Unsupervised Learning of Lexical Information for Language Processing Systems

Graham Neubig

Abstract

Natural language processing systems such as speech recognition and machine translation conventionally treat words as their fundamental unit of processing. However, in many cases the definition of a "word" is not obvious, such as in languages without explicit white space delimiters, in agglutinative languages, or in streams of continuous speech.

This thesis attempts to answer the question of which lexical units should be used for these applications by acquiring them through unsupervised learning. This has the potential to lead to improvements in accuracy, as it can choose lexical units flexibly, using longer units when justified by the data, or falling back to shorter units when faced with data sparsity. In addition, this approach allows us to re-examine our assumptions of what units we should be using to recognize speech or translate text, which will provide insights to the designers of supervised systems. Furthermore, as the methods require no annotated data, they have the potential to remove the annotation bottleneck, allowing for the processing of under-resourced languages for which no human annotations or analysis tools are available.

Chapter 1 provides an overview of the general topics of word segmentation and morphological analysis, as well as previous research on learning lexical units from raw text. It goes on to discuss the problems with the existing approaches, and lays out the general motivation for and techniques used in the work presented in the following chapters.

Chapter 2 describes the overall learning framework adopted in this thesis, which consists of models created using non-parametric Bayesian statistics, and inference procedures for the models using Gibbs sampling. Non-parametric Bayesian statistics are useful because they allow for automatically discovering the appropriate balance between model complexity and expressive power. We adopt Gibbs sampling as an inference procedure because it is a principled, yet flexible learning method that can be used with a wide variety of models. Within this framework, this thesis presents models for lexical learning for speech recognition and machine translation.

With regards to speech recognition, Chapter 3 presents a method that

can learn a language model and lexicon directly from continuous speech with no text. This is performed using the hierarchical Pitman-Yor language model, a non-parametric Bayesian formulation of standard language modeling techniques based on the Pitman-Yor process, which allows for principled and effective modeling and inference. With regards to modeling, the non-parametric formulation allows for learning of appropriately sized lexical units that are long enough to be useful, but not so long as to cause sparsity problems. Inference is performed using Gibbs sampling with dynamic programming over weighted finite states transducers (WFSTs). This makes it straight-forward to learn over lattices, allowing for language model learning in the face of acoustic uncertainty. Experiments demonstrate that the proposed method is able to reduce the phoneme error rate on a speech recognition task, and is also able to learn a number of intuitively reasonable lexical units.

In the work on machine translation, Chapter 4 presents a model that, given a parallel corpus of sentences in two languages, aligns words or multi-word phrases in each sentence for use in machine translation. The model is hierarchical, allowing for the inclusion of overlapping phrases of multiple granularities, which is essential for achieving high accuracy when using the phrases in translation. Inference is performed using Gibbs sampling over trees expressed using inversion transduction grammars (ITGs), a particular form of synchronous context-free grammar that allows for the expression of reordering between languages and polynomial-time alignment through the process of biparsing. Experiments show that this model is able to achieve translation accuracy that is competitive with the process used in traditional systems while reducing the model to a fraction of its original size.

Chapter 5 extends this model to perform alignment over multi-character substrings, learning a model that directly translates character strings from one language to another. In order to do so, two changes are made to improve alignment. The first improvement is based on aggregating substring co-occurrence statistics over the entire corpus and using these to seed the probabilities of the ITG model. The second improvement is based on introducing a look-ahead score similar to that of A* search to the ITG biparsing algorithm, which allows for more effective pruning of the search space. An experimental evaluation finds that character-based translation with automatically learned units is able to provide comparable results to word-based translation while handling linguistic phenomena such as productive morphology, proper names, and unsegmented text.

Chapter 6 concludes the thesis with an overview of the task of lexical learning for practical applications and directions for future research.

Acknowledgements

First and foremost, thank you to Professor Tatsuya Kawahara for welcoming me into his lab, and for showing me by example what it takes to be both an excellent researcher and teacher. I am so grateful for everything he has taught me, whether it is about technical topics such as WFSTs and speech recognition, or about the basics of being a researcher such as the attention to detail needed to write a top-class paper or give a good academic presentation.

I would also like to thank Professor Shinsuke Mori, from whom I have learned a large amount of my knowledge about both natural language processing, and about language in general. I am grateful for not only his lessons and discussions, but also particularly for the experiences that he provided by encouraging me to create open source tools and getting other people to use them.

Thank you to Professor Sadao Kurohashi and Professor Toshiyuki Tanaka for agreeing to serve as members of my doctoral committee, and for carefully checking this thesis and providing advice during my presentation.

Dr. Taro Watanabe was in a way the second advisor for most of my thesis work, and I am not only grateful for all his advice, but eternally impressed by his depth and breadth of knowledge about all things related to machine translation. In addition, I would like to thank Dr. Eiichiro Sumita for allowing me to visit NICT, as well as Dr. Andrew Finch, Dr. Michael Paul, Dr. Masao Utiyama, and all of the members at NICT for the helpful advice and discussions that I received there.

I would also like to thank Dr. David Talbot for welcoming me as an intern at Google, and for many enlightening discussions about not only machine translation, but also language in general. The internship was a precious experience, and I am also particularly indebted to Mr. Colin Young, Mr. Jan Pfeifer, Mr. Hiroshi Ichikawa, Mr. Jason Katz-Brown, Dr. Ashish Venugopal, Dr. Hideto Kazawa, Dr. Taku Kudo, and countless others for all the help and advice they gave me while I was there.

Of course my time as a graduate student would not have been complete with the members of the Kawahara lab. Thank you for all your help, all your advice, and all the lunchtime banter over mini-karaage (or just the table in the student room).

There are so many others that have helped me along my way that it would be impossible to name them all, but there are a few people that deserve special mention. I would like to thank Dr. Shinji Watanabe, Dr. Mamoru Komachi, and Dr. Hisami Suzuki for giving me an opportunity to present my work at NTT and MSR. Thank you to Dr. Daichi Mochihashi, Dr. Yugo Murawaki, Dr. Fabien Cromieres, Dr. Phil Blunsom, and Dr. Sharon Goldwater for taking the time to answer my questions. Thank you to all the members of the ANPI_NLP project for doing what we could to help after that great Eastern Japan earthquake, and to Dr. Koji Murakami, Dr. Masato Hagiwara, Dr. Yuichiroh Matsubayashi, Professor Atsushi Fujita, and Professor Taiichi Hashimoto for helping to spread the word of this project to the rest of the world.

Mom, Dad, Maia, and all of the rest of the family have been endlessly encouraging and understanding, even when I decided to study so far away from home. And finally, thank you Yuko, for always being there with me and reminding me about the things that are really important in life.

Contents

Chapter 1

Introduction

This thesis is concerned with the most fundamental, and the most important unit in natural language processing: the word. In many cases, the word is taken for granted. Previous works on machine translation usually start with assumption that we will be turning sequences of *words* from one language to another, while previous works on speech recognition assume that we are handling the task of accurately transcribing the *words* that someone speaks.

But what is a word anyway? In English, the answer may be simple, the previous sentence has six words, each of which is separated by a white space. But let us ask the same question in Japanese: "単語とは一体なんでしょう?" Suddenly things become more complicated, as there are no explicit boundaries between the words. And if we ask in Korean, we find we are somewhere in the middle: "단어란 도대체 무엇일까요?" There are white spaces, but much less frequently than in English, with concepts that would require multiple white-space separated segments in English being packed into a single segment.

Despite these conceptual difficulties, before we begin to build systems that can process language, it is necessary to decide what unit we will treat as the fundamental element for our further analysis, and finding a proper answer to this question is paramount to the creation of effective language processing systems.

1.1 Supervised Lexical Processing Systems

One answer to the question of how we define lexical units for Japanese can be found in the annotation standard for the Balanced Corpus of Con-

temporary Written Japanese (Ogura et al., 2011), a 359-page effort detailing standards of segmentation and tag annotation that was the result of careful consideration by professional linguists. With this standard in hand, we proceed to create automatic systems for word segmentation or morphological analysis, which are able to analyze new, unsegmented text.

As word segmentation and morphological analysis are fundamental problems in natural language processing, there has been a significant amount of research into methodologies to perform these tasks. These methodologies fall into two general categories: those based on dictionaries or pattern matching, and those using boundary prediction.

The first examples of word segmentation and morphological analysis systems were dictionary-based methods that represent each sentence as a sequence of words (or morphemes) in a dictionary, and try to decide which sequence is most appropriate. This can be done using anything from simple techniques that match the dictionary units of maximal length (Yoshimura et al., 1983), or other methods with more finely hand-tuned scores (Kurohashi et al., 1994). There are also data-driven methods for dictionary prediction using n-gram models (Nagata, 1994; Sproat et al., 1996), HMMs (Chang and Chen, 1993; Takeuchi and Matsumoto, 1995), or discriminative methods such as conditional random fields (CRFs) (Kudo et al., 2004).

In addition, there have also been methods proposed to perform segmentation by simply predicting whether each character lies on a word boundary or not. This is done by predicting the presence or absence of word boundaries between each pair of characters in the input sentence as a binary classification problem (Sassano, 2002; Neubig et al., 2011), or treating word segmentation as a chunking problem using a "left-middle-right" tagging scheme (Xue and Shen, 2003; Peng et al., 2004). Finally, there has also been significant research on combining dictionary-based and boundary-based prediction methods for increased accuracy (Nakagawa, 2004; Kruengkrai et al., 2009).

The previously introduced works all concern themselves with the segmentation of languages such as Japanese or Chinese, which are written without explicit boundaries between words or morphemes. In addition, there has also been work on morphological analysis for segmented, but morphologically productive languages such as Finnish and Arabic (Koskenniemi, 1984; Beesley, 1996). Unlike word segmentation, which simply splits the text stream into words, these systems do more complicated pattern matching and base form recovery, which cannot be achieved by simple segmentation of the input text.

1.2 Unsupervised Learning of Lexical Units and Morphology

In contrast to supervised methods for morphology and analysis, there has also been a large amount of work on the unsupervised acquisition of lexicons and morphological patterns (Harris, 1954; de Marcken, 1996; Brent, 1999; Goldwater et al., 2009; Mochihashi et al., 2009; Räsänen, 2011). First considering the problem of learning lexical units from unsegmented text without the concept of morphological patterns, there are two problems that must be solved. The first is a problem of modeling: how do we create a model that assigns a high score to units that are of appropriate length? This problem is difficult because it requires a balance between models that assign longer units but may be prone to over-fit the training data, and models that assign shorter units but lack the expressiveness to accurately model the phenomena that we are interested in. The second is a problem of inference: given our model, how do we find a segmentation of maximal score? This is also difficult in that the number of possible segmentations grows exponentially in the length of the corpus.

One of the first works to handle both of these issues in a formal probabilistic framework is (de Marcken, 1996), who handles the modeling problem using the principle of *minimum description length* (MDL), which attempts to maximize likelihood but also penalizes overly complex models. Inference is performed using a hill-climbing technique, merging and separating lexical units based on their contribution to description length.

Another method that has received much attention recently is the Bayesian word segmentation approach of (Goldwater et al., 2009). Modeling is performed using Bayesian techniques, which help to prevent over-fitting, while inference is performed using Gibbs sampling, both of which are described in detail in Chapter 2. (Mochihashi et al., 2009) describes how to train this model more efficiently using dynamic programming.

There has also been work on learning morphology for languages that have spaces between words, but with productive morphology that combines multiple morphemes into single words. Some models deal with *concatenative* morphology, which is similar to word segmentation as it assumes that each word is a simple concatenations of its component parts (Creutz and Lagus, 2007; Snyder and Barzilay, 2008; Poon et al., 2009), an assumption also made in this thesis. Others concern themselves with *non-concatenative* morphology, learning conjugation patterns or even irregular constructions such as “take/took” through the use of string similarity, clustering, and syntactic information (Yarowsky and Wicentowski, 2000; Schone and Jurafsky,

2001; Naradowsky and Goldwater, 2009; Dreyer and Eisner, 2011).

1.3 Problems

There are two major problems concerning the choice of lexical units with these existing frameworks: how are we able to get data to train supervised classifiers, and how do we know that the lexical units we have chosen are actually proper for the task at hand?

1.3.1 The Data Bottleneck

Within the framework of supervised segmentation, the move to data-driven methods has brought improvements in the accuracy, flexibility, and coverage of word segmentation and morphological analysis systems. However, it has also exchanged the difficulty of creating and tuning rules for the difficulty of creating training data. It is a well known fact that if we do not have enough in-domain data, either in the form of dictionaries or training corpora, supervised analysis systems will perform poorly, particularly when encountering unknown words (Neubig and Mori, 2010).

There has been significant work on efficient creation of data through active learning for lexical analysis. Methods have been proposed for creating data both from scratch (Sassano, 2002), and in the context of domain adaptation, where there exists an annotated corpus of text in a certain domain (such as newspaper text), but not in the domain of the text that we would like to analyze (such as medical text) (Neubig and Mori, 2010; Neubig et al., 2011). Even with these techniques, however, there is still a need to spend a fixed amount of effort for each domain of concern.

There has also been work on semi-supervised learning (Xu et al., 2008; Wang et al., 2011), which can use unsegmented text to improve the accuracy of segmenters originally trained on manually annotated data. This is a promising approach as it requires no human effort to improve the system accuracy, but it does require seed data created according to some segmentation standard. This has its own potential pitfalls, as the following section explains.

1.3.2 The Problem with Standards

All of the previously mentioned works are performed and evaluated based on a single premise: that we have some "correct" annotation for our corpus of interest, and our goal is to develop a system that is able to accurately

recover this correct annotation. But the question of what is correct is notoriously hard to answer for human annotators. For example, when native speakers of Chinese were asked to segment text into "words" with no additional instruction, the agreement between annotators was a mere 87.6% (Xia et al., 2000). This is a problem of linguistic annotation in general, with more difficult tasks such as word sense disambiguation and semantic structure annotation seeing agreements as low as 50-60% (Ng et al., 1999; Passonneau et al., 2006).

In order to reduce some of this inconsistency, linguists create detailed and voluminous annotation standards when attempting to annotate new linguistic data. However, while these standards do provide a level of internal consistency to the annotations, they have turned out to not necessarily be ideal for practical applications such as speech recognition (Hirsimäki et al., 2006) and machine translation (Carpuat and Wu, 2005; Chang et al., 2008).

Let us take the example of choosing a segmentation of the English word "uninspiring" do we treat this as a single unit, do we separate it into "un inspir ing," or do we separate only the inflectional prefix and keep together the derivational suffix leaving us with "un inspiring?" Do we normalize "inspir" into "inspire?"

In the context of machine translation, if we assume the longest unit "uninspiring" does not appear in our training corpus, the machine translation system will not be able to generate a translation in the target language, leaving the word as-is. On the other hand, if we choose to process all the morphemes separately, there is a possibility that "inspir ing" will be misinterpreted as the present progressive form of the verb "inspire," instead of being interpreted as the adjective that it actually is, resulting in a mistranslation

Similarly, for speech recognition, most modern speech recognition systems are only able to recognize in-vocabulary words, so if "uninspiring" (with the correct corresponding pronunciation) does not exist in the lexicon, we will not be able to recognize it. On the other hand, if we segment too finely and introduce very short units into the lexicon, there is a good chance that this will confuse the recognizer, causing mistakes such as between the morphological prefix "un" and the filler "um."

What is interesting to note here is that "un," which simply turns a verb or adjective into the negative form, may be relatively easy to handle for machine translation. However, as it is acoustically confusable with "um," it can be expected to cause problems for speech recognition. Thus, we can see that the lexical units that provide the highest accuracy depend both on the application and the amount of data we have available, indicating that

no single segmentation standard, no matter how well thought out, will be the answer to all of our lexical processing needs.

1.4 Lexical Learning for Language Processing Systems

This thesis presents techniques that perform unsupervised lexical learning with the specific purpose of learning units that are able to improve the accuracy of practical applications such as speech recognition and machine translation. This helps resolve the data bottleneck, as unsupervised learning techniques function directly on unlabeled data, which can be gathered in large quantities from the internet for many of the world's languages. This also has the potential to resolve the problem of which units we use in our language processing systems by learning them automatically from the text according to an objective function that is correlated with system performance.

The objective function that is used throughout this thesis is likelihood according to models rooted in non-parametric Bayesian statistics. As mentioned previously, non-parametric Bayesian models have been shown effective for lexical learning tasks, and thus are a natural choice for application-driven lexical learning models as well. Chapter 2 provides an overview of non-parametric Bayesian statistics, focusing on models for discrete variables, and describing how to perform both modeling and inference.

Chapter 3 describes a model for learning lexical information and rudimentary contextual information in the form of an n-gram language model for use in automatic speech recognition. The proposed technique builds upon the language-model-based word segmentation work of (Mochihashi et al., 2009), formalizing the model using finite state machines, which allows for the use of noisy input such as speech in the learning process. As this work uses no transcribed text data, it offers a solution to the data bottleneck, allowing learning from raw speech in languages or domains with no text resources. It is also able to automatically adjust the length of the lexical units used in language modeling. More interestingly, it is able to learn pronunciations directly from speech, allowing for the discovery of non-traditional pronunciations that do not exist in human-created lexicons.

In the context of machine translation, this thesis presents a method to learn the appropriate lexical units for a translation model directly from bilingual data without referencing explicit word boundaries or human tokenization standards. This is done through a bi-text alignment method based

on Inversion Transduction Grammars (ITGs), which is described fully in Chapter 4. This model is inspired by the previous work of (de Marcken, 1996), but is extensively modified by replacing MDL with Bayesian statistics, replacing hill-climbing with sampling, and porting the model to allow for bilingual phrases. An experimental evaluation demonstrates that this technique is able to learn a compact translation model using a fully probabilistic approach, with none of the heuristics used in previous research.

Chapter 5 then applies this model to learning lexical units for machine translation. Specifically, the model is used to learn alignments not over strings of words, but over strings of characters, and modified with two improvements that allow for effective alignment of character strings. This offers a solution to the data bottleneck, allowing for the automatic acquisition of lexical units with no annotated resources. In addition, as the units used in translation are automatically learned, the model is able to choose unit lengths that are appropriate for the translation task.

Finally, Chapter 6 discusses overall findings, and points out future directions for research in the area of lexical learning for practical applications.

Chapter 2

Modeling and Inference using Non-Parametric Bayesian Statistics

This chapter introduces the preliminaries of non-parametric Bayesian statistics that are used as a learning framework throughout this thesis. In particular, it focuses on distributions over discrete variables modeled using the Pitman-Yor process, and techniques to approximate this distribution using Gibbs sampling. As the focus of the thesis is on the use of this framework to model language, this chapter provides an introduction to the general properties of the models and learning framework, but leaves more rigorous mathematical discussion to the references. In addition, the focus will be put on discrete distributions, which are useful for modeling language, as opposed to non-parametric techniques for continuous or relational data (Rasmussen, 2004; Roy and Teh, 2009).

2.1 Statistical Modeling for Discrete Distributions

Assume we have training data $\mathcal{X} = \{x_1, \ldots, x_I\}$, which is a collection of discrete samples, the values of which we assume to be generated independently from some distribution (in other words, the values are *independent and identically distributed, i.i.d.*). For the purpose of demonstration, assume there is an example sequence with the following values

$$\mathcal{X} = \{1, 2, 4, 5, 2, 1, 4, 4, 1, 4\} \tag{2.1}$$

which we assume were i.i.d. according to some distribution over the natural numbers from 1 to 5.

The question that modeling and inference must solve is, how do we estimate the underlying distribution that generated this data for the purpose of modeling new phenomena that are not included in the training data? In other words, let us assume that we have a collection of variables G that parameterizes the underlying distribution, and element g_k represents the true probability of generating the value k according to this distribution:

$$g_k = P(x = k|G). \tag{2.2}$$

We will attempt to estimate G given the data $\mathcal{X}$.

2.1.1 Maximum Likelihood Estimation

The most straight-forward way of estimating G is *maximum likelihood* estimation, which chooses G to maximize the likelihood over the training data. In order to do so, as each element of $\mathcal{X}$ is a discrete sample, we first define a count variable $c_{\mathcal{X},k}$, where k is an arbitrary value that may be generated by the underlying distribution, and $c_{\mathcal{X},k}$ is the number of samples in $\mathcal{X}$ that took the value k. We will omit the first subscript indicating the collection of samples that we are counting over (in this case, $\mathcal{X}$) when it is obvious from context.

Given the data shown in Equation (2.1), we are able to acquire counts as follows:

$$C = \{c_1, \ldots, c_5\} = \{3, 2, 0, 4, 1\}. \tag{2.3}$$

Given these counts, the parameterization that maximizes the likelihood of $\mathcal{X}$ can be estimated as follows

$$g_k = \frac{c_k}{\sum_{\tilde{k}} c_{\tilde{k}}}. \tag{2.4}$$

In the running example, this gives us a *multinomial* distribution over the discrete variables as follows:

$$G = \{g_1, \ldots, g_5\} = \{0.3, 0.2, 0.0, 0.4, 0.1\}. \tag{2.5}$$

There are two fundamental problems with maximum likelihood estimation. The first is that pure maximum likelihood estimation has no constraint to prevent it from reaching degenerate parameter configurations that assign unreasonably low probabilities to events not observed in the training data,

or unreasonably high probabilities to events that are observed in the data. An example of this can be seen in the fact that the zero count of c_3 results in a probability of $g_3 = 0$ in Equation (2.5). Given this probability, any new input that happens to contain an instance of $x_i = 3$ will be given a probability of 0.

The second problem is that maximum likelihood estimation chooses a single unique solution for G, even though we are not actually certain of G's actual value.

2.1.2 Bayesian Estimation

Bayesian statistics alleviate these two problems by working with not a single value of G, but instead considering the entire distribution over the parameters given the data $P(G|\mathcal{X})$. In turn, the expectation of this distribution can be used as our predictive distribution for new values of x

$$P(x = k|\mathcal{X}) = \mathbb{E}[g_k] = \int g_k P(G|\mathcal{X}) dG. \tag{2.6}$$

Given this definition, the next question becomes: how do we estimate the parameters G in this framework given that we have no direct definition of $P(G|\mathcal{X})$? The answer comes in the form of Bayes's law (Bayes and Price, 1763), which allows us to decompose the probability $P(G|\mathcal{X})$ as follows

$$P(G|\mathcal{X}) = \frac{P(\mathcal{X}|G)P(G)}{P(\mathcal{X})}. \tag{2.7}$$

Here, $P(\mathcal{X}|G)$ is the *likelihood*, which can be calculated trivially according to the parameters of the multinomial distribution

$$P(\mathcal{X}|G) = \prod_{x=k\in\mathcal{X}} P(x = k|G) \tag{2.8}$$

$$= \prod_{x=k\in\mathcal{X}} g_k \tag{2.9}$$

which can be simplified using the counts c_k into

$$P(\mathcal{X}|G) = \prod_{k=1}^{K} g_k^{c_k}. \tag{2.10}$$

$P(G)$ is the *prior* probability over the parameters, which can be specified according to our prior belief about which parameter configurations are likely.

$P(G)$ is particularly useful in solving the problem of degenerate parameter configurations, as we can define a prior that assigns low probabilities to degenerate configurations, and high probabilities to configurations that are likely according to our prior knowledge about the distribution we are trying to model.

For example, if we know the data $\mathcal{X}$ that we are modeling represents the frequency of words, we will want to define a prior that assigns at least some probability to all possible words (i.e. all sequences of one or more letters), as we don't have any *a priori* knowledge of which words will appear in our training data. We may also want to define the prior so that it gives a low probability to words that we know are highly unlikely, such as extremely long words. Finally, we may want to define a prior that prefers distributions where the most common words are given most of the probability, but with a long tail spreading small amounts of probability over many less common words in accordance to the power-law distribution, as this is a common characteristic of linguistic data (Zipf, 1949; Manning and Schütze, 1999).

