Perspective on Development*. MIT Press/Bradford Books, Cambridge, MA.

J. Elman. 1990. Finding structure in time. *Cognitive Science*, 14:179–211.

J. Elman. 2003. Generalization from sparse input. In *Proceedings of the 38th Annual Meeting of the Chicago Linguistic Society*.

J. Elman. 2004. An alternative view of the mental lexicon. *Trends in Cognitive Science*, 7:301–306.

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

J. Estoup. 1916. *Gammes Stenographiques*. Institut Stenographique de France, Paris.

H. Feng, K. Chen, X. Deng, and W. Zheng. 2004. Accessor variety criteria for Chinese word extraction. *Computational Lingustics*, 30(1).

T. Gambell and C. Yang. In submission. Mechanisms and constraints in word segmentation.

X. Ge, W. Pratt, and P. Smyth. 1999. Discovering Chinese words from unsegmented text. In *Proceedings of SIGIR*.

S. Geman and D. Geman. 1984. Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741.

L. Gerken. 2006. Decisions, decisions: Infant language learning when multiple generalizations are possible. *Cognition*, 98(3).

E. Gibson and K. Wexler. 1994. Triggers. *Linguistic Inquiry*, 25:407–454.

W.R. Gilks, S. Richardson, and D. J. Spiegelhalter, editors. 1996. *Markov Chain Monte Carlo in Practice*. Chapman and Hall, Suffolk.

E. M. Gold. 1967. Language identification in the limit. *Information and Control*, 10:447–474.

J. Goldsmith. 2001a. Linguistica. Executable available at http://humanities.uchicago.edu/faculty/goldsmith.

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

J. Goldsmith. In press. An algorithm for the unsupervised learning of morphology. *Journal of Natural Language Engineering*.

S. Goldwater and M. Johnson. 2004. Priors in Bayesian learning of phonological rules. In *Proceedings of the Seventh Meeting of the ACL Special Interest Group in Computational Phonology (SIGPHON '04)*.

S. Goldwater, T. Griffiths, and M. Johnson. 2006. Interpolating between types and tokens by estimating power-law generators. In *Advances in Neural Information Processing Systems 18*.

T. Griffiths and J. Tenenbaum. In press. Optimal predictions in everyday cognition. *Psychological Science*.

T. Griffiths. 2005. Zipf plots can be dangerous. Unpublished manuscript.

T. Griffiths. 2006. Power-law distributions and nonparametric Bayes. Unpublished manuscript.

J. Hajič and B. Hladká. 1998. Tagging inflective languages: Prediction of morphological categories for a rich, structured tagset. In *Proceedings of COLING-ACL*, pages 483–490.

J. Hajič. 2000. Morphological tagging: Data vs. dictionaries. In *Proceedings of ANLP-NAACL*, pages 94–101.

D. Hakkani-Tür, K. Oflazer, and G. Tür. 2000. Statistical morphological disambiguation for agglutinative languages. In *Proceedings of COLING*.

Z. Harris. 1954. Distributional structure. *Word*, 10:146–162.

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

W. Hastings. 1970. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57:97–109.

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

B. Hayes. 1995. *Metrical Stress Theory: Principles and Case Studies*. University of Chicago Press, Chicago.

Y. Hu, I. Matveeva, J. Goldsmith, and C. Sprague. 2005. Using morphology and syntax together in unsupervised learning. In *Proceedings of the Workshop on Psychocomputational Models of Language Acquisition at ACL'05*.

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

F. Jelinek. 1997. *Statistical methods for speech recognition*. MIT press, Cambridge, MA.

E. Johnson and P. Jusczyk. 2001. Word segmentation by 8-month-olds: When speech cues count more than statistics. *Journal of Memory and Language*, 44:548–567.

M. Johnson, T. Griffiths, and S. Goldwater. In preparation. Bayesian inference for PCFGs via Markov chain Monte Carlo.

K. Johnson. 2004. Gold's theorem and cognitive science. *Philosophy of Science*, 71:571–592.

P. Jusczyk, E. Hohne, and A. Bauman. 1999a. Infants' sensitivity to allophonic cues for word segmentation. *Perception and Psychophysics*, 61(8):1465–1476.

P. Jusczyk, D. Houston, and M. Newsome. 1999b. The beginnings of word segmentation in English-learning infants. *Cognitive Psychology*, 39:159–207.

P Jusczyk. 1997. *The Discovery of Spoken Language*. MIT Press, Cambridge, MA.

L. Karttunen, R. Kaplan, and A. Zaenen. 1992. Two-level morphology with composition. In *Proceedings of COLING*.

R. Kneser and H. Ney. 1995. Improved backing-off for n-gram language modeling. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*.

K. Koskenniemi. 1983. Two-level morphology: A general computational model for word-form recognition and production. Technical Report 11, University of Helsinki, Department of General Linguistics.

W. Kraaij and R. Pohlmann. 1996. Viewing stemming as recall enhancement. In *Proceedings of the 19th International Conference on Research and Development in Information Retrieval (SIGIR)*.