Finally, we have the *evidence*, or *normalization* term $P(\mathcal{X})$, which is the likelihood of the data given all possible parameter settings

$$P(\mathcal{X}) = \int P(\mathcal{X}|G)P(G)dG. \tag{2.11}$$

Calculating the normalization term is generally the bottleneck in finding $P(G|\mathcal{X})$, as calculating integrals over arbitrary distributions is computationally intractable. Fortunately, there are special cases where this integral can be calculated efficiently, as described in the following section.

2.2 Conjugate Priors and Stochastic Processes

One tractable way of allowing for calculation of the normalization term is through the use of *conjugate* priors for the distribution that we would like to model. Priors that are conjugate have the favorable property that the product of the prior probability and the likelihood takes the same form as the prior itself. Because the product of these two takes a known form, the normalization term can be calculated analytically without complicated integration. Most common probability distributions have a conjugate prior that can be used in Bayesian inference (Fink, 1997). The multinomial distribution is no exception, using a conjugate prior defined by the Dirichlet distribution, which is explained in more detail in the following section.

2.2.1 The Dirichlet Distribution

The conjugate prior for the K dimensional multinomial distribution is the K dimensional Dirichlet distribution. The Dirichlet distribution is defined over the space of parameters $G = \{g_1, \ldots, g_K\}$ that form legal multinomial distributions. Specifically, the elements of G must all be legal probabilities

$$\forall_{g_k \in G}(0 \leq g_k \leq 1) \tag{2.12}$$

and the probabilities must sum to one

$$\sum_{g_k \in G} g_k = 1. \tag{2.13}$$

The Dirichlet distribution takes the form:

$$P(G; A) = \frac{1}{Z} \prod_{k=1}^{K} g_k^{\alpha_k - 1}. \tag{2.14}$$

The parameters $A = \{\alpha_1, \ldots, \alpha_K\}$ are proportional to the expected probability of elements of G. The normalization term Z can be calculated in closed form as follows (Ferguson, 1973):

$$Z = \frac{\prod_{k=1}^{K} \Gamma(\alpha_k)}{\Gamma(\sum_{k=1}^{K} \alpha_k)} \tag{2.15}$$

where $\Gamma()$ is the gamma function, an extension of the factorial function that can be applied to all real numbers instead of only integers.

The fact that the Dirichlet distribution is conjugate to likelihoods generated by the multinomial distribution can be easily confirmed by multiplying the likelihood in Equation (2.10) with the Dirichlet distribution:

$$\prod_{k=1}^{K} g_k^{c_k} * \frac{1}{Z} \prod_{k=1}^{K} g_k^{\alpha_k - 1} = \frac{1}{Z} \prod_{k=1}^{K} g_k^{c_k + \alpha_k - 1} \tag{2.16}$$

$$\propto \frac{1}{Z_{new}} \prod_{k=1}^{K} g_k^{c_k + \alpha_k - 1}. \tag{2.17}$$

It can be seen that the product of the two is proportional to a new Dirichlet distribution with the counts c_k in the likelihood added to α_k parameters of the prior distribution. This can be normalized appropriately by substituting in a new normalization constant Z_{new} for the old constant Z

$$Z_{new} = \frac{\prod_{k=1}^{K} \Gamma(\alpha_k + c_k)}{\Gamma(\sum_{k=1}^{K} \alpha_k + c_k)}. \tag{2.18}$$

Another important feature of a distribution over parameters of the multinomial distribution is that we should be able to predict the probability of the value of a new data point x_{new} using the expectation of the parameters

$$P(x_{new} = k; A) = \int_G g_k P(G; A) dG \tag{2.19}$$

$$= \mathbb{E}[g_k]. \tag{2.20}$$

In order to calculate this expectation, we first introduce the sum $\alpha_0 = \sum_{k=1}^{K} \alpha_k$, which can be used to find the expectation according to the following equation (a detailed derivation of this expectation is given in (Gelman, 1995))

$$P(x_{new} = k; A) = \frac{\alpha_k}{\alpha_0}. \tag{2.21}$$

Equation (2.21) indicates that the expected value of g_k is proportional to α_k. From a modeling point of view, this means that we can adjust α_k according to our prior knowledge of which g_k is likely to be higher. If we have no prior knowledge, we can set all α_k to equal values, which will result in the expectations of g_k forming a uniform distribution over the space of possible values k. If we believe that a certain value of k is more likely than others, we can set its corresponding α_k to a higher value than the others accordingly.

If we instead want to find the posterior expectation of g_k given the observed data and the Dirichlet prior, we can use the fact that the posterior in Equation (2.16) is also in the form of a Dirichlet distribution, which gives us an expectation as follows:

$$P(x_{new} = k | \mathcal{X}; A) = \frac{c_k + \alpha_k}{\sum_{\tilde{k}=1}^{K} c_{\tilde{k}} + \alpha_0}. \tag{2.22}$$

It should be noted that this use of the predictive probability of a multinomial distribution with a Dirichlet prior is identical to the widely used heuristic technique of *additive smoothing*, where a fixed pseudo-count is added to observation counts before calculating probabilities (Mackay and Petoy, 1995), with the parameter of α_k functioning as the pseudo-count for element k.

Again, taking a look from the modeling perspective, the sum α_0 has important connotations for the value of this posterior probability. If we choose a small value of α_0, our posterior expectation will be easily influenced by even small amounts of data, with the extreme value $\alpha_0 = 0$ reducing to maximum likelihood estimation. On the other hand, if α_0 is large, we will need to see large amounts of data before there is a significant effect on the posterior expectations.

2.2.2 The Dirichlet Process

The previous section described the properties of the K-dimensional Dirichlet distribution, which can be used to define prior probabilities over K-dimensional multinomial distributions. However, it is possible to think of cases where K is essentially infinite. For example, when attempting to define a probability distribution over words in a language, there are an infinite number of combinations of letters that could form a word. In order to create a robust model that can properly handle unknown words, we would like to assign at least a small amount of probability to every possible word that we may see in the future. Models that are formulated in this way are referred to as *non-parametric*, which is somewhat of a misnomer, as the models do not have no parameters, but a potentially infinite number of parameters as K is not explicitly set in advance.

The Dirichlet process is a framework that allows us to model these non-parametric distributions. The main practical difference between the Dirichlet process and the standard Dirichlet distribution is how they are parameterized. While the Dirichlet distribution has a fixed set of parameters $\alpha_1, \ldots, \alpha_K$, the Dirichlet process replaces these with a single parameter α_0 and the *base measure* P_{base}

$$\alpha_k = \alpha_0 P_{base}(x = k). \tag{2.23}$$

It can be seen that P_{base} is equal to the expectation as shown in Equation (2.21).

This may seem like a superficial change, but it is actually quite important in practice, as it allows us to easily define Dirichlet priors over elements of any space that can be given a probability according to a distribution P_{base}, including infinite discrete spaces, or even continuous spaces. For example, let us consider a model for which P_{base} is generated according to the Poisson distribution parameterized by λ

$$P_{base}(k; \lambda) = \frac{(\lambda - 1)^{k-1}}{(k-1)!} e^{-(\lambda - 1)}. \tag{2.24}$$

In this model the base measure will give some probability to all natural numbers, resulting in a multinomial distribution with an expected value equal to the Poisson base measure, but instead of using an infinite number of hyper-parameters to represent each natural number, we have only two hyper-parameters α_0 and λ.

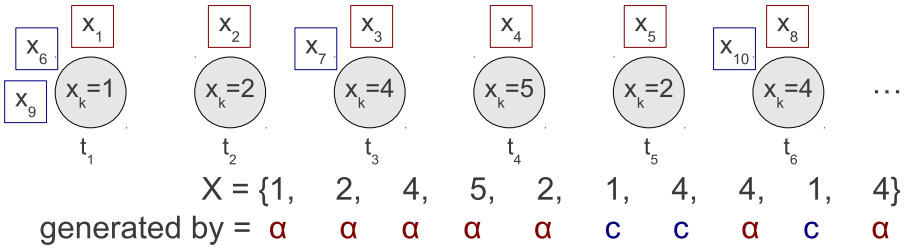

Figure 2.1: An example of a configuration of the Chinese restaurant process. Circles represent tables, labeled with the food (value k) that they generate. Squares represent customers, labeled with their index in $\mathcal{X}$. Each value of $\mathcal{X}$ is also marked with whether it was generated as a new table from α, or as an existing table from the cache c.

2.2.3 The Chinese Restaurant Process

One other way of looking at the posterior probability of the Dirichlet process (or other stochastic process) is based on a representation scheme called the *Chinese Restaurant Process* (CRP) (Pitman, 1995). This representation is useful for a number of reasons, in that it allows us to calculate the marginal probability of the observed data $\mathcal{X}$ given the parameters α_0 and P_{base}, and also allows for intuitive representation of more complex models during the process of Gibbs sampling. In order to describe this process, Figure 2.1 shows an example of one configuration for the Chinese restaurant process for $\mathcal{X} = \{1, 2, 4, 5, 2, 1, 4, 4, 1, 4\}$.

The basic concept of the Chinese restaurant process is that there is a Chinese restaurant with a potentially unlimited number of tables. Each time a customer enters the restaurant, he or she will, according to some probability, choose to sit either at any of the existing tables in $T = \{t_1, \ldots, t_J\}$ that already has at least one customer, or at a new table t_{J+1}. For the Dirichlet process, these probabilities are:

$$P(t_j) = \frac{c_{t_j}}{\alpha_0 + \sum_{\tilde{j}=1}^{J} c_{t_{\tilde{j}}}} \tag{2.25}$$

$$P(t_{J+1}) = \frac{\alpha_0}{\alpha_0 + \sum_{\tilde{j}=1}^{J} c_{t_{\tilde{j}}}} \tag{2.26}$$

where c_{t_j} is the number of customers sitting at table t_j. If the customer sits at a new table, we decide which type of food will be served at that table according to the base measure P_{base}, which the customer will then proceed

to eat. If the customer decides to sit at an existing table, the customer will proceed to eat the food that is already being served at the table. Here, the number of customers eating a particular food k is equal to c_k.

While it may be difficult to tell the relation between Chinese restaurants and Dirichlet processes at first glance, there is a very clear connection between this process and the calculation of the marginal probability of the data $P(\mathcal{X}; \alpha_0, P_{base})$ after we have marginalized out the multinomial distribution parameters G. First, note that the probability of $\mathcal{X}$ can be decomposed into the product of the conditional probabilities using the chain rule

$$P(\mathcal{X}; \alpha_0, P_{base}) = \prod_{i=1}^{I} P(x_i | x_1^{i-1}; \alpha_0, P_{base}) \tag{2.27}$$

where x_1^{i-1} is used as short hand for $\{x_1, \ldots, x_{i-1}\}$. We then substitute in the posterior probability of Equation (2.16)

$$P(\mathcal{X}; \alpha_0, P_{base}) = \prod_{i=1}^{I} \frac{c_{x_1^{i-1}, x_i} + \alpha_0 P_{base}(x_i)}{i - 1 + \alpha_0} \tag{2.28}$$

$$= \prod_{i=1}^{I} \left(\frac{c_{x_1^{i-1}, x_i}}{i - 1 + \alpha_0} + \frac{\alpha_0 P_{base}(x_i)}{i - 1 + \alpha_0} \right). \tag{2.29}$$

Note that here we are using counts $c_{x_1^{i-1}, x_i}$ not over all of $\mathcal{X}$, but over the previously generated elements x_1^{i-1}, as we are conditioning on only the previously generated elements as shown in Equation (2.27).

After this transformation, the correspondence between the first of the two elements of Equation (2.29) and Equation (2.25) is clear. In addition, it can be seen that the second of the two elements in Equation (2.29) is equal to the probability of choosing a new table in Equation (2.26) multiplied by the base measure probability P_{base}, with which we choose which food must be served at that table.

While this equivalence is interesting, it still does not make clear the motivation for the Chinese restaurant process. The true power of the Chinese restaurant process lies in the fact that it allows us to keep track of the number of times that a particular value of $x = k$ was generated by the contribution of the base distribution $\alpha_0 P_{base}$, as opposed to the counts (or cache) c_k. This is done by keeping track of the number of tables for which $x = k$, which will be represented using the function t_k. These table counts, while not important for the simple Dirichlet process, are essential for accurately calculating the probabilities of other models such as the Pitman-Yor

process explained in the next section, or the hierarchical models described in the rest of this thesis.

One other important thing to notice about the Chinese restaurant process is that the joint probability in Equation (2.29) is the same regardless of the order in which we add the customers or tables. The denominator depends only on the total number of customers, while the numerator depends only on the number of customers sitting at tables serving a particular dish (as well as the number of tables for the Pitman-Yor process described in Section 2.2.4). This property of probabilities being agnostic to order is called *exchangeability*, and is useful for learning using Gibbs sampling as described in Section 2.3.

2.2.4 The Pitman-Yor Process

The Pitman-Yor process is a generalization of the Dirichlet process that allows for more expressive modeling (Pitman and Yor, 1997; Ishwaran and James, 2001). The Pitman-Yor process has three parameters: a discount d, strength s, and base measure P_{base}. Given these parameters, observed data $\mathcal{X}$, and the table distribution according to the Chinese restaurant process, the posterior probability of x given a Pitman-Yor process prior is

$$P(x|\mathcal{X}, T; d, s, P_{base}) = \frac{c_x - d * t_x + (s + d\sum_{\tilde{x}} t_{\tilde{x}})P_{base}(x)}{\sum_{\tilde{x}} c_{\tilde{x}} + s}. \tag{2.30}$$

When compared to the posterior of the multinomial distribution with a Dirichlet prior (Equation (2.16)) we can see that the Pitman-Yor s directly corresponds with α_0, and the only difference is the addition of the discount d, which is subtracted once for every table that corresponds to x.

However, this discount is extremely important for modeling language (Kneser and Ney, 1995; Teh, 2006; Durrett and Klein, 2011). Figure 2.2 gives an intuition of why this is true. The figure details an experiment where a large corpus of text was divided exactly into two, one training set and one testing set, so that each set contains C words. The words are sorted into buckets based on their frequency in the training set c_{train}, and c_{test} measures the actual average frequency of the words in each bucket using the testing set. The correspondence between the training and the testing frequency gives us an idea of how much probability we should give to each word in the predictive distribution given its count in the training data.

In the case of the Dirichlet prior, our estimate for the testing set fre-

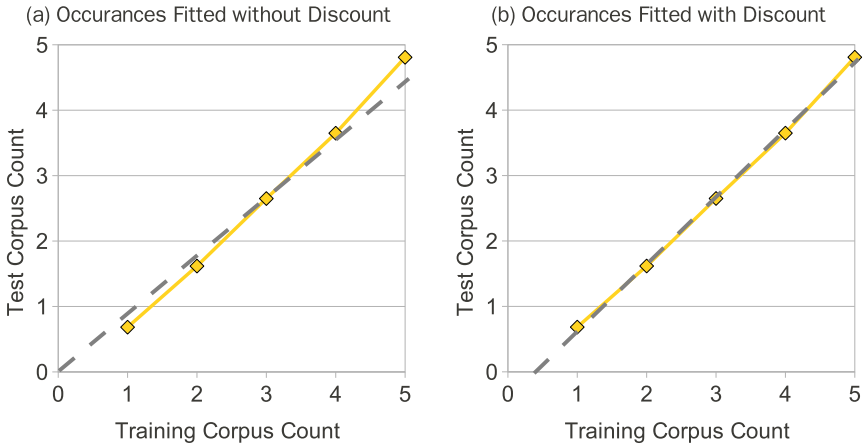

Figure 2.2: The actual (solid line) and estimated (dashed line) number of times a word occurred in the test corpus based on how many times it occurred in the training corpus fitted (a) without a discount (Dirichlet prior), and (b) with a discount (Pitman-Yor prior).

quency $\tilde{c}_{test}$ is as follows:

$$\tilde{c}_{test,x} = P_{train}(x)C \tag{2.31}$$

$$= \frac{c_{train,x} + \alpha_0 P_{base}(x))}{C + \alpha_0} C \tag{2.32}$$

$$= (c_{train,x} + \alpha_0 P_{base}(x)) \frac{C}{C + \alpha_0} \tag{2.33}$$

If we ignore the contribution of the base measure probability[1] we can see that the estimated test set frequency is simply the training set frequency multiplied by a constant $\frac{C}{C+\alpha_0}$. This sort of discounting allows us to fit the frequencies using estimates similar to Figure 2.2 (a), with a straight line that passes through the origin, with the value of the constant modifying the slope of this line. However, it can be seen that this does not provide a good fit for the actual testing counts, overestimating the frequency of less frequent words, and underestimating the frequency of more frequent words.

In contrast, when we examine the same frequency with the additional

[1]When $C \gg \alpha_0$, which is true in most cases, the counts contributed by the base measure are generally significantly smaller than those contributed by the cache counts.

discount allowed by the Pitman-Yor prior

$$\tilde{c}_{test}(x) = (c_{train,x} - d * t_{train}(x) + sP_{base}(x))\frac{C}{C+s}, \tag{2.34}$$

and assume for simplicity that $t_{train}(x) = 1$, we are able to create discounts that correspond to any straight line that either travels through the origin, or anywhere below it. As can be seen in Figure 2.2 (b), this provides a much better fit for word frequencies. This discount allows for modeling of distributions with a few common instances and many uncommon instances (power-law distributions). Power-law distributions are known to be good models of not only word counts (Zipf, 1949), but also parts of speech (Goldwater and Griffiths, 2007), syntactic structures (Johnson et al., 2007b), and other aspects of natural language.

2.3 Gibbs Sampling

The previous section described how to calculate probabilities for Bayesian probabilistic models given an observed data set $\mathcal{X}$. However, for most models of interest, in addition to our observed variables, we also have a set of latent variables $\mathcal{Y}$. Given that the goal of Bayesian learning is to discover the distribution over the parameters $P(G|\mathcal{X})$, introducing additional hidden variables indicates that we will have to take the integral over possible configurations of these hidden variables

$$P(G|\mathcal{X}) = \int P(G|\mathcal{Y},\mathcal{X})P(\mathcal{Y}|\mathcal{X})d\mathcal{Y}. \tag{2.35}$$

Unfortunately, these integrals are often intractable and need to be approximated. There are two main techniques for approximating these integrals, variational Bayes (Beal, 2003) and Gibbs sampling (Geman and Geman, 1984). This thesis focuses on the latter, as it allows for relatively straightforward implementation of the more complex models introduced therein.

2.3.1 Latent Variable Models

In order to demonstrate the basics of Gibbs sampling, we can take an example from unsupervised word segmentation, which will be a recurring theme throughout the rest of this thesis. In unsupervised word segmentation, we assume we have an observed corpus of unsegmented text $\mathcal{X}$ that was generated as a word sequence specified by latent variables $\mathcal{Y}$ from some

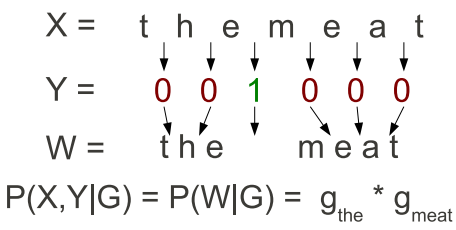

Figure 2.3: Word segmentation with a probabilistic model with characters $\mathcal{X}$, word boundary indicators $\mathcal{Y}$, words W, and likelihood $P(\mathcal{X}, \mathcal{Y}|G)$.

word-based generative model with parameters G. Figure 2.3 shows an example of a sentence within this framework, which we will assume is part of a larger corpus. Here, the observed variables $X \in \mathcal{X}$ indicate characters of a single sentence and hidden variables $Y \in \mathcal{Y}$ indicate whether a word boundary exists between the corresponding characters. Finally, for convenience, we define a set of variables $W \in \mathcal{W}$ that represent the sequence of words that will be created when X is segmented according to Y. The pair $\langle X, Y \rangle$ and the words W uniquely determine each other, so they can be used interchangeably.

As the likelihood for this model we use a multinomial distribution over words and assume that each word is generated independently, giving us the following likelihood of the entire training data:

$$P(\mathcal{X}, \mathcal{Y}|G) = P(\mathcal{W}|G) \tag{2.36}$$

$$= \prod_{w \in \mathcal{W}} g_w. \tag{2.37}$$

In order to calculate the posterior distribution of the parameters given the observed data $P(G|X)$, we can perform the familiar transformation using Bayes's law:

$$P(G|\mathcal{X}) = \frac{P(\mathcal{X}|G)P(G)}{P(\mathcal{X})} \tag{2.38}$$

and introduce the likelihood term $P(\mathcal{X}, \mathcal{Y}|G)$ by marginalizing over $\mathcal{Y}$

$$P(G|\mathcal{X}) = \int \frac{P(\mathcal{X}, \mathcal{Y}|G)P(G)}{P(\mathcal{X})} d\mathcal{Y}. \tag{2.39}$$

However, here we run into a problem. As changes in the value of $\mathcal{Y}$ will affect the distribution over all G and changes in the value of G will affect

the distribution over $\mathcal{Y}$, it becomes impossible to calculate this integral in closed form. Instead, we turn to methods that can approximate this integral in a computationally tractable fashion.

2.3.2 Basic Gibbs Sampling

The most widely used method for approximating this integral over $\mathcal{Y}$ is Gibbs sampling. The basic premise of sampling is based on the law of large numbers: if we make enough observations of a random variable that is generated according to some distribution, the distribution over observations will eventually approach the true distribution. This means that instead of analytically solving the integral in Equation (2.39), if we can generate samples of $\mathcal{Y}$, we can instead approximate this integral with the average over each sample $\{\mathcal{Y}_1, \ldots, \mathcal{Y}_N\}$:

$$P(G|\mathcal{X}) \approx \frac{1}{N} \sum_{n=1}^{N} \frac{P(\mathcal{X}, \mathcal{Y}_n|G)P(G)}{P(\mathcal{X})} \tag{2.40}$$

However, while it is easy to generate each sample $\mathcal{Y}_n$ from simple distributions such as the multinomial, it is often not possible to directly sample from multivariate distributions with complex interactions between the component variables, as in the previous word segmentation example.

Gibbs sampling is a technique that allows for the sampling from multivariate distributions, relying on the fact that if we sample one variable at a time conditioned on all the other variables, we can simulate the true distribution (Geman and Geman, 1984). The intuition behind this is based on the fact that if we assume that all variables in $\mathcal{Y}$ except y (denoted $\mathcal{Y}\backslash y$) are already distributed according to the true joint distribution, and sample y according to its conditional distribution given $\mathcal{Y}\backslash y$, we will recover the true distribution over all variables in $\mathcal{Y}$

$$P(y|\mathcal{Y}\backslash y)P(\mathcal{Y}\backslash y) = P(\mathcal{Y}). \tag{2.41}$$

This indicates the distribution is *invariant*, which is one of the conditions for convergence of Gibbs sampling.

Even if we cannot assume the current $\mathcal{Y}\backslash y$ was distributed correctly according to $P(\mathcal{Y}\backslash y)$, each time we draw a new sample, we will get slightly closer to the true distribution, and will approach the true distribution in the limit as long as the distribution is *ergodic*. Ergodicity is the property that every configuration of $\mathcal{Y}$ is reachable from every other configuration of $\mathcal{Y}$ with non-zero probability. A sufficient (but not necessary) condition for

ergodicity in the Gibbs sampling framework is that each time we sample a value for y, all of its possible values are given a non-zero probability. If this is the case, any configuration $\mathcal{Y}$ can be reached with non-zero probability by sampling the values of each of its corresponding y one-by-one.

In addition, the property of exchangeability mentioned in Section 2.2.3 is particularly useful for Gibbs sampling. The reason for this is that every time we sample a new value for y, if the values of $\mathcal{Y}$ are exchangeable we can assume that current y of interest was the last value generated from the distribution. For example, for multinomials with Dirichlet process priors, this makes calculating the probability $P(y|\mathcal{Y}\backslash y)$ as simple as calculating the probability of generating one more value from the posterior in Equation (2.16). On the other hand, if exchangeability does not hold, it may be necessary consider the probability of y itself, along with the effect that y has upon all following values in $\mathcal{Y}$, a much more computationally intensive task.

2.3.3 Gibbs Sampling for Word Segmentation

To further illustrate the sampling process, let us take word segmentation as an example and imagine the situation where we want to sample y_5, which determines whether there is a boundary between "e" and "a" in "meat" (or "me at"). To find the probability that the boundary exists, we first remove the effect that this word boundary has on the model by subtracting all of the counts that are influenced by the choice of y_5. In this case, y_5 lies within the word "meat" so we subtract one from its count:

$$c_{\mathcal{W}\backslash w,\text{“meat”}} \leftarrow c_{\mathcal{W},\text{“meat”}} - 1. \tag{2.42}$$

Next, we calculate the probability of whether this new boundary exists according to the posterior probability of the Dirichlet process

$$P(y_5 = 0) \propto \frac{c_{\mathcal{W}\backslash w,\text{“meat”}} + \alpha_0 P_{base}(w = \text{“meat”})}{\sum_{\tilde{w}} c_{\mathcal{W}\backslash w,\tilde{w}} + \alpha_0} \tag{2.43}$$

$$P(y_5 = 1) \propto \frac{c_{\mathcal{W}\backslash w,\text{“me”}} + \alpha_0 P_{base}(w = \text{“me”})}{\sum_{\tilde{w}} c_{\mathcal{W}\backslash w,\tilde{w}} + \alpha_0} * \frac{c_{\mathcal{W}\backslash w,\text{“at”}} + \alpha_0 P_{base}(w = \text{“at”})}{\sum_{\tilde{w}} c_{\mathcal{W}\backslash w,\tilde{w}} + \alpha_0 + 1}. \tag{2.44}$$

Note that if $y_5 = 0$, we have generated a single word "meat" from the distribution, while in the case of $y_5 = 1$ we have generated two words, "me" and "at". It should also be noted that we are adding a count of 1

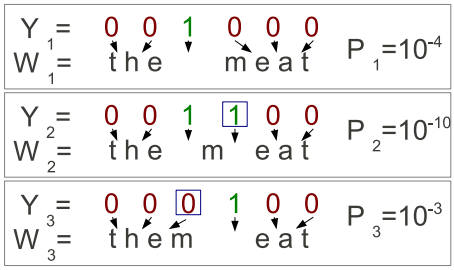

Figure 2.4: An example in which sampling must pass through a low probability region to reach a high probability region.

to the denominator of "at" in the case of $y_5 = 1$. This corresponds to the newly added count of "me," which must be considered, as it was generated separately from "at."

Once the above equations have been calculated, we randomly pick a new value for y_5 according to these probabilities. If we choose $y_5 = 0$, we add one to the count $c_{\text{"meat"}}$, and if we choose $y_5 = 1$, we add one to each of the counts for $c_{\text{"me"}}$ and $c_{\text{"at"}}$. With the value of y_5 and its corresponding counts updated, we can proceed to the next value in $\mathcal{Y}$ (in arbitrary order) and continue the process. By proceeding to sample every value in $\mathcal{Y}$ for a number of iterations, we will achieve an approximation that will approach the true distribution as the number of samples goes to infinity.

2.3.4 Blocked Gibbs Sampling

While any form of Gibbs sampling is guaranteed to approach the true distribution as the number of samples goes to infinity, in practice we will never have infinite time and computing power with which to train our models. In the scenario where we have limited time, simple Gibbs sampling has the undesirable feature of getting stuck in local maxima.

An example of this can be found in our word segmentation example, which is demonstrated in Figure 2.4 with an example of three possible segmentations for a single sentence. In this case, the current configuration Y_1 is ten times less likely than the configuration Y_3, so we would expect the sampler to travel to Y_3 and spend ten times as much time there than in configuration Y_1. However, in order to travel from Y_1 to Y_3, we must change the values of two separate variables. As the traditional Gibbs sampler is only able to change a single variable at a time, this means that we will have

to choose each of these two changes independently. If the intermediate hypotheses such as Y_2 are highly unlikely, this means that it may take a large number of samples to make one of the two required changes. In the case of the example, the probability of Y_2 is 10^{-6} less than Y_1 (which is not an altogether unreasonable number in most models of unsupervised learning), and thus it would require an average of one million iterations to escape from the local maximum.

There have been several solutions proposed to this problem (Hukushima and Nemoto, 1996; Liang et al., 2010), but here we focus on blocked sampling (Jensen et al., 1995; Ishwaran and James, 2001; Scott, 2002), which is used extensively in this thesis. Blocked Gibbs sampling is based on the idea that while it is generally not possible to acquire a sample for all variables in $\mathcal{Y}$ simultaneously, it is often possible to acquire a sample for some subset $Y \subset \mathcal{Y}$ according to the true distribution where $|Y| > 1$. One example would be that $\mathcal{Y}$ describes the word boundaries for an entire corpus of observed data, while Y describes the word boundaries for a single sentence within this corpus. If we are able to explicitly sample over all possible configurations of Y, the problem of intermediate states disappears, allowing us to jump from Y_1 to Y_3 or vice-versa in a single step.

In the case of word segmentation, we can acquire a sample of Y for a single sentence by recasting the problem as the task of finding W, assuming independence between each $w \in W$, and performing dynamic programming to allow for efficient calculation. This process is described fully in Section 3.3, but the important point to note is the independence assumption between words in W, which results in a sample from the *proposal* distribution:

$$P_{prop}(W|\mathcal{W}\backslash W) = \prod_{w \in W} \frac{c_{\mathcal{W}\backslash W,w} + \alpha_0 P_{base}(w)}{\sum_{\tilde{w}} c_{\mathcal{W}\backslash W,\tilde{w}} + \alpha_0} \tag{2.45}$$

where $\mathcal{W}\backslash W$ and $c_{\mathcal{W}\backslash W}$ respectively indicate the corpus and counts with sentence W removed.

While this particular proposal distribution is a close approximation to the actual probability of the word segmentation, it is not exact. As noted previously in Equation (2.44), we must actually add the counts of each word to the denominator before generating the next word to achieve the true marginal probabilities of the multinomial distribution with a Dirichlet prior. In addition, if a particular word w is generated more than once in a sentence, we must add the additional counts to the c_w term in the numerator as well. In order to express this, we define uniq(W) as the collection of all unique words in W, and get the following true conditional probability for

W given $\mathcal{W}\backslash W$:

$$P_{true}(W|\mathcal{W}\backslash W) = \frac{\left(\prod_{x\in \text{uniq}(W)} \prod_{i=0}^{c_{W,x}-1} c_{\mathcal{W}\backslash W,x} + i + \alpha_0 P_{base}(x)\right)}{\left(\prod_{j=0}^{|W|-1} |\mathcal{W}\backslash W| + j\right)}. \quad (2.46)$$

The numerator here takes into account words that are generated more than one time in W, and the denominator takes into account the fact that the number of words in $\mathcal{W}$ increases as we generate each word in W.

It can be seen that there is a small gap between the proposal distribution P_{prop} and the true distribution P_{true}. Fortunately, there exists a method called Metropolis-Hastings sampling that allows us to close this gap and ensure that we are sampling from the correct distribution (Hastings, 1970; Johnson et al., 2007b). This technique is based on *rejection sampling* where we choose a sample from the proposal distribution, then decide to accept or reject it based on an acceptance probability A. In the case of Metropolis-Hastings sampling, this acceptance probability is defined using the probabilities P_{prop} and P_{true} for the old and new values of W, which are labeled W_{old} and W_{new} respectively[2]

$$A(W_{old}, W_{new}) = \min\left(\frac{P_{true}(W_{new})}{P_{true}(W_{old})}\frac{P_{prop}(W_{old})}{P_{prop}(W_{new})}, 1\right). \quad (2.47)$$

Mathematical details of this result can be found in the references (Hastings, 1970), but the basic intuition lies in the comparison between the ratios of the P_{true} and P_{prop}. When the probability ratio of W_{new} to W_{old} is higher for P_{prop} than it is for P_{true}, we will be acquiring more samples of W_{new} than are justified by P_{true}. The Metropolis-Hastings method adjusts for this fact by reducing the acceptance probability accordingly, so we reject some of these over-produced samples, allowing us to remain faithful to the true distribution. If this method is properly applied, we can be guaranteed to be sampling from the proper distribution, as the distribution will satisfy the property of *detailed balance*, which is a sufficient condition for the distribution being invariant (Bishop, 2006).

2.3.5 Gibbs Sampling as Stochastic Search

Up until this point we have presented Gibbs sampling as a method for approximating Equation (2.39)'s integral over $\mathcal{Y}$ to estimate the posterior

[2]Simple Gibbs sampling is actually a specific case of Metropolis-Hastings sampling where $P_{prop} = P_{true}$.

probability of G. However, in many cases (perhaps the majority in natural language processing), we are more interested in the actual values of $\mathcal{Y}$ than the distribution over G. In our running word segmentation example, this would indicate that we are interested in the word sequence $\mathcal{W}$ of our corpus more than the word probabilities G, a perfectly reasonable proposition if this corpus is to be used, for example, as data for training of machine translation or speech recognition systems. This indicates that Gibbs sampling can be used as a method not for approximating the integral over $\mathcal{Y}$, but instead as a stochastic search method to find some value for the latent variables $\mathcal{Y}$ that lies in a high-probability section of the space over all possible $\mathcal{Y}$.

Within this context, it is often better to focus on implementation motivated considerations, even at the cost of some degree of mathematical correctness, when and only when they are justified. For example, when estimating the parameters G using sampling, it is technically necessary to average over values obtained at each sample to obtain the true distribution. In the speech recognition experiments of Section 3.4, this provides significant gains in accuracy, so practical concerns suggest that we adopt a method to take this sample averaging into account. However, Section 4.7 finds that for machine translation, this can lead to larger models without providing significant gains in down-stream accuracy, so we are justified by practical considerations in using only a single sample.

In addition, in the case of block sampling where the most efficiently computable proposal distributions differ significantly from the true distribution (such as the ITG-based sampling model proposed in Chapter 4), we may be faced with prohibitively high rejection rates when performing the Metropolis-Hastings step. This can hurt learning, particularly in the early stages, as in most cases rejected samples of W_{new} are still generally higher in probability than W_{old} according to both P_{prop} and P_{true}, but rejected because of mismatches between the two.

One empirical example of this is shown in Figure 2.5, which graphs the likelihoods of the samples obtained by the 2-gram word segmentation model described by (Mochihashi et al., 2009) on 36,000 sentences with a 98% acceptance rate. Despite the fact that the acceptance rate is relatively high, there is still a significant gap in the likelihoods of the samples obtained by the two methods. Thus, in many cases it is better to sacrifice correctness by skipping the Metropolis-Hastings step to remove the effect of rejection rates and achieve a faster convergence to a high-probability configuration of $\mathcal{Y}$.[3] In fact, this was found to be true for all of the models considered in this

[3]On the other hand, the Metropolis-Hastings step is very useful for detecting mistakes

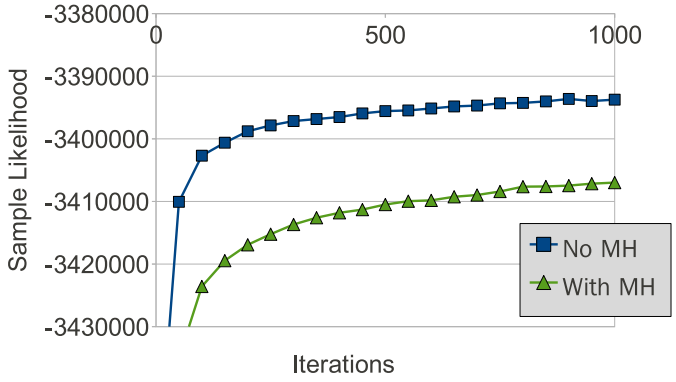

Figure 2.5: Likelihoods with and without Metropolis-Hastings sampling.

thesis, so the Metropolis-Hastings rejection step is never used, even when it is necessary for strict mathematical correctness.

This thesis mainly uses Gibbs sampling as a tool for stochastic search, and will be making these sort of approximations when they are motivated.

in the implementation of sampling algorithms. In general, the calculation of P_{prop} and P_{true} must be performed separately, and mistakes in the implementation of either will lead to unrealistically high rejection rates, which are easy to notice and debug.

Chapter 3

Learning a Language Model from Continuous Speech

A language model (LM) is an essential part of automatic speech recognition (ASR) systems, providing linguistic constraints on the recognizer and helping to resolve the ambiguity inherent in the acoustic signal. Traditionally, these LMs are learned from digitized text, preferably text that is similar in style and content to the speech that is to be recognized. In addition, this text is generally annotated with word or morpheme boundaries using the supervised techniques introduced in Section 1.1, and a dictionary mapping the surface form of words to their pronunciations must be prepared.

This chapter proposes a new paradigm for LM learning, using not digitized text but audio data of continuous speech. The proposition of learning an LM from continuous speech is motivated from a number of viewpoints. First, the properties of written and spoken language are very different (Tannen, 1982), and LMs learned from continuous speech can be expected to naturally model spoken language. In contrast, when language models are learned from written text, it is often necessary to manually transcribe speech or compensate for these differences by transforming written-style text into spoken-style text when creating an LM for ASR (Vergyri and Kirchhoff, 2004; Akita and Kawahara, 2010). Second, learning lexical units and their context from speech can allow for out-of-vocabulary word detection and acquisition, which has been shown to be useful in creating more adaptable and robust ASR or dialog systems (Bazzi and Glass, 2001; Hirsimäki et al., 2006). Learning LMs from speech can also provide a powerful tool in efforts for technology-based language preservation (Abney and Bird, 2010), particularly for languages that have a rich oral, but not written tradition.

Finally, as human children learn language from speech, not text, computational models for learning from speech are of great interest in the field of cognitive science (Roy and Pentland, 2002).

For all of the previous reasons, there has been a significant amount of work on learning lexical units from speech data. These include statistical models based on the minimum description length or maximum likelihood frameworks, which have been trained on one-best phoneme recognition results (de Marcken, 1995; Deligne and Bimbot, 1997; Gorin et al., 1999) or recognition lattices (Driesen and Hamme, 2008). There have also been a number of works that use acoustic matching methods combined with heuristic cutoffs that may be adjusted to determine the granularity of the units that need to be acquired (ten Bosch and Cranen, 2007; Park and Glass, 2008; Jansen et al., 2010). Finally, many works, inspired by the multimodal learning of human children, use visual and audio information (or at least abstractions of such) to learn lexical units without text (Roy and Pentland, 2002; Iwahashi, 2003; Yu and Ballard, 2004).

The work presented in this chapter is different from these other approaches in that it is the first model that is able to learn a full n-gram language model from raw audio, and demonstrate that this model can be used to reduce the phoneme error rate of speech in an ASR system. The first step in learning an LM from continuous speech is to generate lattices of phonemes without any linguistic constraints using a standard ASR acoustic model. To learn an LM from this data, this chapter builds on recent work in unsupervised word segmentation of text (Mochihashi et al., 2009), proposing a novel inference procedure that allows for models to be learned over lattice input. For LM learning, the proposed technique uses the hierarchical Pitman-Yor LM (HPYLM) (Teh, 2006), a variety of LM that is based on non-parametric Bayesian statistics. As mentioned in Chapter 2, non-parametric Bayesian statistics are well suited to this learning problem, as they allow for automatically balancing model complexity and expressiveness, and have a principled framework for learning through the use of Gibbs sampling.

To perform sampling over phoneme lattices, all models are represented using weighted finite state transducers (WFSTs), which allow for simple and efficient combination of the phoneme lattices with the LM. Using this combined lattice, we can use a variant of the forward-backward algorithm to efficiently sample a phoneme string and word segmentation according to the model probabilities. By performing this procedure on each of the utterances in the corpus for several iterations, it is possible to effectively discover phoneme strings and lexical units appropriate for LM learning,

even in the face of acoustic uncertainty.

Finally, experiments are performed to test the feasibility of learning an LM from only audio files of fluent adult-directed meeting speech with no accompanying text. The results of the experiment show that, despite the lack of any text data, the proposed model is able to both decrease the phoneme recognition error rate over a separate test set and acquire a lexicon with many intuitively reasonable lexical entries. Moreover, the proposed lattice processing approach proves effective for overcoming acoustic ambiguity present during the training process.

Section 3.1 briefly overviews the process of speech recognition, including language modeling and representation of ASR models in the WFST framework. Section 3.2 describes previous research on LM-based unsupervised word segmentation in more detail. Section 3.3 proposes a method for formulating LM-based unsupervised word segmentation using a combination of WFSTs and Gibbs sampling. The description concludes in Section 3.3.3 by showing that the WFST-based formulation allows for LM learning directly from speech, even in the presence of acoustic uncertainty. Section 3.4 describes the results of an experimental evaluation demonstrating the effectiveness of the proposed method, and Section 3.5 concludes the chapter and discusses future directions.

3.1 Speech Recognition and Language Modeling

This section provides an overview of ASR and language modeling and provides definitions that will be used in the rest of the chapter.

3.1.1 Speech Recognition

ASR can be formalized as the task of finding a series of words W given acoustic features U of a speech signal containing these words. Most ASR systems use statistical methods, creating a model for the posterior probability of the words given the acoustic features, and searching for the word sequence that maximizes this probability

$$\hat{W} = \underset{W}{\operatorname{argmax}} P(W|U). \tag{3.1}$$

As this posterior probability is difficult to model directly, Bayes's law is

used to decompose the probability

$$\hat{W} = \underset{W}{\operatorname{argmax}} \frac{P(U|W)P(W)}{P(U)} \tag{3.2}$$

$$= \underset{W}{\operatorname{argmax}} P(U|W)P(W). \tag{3.3}$$

Here, $P(U|W)$ is computed by the acoustic model (AM), which makes a probabilistic connection between words and their acoustic features. However, directly modeling the acoustic features of the thousands to millions of words in large-vocabulary ASR systems is not realistic due to data sparsity issues. Instead, AMs are trained to recognize sequences of phonemes X, which are then mapped into the word sequence W. Phonemes are defined as the smallest perceptible linguistic unit of speech. Thus, the entire ASR process can be described as finding the optimal word sequence according to the following formula

$$\hat{W} = \underset{W}{\operatorname{argmax}} \sum_{X} P(U|X)P(X|W)P(W). \tag{3.4}$$

This is usually further approximated by choosing the single most likely phoneme sequence to allow for efficient search:

$$\hat{W} = \underset{W,X}{\operatorname{argmax}} P(U|X)P(X|W)P(W). \tag{3.5}$$

Here, $P(U|X)$ indicates the AM probability and $P(X|W)$ is a lexicon probability that maps between words and their pronunciations. $P(W)$ is computed by the LM, which will be described in more detail in the following section. It should be noted that in many cases a scaling factor α is used

$$\hat{W} = \underset{W,X}{\operatorname{argmax}} P(U|X)P(X|W)P(W)^{\alpha}. \tag{3.6}$$

This allows for the adjustment of the relative weight put on the LM probability, with a higher weight indicating that the LM will have a large influence on the recognition results, preferring to generate well-formed sentences, while a lower weight will indicate that the acoustic model will have a large influence, resulting in sentences that closely match the acoustic features.

3.1.2 Language Modeling

The goal of the LM probability $P(W)$ is to provide a preference towards "good" word sequences, assigning high probability to word sequences that

the speaker is likely to say, and low probability to word sequences that the speaker is unlikely to say. By doing so, this allows the ASR system to select linguistically proper sequences when purely acoustic information is not enough to correctly recognize the input.

The most popular form of LM is the n-gram, which is notable for its simplicity, computational efficiency, and surprising power (Goodman, 2001). n-gram LMs are based on the fact that it is possible to calculate the joint probability of $W = w_1^I$ sequentially by conditioning on all previous words in the sequence using the chain rule

$$P(W) = \prod_{i=1}^{I} P(w_i|w_1^{i-1}). \tag{3.7}$$

Conditioning on previous words in the sequence allows for the consideration of contextual information in the probabilistic model. However, as few sentences will contain exactly the same words as any other, conditioning on all previous words in the sentence quickly leads to data sparseness issues. n-gram models resolve this problem by only conditioning on the previous $(n-1)$ words when choosing the next word in the sequence

$$P(W) \approx \prod_{i=1}^{I} P(w_i|w_{i-n+1}^{i-1}). \tag{3.8}$$

The conditional probabilities are generally trained from a large corpus of word sequences $\mathcal{W}$. From $\mathcal{W}$ we can calculate the counts of each subsequence of n words w_{i-n+1}^{i} (an "n-gram"). From these counts, it is possible to compute conditional probabilities using maximum likelihood estimation

$$P_{ml}(w_i|w_{i-n+1}^{i-1}) = \frac{c_{w_{i-n+1}^{i}}}{c_{w_{i-n+1}^{i-1}}}. \tag{3.9}$$

However, even if we set n to a relatively small value, we will never have a corpus large enough to exhaustively cover all possible n-grams. In order to deal with this data sparsity issue, it is common to use a framework that references higher order n-gram probabilities when they are available, and falls back to lower order n-gram probabilities according to a *fallback* probability $P(FB|w_{i-n+1}^{i-1})$:

$$P(w_i|w_{i-n+1}^{i-1}) = \begin{cases} P_s(w_i|w_{i-n+1}^{i-1}) & \text{if } c_{w_{i-n+1}^{i}} > 0, \\ P(FB|w_{i-n+1}^{i-1})P(w_i|w_{i-n+2}^{i-1}) & \text{otherwise.} \end{cases}$$

By combining more accurate but sparse higher-order n-grams with less accurate but more reliable lower-order n-grams, it is possible to create LMs that are both accurate and robust. To reserve some probability for $P(FB|w_{i-n+1}^{i-1})$, we replace P_{ml} with the smoothed probability distribution P_s. P_s can be defined according to a number of smoothing methods, which are described thoroughly in (Chen and Goodman, 1996).

3.1.3 Bayesian Language Modeling

While traditional methods for LM smoothing are based on heuristics (often theoretically motivated), it is also possible to motivate language modeling from the perspective of Bayesian statistics (Mackay and Petoy, 1995; Teh, 2006). In order to perform smoothing in the Bayesian framework, we first define a variable $g_{w_i|w_{i-m+1}^{i-1}}$ that specifies n-gram probabilities

$$g_{w_i|w_{i-m+1}^{i-1}} = P(w_i|w_{i-m+1}^{i-1}) \tag{3.10}$$

where $0 \leq m \leq n-1$ is the length of the context being considered.

As we are not sure of the actual values of the n-gram probabilities due to data sparseness, the standard practice of Bayesian statistics suggests we treat all probabilities as random variables G that we can learn from the training data $\mathcal{W}$. Formally, this learning problem consists of estimating the posterior probability $P(G|\mathcal{W})$. This can be calculated in a Bayesian fashion by placing a prior probability $P(G)$ over G and combining this with the likelihood $P(\mathcal{W}|G)$ and the evidence $P(\mathcal{W})$

$$P(G|\mathcal{W}) = \frac{P(\mathcal{W}|G)P(G)}{P(\mathcal{W})} \tag{3.11}$$

$$\propto P(\mathcal{W}|G)P(G) \tag{3.12}$$

$$= \left(\prod_{W \in \mathcal{W}} P(W|G) \right) P(G). \tag{3.13}$$

We can generally ignore the evidence probability, as the training data is fixed throughout the entire training process, and we assume that each of the sentences in the corpus was generated independently from the probability distribution specified by G.

It should be noted that LMs are a collection of multinomial distributions $G_{w_{i-m+1}^{i-1}} = \{g_{w_i=1|w_{i-m+1}^{i-1}}, \ldots, g_{w_i=N|w_{i-m+1}^{i-1}}\}$ where N is the number of words in the vocabulary. There is one multinomial for each history w_{i-m+1}^{i-1},

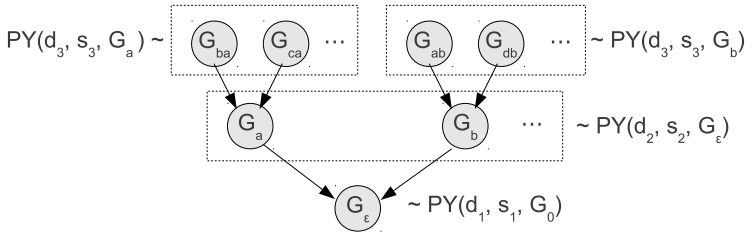

Figure 3.1: An example of the hierarchical structure of the HPYLM.

with the length of w_{i-m+1}^{i-1} being 0 through $n-1$. As the variables in $G_{w_{i-m+1}^{i-1}}$ belong to a multinomial distribution, it is natural to use priors based on the Pitman-Yor process described in Section 2.2.4.