L. Larkey, L. Ballesteros, and M. Connell. 2002. 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.

Y. Lee, K. Papineni, S. Roukos, O. Emam, and H. Hassan. 2003. Language model based Arabic word segmentation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*.

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

P. Luce and D. Pisoni. 1998. Recognizing spoken words: the Neighborhood Activation Model. *Ear and Hearing*, 19:1–36.

P. Luce, S. Goldinger, E. Auer, and M. Vitevitch. 2000. Phonetic priming, neighborhood activation, and PARSYN. *Perception and Psychophysics*, 62(3):615–625.

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

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

R. Madsen, D. Kauchak, and C. Elkan. 2005. Modeling word burstiness using the Dirichlet distribution. In *Proceedings of the 22nd International Conference on Machine Learning*.

C. Manning and H. Schütze. 1999. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA.

M. Marcus, B. Santorini, and M. A. Marcinkiewicz. 1993. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2):331–330.

D. Marr. 1982. *Vision: A Computational Approach*. Freeman & Co., San Francisco.

W. Marslen-Wilson. 1987. Functional parallelism in spoken word-recognition. *Cognition*, 25:71–102.

S. Mattys, P. Jusczyk, P. Luce, and J. Morgan. 1999. Phonotactic and prosodic effects on word segmentation in infants. *Cognitive Psychology*, 38:465–494.

J. McClelland and J. Elman. 1986. The TRACE model of speech perception. *Cognitive Psychology*, 18:1–86.

B. Merialdo. 1994. Tagging English text with a probabilistic model. *Computational Linguistics*, 20(2):155–172.

N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller. 1953. Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, 21:1087–1092.

T. Mintz, E. Newport, and T. Bever. 2002. The distributional structure of grammatical categories in speech to young children. *Cognitive Science*, 26:393–424.

M. Mitzenmacher. 2003. A brief history of generative models for power law and lognormal distributions. *Internet Mathematics*, 1(2):226–251.

C. Monson, A. Lavie, J. Carbonell, and L. Levin. 2004. Unsupervised induction of natural language morphology inflection classes. In *Proceedings of the Seventh Meeting of the ACL Special Interest Group in Computational Phonology (SIGPHON '04)*, pages 52–61.

J. Morgan, K. Bonamo, and L. Travis. 1995. Negative evidence on negative evidence. *Developmental Psychology*, 31:180–197.

D. Navarro, T. Griffiths, M. Steyvers, and M. Lee. 2006. Modeling individual differences using Dirichlet processes. *Journal of Mathematical Psychology*.

R. Neal and G. Hinton. 1998. A new view of the EM algorithm that justifies incremental and other variants. In *Learning in Graphical Models*, pages 355–368. Kluwer.

R. Neal. 1993. Probabilistic inference using Markov chain Monte Carlo methods. Technical Report CRG-TR-93-1, University of Toronto, Department of Computer Science.

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

S. Neuvel and S. Fulop. 2002. Unsupervised learning of morphology without morphemes. In *Proceedings of the Workshop on Morphological and Phonological Learning (ACL SIGPHON)*.

E. Newport and R. Aslin. 2004. Learning at a distance I. Statistical learning of non-adjacent dependencies. *Cognitive Psychology*, 48:127–162.

J.-Y. Nie, W. Jin, and M.-L. Hannan. 1994. A hybrid approach to unknown word detection and segmentation of Chinese. In *Proceedings of the International Conference on Chinese Computing*, pages 326–335.

D. Norris. 1994. Shortlist: A connectionist model of continuous speech recognition. *Cognition*, 52:189–234.

F. Och. 2005. Statistical machine translation: The fabulous present and future. Invited talk at the Workshop on Building and Using Parallel Texts at ACL'05.

F. Peng and D. Schuurmans. 2001. Self-supervised Chinese word segmentation. In *Proceedings of the Fourth International Conference on Advances in Intelligent Data Analysis (IDA)*, Portugal.

A. Peters. 1983. *The Units of Language Acquisition*. Cambridge University Press, New York.

J. Pierrehumbert. 2001. Stochastic phonology. *Glot International*, 5(6):195–207.

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

S. Pinker. 1988. On language and connectionism: Analysis of a parallel distributed processing model of language acquisition. *Cognition*, 26:195–267.

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

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

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

M. Popovič and P. Willett. 1992. The effectiveness of stemming for natural-language access to Slovene textual data. *Journal of the American Society for Information Science*, 43(5):384–390.

M. Porter. 1980. An algorithm for suffix stripping. *Program*, 14(3):130–137.

A. Prince and P. Smolensky. 1993. Optimality theory: Constraint interaction in generative grammar. Technical Report TR-2, Rutgers Center for Cognitive Science, Rutgers Univ.

J. Randall. 1992. The catapult hypothesis: an approach to unlearning. In J. Weissenborn, H. Goodluck, and T. Roeper, editors, *Theoretical Issues in Language Acquisition: Continuity and Change in Development*. Lawrence Erlbaum Associates, New Jersey.