As mentioned previously, the Pitman-Yor process has three parameters: the discount parameter d_m, the strength parameter s_m, and the base measure $G_{w_{i-m+2}^{i-1}}$

$$G_{w_{i-m+1}^{i-1}} \sim PY(d_m, s_m, G_{w_{i-m+2}^{i-1}}). \tag{3.14}$$

Here, the most important parameter is the base measure $G_{w_{i-m+2}^{i-1}}$, which indicates the expected value of the probability distribution generated by the process. In other words, it is essentially the "default" value used when there are no words in the training corpus for context w_{i-m+1}^{i-1}.

The important thing to note is that when creating an n-gram language model using the Pitman-Yor process, we set the base measure of each $G_{w_{i-m+1}^{i-1}}$ to be the distribution of its parent context $G_{w_{i-m+2}^{i-1}}$. This forms a hierarchical structure that is referred to as the hierarchical Pitman-Yor LM (HPYLM, (Teh, 2006)) and shown in Figure 3.1. This hierarchical structure implies that each set of m-gram (e.g., trigram) probabilities will be using its corresponding $(m-1)$-gram (e.g., bigram) probabilities as a starting point when no or little training data is available. As a result, we achieve a principled probabilistic interpolation of m-gram and $(m-1)$-gram smoothing similar to the heuristic methods described in Section 3.1.2. Finally, the base measure of the unigram model G_0 indicates the prior probability over words in the vocabulary. If we have a vocabulary of all the words that the HPYLM is expected to generate, we can simply set this so that a uniform probability is given to each word in the vocabulary.

For the Pitman-Yor language model, the actual probabilities can be calculated through Gibbs sampling and the Chinese Restaurant Process (CRP)

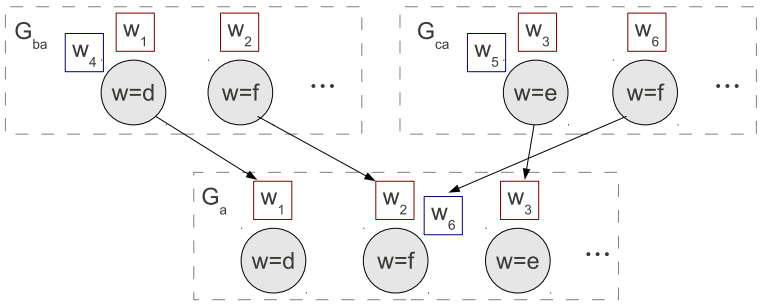

Figure 3.2: An example of the Chinese restaurant process for the HPYLM. Boxes are drawn around the tables for each individual distribution, and arrows are drawn between the tables of the 3-gram distribution and the corresponding customers in the 2-gram base measure.

formulation (Teh, 2006). The formulation of the CRP used in the Pitman-Yor language model is slightly different from that introduced in Section 2.2.3 in that we are now dealing with a hierarchical model with shared base measures that are also given Pitman-Yor priors.

An example of a segment of the previously shown hierarchical model and its corresponding table configurations for the 3-gram and 2-gram probabilities is shown in Figure 3.2. The most important point to notice here is that at the top of the tree, tables corresponding to 3-gram probabilities are given one customer for each word in the training corpus. However, when we look at the 2-gram probabilities it can be seen that the tables are not given one customer for each word, but instead one customer for each table in the 3-gram configurations. This property is beneficial for language modeling, as it is desirable for lower-order n-grams to simulate the distribution of words that occur only when higher-order n-grams are not present (Kneser and Ney, 1995; Chen and Goodman, 1996).

With regards to inference, it is also important to note that for each n-gram probability, it is possible to calculate the expectation of the probability given a set of sufficient statistics S

$$P(w_i|w_{i-n+1}^{i-1}, S) = \int_0^1 g_{w_i|w_{i-n+1}^{i-1}} P(g_{w_i|w_{i-n+1}^{i-1}}|S) dg_{w_i|w_{i-n+1}^{i-1}}. \tag{3.15}$$

The statistics S consist of the customer counts and table counts that summarize the configuration of the CRP. The practical implication of this is that we do not need to directly estimate the parameters G, but only need

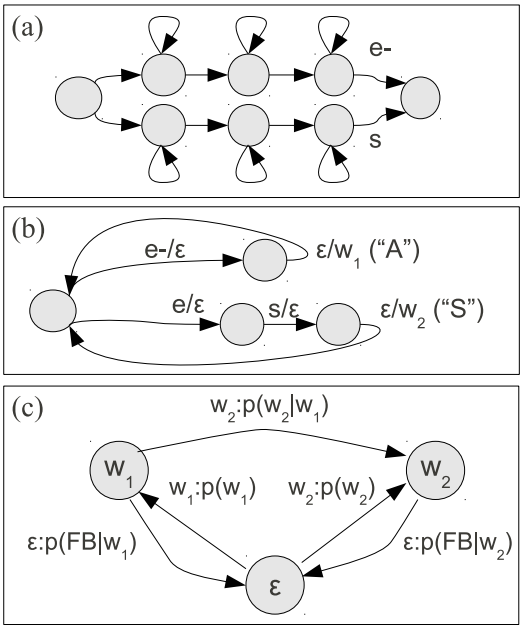

Figure 3.3: The WFSTs for ASR including (a) the acoustic model A, (b) the lexicon L, and (c) the language model G.

to keep track of the sufficient statistics needed to calculate this expectation of $P(w_i|w_{i-n+1}^{i-1}, S)$. This fact becomes useful when using this model in unsupervised learning, as described in later sections.

3.1.4 Weighted Finite State ASR

In recent years, the paradigm of weighted finite state transducers (WFSTs) has brought about great increases in the speed and flexibility of ASR systems (Mohri et al., 2008). Finite state transducers are finite automata with transitions labeled with input and output symbols. WFSTs also assign a weight to transitions, allowing for the definition of weighted relations between two strings. These weights can be used to represent probabilities of each model for ASR including the AM, lexicon, and the LM, examples of which are shown in Figure 3.3. In figures of the WFSTs, edges are labeled as "a/b:c", where a indicates the input, b indicates the output, and c indicates the weight. b may be omitted when a and b are the same value, and c will be omitted when it is equal to 1.

The standard AM for $P(U|X)$ in most ASR systems is based on a Hidden Markov Model (HMM), and its WFST representation, which will be called A. A simplified example of this model is shown in Figure 3.3 (a). As input, this takes acoustic features, and after several steps through the HMM outputs a single phoneme such as "e-" or "s." The transition and emission probabilities are identical to the standard HMM used in ASR acoustic models, but they are omitted from the figure for simplicity.

The WFST formulation for the lexicon L, shown in Figure 3.3 (b), takes phonemes as input and outputs words along with their corresponding lexicon probability $P(X|W)$. Excluding the case of homographs (words with the same spelling but different pronunciations), the probability of transitions in the lexicon will be 1.

Finally, the LM probability $P(W)$ can also be represented in the WFST format. Figure 3.3 (c) shows an example of a bigram LM with only two words w_1 and w_2 in the vocabulary. Each node represents a unique n-gram context w_{i-m+1}^{i-1}, and the outgoing edges from the node represent the probability of symbols given this context $P(w_i|w_{i-m+1}^{i-1})$. In order to handle the fallback to lower-order contexts as described in Section 3.1.2, edges that transition from w_{i-m+1}^{i-1} to w_{i-m+2}^{i-1} are added, weighted with the fallback probability (marked with "FB" in the figure). The label ϵ on these edges indicates the empty string, which means they can be followed at any time, regardless of the input symbol.

The main advantage of using WFSTs to describe the ASR problem is the existence of efficient algorithms for operations such as composition, intersection, determinization, and minimization. In particular, composition (designated with the $\circ$ operator) allows the combination of two WFSTs in sequence, so if we compose $A \circ L \circ G$ together, we can create a single WFST that takes acoustic features as input and outputs weighted strings of words entailed by the acoustic features. This property of WFSTs will be useful later to facilitate the implementation of learning of LMs from continuous speech.

3.2 Learning LMs from Unsegmented Text

While Sections 3.1.2 and 3.1.3 described how to learn LMs when we are given a corpus of word sequences $\mathcal{W}$, there are some cases when the word sequence is not obvious. For example, when human babies learn words they do so from continuous speech, even though there often are not explicit boundaries between words in the phoneme stream. In addition, as mentioned

previously, many languages such as Japanese, Chinese, and Thai are written without boundaries between words, and thus the definition of words is not uniquely fixed. This section describes formally and in detail a method for jointly learning lexical units and a language model from unsegmented text.

3.2.1 Unsupervised WS Modeling

This work follows (Mochihashi et al., 2009) in taking an LM-based approach to unsupervised WS, learning a word-based LM G from a corpus of unsegmented phoneme strings $\mathcal{X}$. The LM-based approach is attractive for the task of learning from speech, as LMs are an integral part of speech recognition systems, and the resulting LM can thus be plugged directly back in to ASR for use on text that is not included in our training data.

The problem of learning an LM from unsegmented phoneme strings can be specified as finding a model according to the posterior probability of the LM $P(G|\mathcal{X})$, which can be decomposed using Bayes's law

$$P(G|\mathcal{X}) \propto P(\mathcal{X}|G)P(G). \tag{3.16}$$

However, as G is a word-based LM, we also assume that there are hidden word sequences $\mathcal{W}$, and model the probability given these sequences

$$P(G|\mathcal{X}) \propto \sum_{\mathcal{W}} P(\mathcal{X}|\mathcal{W})P(\mathcal{W}|G)P(G). \tag{3.17}$$

Here, $P(\mathcal{X}|\mathcal{W})$ indicates that the words in $\mathcal{W}$ must correspond to the phonemes in $\mathcal{X}$, and will be 1 if and only if $\mathcal{X}$ can be recovered by concatenating the words in $\mathcal{W}$ together. $P(\mathcal{W}|G)$ is the likelihood given the LM probabilities, and is identical to that described in Equation (3.8).

$P(G)$ can be set using the previously described HPYLM, with one adjustment. With the model described in Section 3.1.3, it was necessary to know the full vocabulary in advance so that we could set the base measure G_0 to a uniform distribution over all the words in the vocabulary. However, when learning an LM from unsegmented text, $\mathcal{W}$ is not known in advance, and thus it is impossible to define a closed vocabulary before training starts. As a result, it is necessary to find an alternative method of defining G_0 that allows the model to flexibly decide which words to include in the vocabulary as training progresses.

In order to do so, (Mochihashi et al., 2009) use a "spelling model" H, which assigns prior probabilities over words by using an LM specified over phonemes. If we have a word w_i that consists of phonemes, $x_1, \ldots, x_J$, we

define the spelling model probability of w_i according to the n-gram probabilities of H:

$$G_0(w_i) = P(w_i = x_1, \ldots, x_J|H) = \prod_{j=1}^{J} h_{x_j|x_{j-n+1}^{j-1}} \tag{3.18}$$

We assume that H is also distributed according to the HPYLM, and that the set of phonemes is closed and thus we are able to define a uniform distribution over phonemes H_0. The probabilities of H can be calculated from the set of phoneme sequences of words generated from the spelling model, much like the probabilities of G can be calculated from the set of word sequences contained in the corpus.

This gives us a full generative model for the corpus $\mathcal{X}$ that first generates the LM probabilities

$$H \sim HPYLM(\boldsymbol{d}_H, \boldsymbol{s}_H, H_0) \tag{3.19}$$

$$G \sim HPYLM(\boldsymbol{d}_G, \boldsymbol{s}_G, P(w|H)) \tag{3.20}$$

then generates each word sequence $W \in \mathcal{W}$ and concatenates it into a phoneme sequence

$$W \sim P(W|G) \tag{3.21}$$

$$X \leftarrow \text{concat}(W). \tag{3.22}$$

This generative story is important in that it allows for the creation of LMs that are both highly expressive and compact (and thus have high generalization capacity). The HPYLM priors for H and G have a preference for simple models, and thus will tend to induce compact models, while the likelihoods for $\mathcal{W}$ bias towards larger and more expressive models that describe the data well. It should be noted that in contrast, if maximum likelihood estimation is used without a prior that biases against degenerate solutions, the model will have a tendency to maximize the likelihood by over-fitting the training data, memorizing each training sentence as a single word.

3.2.2 Inference for Unsupervised WS

The main difficulty in learning LM G from the phoneme string $\mathcal{X}$ is solving Equation (3.17). Here, it is necessary to sum over all possible configurations of $\mathcal{W}$, which represent all possible segmentations of $\mathcal{X}$. However, for all but the smallest of corpora, the number of possible segmentations is

Input: Unsegmented corpus $\mathcal{X}$
Output: Word segmented corpus $\mathcal{W}$
  for all Iterations i in $\{1, \ldots, I\}$ **do**
    for all Sentence k in $\{1, \ldots, |\mathcal{X}|\}$ **do**
      if $i \neq 1$ **then**
        Remove sufficient statistics obtained from W_k from S
      end if
      Sample a new value of W_k from $P(W_k|X_k, S \backslash W_k)$
      Add the sufficient statistics of the new W_k back to S
    end for
    Save a sample S_i and $\mathcal{W}_i$
  end for

Figure 3.4: The algorithm for Gibbs sampling of the word corpus $\mathcal{W}$ and the sufficient statistics S necessary for calculating LM probabilities.

astronomical and thus it is impractical to explicitly enumerate all possible $\mathcal{W}$.

Instead, we can turn to Gibbs sampling, as described in detail in Section 2.3. As we are interested in calculating $\mathcal{W}$, for each step of the algorithm we take a single sentence $W_k \in \mathcal{W}$ and sample it according to a distribution $P(W_k|X_k, S \backslash W_k)$. S indicates the sufficient statistics calculated from the current configuration of $\mathcal{W}$ required to calculate language model probabilities (as described in Section 3.1.3). $S \backslash W_k$ indicates the sufficient statistics after subtracting the n-gram counts and corresponding CRP configurations that were obtained from the sentence W_k.[1] These sufficient statistics allow us to calculate the conditional probability of W_k given all other sentences, a requirement to properly perform Gibbs sampling. It should be noted that each W_k contains multiple variables (words), so this is in fact a variant of blocked Gibbs sampling. The full sampling procedure is shown in Figure 3.4, and the following section further details how a single sentence W_k can be sampled according to this distribution.

By repeating Gibbs sampling for many iterations, the sampled values of each sentence W_k, and the LM sufficient statistics S calculated therefrom, will gradually approach the high-probability areas specified by the model. As mentioned previously, the HPYLM-based formulation prefers highly expressive, compact models. Lexicons that contain many words are penalized

[1] On the first iteration, we start with an empty S, and gradually add the statistics for each sentence as they are sampled.

by the HPYLM prior, preventing segmentations of $\mathcal{W}$ that result in a large number of unique words. On the other hand, if the lexicon is too small, it will result in low descriptive power. Thus the sampled values are expected to be those with a consistent segmentation for words, and with common phoneme sequences grouped together as single words.

3.2.3 Calculating Predictive Probabilities

As the main objective of an LM is to assign a probability to an unseen phoneme string X, we are interested in calculating the predictive distribution

$$P(X|\mathcal{X}) = \int_G \sum_{W \in \{\tilde{W}:\text{concat}(\tilde{W})=X\}} P(W|G)P(G|\mathcal{X})dG. \tag{3.23}$$

However, computing this function directly is computationally difficult. To reduce this computational load we can approximate the summation over W with the maximization, assuming that the probability of X is equal to that of its most likely segmentation.

In addition, assume we have I effective samples of the sufficient statistics obtained after iterations of the previous sampling process.[2] Using these samples, we can approximate the integral over G with the mean of the probabilities given the sufficient statistics $\{S_1, \ldots, S_I\}$

$$P(X|\mathcal{X}) \approx \frac{1}{I} \sum_{i=1}^{I} \max_{W \in \{\tilde{W}:\text{concat}(\tilde{W})=X\}} P(W|S_i). \tag{3.24}$$

While Equation (3.24) approximates the probability using the average maximum-segmentation probability of each S_i, search for such a solution at decoding time is a non-trivial problem. As an approximation to this sum, we find the one-best solution mandated by each of the samples, and combine the separate solutions using ROVER (Fiscus, 1997).

3.3 WFST-based Sampling of Word Sequences

While the previous section described the general flow of the inference process, we still require an effective method to sample the word sequence W according to the probability $P(W|X, S\backslash W)$. One way to do so would

[2] Some samples may be skipped during the early stages of sampling (a process called "burn-in") to help ensure that samples are likely according to the HPYLM.

be to explicitly enumerate all possible segmentations for X, calculate their probabilities, and sample based on these probabilities. However, as the number of possible segmentations of X grows exponentially in the length of the sentence, this is an unrealistic solution. Thus, the most difficult challenge of the algorithm in Figure 3.4 is efficiently obtaining a word sequence W given a phoneme sequence X according to the language model probabilities specified by $S\backslash W$.

One solution is proposed by (Mochihashi et al., 2009), who use a dynamic programming algorithm that allows for efficient sampling of a value for W according to the probability $P(W|X, S\backslash W)$. While this method is applicable to unsegmented text strings, it is not applicable to situations where uncertainty exists in the input, such as the case of learning from speech. This section proposes an alternative formulation that uses the WFST framework. This is done by first creating a WFST-based formulation of the WS model (Section 3.3.1), then describing a dynamic programming method for sampling over WFSTs (Section 3.3.2). This formulation is critical for learning from continuous speech, as it allows for sampling a word string W from not only one-best phoneme strings, but also phoneme lattices that are able to encode the uncertainty inherent in acoustic matching results.

3.3.1 A WFST Formulation for Word Segmentation

Our formulation for sampling word sequences consists of first generating a lattice of all possible segmentation candidates using WFSTs, then performing sampling over this lattice. The three WFSTs used for WS (Figure 3.5) are quite similar to the ASR WFSTs shown in Figure 3.3.

In place of the acoustic model WFST used in ASR, we simply use a linear chain representing the phonemes in X, as shown in Figure 3.5 (a). The lexicon WFST L in Figure 3.5 (b) is identical to the lexicon WFST used in ASR, except that in addition to creating words from phonemes, it also allows all phonemes in the input to be passed through as-is. This allows words in the lexicon to be assigned word-based probabilities according to the language model G, and all words (in the lexicon or not) to be assigned probabilities according to the spelling model H. This is important in the unsupervised WS setting, where the lexicon is not defined in advance, and words outside of the lexicon are still assigned a small probability.

The training process starts with an empty lexicon, and thus no paths emitting words are present. When a word that is not in the lexicon is sampled as a phoneme sequence, L is modified by adding a path that converts the new word's phonemes into its corresponding word token. Conversely,

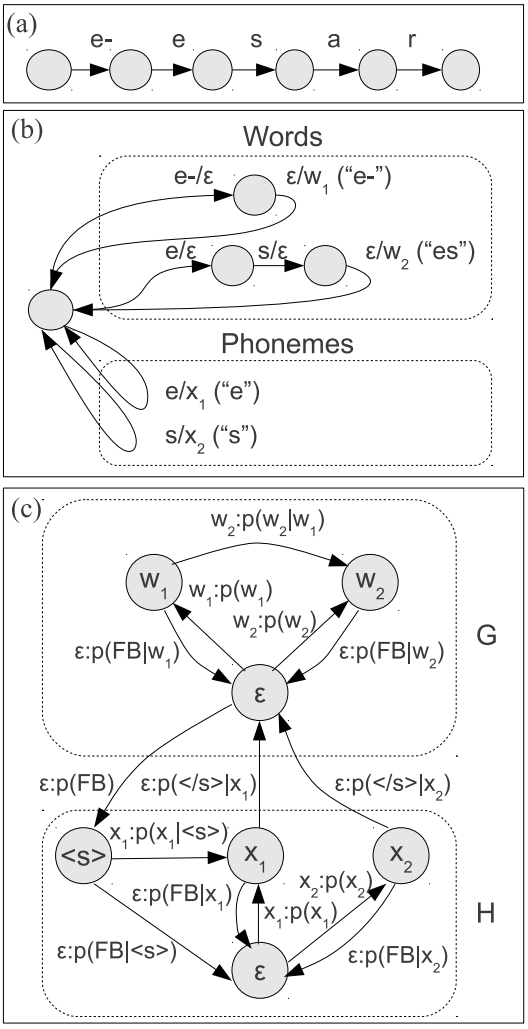

Figure 3.5: The WFSTs for word segmentation including (a) the input X, (b) the lexicon L, and (c) the language model GH.

when the last sample containing a word in the lexicon is subtracted from the distribution and the word's count becomes zero, its corresponding path is removed from L. It should be noted that here we are making the assumption that each word can be mapped onto a single spelling, so $P(X|W)$ will always be 1.[3]

More major changes are made to the LM WFST, which is shown in Figure 3.5 (c). Unlike the case in ASR, where we are generally only concerned with words that exist in the vocabulary, it is necessary to model unknown words that are not included in the vocabulary. The key to the representation is that the word-based LM G and the phoneme-based spelling model H are represented in a single WFST, which we will call GH. GH has weighted edges falling back from the base state of G to H, and edges accepting the terminal symbol for unknown words and transitioning from H to the base state of G. This allows for the WFST to transition as necessary between the known word model and the spelling model.

By composing together these three WFSTs as $X \circ L \circ GH$, it is possible to create a WFST representing a lattice of segmentation candidates weighted with probabilities according to the LM.

3.3.2 Sampling over WFSTs

Once we have a WFST lattice representing the model probabilities, we can sample a single path through the WFST according to the probabilities assigned to each edge. This is done using *forward-filtering/backward-sampling*, a technique similar to that of the forward-backward algorithm for hidden Markov models (HMM). This algorithm can be used to sample a single path from all probabilistically weighted, acyclic WFSTs defined by a set of states S and a set of edges E.

The first step of the algorithm consists of choosing an ordering for the states in S, which will be written $s_1, \ldots, s_I$. This ordering must be chosen so that all states included in paths that travel to state s_i should be processed before s_i itself. Each edge in E is defined as $e_k = \langle s_i, s_j, w_k \rangle$ traveling from s_i to s_j and weighted by w_k. Assuming the graph is acyclic, we can choose the ordering so that for all edges in E, $i < j$. Given this ordering, if all states are processed in ascending order, we can be ensured that all states will be processed after their predecessors.

Next, we perform the forward filtering step, identical to the forward

[3]This work assumes that all words are represented by their phonetic spelling, not considering the graphemic representation used in usual text. For example, the word "ASR" will be transcribed as "e-esar" in the learned model.

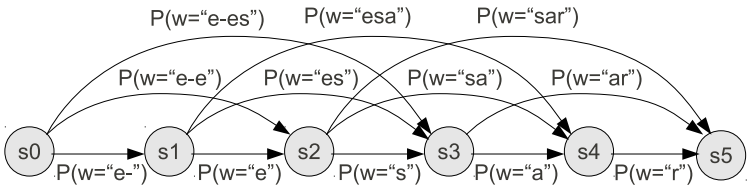

Figure 3.6: A WFSA representing a unigram segmentation (words of length greater than three are not displayed).

pass of the forward-backward algorithm for HMMs, where probabilities are accumulated from the start state to following states. The initial state s_0 is given a forward probability $f_0 = 1$, and all following states are updated with the sum of the forward probabilities of each of the incoming states multiplied by the weights of the edges to the current state

$$f_j = \sum_{e_k=\langle s_i, s_{\tilde{j}}, w_k\rangle \in \{E:\tilde{j}=j\}} f_i w_k. \tag{3.25}$$

This forward probability can be interpreted as the total probability of all paths that travel to f_j from the initial state.

Figure 3.6 provides an example of this process using a weighted finite state acceptor (WFSA) for the unigram segmentation model of "e- e s a r" ("ASR"). In this case, the forward step will push probabilities from the first state as follows:

$$f_1 = P(w = \text{"e-"}) f_0 \tag{3.26}$$

$$f_2 = P(w = \text{"e-e"}) f_0 + P(w = \text{"e"}) f_1 \tag{3.27}$$

$$\vdots$$

The backward sampling step of the algorithm consists of sampling a path starting at the final state s_I of the WFST. For the current state, s_j, we can calculate the probability of all incoming edges

$$P(e_k = \langle s_i, s_j, w_k\rangle) = \frac{f_i w_k}{f_j}, \tag{3.28}$$

and sample a single incoming edge according to this probability. Here w_k considers the likelihood of e_k itself, while f_i considers the likelihood of all paths traveling up to s_i, allowing for the correct sampling of an edge e_k

according to the probability of all paths that travel through it to the current state s_j. In the example, the edge incoming to state s_5 is sampled according to

$$P(s_4 \rightarrow s_5) = P(w = \text{“r”}) f_4 \tag{3.29}$$

$$P(s_3 \rightarrow s_5) = P(w = \text{“ar”}) f_3 \tag{3.30}$$

$$\vdots$$

Through this process, a path representing the segmentation of the phoneme string can be sampled according to the probability of the models included in the lattice. Given this path, it is possible to recover X and W by concatenating the phonemes and words represented by the input and output of the sampled path respectively.

3.3.3 Extension to Continuous Speech Input

When learning from continuous speech, the input is not a set of phoneme strings $\mathcal{X}$, but a set of spoken utterances $\mathcal{U}$. As a result, instead of sampling just the word sequences $\mathcal{W}$, we now need to additionally sample the phoneme strings $\mathcal{X}$. If we can create a single lattice representing the probability of both W and X for a particular U, it is possible to use the forward-filtering/backward-sampling algorithm to sample phoneme strings and their segmentations together.

With the WFST-based formulation described in the previous section, it is straight-forward to create this lattice representing candidates for X and W. In fact, all we must do is replace the string of phonemes X that was used in the WS model in Figure 3.5 (a) with the acoustic model HMM A used for ASR in Figure 3.3. As a result, the composed lattice $A \circ L \circ GH$ can take acoustic features as input, and includes both the acoustic and language model probabilities. Using this value, we can sample appropriate new values of X and W, and plug this into the learning algorithm of Figure 3.4.

However, as with traditional ASR, if we simply expand all hypotheses allowed by the acoustic model during the forward-filtering step, the hypothesis space will grow unmanageably large. As an approximation to the full expansion of the search space, before starting training we first perform ASR using only the acoustic model and no linguistic information, generating trimmed phoneme lattices representing candidates for each X such as those shown in Figure 3.7.

It should be noted that this dependence on an acoustic model to estimate $P(U|X)$ indicates that this is not an entirely unsupervised method. How-

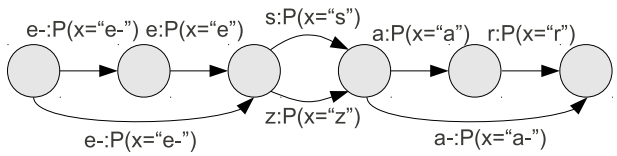

Figure 3.7: A WFSA representing a phoneme lattice, with conditioning on acoustic features U omitted for simplicity.

ever, some work has been done on language-independent acoustic model training (Lamel et al., 2002), as well as the unsupervised discovery and clustering of acoustic units from raw speech (Glass, 1988). The proposed LM acquisition method could be used in combination with these acoustic model acquisition methods to achieve fully unsupervised speech recognition, a challenge that may be tackled in future work.

3.4 Experimental Evaluation

This section evaluates the feasibility of the proposed method on continuous speech from meetings of the Japanese Diet (Parliament). This was chosen as an example of naturally spoken, interactive, adult-directed speech with a potentially large vocabulary, as opposed to the simplified grammars or infant-directed speech used in some previous work (Iwahashi, 2003; Roy and Pentland, 2002).

3.4.1 Experimental Setup

Phoneme lattices were created using a triphone acoustic model, performing decoding with a vocabulary of 385 syllables that represent the phoneme transitions allowed by the syllable model.[4] No additional linguistic information was used during the creation of the lattices, with all syllables in the vocabulary being given a uniform probability.

In order to assess the amount of data needed to effectively learn an LM, experiments were performed using five different corpora of varying sizes: 7.9, 16.1, 31.1, 58.7, and 116.7 minutes. The speech was separated into utterances, with utterance boundaries being delimited by short pauses of 200ms or longer. According to this criterion, the training data consisted of

[4]Syllable-based decoding was a practical consideration due to the limits of the decoding process, and is not a fundamental part of the proposed method.

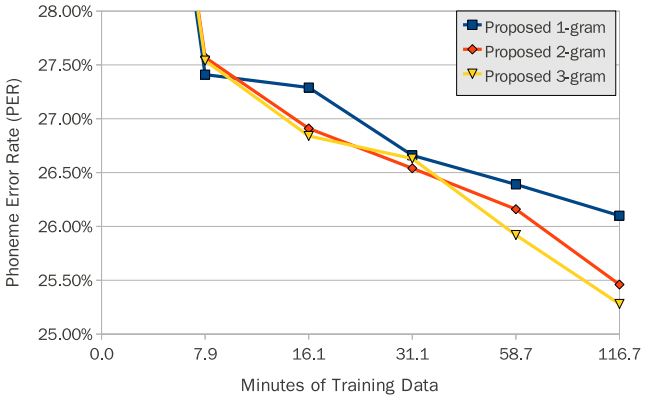

Figure 3.8: Phoneme error rate by model order.

119, 238, 476, 952, and 1,904 utterances respectively. An additional 27.2 minutes (500 utterances) of speech were held out as a test set.

As a measure of the quality of the LM learned by the training process, we adopt phoneme error rate (PER) when the LM was used to rescore the phoneme lattices of the test set. PER is an appropriate measure as word-based accuracy may depend heavily on a particular segmentation standard. Given no linguistic information, the PER on the test set was 34.20%. The oracle PER of the phoneme lattice was 8.10%, indicating the lower bound possibly obtainable by LM learning.

Fifty samples of the word sequences $\mathcal{W}$ for each training utterance (and the resulting sufficient statistics S) were taken after 20 iterations of burn-in, the first 10 of which were annealed according to the technique presented by (Goldwater et al., 2009). For the LM scaling factor of Equation (3.6), α was set arbitrarily to 5, with values between 5 and 10 producing similar results in preliminary tests.

3.4.2 Effect of n-gram Context Dependency

In the first experiment, the effect of using context information in the learning process was examined. The n of the HPYLM language model was set to 1, 2, or 3, and n of the HPYLM spelling model was set to 3 for all models. The results with regards to PER are shown in Figure 3.8.

First, it can be seen that an LM learned directly from speech was able to improve the accuracy by 7% absolute PER or more compared to a baseline

Table 3.1: The size of the vocabulary, and the number of n-grams in the word-based model G, and the phoneme-based model H when trained on 116.7 minutes of speech.

	1-gram	2-gram	3-gram
Vocabulary size	4480	1351	708
G entries	4480	16150	38759
H entries	9624	3869	2426

using no linguistic information. This is true even with only 7.9 minutes of training speech. In addition, the results show that the bigram model outperforms the unigram, and the trigram model outperforms the bigram, particularly as the size of the training data increases. The experiments were also able to confirm the observation of (Goldwater et al., 2009) that the unigram model tends to undersegment, grouping together "multi-word" phrases instead of actual words. This is reflected in the vocabulary and n-gram sizes of the three models after the final iteration of the learning process, which are displayed in Table 3.1. It can also be seen that the vocabulary size increases when the LM is given a smaller n, with the lack of complexity in the word-based LM being transferred to the phoneme-based spelling model.

3.4.3 Effect of Joint and Bayesian Estimation

The proposed method has two major differences from previous methods such as (Driesen and Hamme, 2008), which estimates multigram models from speech lattices. The first is that we are performing joint learning of the lexicon and n-gram context, while multigram models do not consider context, similarly to the 1-gram model presented in this chapter (Bimbot et al., 1995). However, it is conceivable that a context insensitive model could be used for learning lexical units, and its results used to build a traditional LM. In order to test the effect of context-sensitive learning, experiments are performed with not only the proposed 1-gram and 3-gram models from Section 3.4.2, but also using the 1-gram model to acquire samples of $\mathcal{W}$ and using these to train a standard 3-gram LM.

The second major difference is that we are performing learning using Bayesian methods. This allows us to consider the uncertainty of the acquired W through the sum in Equation (3.24). Previous multigram approaches are based on maximum likelihood estimation, which only allows for a unique

Table 3.2: The effects on accuracy of the n-gram length used to acquire the lexicon and train the language model, as well as whether a single sample is used or multiple samples are combined. The proposed method significantly exceeds italicized results according to the two-proportions z-test ($p < 0.05$).

Lexicon	LM	Single	Combined
1-gram	1-gram	*26.28%*	*26.08%*
1-gram	3-gram	*26.06%*	25.41%
3-gram	3-gram	25.85%	**25.28%**

solution to be considered. To test the effect of this, we take the one-best results acquired by the sampled LMs, but instead of combining them together to create a better result as explained in Section 3.2.3, simply report the average PER of these one-best results.

Table 3.2 shows the results of the evaluation (performed on the 116.7 minute training data). It can be seen that the proposed method using Bayesian sample combination and incorporating LMs directly into training (3-gram/3-gram/combined) is effective in reducing the error rate compared to a model that does not use these proposed improvements (1-gram/3-gram/single).

3.4.4 Effect of Lattice Processing

This section compares the proposed lattice processing method with four other LM construction methods. The first baseline trains a model using the proposed method, but instead of using word lattices, used one-best ASR results to provide a comparison with previous methods that have used one-best results (de Marcken, 1995; Gorin et al., 1999). Second, to examine whether the estimation of word boundaries is necessary when acquiring an LM from speech, results using a syllable trigram LM trained on these one-best results are also shown. Two other performance results are also shown for reference. One is an LM that was built using a human-created verbatim transcription of the utterances. WS and pronunciation annotation were performed with the KyTea toolkit (Neubig and Mori, 2010), and pronunciations of unknown words were annotated by hand. Trigram language and spelling models were created on the segmented word and phoneme strings using interpolated Kneser-Ney smoothing. The second reference is an "oracle" model created by training on the lattice path with the lowest possible PER for each utterance. This demonstrates an upper bound of the accuracy

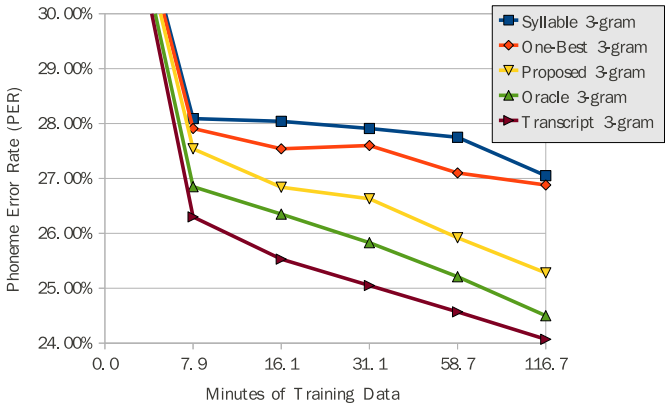

Figure 3.9: Phoneme error rate for various training methods.

achievable by the proposed model if it picks all the best phoneme sequences in the training lattice.

The PER for the four methods is shown in Figure 3.9. It can be seen that the proposed method outperforms the model trained on one-best results, demonstrating that lattice processing is critical in reducing the noise inherent in acoustic matching results. It can also be seen that on one-best results, the model using acquired units achieves slightly but consistently better results than the syllable-based LM for all data sizes.

As might be expected, the proposed method does not perform as well as the model trained on gold-standard transcriptions. However, it appears to improve at approximately the same rate as the model trained on the gold-standard transcriptions as more data is added, which is not true for one-best transcriptions. Furthermore, it can be seen that the oracle results fall directly between those achieved by the proposed model and the results on the gold-standard transcriptions. This indicates that approximately one half of the difference between the model learned on continuous speech and that learned from transcripts can be attributed to the lattice error. By expanding the size of the lattice, or directly integrating the calculation of acoustic scores with sampling, it will likely be possible to further close this gap.

Another measure commonly used for evaluating the effectiveness of LMs is cross-entropy on a test set (Goodman, 2001). Entropy per syllable for the LMs learned with each method is shown in Figure 3.10. It can be seen that the proposed method only slightly outperforms the model trained on

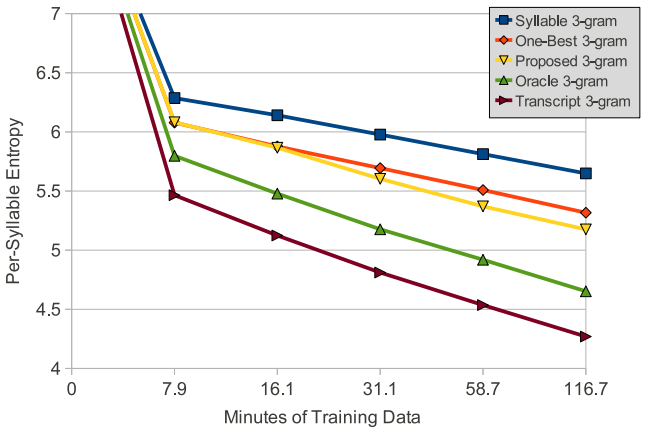

Figure 3.10: Entropy comparison for various LM learning methods.

Table 3.3: An example of words learned from continuous speech.

Function Words	*no* (genitive marker), *ni* (locative marker), *to* (“and”)
Subwords	*ka* (*kyoka* “reinforcement”, interrogative marker) *sai* (*kokusai* “international”, *seisai* “sanction”)
Content Words	*koto* (“thing”), *hanashi* (“speak”), *kangae* (“idea”), *chi-ki* (“region”), *shiteki* (“point out”)
Spoken Expressions	*yu-* (“say (colloquial)”), *e-* (filler), *desune* (filler), *mo-shiage* (“say (polite)”)

one-best phoneme recognition results. This difference can be explained by systematic pronunciation variants that are not accounted for in the verbatim transcript. For example, *kangaeteorimasu* (“I am thinking”) is often pronounced with a dropped *e* as *kangaetorimasu* in fluent conversation. As a whole word will fail to match the reference, this will have a large effect on entropy results, but less of an effect on PER as only a single phoneme was dropped. In fact, for many applications such as speech analysis or data preparation for acoustic model training, the proposed method, which managed to properly learn pronunciation variants, is preferable to one that matches the transcript correctly.

3.4.5 Lexical Acquisition Results

Finally, this section presents a qualitative evaluation of the lexical acquisition results. Typical examples of the words that were acquired in the process of LM learning are shown in Table 3.3. These are split into four categories: function words, subwords, content words, spoken language expressions.

In the resulting vocabulary, function words were the most common of the acquired words, which is reasonable as function words make the majority of the actual spoken utterances. Subwords are the second most frequent category, and generally occur when less frequent content words share a common stem.

An example of the content words discovered by the learning method shows a trend towards the content of discussions made in meetings of the Diet. In particular, *chi-ki* ("region") and *shiteki* ("point out") are good examples of words that are characteristic of Diet speech and acquired by the proposed model. While this result is not surprising, it is significant in that it shows that the proposed method is able to acquire words that match the content of the utterances on which it was trained. In addition to learning the content of the utterances, the proposed model also learned a number of stylistic characteristics of the speech in the form of fillers and colloquial expressions. This is also significant in that these expressions are not included in the official verbatim records in the Diet archives, and thus would not be included in an LM that was simply trained on these texts.

3.5 Conclusion

This chapter presented a method for unsupervised learning of an LM given only speech and an acoustic model. Specifically, a Bayesian model for word segmentation and LM learning was adapted so that it could be applied to speech input. This was achieved by formulating all elements of LM learning as WFSTs, which allows for lattices to be used as input to the learning algorithm, and then formulating a Gibbs sampling algorithm that allows for learning over composed lattices that represent acoustic and LM probabilities.

An experimental evaluation showed that LMs acquired from continuous speech with no accompanying transcriptions were able to significantly reduce the error rates of ASR over when no such models were used. It also showed that the proposed technique of joint Bayesian learning of lexical units and an LM over lattices significantly contributes to this improvement.

This work contributes a basic technology that opens up a number of possible directions for future research into practical applications. The first and most immediate application of the proposed method would be for use in semi-supervised learning. In the semi-supervised setting, we have some text already available, but want to discover words from untranscribed speech that may be in new domains, speaking styles, or dialects. This can be formulated in the proposed model by treating the phoneme sequences X (and possibly word boundaries W) of existing text as observed variables and the X and W of untranscribed speech as hidden variables. In addition, if it is possible to create word dictionaries but not a training corpus, these dictionaries could be used as a complement or replacement to the spelling model, allowing the proposed method to favor words that occur in the dictionary.

The combination of the proposed model with information from modalities other than speech is another promising future direction. For example, while the model currently learns words as phoneme strings, it is important to learn the orthographic forms of words for practical use in ASR. One possibility is that speech could be grounded in text data such as television subtitles to learn these orthographic forms. In order to realize this in the proposed model, an additional FST layer that maps between phonetic transcriptions and their orthographic forms could be introduced to allow for a single phonetic word to be mapped into multiple orthographic words and vice-versa.

In addition, the proposed method could be used to discover a lexicon and LM for under-resourced languages with little or no written text. In order to do so, it will be necessary to train not only an LM, but also an acoustic model that is able to recognize the phonemes or tones in the target language. One promising approach is to combine the proposed method with cross-language acoustic model adaptation, an active area of research that allows for acoustic models trained in more resource-rich languages to be adapted to resource-poor languages (Schultz and Waibel, 2001; Lamel et al., 2002).

The proposed method is also of interest in the framework of computational modeling of lexical acquisition by children. In its current form, which performs multiple passes over the entirety of the data, the proposed model is less cognitively plausible than previous methods that have focused on incremental learning (Pearl et al., 2010; McInnes and Goldwater, 2011; Räsänen, 2011)[5] However, work by (Pearl et al., 2010) has demonstrated that similar

[5]On the other hand, phonemic acquisition is generally considered to occur in the early stages of infancy, prior to lexical acquisition (Eimas et al., 1971; Roy and Pentland, 2002), and thus our reliance on a pre-trained acoustic model is largely plausible.

Bayesian methods (which were evaluated on raw text, not acoustic input) can be adapted to an incremental learning framework. This sort of incremental learning algorithm is compatible with the proposed method as well, and may be combined to form a more cognitively plausible model.

Chapter 4

Phrase Alignment for Statistical Machine Translation

Statistical machine translation (SMT) has seen great improvements over the past decade thanks largely to the introduction of phrase-based translation, which helps resolve lexical ambiguity and short-distance reordering by translating multi-word phrases as single chunks. The most important element of phrase-based SMT systems is the "phrase table," a list of bilingual phrase pairs that are translations of each other, each annotated with feature functions indicating certain properties of the phrase. This phrase table is generated from a parallel corpus of translated sentences that are aligned at the sentence level, but not at the word or phrase level.

Traditional systems construct phrase tables by going through a two-step pipeline. The first step consists of finding alignments between words or minimal phrases in both sentences, while the second step extracts an expanded phrase table from these alignments through heuristic combination of words or minimal phrases into longer units. The ability to use both short single-word units and longer phrases is one of the major reasons why phrase-based translation achieves superior results to word-based methods. However, it has been shown in previous research (DeNero and Klein, 2010) that this two step approach results in word alignments that are not optimal for the final task of generating phrase tables that are used in translation. In addition, exhaustively extracted phrase tables are often unnecessarily large, which results in an increase in the amount of time and memory required to run machine translation systems.

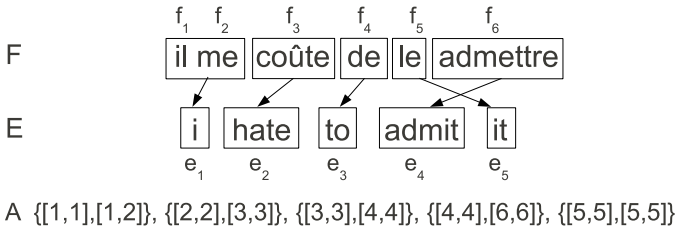

Figure 4.1: The target sentence F, source sentence E, and alignment A.

This chapter proposes an approach that is able to reduce the two steps of alignment and extraction into a single step by including phrases of multiple granularities in a probabilistic alignment model. The model is based on inversion transduction grammars (ITGs (Wu, 1997)), a variety of synchronous context-free grammars (SCFGs). ITGs allow for efficient word or phrase alignment (Cherry and Lin, 2007; Zhang et al., 2008a; Blunsom et al., 2009) through the use of bilingual chart parsing, similar to parsing algorithms used widely for the parsing of monolingual CFGs.

In contrast to previous approaches, which generally only attempt to model word (or minimal phrase) alignments, the proposed method models phrases at multiple levels of granularity through a novel recursive formulation, where larger phrase pairs are probabilistically constructed from two smaller phrase pairs. The model uses methods from non-parametric Bayesian statistics, which favor simpler models, preventing the over-fitting that occurs in some previous alignment approaches (Marcu and Wong, 2002).

An evaluation of this model is performed using machine translation experiments over four language pairs. The experiments demonstrate that the proposed hierarchical model is able to meet or exceed results attained by the traditional combination of word alignment and heuristic phrase extraction with a significantly smaller phrase table size. They also show that in contrast, previously proposed ITG-based phrase alignment approaches are not able to achieve competitive accuracy without heuristic phrase extraction and the accompanying increase in phrase table size.

4.1 Phrase-Based Statistical Machine Translation

Machine translation is the process of translating text in a source language into text in a target language. A source language sentence is represented as F, and the equivalent sentence in the target language is represented as E.

For now, we will assume that each of these sentences is separated into words, deferring a discussion of automatic acquisition of lexical units to the next chapter. Many modern machine translation (MT) systems utilize phrase-based MT (Koehn et al., 2003) techniques, which break F into phrases of one or more words, each of which is individually translated and reordered to form E. An example of a phrase-based translation is shown in Figure 4.1. It should be noted that in addition to F and E, there is a string A of alignment spans that indicates which parts of F were translated into which parts of E. Each element of A takes the form $\{[s,t],[u,v]\}$ indicating a single pair of phrases in the source and target sentences. The variables s and t indicate the position of the first and last words of the target phrase, while u and v indicate the position of the first and last words of the source phrase, respectively.

For any particular source sentence F there are many possible translations, some more natural or semantically correct than others. Statistical machine translation (SMT) attempts to resolve this ambiguity by creating a statistical model for the target sentence and alignment given the source sentence, and finding the target sentence that maximizes this probability:

$$\hat{E} = \underset{E}{\text{argmax}} P(E, A|F). \tag{4.1}$$

The predominant paradigm for calculating this probability is the log-linear model of (Och and Ney, 2002). This model defines the logarithm of the translation probability as a linear combination of a set of feature functions $\phi_1, \ldots, \phi_I$ over E, F, and A, weighted with weights $\lambda_1, \ldots, \lambda_I$

$$\log P(E, A|F) = \sum_{i=1}^{I} \lambda_i \phi_i(E, F, A). \tag{4.2}$$

This formulation allows arbitrary features of E, F, and A to be used in determining the translation probabilities. Commonly used feature functions include log language model probabilities, which attempt to capture the fluency of E, or reordering probabilities over A, which attempt to ensure that the word order in the target language is appropriate.