C. Rasmussen. 2000. The infinite Gaussian mixture model. In *Advances in Neural Information Processing Systems 12*.

A. Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *Proceedings of the First Conference on Empirical Methods in Natural Language Processing*, pages 133–142.

T. Regier, B. Corrigan, R. Cabasaan, A. Woodward, M. Gasser, and L. Smith. 2001. The emergence of words. In *Proceedings of the Cognitive Science Conference*, Mahwah, NJ. Erlbaum.

J. Rissanen. 1989. *Stochastic Complexity and Statistical Inquiry*. World Scientific Co., Singapore.

B. Roark and K. Demuth. 2000. Prosodic constraints and the learner's environment: a corpus study. In *Proceedings of the 24th Annual Boston University Conference on Language Development*, pages 597–608.

D. Rumelhart and J. McClelland. 1986. On learning the past tenses of English verbs. In D. Rumelhart and J. McClelland, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, chapter 18. MIT Press.

J. Saffran, R. Aslin, and E. Newport. 1996a. Statistical learning in 8-month-old infants. *Science*, 274:1926–1928.

J. Saffran, E. Newport, and R. Aslin. 1996b. Word segmentation: the role of distributional cues. *Journal of Memory and Language*, 35:606–621.

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

L. Santelmann and P. Jusczyk. 1998. Sensitivity to discontinuous dependencies in language learners: Evidence for limitations in processing space. *Cognition*, 69(2):105–134.

P. Schone and D. Jurafsky. 2001. Knowledge-free induction of inflectional morphologies. In *Proceedings of the North American chapter of the ACL*, pages 183–191.

M. Seidenberg and L. Gonnerman. 2000. Explaining derivational morphology as the convergence of codes. *Trends in Cognitive Sciences*, 4:353–361.

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

M. Snover and M. Brent. 2003. A probabilistic model for learning concatenative morphology. In *Advances in Neural Information Processing Systems 15*.

R. Sproat and C. Shih. 1990. A statistical method for finding word boundaries in Chinese text. *Computer Processing of Chinese and Oriental Languages*, 4:336–351.

R. Sproat, C. Shih, W. Gale, and N. Chang. 1996. A stochastic finite-state word-segmentation algorithm for Chinese. *Computational Linguistics*, 22(3).

M. Sun, D. Shen, and B. Tsou. 1998. Chinese word segmentation without using lexicon and hand-crafted training data. In *Proceedings of COLING-ACL*.

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

Y. Teh, M. Jordan, M. Beal, and D. Blei. 2005. Hierarchical Dirichlet processes. In *Advances in Neural Information Processing Systems 17*. MIT Press, Cambridge, MA.

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

J. Tenenbaum and F. Xu. 2000. Word learning as Bayesian inference. In *Proceedings of the 22nd Annual Conference of the Cognitive Science Society*.

B. Tesar and P. Smolensky. 1996. Learnability in Optimality Theory (long version). Technical Report CU–CS–678–93, Department of Computer Science, University Of Colorado, Boulder.

B. Tesar and P. Smolensky. 2000. *Learnability in Optimality Theory*. MIT press, Cambridge, MA.

B. Tesar. 1998. Error-driven learning in Optimality Theory via the efficient computation of optimal forms. In P. Barbosa, D. Fox, P. Hagstom, M. McGinnis, and D. Pesetsky, editors, *Is the Best Good Enough? Optimality and Competition in Syntax*, pages 421–435. MIT press, Cambridge, MA.

E. Thiessen and J. Saffran. 2003. When cues collide: Use of stress and statistical cues to word boundaries by 7- to 9-month-old infants. *Developmental Psychology*, 39(4):706–716.

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

M. Vitevitch and P. Luce. 1999. Probabilistic phonotactics and neighborhood activation in spoken word recognition. *Journal of Memory and Language*, 40:374–408.

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

R. Wicentowski. 2004. Multilingual noise-robust supervised morphological analysis using the WordFrame model. In *Proceedings of the Seventh Meeting of the ACL Special Interest Group in Computational Phonology (SIGPHON '04)*, pages 70–77.

F. Wood, S. Goldwater, and M. Black. 2006. A non-parametric Bayesian approach to spike sorting. In *IEEE Engineering Medicine Biologicial Systems*.

D. Yarowsky and R. Wicentowski. 2000. Minimally supervised morphological analysis by multimodal alignment. In *Proceedings of the ACL*, pages 207–216.

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

Appendix A

Phonemic Transcription Symbols

A.1 Brown-Morgan corpus

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 Chapter 4.

Consonants

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

Vowels

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

A.2 Bernstein-Ratner-Brent corpus

The following ASCII characters are used in the phonemic transcriptions in the Bernstein-Ratner-Brent corpus, which was used as input to the word segmentation system in Chapter 5.

Consonants

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

Vowels

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

Rhotic Vowels

ASCII	Example
#	ARe
%	fOR
(	hERE
)	lURE
*	hAIR
3	bIRd
R	buttER