However, the features that most directly affect the translation quality are those that belong to the *phrase table*. As shown in the example in Figure 4.2, the phrase table is a collection of phrase pairs, consisting of equivalent source and target language phrases ($\boldsymbol{f}$ and $\boldsymbol{e}$ respectively). Each phrase pair is additionally scored with several feature functions, which will be explained in more detail in Section 4.5. These feature functions are used to provide

$\boldsymbol{e}$	$\boldsymbol{f}$	$\phi_1(\boldsymbol{e},\boldsymbol{f})$	$\phi_2(\boldsymbol{e},\boldsymbol{f})$	
admit	admettre	0.5	0.3	...
admit	avouer	0.5	0.8	
admit it	le admettre	1.0	0.28	
...				

Figure 4.2: An example of part of the phrase table with source phrases $\boldsymbol{e}$, target phrases $\boldsymbol{f}$, and feature functions ϕ_i.

an indication of the reliability or frequency of each phrase pair, and can be learned from a corpus consisting of translated pairs of sentences in the source and target languages.

4.2 Inversion Transduction Grammars (ITGs)

The first step in creating a phrase table from a sentence-aligned parallel corpus is *alignment*, the process of finding which words or phrases in the source and target sides of the training data correspond to each other. Following the definitions presented in the previous section, this means that we are given a parallel training corpus consisting of $\mathcal{F} = F_1, \ldots, F_n$ and $\mathcal{E} = E_1, \ldots, E_n$, and we must find the corresponding alignments $\mathcal{A} = A_1, \ldots, A_n$. One framework for learning these alignments that has been used in a number of recent works (Cherry and Lin, 2007; Zhang et al., 2008a; Blunsom et al., 2009) is the inversion transduction grammar (ITG) (Wu, 1997).

4.2.1 ITG Structure and Alignment

ITGs are generative models that were designed to simultaneously describe the generative process of equivalent strings of tokens $\boldsymbol{e}$ and $\boldsymbol{f}$ in two different languages. They are a limited form of synchronous context-free grammar (SCFG) in Chomsky normal form (Chomsky, 1956), where "synchronous" indicates that the grammar is defined over two languages instead of one. Figure 4.3 (a) shows an example of the ITG derivation that has generated two phrases "to admit it" and "de le admettre" in English and French, which we will use to demonstrate how ITGs work. The ITG describes how these two equivalent sentences were created through a recursive process that passes through two phases.

The first phase consists of generating the sentence structure, which in the case of ITGs is particularly important for specifying the reordering that

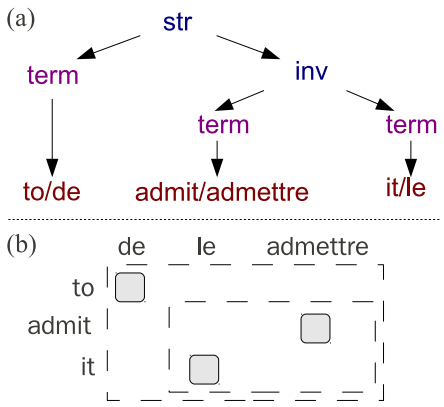

Figure 4.3: An example of (a) an inversion transduction grammar (ITG) derivation tree, (b) its corresponding alignment matrix.

occurs between the sentences in the two languages. It can be seen that from the reordering matrix in Figure 4.3 (b) that for some phrase pairs the word order is the same in both languages ("to" precedes "admit it," and "de" precedes "le admettre"). On the other hand, there are also some places where the order is inverted ("admit" precedes "it" while "admettre" succeeds "le"). ITGs represent this reordering structure as a binary tree, with each internal node labeled as *straight* (str) or *inverted* (inv), where each of these node types represents the case where the order is the same or inverted in both languages, respectively.[1] Much like standard CFGs, each leaf node is labeled with the *pre-terminal* (term) to indicate that we have finished the first step of generating the sentence structure.

This method of expressing reordering through an underlying derivation is both intuitive and flexible. Figures 4.4 (a-d) demonstrate a number of ITG derivations and their corresponding alignments. On the other hand, it should be noted that there are some patterns that are not able to be expressed in this framework. Some examples of these include the *pinwheel* pattern shown in Figure 4.4 (e), and patterns where a single word in one language is aligned to multiple discontiguous words in the other language

[1]Here we are specifically referring to a special case of ITGs with only a single symbol each for straight and inverted productions, which is also known as the *bracketing* ITG. ITGs with multiple straight and inverted terminals are also conceivable, but are generally not used in alignment as they significantly increase the computational burden of learning the ITG.

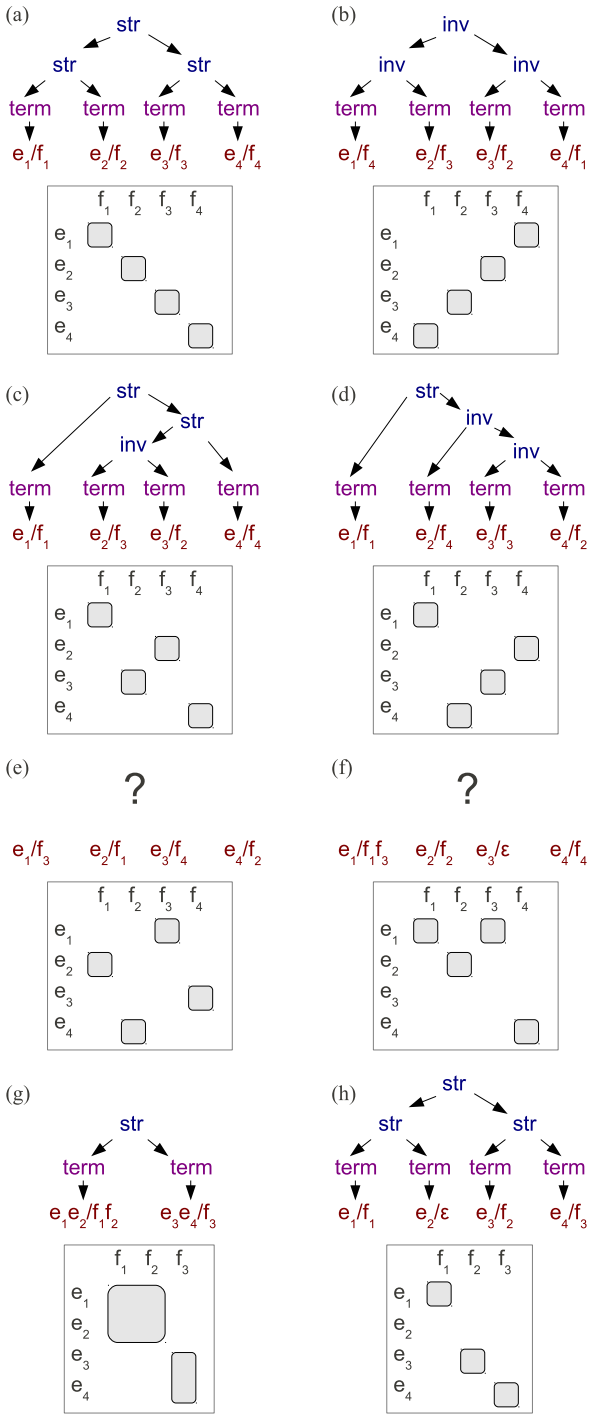

Figure 4.4: Eight different inversion transduction grammar (ITG) derivations and their corresponding alignment matrices.

shown in Figure 4.4 (f). However, these are known to be relatively rare, with ITGs being reported to cover 98.8-100% of the reordering patterns in actual parallel corpora (Wu, 1997; Haghighi et al., 2009).

The second phase takes place after generating the pre-terminal symbol, and consists of generating short parallel phrases. While these are generally word pairs such as "to/de," "admit/admettre," and "it/le," they can also be one-to-many or many-to-many alignments such as "admit it/le admettre," or one-to-zero or many-to-zero alignments, where words in one language have no translation in the other language. These allow for the representation of alignments that are not strictly one-to-one such as those shown in Figures 4.4 (g-h).

4.2.2 Probabilistic ITGs

By assigning a probability to each of the ITG productions, it is possible to create a generative model for parallel phrase pairs. The traditional ITG generative probability for a particular phrase pair $P_{flat}(\langle \boldsymbol{e}, \boldsymbol{f} \rangle; \theta_x, \theta_t)$ is parameterized by θ_t, which defines a distribution over terminal phrase pairs (the minimal phrase pairs that are generated after the pre-terminal symbol), and θ_x, which specifies a probability distribution over the non-terminal and pre-terminal symbols that define the derivation structure. A number of small variations of this traditional ITG model have been proposed in the literature, but the following generative story can be used as a representative of previously proposed models.

1. Generate symbol x from the multinomial distribution $P_x(x; \theta_x)$. x can take the values TERM, STR, or INV.

2. According to the value of x, take the following actions.

   (a) If $x =$ TERM, the pre-terminal, generate a phrase pair from the terminal phrase distribution $P_t(\langle \boldsymbol{e}, \boldsymbol{f} \rangle; \theta_t)$.

   (b) If $x =$ STR, a straight ITG non-terminal, generate phrase pairs $\langle \boldsymbol{e}_1, \boldsymbol{f}_1 \rangle$ and $\langle \boldsymbol{e}_2, \boldsymbol{f}_2 \rangle$ from P_{flat}, and concatenate them into a single phrase pair $\langle \boldsymbol{e}_1\boldsymbol{e}_2, \boldsymbol{f}_1\boldsymbol{f}_2 \rangle$.

   (c) If $x =$ INV, an inverted ITG non-terminal, follow the same process as (b), but concatenate $\boldsymbol{f}_1$ and $\boldsymbol{f}_2$ in reverse order $\langle \boldsymbol{e}_1\boldsymbol{e}_2, \boldsymbol{f}_2\boldsymbol{f}_1 \rangle$.

The result of this generative process is a bilingual phrase pair, along with its corresponding generative probability. Hereafter, this model will be referred to as FLAT.

ITG-based models can be used to find alignments for words in parallel sentences through the process of biparsing (Wu, 1997). Within the previously described ITG framework, a sentence pair $\langle E, F \rangle$ can be defined as the phrase pair that is generated by the node at the top of the derivation tree. Biparsing for ITGs finds the most likely derivation for this sentence pair given the ITG probabilities. Once we have this most likely derivation, we treat all phrase pairs that were generated from the same terminal symbols as aligned (for example, in Figure 4.3: "to/de," "admit/admettre," and "it/le").

4.3 Bayesian Modeling for Inversion Transduction Grammars

The probabilities of ITG models can be calculated in the same manner as traditional unsupervised PCFGs using the expectation-maximization algorithm and maximum likelihood estimation. However, as noted by (DeNero et al., 2006), when many-to-many alignments are allowed, the solution that maximizes the likelihood is often to simply memorize every sentence as a single phrase pair, a degenerate solution that defeats the purpose of performing alignment. (Zhang et al., 2008a) and others propose dealing with this problem by putting a prior probability $P(\theta_x, \theta_t)$ on the parameters, which allows us to bias towards compact models and prevent this degenerate solution.

Priors based on the Bayesian statistics introduced in Chapter 2 have proven useful for controlling model complexity in previous work, and a similar approach will be adopted here. The symbol distribution parameters θ_x specify a multinomial distribution over three elements. Because of this it is natural to use a Dirichlet distribution as a prior for θ_x, as the Dirichlet distribution is the conjugate prior of the multinomial distribution.

$$\theta_x \sim Dirichlet(\alpha). \tag{4.3}$$

α is the concentration hyper-parameter controlling the sparsity of the distribution, but this has little empirical effect on the results, so we can arbitrarily set $\alpha = 1$.

The phrase table parameters θ_t specify a multinomial distribution over an undetermined number of elements (every possible phrase pair). Previous work on both word alignment (Zhang et al., 2008a; Blunsom et al., 2009) and other natural language processing tasks has used non-parametric Bayesian techniques to specify priors over these sort of infinite multinomial distributions. In particular we use a prior based on the non-parametric Pitman-Yor

process described in Section 2.2.4. Remember that the prior is expressed as

$$\theta_t \sim PY(d, s, P_{base}), \tag{4.4}$$

where d is the discount parameter, s is the strength parameter, and P_{base} is the base measure. The most important element of this distribution for the task at hand is how we define the base measure P_{base}, which assigns the prior probability of generating a particular phrase pair. The following section describes this in more detail.

Non-parametric priors are appropriate for modeling the phrase distribution because every time a phrase is generated by the model, it is "memorized" and given a higher probability. Within the framework of the ITG model, this indicates that phrase pairs that are generated by P_t many times are more likely to be reused (the *rich-get-richer* effect), which results in the induction of phrase tables with fewer, but more helpful phrases. In the FLAT model, non-terminal nodes are first generated from P_x, reducing the sentence to manageable chunks, followed by the generation of the pre-terminal from P_x, then the generation of a minimal phrase pair from P_t. As P_t will only generate a phrase pair at the end of the generative process, only phrase pairs of the smallest level of granularity will be memorized and given a higher probability by the model.

As previous works had used the Dirichlet process to specify the phrase distribution (DeNero et al., 2008), we performed preliminary experiments (using the data described in Section 4.7) to test whether the Pitman-Yor process was actually necessary. The results of these experiments confirmed that the Pitman-Yor process with automatically adjusted parameters results in superior alignment results, outperforming the sparse Dirichlet process priors used in previous research.[2] The average gain across all data sets was approximately 0.8 BLEU points.

4.3.1 Base Measure

P_{base} in Equation (4.4) is the base measure, the prior probability of phrase pairs according to the model. By choosing this probability appropriately, we can incorporate prior knowledge of what phrases tend to be aligned to each other. In particular, there are three pieces of prior knowledge that

[2]Following (Teh, 2006), we put priors on s ($Gamma(\alpha = 2, \beta = 1)$) and d ($Beta(\alpha = 2, \beta = 2)$) for the Pitman-Yor process, and sample their values. These priors do not provide a strong bias towards any particular value of s or d, allowing the model freedom to choose values that maximize the likelihood of the training data. For the Dirichlet process α was set to 10^{-10}.

we would like to provide through the base measure. First, we would like to minimize the number of phrases that are not aligned to any phrase in the other language, as we can assume that most of the phrases will have some corresponding translation. Second, we would like to bias against overly long phrases, as these are likely to cause sparsity and hurt generalization performance when the model is tested on new data. Finally, when aligning multi-word phrases, it makes sense to align phrases that are composed of words that are good translations of each other.

This section adopts a formulation similar to that of (DeNero et al., 2008) that is able to satisfy all of these desiderata. P_{base} is first calculated by choosing whether to generate an unaligned phrase pair (where $|\boldsymbol{e}| = 0$ or $|\boldsymbol{f}| = 0$) according to a fixed probability p_u. p_u should generally be a small value to minimize the number of unaligned phrases.[3] Based on this choice, we next generate an aligned phrase pair from P_{ba}, or an unaligned phrase pair from P_{bu}

For P_{ba}, we use the following probability:

$$P_{ba}(\langle \boldsymbol{e}, \boldsymbol{f} \rangle) = M_0(\langle \boldsymbol{e}, \boldsymbol{f} \rangle) P_{pois}(|\boldsymbol{e}|; \lambda) P_{pois}(|\boldsymbol{f}|; \lambda) \tag{4.5}$$

$$M_0(\langle \boldsymbol{e}, \boldsymbol{f} \rangle) = (P_{m1}(\boldsymbol{f}|\boldsymbol{e}) P_{uni}(\boldsymbol{e}) P_{m1}(\boldsymbol{e}|\boldsymbol{f}) P_{uni}(\boldsymbol{f}))^{\frac{1}{2}}. \tag{4.6}$$

P_{pois} is the Poisson distribution with the average length parameter λ, where k represents the phrase length $|\boldsymbol{f}|$ or $|\boldsymbol{e}|$.

$$P_{pois}(k|\lambda) = \frac{(\lambda - 1)^{k-1}}{(k-1)!} e^{-(\lambda - 1)}. \tag{4.7}$$

λ was set to a relatively small value, which allows us to bias against overly long phrases.[4]

P_{uni} is the unigram probability of a particular phrase, and P_{m1} is the word-based Model 1 (Brown et al., 1993) probability of one phrase given the other. Model 1 probabilities are word-based translation probabilities that help to indicate whether the words in each phrase are good translations of each other. The phrase-based Model 1 probability is calculated according to the following equation:

$$P_{m1}(\boldsymbol{e}|\boldsymbol{f}) = \prod_{i=1}^{|\boldsymbol{e}|} \frac{1}{|\boldsymbol{f}|} \sum_{j=1}^{|\boldsymbol{f}|} P_{m1}(e_i|f_j) \tag{4.8}$$

[3] One of the three values 10^{-2}, 10^{-3}, or 10^{-10} was chosen based on which value gave the best translation accuracy on the development set.

[4] We tune λ to 1, 0.1, or 0.01 based on which value gives the best translation accuracy on the development set.

where e_i and f_j are the i-th and j-th words in phrases $\boldsymbol{e}$ and $\boldsymbol{f}$ respectively. The word-based probabilities $P_{m1}(e_i|f_j)$ and $P_{m1}(f_j|e_i)$ are parameters of the model, and can be calculated efficiently using the expectation maximization algorithm (Brown et al., 1993) before starting phrase alignment.

Following (Liang et al., 2006), we combine the Model 1 probabilities in both directions using the geometric mean. It should be noted that the probabilities of the geometric mean do not add to one, and are thus not, strictly speaking, proper probabilities. However empirically, even when left unnormalized, they provided much better results than the model using the arithmetic mean, which is mathematically correct. This is because taking the geometric mean favors alignments that are supported by both models, and alignments for which both models agree are generally highly reliable. On the other hand, the arithmetic mean favors alignments that are supported by either of the two models, which indicates that the less reliable alignments where the two models disagree will still receive a relatively high probability, weakening the ability of the base measure to penalize bad alignment candidates.

For P_{bu}, in the case of $|\boldsymbol{f}| = 0$, we calculate the probability as follows:

$$P_{bu}(\langle \boldsymbol{e}, \boldsymbol{f} \rangle) = P_{uni}(\boldsymbol{e}) P_{pois}(|\boldsymbol{e}|; \lambda)/2. \tag{4.9}$$

The probability can be calculated similarly when $|\boldsymbol{e}| = 0$. Note that P_{bu} is divided by 2 as the probability is considering null alignments in both directions.

4.4 Hierarchical ITG Model

While in FLAT only minimal phrases were memorized by the model, as (DeNero et al., 2008) note and the experiments in Section 4.7 confirm, using only minimal phrases leads to inferior results for phrase-based translation. Because of this, previous research has combined FLAT with heuristic phrase extraction, which exhaustively combines all adjacent phrases permitted by the word alignments (Och et al., 1999). This section proposes an alternative, fully statistical approach that directly models phrases at multiple granularities, which will be referred to as HIER. By doing so, we are able to do away with heuristic phrase extraction, creating a phrase table that is able to achieve competitive accuracy in a single step through a fully probabilistic process.

Similarly to FLAT, HIER assigns a probability $P_{hier}(\langle \boldsymbol{e}, \boldsymbol{f} \rangle; \theta_x, \theta_t)$ to phrase pairs, and is parameterized by a phrase table θ_t and a symbol distribution

θ_x. The main difference between the two models is that non/pre-terminal symbols and phrase pairs are generated in reverse order. While FLAT first generates branches of the derivation tree using P_x, then generates leaves using the phrase distribution P_t, HIER first attempts to generate the full sentence as a single phrase pair from P_t, then falls back to ITG-style derivations to cope with sparsity. The proposed model accounts for this within the Bayesian ITG context by defining a new base measure P_{dac} ("divide-and-conquer") to replace P_{base} in Equation (4.4), resulting in the following distribution for θ_t.

$$\theta_t \sim PY(d, s, P_{dac}) \tag{4.10}$$

P_{dac} essentially generates a single longer phrase through two generations and a combination of shorter phrases, allowing even long phrase pairs to be given significant amounts of probability when justified. The generative process of P_{dac}, similar to that of P_{flat} from the previous section, is as follows:

1. Generate symbol x from $P_x(x; \theta_x)$. x can take the values BASE, STR, or INV.
2. According to x, take the following actions.
   (a) If $x =$ BASE, generate a new phrase pair directly from P_{base} of Section 4.3.1.
   (b) If $x =$ STR, generate $\langle \boldsymbol{e}_1, \boldsymbol{f}_1 \rangle$ and $\langle \boldsymbol{e}_2, \boldsymbol{f}_2 \rangle$ from P_{hier}, and concatenate them into a single phrase pair $\langle \boldsymbol{e}_1\boldsymbol{e}_2, \boldsymbol{f}_1\boldsymbol{f}_2 \rangle$.
   (c) If $x =$ INV, follow the same process as (b), but concatenate $\boldsymbol{f}_1$ and $\boldsymbol{f}_2$ in reverse order $\langle \boldsymbol{e}_1\boldsymbol{e}_2, \boldsymbol{f}_2\boldsymbol{f}_1 \rangle$.

A comparison of derivation trees for FLAT and HIER is shown in Figure 4.5. As previously described, FLAT first generates from the symbol distribution P_x, then from the phrase distribution P_t. On the other hand, HIER generates directly from P_t, which falls back to divide-and-conquer based on P_x when necessary. The minimal and non-minimal phrase pairs that are generated by P_t are surrounded by solid and dotted lines respectively. It can be seen that while P_t in FLAT only generates minimal phrases, P_t in HIER generates (and thus memorizes) phrases at all levels of granularity.

4.4.1 Length-based Parameter Tuning

There are still two problems with HIER, one theoretical, and one practical. Theoretically, HIER contains itself as its base measure, and stochastic

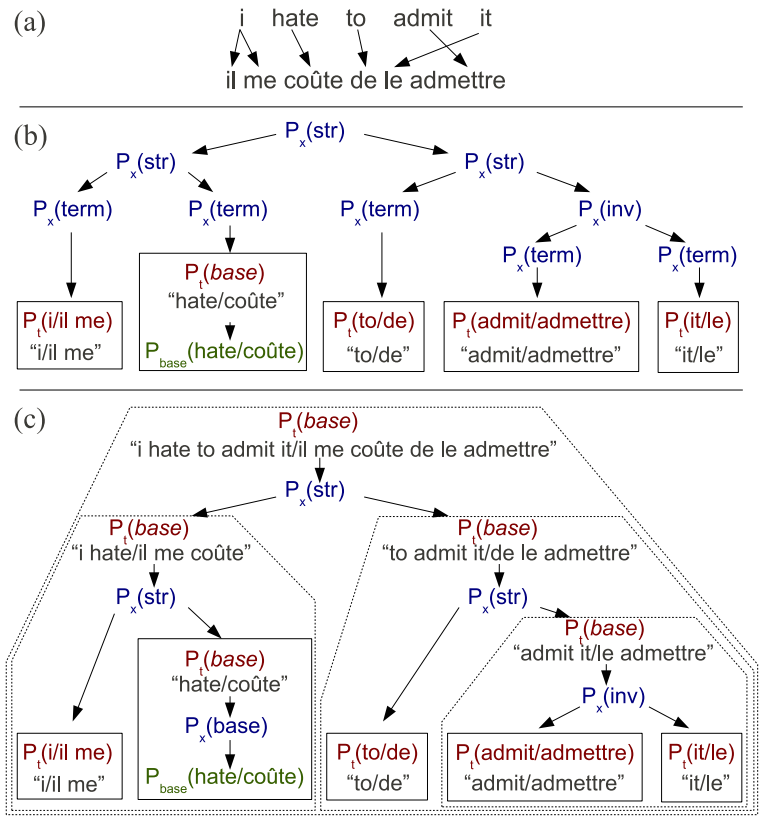

Figure 4.5: A word alignment (a), and its derivation according to FLAT (b), and HIER (c). Solid and dotted lines indicate minimal and non-minimal pairs respectively, and phrases memorized by the model are written in quotes under their corresponding instance of P_t. The pair hate/coûte is generated due to the contribution P_{base}.

process models that include themselves as base measures are technically deficient, as noted in (Cohen et al., 2010). Practically, while the Pitman-Yor process in HIER shares the parameters s and d over all phrase pairs in the model, long phrase pairs are much more sparse than short phrase pairs, and thus it is desirable to appropriately adjust the parameters of Equation (4.4) according to the phrase pair length.

In order to solve these problems, we reformulate the model so that each phrase length $l = |\boldsymbol{f}| + |\boldsymbol{e}|$ has its own phrase parameters $\theta_{t,l}$ and symbol parameters $\theta_{x,l}$, which are given separate priors:

$$\theta_{t,l} \sim PY(d, s, P_{dac,l}) \tag{4.11}$$

$$\theta_{x,l} \sim Dirichlet(\alpha) \tag{4.12}$$

This model will be referred to as HLEN.

The generative story is largely similar to HIER with a few minor changes. When we generate a sentence, we first choose its length l according to a uniform distribution over all possible sentence lengths

$$l \sim Uniform(1, L), \tag{4.13}$$

where L is the size $|E| + |F|$ of the longest sentence in the corpus. As noted by (Brown et al., 1993), defining a distribution over sentence lengths is necessary to create a true generative model, but has no effect on alignments as the length of both sentences $|E|$ and $|F|$ are known before learning begins.

We then generate a phrase pair from the probability $P_{t,l}(\langle \boldsymbol{e}, \boldsymbol{f} \rangle)$ for length l. The base measure for HLEN is identical to that of HIER, with one minor change: when we fall back to two shorter phrases, we must choose the length of the shorter phrases so we know which distribution $P_{t,l}$ from which to generate them. In order to do so in a probabilistic manner, we first choose the length of the left phrase from $l_l \sim Uniform(1, l-1)$, set the length of the right phrase to $l_r = l - l_l$, and generate the smaller phrases from P_{t,l_l} and P_{t,l_r} respectively. Here, the choice of distribution does have an effect on alignment results; if we choose a distribution that prefers numbers closer to the middle of the range (close to $l/2$) the derivation trees will be more balanced, while if we choose a distribution that prefers small or large numbers the derivation trees will tend to be left-branching or right-branching, respectively. However, there is no intuitive reason why any of these derivation structures would be preferable over the others, so we choose to use the uninformative uniform distribution and let the other parts of the model resolve this ambiguity.

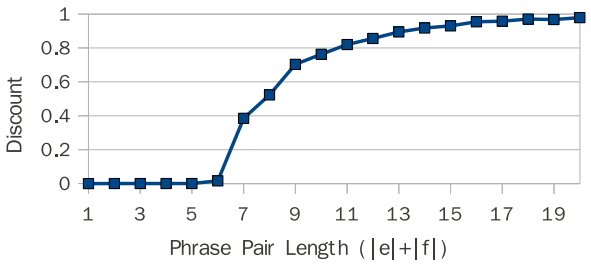

Figure 4.6: Learned discount values by phrase pair length.

In this framework, it can be seen that phrases at each length are generated from different distributions, and thus the parameters for the Pitman-Yor process will be different for each distribution. Furthermore, as l_l and l_r must be smaller than l, $P_{t,l}$ no longer contains itself as a base measure, and is thus not deficient.

An example of the actual discount values learned in one of the experiments described in Section 4.7 is shown in Figure 4.6. It can be seen that, as expected, the discounts for short phrases are lower than those of long phrases. In particular, phrase pairs of length up to six (for example, $|\boldsymbol{e}| = 3$, $|\boldsymbol{f}| = 3$) are given discounts of nearly zero while larger phrases are more heavily discounted. This is likely related to the observation by (Koehn et al., 2003) that using phrases where $\max(|\boldsymbol{e}|, |\boldsymbol{f}|) \leq 3$ cause significant improvements in translation accuracy, while using larger phrases results in diminishing returns.

In addition, the HLEN model has the potential to learn different ITG reordering probabilities for different lengths. An example of the ratio between $P_x(str)$ and $P_x(inv)$ learned for phrases of length 4 to 40 in German, Spanish, French, and Japanese is shown in Figure 4.7. It can be seen that at the shortest phrase length of 4, which generally corresponds to the reordering of two single-word translations, that German has a higher ratio than all other languages. This is intuitive, as both French and Spanish order adjective-noun pairs in the opposite order of English, so there should be more swaps of single words than in German, which places adjective-noun pairs in the same order as English. On the other hand, as sentence length grows longer, French and Spanish surpass German in monotonicity, a result of German having greater divergence in syntax from English. One typical example of this is that sentence-final verbs in German must be reordered over long distances to their natural position in the middle of the sentence

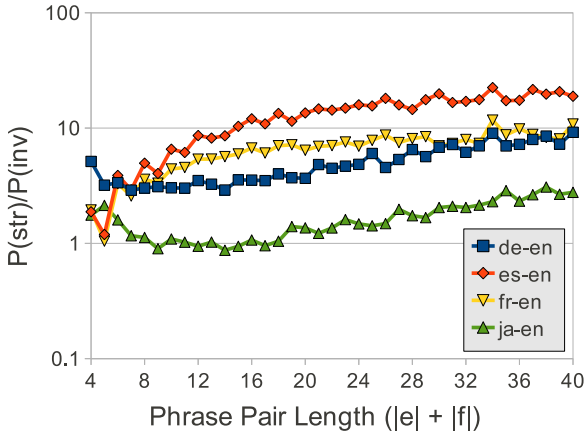

Figure 4.7: The ratio of $P_x(str)$ to $P_x(inv)$ by length. Higher values indicate more monotonic alignments.

for English. Finally, Japanese has significantly lower monotonicity than all of the European languages at almost all phrase pair lengths, a result of the vast differences in sentence structure between Japanese and English. In contrast, HIER can only learn a single value for $P_x(str)$ and $P_x(inv)$. For German, Spanish, French, and Japanese, the values of $P_x(str)/P_x(inv)$ were 4.83, 5.81, 4.99, and 1.83 respectively, showing that the overall preference for monotonicity or non-monotonicity can be learned, although not in the fine-grained manner allowed by HLEN.

4.4.2 Implementation

Previous research has used a variety of methods to learn Bayesian phrase based alignment models, all of which have used Gibbs sampling as their central learning algorithm (DeNero et al., 2008; Blunsom et al., 2009; Blunsom and Cohn, 2010). All of these techniques are applicable to the proposed model, but the sentence-based blocked sampling proposed by (Blunsom and Cohn, 2010) has desirable convergence properties compared to sampling single alignments, and is what we will use here. In this method, the majority of computation in the sampling process takes place in the parsing step where probabilities for each possibly aligned bilingual span are calculated to allow for proper sampling of an ITG parse tree for each sentence.

Exhaustive parsing of ITGs can be performed in $O(n^6)$, but this is too

slow in practical situations for all but the smallest of sentences. One solution to this problem is the beam search algorithm of (Saers et al., 2009), which can be used as an approximation of full exhaustive parsing. This algorithm works by separating each span into a bucket based on its length $l = |\boldsymbol{f}|+|\boldsymbol{e}|$, then performing beam pruning over the spans in each bucket to reduce the number of hypotheses that must be expanded. Instead of the histogram pruning proposed by (Saers et al., 2009), the experiments presented here use a probability beam, trimming spans where the probability is at least 10^{10} times smaller than that of the best hypothesis of identical length, as this was found to give better results in comparable time.

One important implementation detail that is different from previous models is the management of phrase counts. As a phrase pair t_a may have been generated from two smaller component phrases t_b and t_c, when a sample containing t_a is removed from the distribution, it may also be necessary to decrement the counts of t_b and t_c as well. The Chinese Restaurant Process representation of P_t (Teh, 2006) lends itself to a natural and easily implementable solution to this problem. For each table representing a phrase pair t_a, we can maintain not only the number of customers sitting at the table, but also the identities of phrases t_b and t_c that were originally used when generating the table. When the count of the table t_a is reduced to zero and the table is removed, the counts of t_b and t_c are also decremented.

4.5 Phrase Extraction

This section describes both traditional heuristic phrase extraction, and the proposed model-based extraction method.

4.5.1 Heuristic Phrase Extraction

The traditional method for heuristic phrase extraction from word alignments exhaustively enumerates all phrases up to a certain length that are consistent with the alignment (Och et al., 1999). After counts for each phrase pair $\langle \boldsymbol{e}, \boldsymbol{f} \rangle$ have been enumerated, these counts are used to calculate five features used in the phrase table:

- **Phrase conditional probabilities:** These are calculated in both directions using maximum likelihood estimation over phrase pair counts:

$$P_{ml}(\boldsymbol{e}|\boldsymbol{f}) = c_{\langle \boldsymbol{e},\boldsymbol{f} \rangle}/c_{\boldsymbol{f}} \tag{4.14}$$

$$P_{ml}(\boldsymbol{f}|\boldsymbol{e}) = c_{\langle \boldsymbol{e},\boldsymbol{f} \rangle}/c_{\boldsymbol{e}}. \tag{4.15}$$

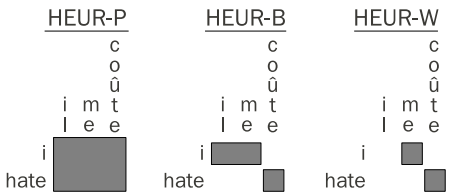

Figure 4.8: The phrase, block, and word alignments used in heuristic phrase extraction.

- **Lexical weighting probabilities:** As many phrases have very low counts, simple phrase conditional probabilities are sparse and often do not provide reliable information about the correctness of the phrase pair. To solve this problem, (Koehn et al., 2003) proposes a method of breaking each phrase down into its respective words, and using the conditional probabilities of the words in the phrase to calculate a more robust estimate of the phrase translation probabilities. The lexical weighting probabilities in both directions are used as two additional features in the model.

- **Phrase penalty:** The last feature is a fixed penalty or bonus for every phrase used. If it is a penalty, the model will prefer to use fewer but longer phrases, and if it is a bonus the model will prefer to use many shorter phrases.

These features are combined in a weighted manner to indicate the overall score of each phrase, with the weights being learned using a training regimen such as minimum error rate training (MERT (Och, 2003)).

Heuristic phrase extraction over the alignments acquired by the FLAT and HIER models is used as a baseline. As the proposed method often aligns relatively long phrases, not words, a variety of alignment granularities can be used to create the phrase table (Figure 4.8). In model HEUR-P, minimal phrases generated from P_t are treated as aligned, and phrase extraction is performed over these alignments. Two other techniques are also tested to create smaller alignment chunks that prevent sparsity. The first method performs regular sampling of the trees, but when a minimal phrase generated from P_t is reached, we continue traveling down the tree until we reach either a one-to-many alignment, which will be called HEUR-B as it creates alignments of "blocks," or an at-most-one alignment, which will be called HEUR-W as it generates word alignments. It should be noted that forcing alignments

smaller than the model suggests is only used for generating alignments for use in heuristic extraction, and does not affect the training process.

4.5.2 Model-Based Phrase Extraction

For our proposed model, we are also able to perform phrase table extraction that directly utilizes the phrase probabilities $P_t(\langle \boldsymbol{e}, \boldsymbol{f} \rangle)$. Similarly to the heuristic phrase tables, we use conditional probabilities $P_t(\boldsymbol{f}|\boldsymbol{e})$ and $P_t(\boldsymbol{e}|\boldsymbol{f})$, lexical weighting probabilities, and a phrase penalty. Here, instead of using maximum likelihood, we calculate conditional probabilities directly from P_t probabilities:

$$P_t(\boldsymbol{f}|\boldsymbol{e}) = P_t(\langle \boldsymbol{e}, \boldsymbol{f} \rangle) / \sum_{\{\tilde{\boldsymbol{f}}: c_{\langle \boldsymbol{e}, \tilde{\boldsymbol{f}} \rangle} \geq 1\}} P_t(\langle \boldsymbol{e}, \tilde{\boldsymbol{f}} \rangle) \tag{4.16}$$

$$P_t(\boldsymbol{e}|\boldsymbol{f}) = P_t(\langle \boldsymbol{e}, \boldsymbol{f} \rangle) / \sum_{\{\tilde{\boldsymbol{e}}: c_{\langle \tilde{\boldsymbol{e}}, \boldsymbol{f} \rangle} \geq 1\}} P_t(\langle \tilde{\boldsymbol{e}}, \boldsymbol{f} \rangle). \tag{4.17}$$

To limit phrase table size, we include only phrase pairs that are aligned at least once in the sample.

Two more features are also added:

- **Model joint probability:** As the proposed method assigns a probability $P_t(\langle \boldsymbol{e}, \boldsymbol{f} \rangle)$ to all phrase pairs, we can use this as an additional feature.
- **Span generative probability:** Use the average generative probability of each span that generated $\langle \boldsymbol{e}, \boldsymbol{f} \rangle$ as computed by the chart parser during training. This is similar to the joint probability, but is more reliable for low-frequency phrases, where the model probability tends to over-estimate the actual probability. The generative probability will be high for common phrase pairs that are generated directly from the model, and also for phrases that, while not directly included in the model, are composed of two high-probability child phrases and thus can be assumed to be more reliable.

It should be noted that while for FLAT and HIER P_t can be used directly, as HLEN learns separate models for each length, we must combine these probabilities into a single value. We can do this by setting

$$P_t(\langle \boldsymbol{e}, \boldsymbol{f} \rangle) = P_{t,l}(\langle \boldsymbol{e}, \boldsymbol{f} \rangle) c_l / \sum_{\tilde{l}=1}^{L} c_{\tilde{l}} \tag{4.18}$$

for every phrase pair, where $l = |\boldsymbol{e}| + |\boldsymbol{f}|$ and c_l is the number of phrases of length l in the sample.

This model-based extraction method is referred to as MOD.

4.5.3 Sample Combination

As has been noted in previous works, (Koehn et al., 2003; DeNero et al., 2006) exhaustive phrase extraction tends to outperform approaches that use syntax or generative models to limit phrase boundaries. (DeNero et al., 2006) states that this is because generative models choose only a single phrase segmentation, and thus throw away many good phrase pairs that are in conflict with this segmentation.

Fortunately, in the Bayesian framework it is simple to overcome this problem by combining phrase tables from multiple samples. In MOD, we do this by taking the average of the joint probability and span probability features, and recalculating the conditional probabilities from the averaged joint probabilities.

4.6 Related Work

While ITGs have been growing in popularity in recent years, they are by no means the only method for word or phrase alignment. In fact, the seminal IBM models presented in (Brown et al., 1993) and the implementation provided by the open-source software GIZA++[5] (Och and Ney, 2003) are still used for word alignment in a large number of systems. The IBM models, while quite powerful, are fundamentally different from the models previously described in this chapter in that they are not able to handle many-to-many alignments. As a result, it is necessary to find one-to-many word alignments in both directions, which allows for the capturing of multi-word units on both the source and target sides. These one-to-many alignments can then be combined using heuristics into a many-to-many alignment (Koehn et al., 2003). Finally, using this alignment, heuristic phrase extraction enumerates all possible phrases that do not conflict with the word alignments (Och et al., 1999). In the next section, we present experimental results comparing alignments acquired using the IBM models with those acquired using ITG-based alignment methods.

In addition to the previously mentioned alignment techniques, there has also been a significant body of work on improving phrase extraction methods

[5] `http://code.google.com/p/giza-pp/`

(such as (Moore and Quirk, 2007) and (Johnson et al., 2007a)). (DeNero and Klein, 2010) presented the first work on joint phrase alignment and extraction at multiple levels. While they take a supervised approach based on discriminative methods, this work presents a fully unsupervised generative model.

The generative probabilistic model where longer units are built through the binary combination of shorter units that we use in this model was inspired by the model proposed by (de Marcken, 1996) for monolingual word segmentation using the minimum description length (MDL) framework. Our work differs in that it uses Bayesian techniques instead of MDL, works on two languages instead of one, and uses words as its basic unit instead of phrases.

Adaptor grammars, models in which non-terminals memorize subtrees that lie below them, have been used for word segmentation or other monolingual tasks (Johnson et al., 2007c). The proposed method could be thought of as a synchronous adaptor grammar over two languages. However, adaptor grammars have generally been used to specify only two or a few levels as in the FLAT model in this chapter, as opposed to recursive models such as HIER or many-leveled models such as HLEN. One exception is the variational inference method for adaptor grammars presented by (Cohen et al., 2010) that is applicable to recursive grammars such as HIER. Applying variational inference to the models proposed here is a promising direction for future work.

4.7 Experimental Evaluation

This section presents experiments on translation tasks from four languages, French, German, Spanish, and Japanese, into English that evaluate the effectiveness of the proposed method.

4.7.1 Experimental Setup

The data for French, German, and Spanish are from the 2010 Workshop on Statistical Machine Translation (Callison-Burch et al., 2010). The news commentary corpus was used for training the phrase table, and the news commentary and EuroParl corpora for training the LM. The Japanese experiments were performed using data from the NTCIR patent translation task (Fujii et al., 2008). The first 100k sentences of the parallel corpus were used to construct the phrase table, and the whole parallel corpus was used to construct the LM. Details of both corpora can be found in Table

Table 4.1: The number of words in each corpus for phrase table (PT) and LM training, tuning, and testing.

	de-en	es-en	fr-en	ja-en
PT (en)	1.80M	1.62M	1.35M	2.38M
PT (other)	1.85M	1.82M	1.56M	2.78M
LM (en)	52.7M	52.7M	52.7M	44.7M
Tune (en)	49.8k	49.8k	49.8k	68.9k
Tune (other)	47.2k	52.6k	55.4k	80.4k
Test (en)	65.6k	65.6k	65.6k	40.4k
Test (other)	62.7k	68.1k	72.6k	48.7k

4.1. Corpora are tokenized, lower-cased, and sentences of over 40 words on either side are removed for phrase table training. For both tasks, weight tuning and testing were performed on specified development and test sets. Case-insensitive BLEU score is used as an evaluation measure (Papineni et al., 2002), a widely used evaluation metric for machine translation.

Next, the accuracy of our proposed method of joint phrase alignment and extraction using the FLAT, HIER and HLEN models is compared with a baseline of using word alignments from GIZA++ (Och and Ney, 2003) and heuristic phrase extraction. Translation is performed using the Moses phrase-based machine translation decoder (Koehn and others, 2007) using the phrase tables learned by each method under consideration. Phrase reordering probabilities are calculated using Moses's standard lexicalized reordering model (Koehn et al., 2005) for all experimental settings. The maximum phrase length is limited to 7 in all models, and the LM is created using an interpolated Kneser-Ney 5-gram model.

GIZA++ is trained using the standard training regimen up to Model 4, and the resulting alignments are combined with the `grow-diag-final-and` heuristic (Koehn et al., 2005). The proposed models were allowed to run 100 iterations, with the final sample acquired at the end of the training process being used to construct the translation model in experiments using a single sample.[6] In addition, results are presented for averaging the phrase tables from the last ten samples as described in Section 4.5.3.

[6]For most models, while likelihood continued to increase gradually for all 100 iterations, BLEU score gains plateaued after 5-10 iterations, likely due to the strong prior information provided by P_{base}. As iterations took 1.3 hours on a single processor, good translation results can be achieved in approximately 13 hours, which could be further reduced using distributed sampling (Newman et al., 2009; Blunsom et al., 2009).

Table 4.2: BLEU score and phrase table size by alignment method and samples combined. Bold numbers are not significantly different from the best result according to the sign test ($p < 0.05$). GIZA uses HEUR-W for phrase extraction and all other models use MOD.

		de-en		es-en		fr-en		ja-en	
Align	Samp	BLEU	Size	BLEU	Size	BLEU	Size	BLEU	Size
GIZA	1	**16.62**	4.91M	**22.00**	4.30M	21.35	4.01M	**23.20**	4.22M
FLAT	1	13.48	136k	19.15	125k	17.97	117k	16.10	89.7k
HIER	1	**16.58**	1.02M	**21.79**	859k	**21.50**	751k	**23.23**	723k
HLEN	1	**16.49**	1.17M	21.57	930k	21.31	860k	**23.19**	820k
HIER	10	**16.53**	3.44M	**21.84**	2.56M	**21.57**	2.63M	**23.12**	2.21M
HLEN	10	**16.51**	3.74M	**21.69**	3.00M	**21.53**	3.09M	**23.20**	2.70M

4.7.2 Experimental Results

The results for these experiments can be found in Table 4.2. From these results it can be seen that when using a single sample, the combination of using HIER and model probabilities achieves results approximately equal to GIZA++ and heuristic phrase extraction. This is the first reported result in which an unsupervised phrase alignment model has built a phrase table directly from model probabilities and achieved results that compare to heuristic phrase extraction. It can also be seen that the phrase table created by the proposed method is approximately 5 times smaller than that obtained by the traditional pipeline.

In addition, HIER significantly outperforms FLAT when using the model probabilities. This confirms that phrase tables containing only minimal phrases are not able to achieve results that compete with phrase tables of multiple granularities.

Somewhat surprisingly, HLEN consistently slightly underperforms HIER. This indicates potential gains to be provided by length-based parameter tuning were outweighed by losses due to the increased complexity of the model. In particular, as examined further in Section 4.7.3, dividing phrases of each length between different models provides an implicit bias to spread phrases evenly across all of the models, even when this is not justified by the data.

It can also be seen that combining phrase tables from multiple samples improved the BLEU score for HLEN, but not for HIER. This suggests that for HIER, most of the useful phrase pairs discovered by the model are included in every iteration, and the increased recall obtained by combining multiple

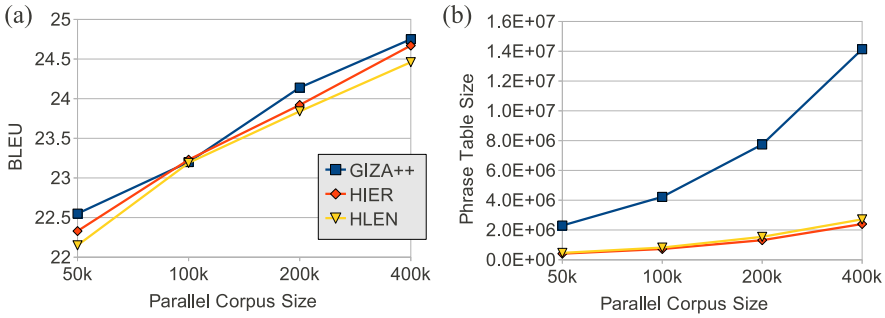

Figure 4.9: The effect of corpus size on the accuracy (a) and phrase table size (b) for each method (Japanese-English).

samples does not consistently outweigh the increased confusion caused by the larger phrase table.

Effect of Corpus Size

In order to ensure that the proposed method works well at all data sizes, experiments were also performed varying the size of the training corpus. As there are not large amounts of in-domain data for the news commentary task, experimental results are presented only on the Japanese-English patent task, varying the number of training sentences from 50k to 400k. The accuracy results are shown in Figure 4.9 (a). It can be seen that the results are largely consistent across all data sizes over, with statistically insignificant differences between HIER and GIZA++, and HLEN lagging slightly behind HIER. Figure 4.9 (b) shows the size of the phrase table induced by each method over the various corpus sizes. It can be seen that the tables created by GIZA++ are significantly larger at all corpus sizes, with the difference being particularly pronounced at larger corpus sizes.

Phrase Alignment/Heuristic Extraction

Table 4.3 shows additional results evaluating the effectiveness of model-based phrase extraction compared to heuristic phrase extraction. Using the alignments from HIER, phrase tables are created using both model probabilities (MOD), and heuristic extraction on words (HEUR-W), blocks (HEUR-B), and minimal phrases (HEUR-P) as described in Section 4.5. It can be seen that model-based phrase extraction using HIER outperforms or insignif-

Table 4.3: Translation results and phrase table size for various phrase extraction techniques (French-English).

	FLAT		HIER	
MOD	17.97	117k	21.50	751k
HEUR-W	21.52	5.65M	21.68	5.39M
HEUR-B	21.45	4.93M	21.41	2.61M
HEUR-P	21.56	4.88M	21.47	1.62M

Table 4.4: Overlap of phrase tables. The numbers indicate the percentage of the phrase table in the column that is also included in the phrase table in the row.

	GIZA	FLAT	HIER	HLEN
GIZA	-	40.46%	47.94%	41.54%
FLAT	1.68%	-	14.84%	12.51%
HIER	9.24%	68.72%	-	31.61%
HLEN	9.59%	69.40%	37.89%	-

icantly underperforms heuristic phrase extraction over all experimental settings, while keeping the phrase table to a fraction of the size of most heuristic extraction methods.

4.7.3 Acquired Phrases

This section presents a quantitative and qualitative analysis of the phrase tables acquired using GIZA, FLAT, HIER, and HLEN for the French-English task. Phrase extraction was performed with HEUR-W for GIZA and MOD for all other alignment methods.

First, Table 4.4 shows the results of an analysis of how much overlap there was between the extracted phrase tables. Interestingly, the GIZA phrase table only covers approximately 40-50% of the acquired phrases in each of the ITG models, despite being much larger. To help understand the difference between GIZA and the ITG-based methods, Table 4.5 shows a sample of the phrases that occurred in only the GIZA phrase table, as well as the phrases that occurred in all of the other phrase tables, but not the GIZA phrase table. For phrases found by all of the ITG models but not GIZA, the majority were rare single-word translations that were mis-aligned by GIZA due to the "garbage-collecting" phenomenon, where rare words are aligned to too many words, and thus not extracted properly. Among the shorter phrases that were found by none of the ITG models, but found by

Table 4.5: Examples of phrases that exist only in GIZA or ITG-based models.

GIZA only

our réduire les	in reducing the
perceptuel implique une	implies a
élections est	elections
des attentes qui	the expectations that
vanterait	might

ITG only

sensationnalisme	sensational
tapageur	flashy
évolué	moving
dégénèrent	degenerate
inscrire	enroll

GIZA, most were the combination of one or several content words with a preposition.

In addition, we show examples of the phrases that are not just included in the phrase table, but actually used in translation by GIZA and HIER, focusing on phrase pairs that were used much more often by one system than the other, as well as less frequent phrase pairs that were used by one system twice, but the other system no times. Phrase pairs more commonly used in the GIZA and HIER systems are shown in Table 4.6 and Table 4.7 respectively. It can be seen that as an overall trend, the GIZA system tends to translate function words and punctuation in phrases together with the neighboring words, while the HIER system tends to translate these words separately, reflecting previous observations about the composition of the respective phrase tables. This combination of function words and punctuation into longer phrases does not change translation results, but increases the size of the phrase table, lending a convincing explanation for why HIER is able to achieve translation results that match GIZA with a smaller phrase table.

In the next most common case, one of the two systems dropped a frequent word in a multi-word translation (such as "de la" above). This was a problem for both systems, and there was not a clear trend favoring either system in these cases. In addition, both GIZA and HIER see words that are unknown for one of the two systems ("opérateur" and "communiqué" respectively) due to missed alignments preventing the creation of phrases for rare words. Overall, GIZA was able to successfully generate phrases for fewer words, resulting in a total of 4,738 untranslatable words in the test set compared to 3,843 for

Table 4.6: Phrase pairs that are used more often by GIZA than HIER. #GIZA and #HIER are the number of times the phrase was used by each system.

Source	Target	#GIZA	#HIER	HIER Phrase
les	the	529	475	*with noun*
qu'	that	74	38	*with verb*
: "	: "	33	0	*separate*
c' est	it is	32	0	*separate*
opérateur	opérateur	32	0	operator
de la	the	33	2	of
2010 .	2010 .	2	0	*separate*
, ou alors	, or	2	0	*separate*
qui sont	who are	2	0	*separate*
il nous	we	2	0	*with comma*
travaillait	"	2	0	depravity (correct: "was working")

Table 4.7: Phrase pairs that are used more often by HIER than GIZA.

Source	Target	#GIZA	#HIER	GIZA Phrase
,	,	2061	2833	*with word*
de	of	685	1366	*with noun*
.	.	1443	2002	*with word*
la	the	495	820	*with noun*
le	the	574	795	*with noun*
définitif	final	0	2	done, means
communiqué	communiqué	0	2	declaration, communicated
fréquente	frequent	0	2	frequently
connaissant	surplus	0	2	moreover (correct: "knowing")
moment où	moment when	0	2	*with "the"*

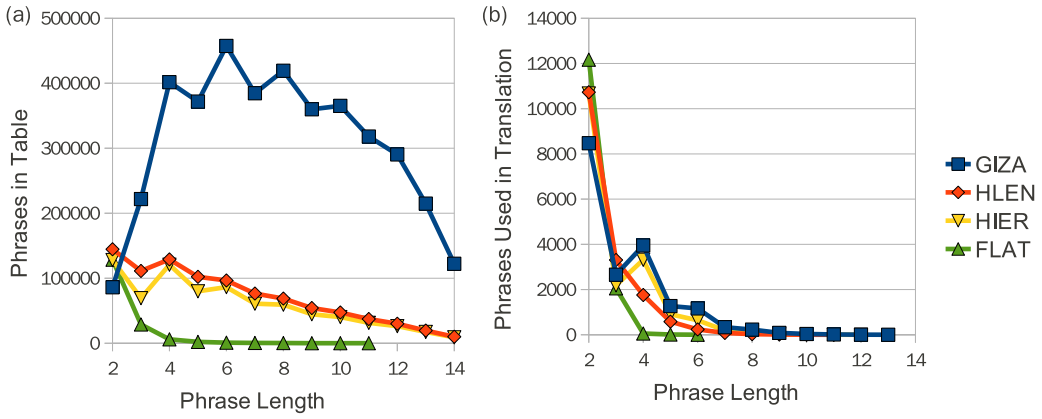

Figure 4.10: The distribution of unique phrases by length (a) included in the phrase table and (b) used in translation.

HIER. Finally, there were a number of examples of equally valid translations with different lexical choice ("définitif") and syntactic form ("fréquente"), as well as examples where neither system was able to create a translation correctly ("travaillait" and "connaissant").

Figure 4.10 shows a break-down by length of the phrases in the acquired phrase table, as well as of the phrases that were actually used during translation. From this graph it can be seen that GIZA creates large numbers of long phrases of length 6 or higher, despite the fact that the majority of used phrases are of length 2 or 4 (for 1-to-1 or 2-to-2 translations respectively). In general the distribution of phrases used by HIER in translation is similar to that of GIZA (with a slight tendency towards using shorter phrases), but the overall distribution of extracted phrases decreases gradually with length. FLAT, as expected, tends to both extract and use very short phrases.

Comparing HIER and HLEN, it can be seen that their patterns are largely similar, with the exception of phrases of length 3. Phrases of length 3 must be 1-to-2 or 2-to-1, and thus should be less common for language pairs such as English and French where one word tends to correspond to one word. HLEN creates more 3-word patterns than HIER because each length of phrase is given its own unique phrase distribution. Specifically, if $P_{t,3}$ has fewer phrases than $P_{t,2}$ and $P_{t,4}$, it also has more probability to "give away" to new 3-word phrases, creating an implicit bias towards creating more new phrases of less common phrase lengths. It is possible that this bias can be corrected by introducing priors that prefer phrase pairs where the number of

words is roughly equal on both sides. However, this will require significant expansions to the current generative story, which does not explicitly keep track of the number of words on each side, and thus we leave this to future work.

4.8 Conclusion

This chapter presented a novel approach to joint phrase alignment and extraction through a hierarchical model using non-parametric Bayesian methods and inversion transduction grammars (ITGs). Unlike previous work, this hierarchical model directly includes phrases of multiple granularities, which allows for the effective use of model probabilities in phrase table construction. Machine translation systems using phrase tables learned by the proposed model were able to achieve accuracy competitive with the traditional pipeline of word alignment and heuristic phrase extraction, the first such result for an unsupervised model.

One of the advantages of the proposed model is that it lends itself to relatively simple extension, allowing for further gains in accuracy through the introduction of more sophisticated alignment models. One promising future direction is the introduction of more intelligent prior knowledge through the base measure P_{base}. For example, P_{base} could be adjusted to take into account spelling similarities, parts of speech, phrase-based translation dictionaries, or bilingually acquired classes such as those proposed by (Och, 1999). It may also be possible to refine HLEN to use a more appropriate model of phrase length than the uniform distribution, particularly by attempting to bias against phrase pairs where one of the two phrases is much longer than the other.

In addition, it is worth testing the use of the proposed model with other forms of translation than simple phrase-based translation. These could include examining the applicability of the proposed model in the context of hierarchical phrases (Chiang, 2007), or in alignment using syntactic structure (Galley et al., 2006). It is also worth examining the plausibility of variational inference as proposed by (Cohen et al., 2010) in the alignment context.

Chapter 5

Lexical Acquisition for Machine Translation

In the previous chapter, we treated statistical machine translation (SMT) as the task of translating a source language sentence F to a target language sentence E, where each element of F and E is assumed to be a word in the source and target languages. However, as noted in Chapter 1, the definition of a "word" is often problematic. In unsegmented languages such as Chinese, Japanese, or Thai, it has been noted that the segmentation standard has a large effect on translation accuracy (Chang et al., 2008). Even for languages with explicit word boundaries, all machine translation systems perform at least some precursory form of tokenization, splitting punctuation and words to prevent the sparsity that would occur if punctuated and non-punctuated words were treated as different entities. Sparsity also manifests itself in a number of other forms, with an extremely large number of rare words existing due to morphological productivity, word compounding, numbers, and proper names. A myriad of methods have been proposed to handle each of these phenomena individually in the context of machine translation, including morphological analysis, stemming, compound breaking, number regularization, optimizing word segmentation, and transliteration, which are outlined in more detail in Section 5.1.

These difficulties stem from the basic premise that we are translating sequences of *words* as our basic unit. On the other hand, (Vilar et al., 2007) examine the possibilities of eschewing the concept of words, treating each sentence as sequences of *characters* to be translated. This method is attractive, as it is theoretically able to handle all sparsity phenomena in a single unified framework, but has only proven feasible between similar lan-

guage pairs such as Spanish-Catalan (Vilar et al., 2007), Swedish-Norwegian (Tiedemann, 2009), and Thai-Lao (Sornlertlamvanich et al., 2008), which have a large number of cognates and a strong co-occurrence between single characters. As (Vilar et al., 2007) and (Xu et al., 2004) state and as is further confirmed here, accurate translations cannot be achieved when applying traditional translation techniques to character-based translation for less similar language pairs.

This chapter proposes improvements to the alignment process tailored to character-based machine translation, and demonstrate that it is, in fact, possible to achieve competitive translation accuracy for distant language pairs using only character strings. This is achieved by adapting the phrase-based alignment method of the previous chapter, applying it not to word strings, but to character strings. As the many-to-many units learned may be at the character, subword, word, or multi-word phrase level, this can be expected to allow for better character alignments than one-to-many alignment techniques, and will also allow for better translation of uncommon words than traditional word-based models by breaking down words into their component parts.

In order to make character-based alignment feasible, two improvements are proposed to the alignment model. The first proposed improvement increases the efficiency of the beam-search technique of (Saers et al., 2009) by augmenting it with look-ahead probabilities in the spirit of A* search. This is important because in the inversion transduction grammar (ITG) framework (Wu, 1997) used in the previous chapter, search is cumbersome for longer sentences, a problem that is further exacerbated when using characters instead of words as the basic unit. The second proposed improvement seeds the search process using counts of all substring pairs in the corpus to bias the phrase alignment model. This is done by defining prior probabilities based on these substring counts within the Bayesian phrasal ITG framework.

The proposed method for character-based translation is evaluated on four language pairs with differing morphological properties. The evaluation shows that for distant language pairs, character-based SMT can achieve translation accuracy that is comparable to word-based systems. In addition, ablation studies show that these results were not possible without the proposed enhancements to the model. Finally, a qualitative analysis shows that the character-based method is able to translate unsegmented text, conjugated words, and proper names in a unified framework with no additional processing.

5.1 Related Work on Lexical Processing in SMT

As traditional SMT systems treat all words as single tokens without considering their internal structure, major problems of data sparsity occur for less frequent tokens. In fact, it has been shown that there is a direct negative correlation between vocabulary size (and thus sparsity) of a language and translation accuracy (Koehn, 2005). Rare words causes trouble for alignment models, both in the form of incorrect alignments, and in the form of garbage collection, where rare words in one language are incorrectly aligned to large segments of the sentence in the other language (Och and Ney, 2003). Unknown words are also a problem during the translation process, and the default approach is to map them as-is into the translated sentence.

This is a major problem in morphologically rich languages such as Finnish and Korean, as well as highly compounding languages such as Dutch and German. Many previous works have attempted to handle morphology, decompounding and regularization through lemmatization, morphological analysis, or unsupervised techniques (Nießen and Ney, 2000; Brown, 2002; Lee, 2004; Goldwater and McClosky, 2005; Talbot and Osborne, 2006; Macherey et al., 2011). Other research has noted that it is more difficult to translate into morphologically rich languages with word-based systems, and methods for modeling target-side morphology have attracted interest in recent years (Bojar, 2007; Subotin, 2011). It is also notable that morphology and compounding remain problematic regardless of the size of the training data, with systems trained on hundreds of millions of words still seeing significant gains in accuracy due to lexical processing (Macherey et al., 2011).

Another major source of rare words in all languages is proper names, which have been handled by using cognates or transliteration to improve translation (Knight and Graehl, 1998; Kondrak et al., 2003; Finch and Sumita, 2007). More sophisticated methods for named entity translation that combine translation and transliteration have also been proposed (Al-Onaizan and Knight, 2002).

Choosing word units is also essential for creating good translation results for languages that do not explicitly mark word boundaries, such as Chinese, Japanese, and Thai. A number of works have addressed this word segmentation problem in translation, mainly focusing on Chinese-to-English translation (Bai et al., 2008; Chang et al., 2008; Zhang et al., 2008b; Chung and Gildea, 2009; Nguyen et al., 2010; Wang et al., 2010), although these works generally assume that a word segmentation exists in one language (English) and attempt to optimize the word segmentation in the other lan-

guage (Chinese).

This enumeration of related works demonstrates the myriad of problems caused by rare words and the large number of proposed solutions to these problems. Character-based translation has the potential to handle all of the phenomena in the previously mentioned research in a single unified framework, requiring no language specific tools such as morphological analyzers or word segmenters. However, while the approach is attractive conceptually, previous research has only been shown effective for closely related language pairs (Vilar et al., 2007; Tiedemann, 2009; Sornlertlamvanich et al., 2008). This work proposes effective alignment and decoding techniques that allow character-based translation to achieve accurate translation results for both close and distant language pairs.

5.2 Look-Ahead Biparsing

In order to perform many-to-many alignment, we represent our target and source sentences as E and F. e_i and f_j represent single elements of the target and source sentences, respectively. These may be words in word-based alignment models or single characters in character-based alignment models.[1] We define our alignment as A, where each element is a span $a_k = \langle s, t, u, v \rangle$ indicating that the target string $\boldsymbol{e}_s^t = e_s, \ldots, e_t$ and source string $\boldsymbol{f}_u^v = f_u, \ldots, f_v$ are alignments of each other.

These alignments can be acquired using the alignment method presented in the previous chapter. As this method has been shown to achieve competitive accuracy with a much smaller phrase table than traditional methods, it is particularly well suited for character-based translation as we would like to use phrases that contain large numbers of characters without creating a phrase table so large that it cannot be used in actual decoding. In this framework, blocked sampling is performed by processing sentences in a corpus one-by-one, acquiring a sample for each sentence by first performing bottom-up biparsing to create a chart of probabilities, then performing top-down sampling of a new tree based on the probabilities in this chart.

We define a chart as a data structure with a single cell for each alignment $a_{s,t,u,v}$ spanning $\boldsymbol{e}_s^t$ and $\boldsymbol{f}_u^v$. Each cell has an accompanying "inside" probability $I(a_{s,t,u,v})$. This probability is the combination of the generative probability of each phrase pair $P_t(\boldsymbol{e}_s^t, \boldsymbol{f}_u^v)$ as well as the sum of the

[1] Some previous work has also performed alignment using morphological analyzers to normalize or split the sentence into morpheme streams (Corston-Oliver and Gamon, 2004).

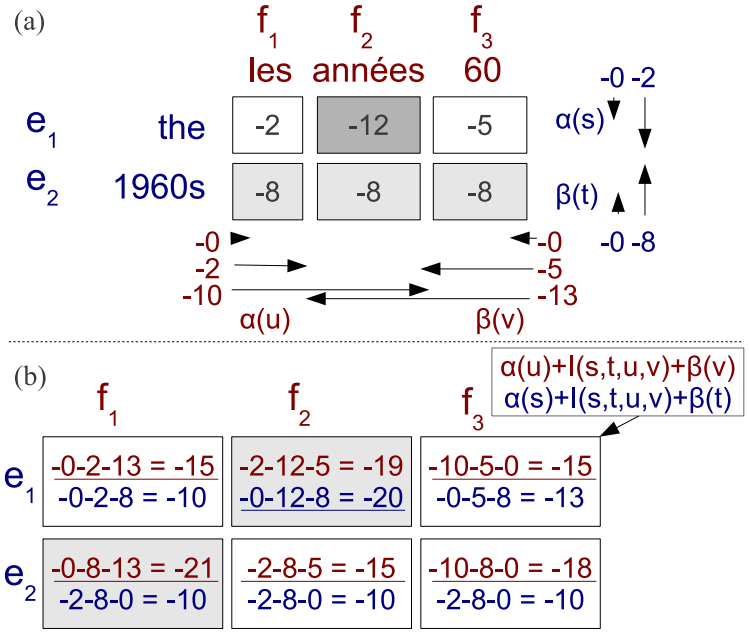

Figure 5.1: (a) A chart with inside log probabilities in boxes and forward/backward log probabilities marking surrounding arrows. (b) Spans with corresponding look-aheads added, and the minimum probability underlined. Lightly and darkly shaded spans will be trimmed when the beam is $\log(P) \geq -3$ and $\log(P) \geq -6$ respectively.

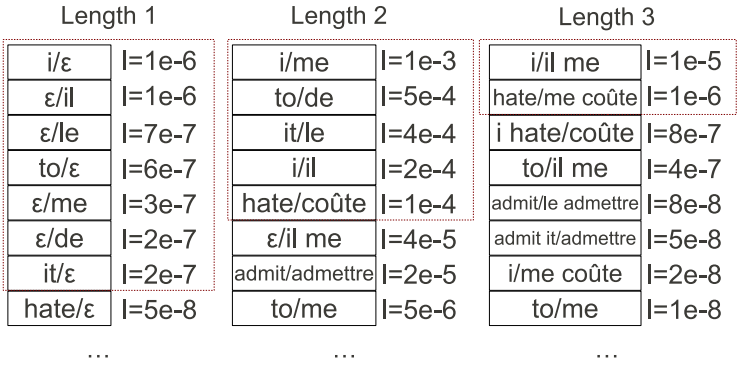

Figure 5.2: An example of the first three queues used in ITG parsing along with their inside probabilities. The hypotheses that would be processed if the beam is set to $c = 1e - 1$ are surrounded by boxes.

probabilities over all shorter spans in straight and inverted order

$$\begin{aligned} I(a_{s,t,u,v}) =& P_t(e_s^t, f_u^v) \\ &+ \sum_{s \le S \le t} \sum_{u \le U \le v} P_x(x = \text{str}) I(a_{s,S,u,U}) I(a_{S,t,U,v}) \\ &+ \sum_{s \le S \le t} \sum_{u \le U \le v} P_x(x = \text{inv}) I(a_{s,S,U,v}) I(a_{S,t,u,U}) \end{aligned} \tag{5.1}$$

where $P_x(x = \text{str})$ and $P_x(x = \text{inv})$ are the probability of straight and inverted ITG productions. An example of part of the chart used in this bottom-up parsing can be found in Figure 5.1 (a), where we show the cells that have one-to-one alignments.

The exact calculation of these probabilities can be performed in $O(n^6)$ time, where $n = \max(I, J)$ is the length of the longer of $\boldsymbol{e}_1^I$ and $\boldsymbol{f}_1^J$ (Wu, 1997). This calculation is performed using a dynamic programming algorithm that separates each of the spans into queues based on their length $l = t - s + u - v$, and queues are processed in ascending order of l. An example of the queues for the first three lengths is shown in Figure 5.2.

The motivation behind this algorithm is that when calculating a particular span's inside probability $I(a_{s,t,u,v})$ according to Equation (5.1), all of the other inside spans that we reference on the right-hand side of the equation are shorter than $a_{s,t,u,v}$ itself. Thus, if we process all spans in ascending order, it is simple to calculate these sums for every span in the chart. The

computational complexity of the algorithm is $O(n^6)$ because Equation (5.1) must be calculated for all of the $O(n^4)$ spans in the sentence, and there are $O(n^2)$ elements in each calculation of the sum.

However, exact computation of these probabilities in $O(n^6)$ time is impractical for all but the shortest sentences. Thus it is necessary to use methods to reduce the search space such as beam-search-based chart parsing (Saers et al., 2009) or slice sampling (Blunsom and Cohn, 2010).

(Saers et al., 2009) note that in order to increase the efficiency of processing, queues can be trimmed based on a fixed histogram beam, only processing the b hypotheses with the highest probability for each queue. Here, we instead utilize a probability beam, expanding only hypotheses that are more than c times as likely as the best hypothesis $\hat{a}$. In other words, we have a queue discipline based on the inside probability, and all spans a_k where $I(a_k) < cI(\hat{a})$ are pruned. c is a constant between 0 and 1 describing the width of the beam, and a smaller constant probability will indicate a wider beam. Figure 5.2 shows an example of this, with boxes surrounding part of each queue showing the hypotheses that fall within the beam when $c = 10^{-1}$.

While this pruning increases the speed of alignment significantly, this method is insensitive to the existence of competing hypotheses when performing pruning. Figure 5.1 (a) provides an example of what a competing hypothesis is, and why it is unwise to ignore them during pruning. Particularly, the alignments "les/1960s" and "les/the" both share the word "les," and thus cannot both exist in a single derivation according to the ITG framework. We will call hypotheses that are mutually exclusive in this manner *competing* hypotheses. As the probability of "les/1960s" is much lower than its competing hypothesis "les/the," it is intuitively unlikely, and thus a good candidate for pruning. However its inside probability is the same as that of "années/1960s," which has no competing hypotheses and thus should not be removed from consideration. This section proposes the use of a look-ahead probability to increase the efficiency of this chart parsing by considering competing hypotheses.

In order to take into account competing hypotheses, we can use for our queue discipline not only the inside probability $I(a_k)$, but also the outside probability $O(a_k)$, the probability of generating all spans other than a_k, as in A* search for CFGs (Klein and Manning, 2003), and tic-tac-toe pruning for word-based ITGs (Zhang and Gildea, 2005). As the calculation of the actual outside probability $O(a_k)$ is just as expensive as parsing itself, it is necessary to approximate this with heuristic function O^* that can be calculated efficiently.

This section proposes a heuristic function that is designed specifically for phrasal ITGs and is computable with worst-case complexity of n^2, compared with the n^3 amortized time of the tic-tac-toe pruning algorithm described by (Zhang et al., 2008a). During the calculation of the phrase generation probabilities P_t, we save the best probability O^* for each monolingual span.

$$O_e^*(s,t) = \max_{\{\tilde{a}=\langle \tilde{s},\tilde{t},\tilde{u},\tilde{v}\rangle;\tilde{s}=s,\tilde{t}=t\}} P_t(\tilde{a}) \tag{5.2}$$

$$O_f^*(u,v) = \max_{\{\tilde{a}=\langle \tilde{s},\tilde{t},\tilde{u},\tilde{v}\rangle;\tilde{u}=u,\tilde{v}=v\}} P_t(\tilde{a}) \tag{5.3}$$

For each language independently, we calculate forward probabilities α and backward probabilities β. For example, $\alpha_e(s)$ is the maximum probability of the span $(0, s)$ of e that can be created by concatenating together consecutive values of O_e^*:

$$\alpha_e(s) = \max_{\{S_1,\ldots,S_x\}} O_e^*(0,S_1)O_e^*(S_1,S_2)\ldots O_e^*(S_x,s). \tag{5.4}$$

Backwards probabilities and probabilities over f can be defined similarly. These probabilities are calculated for e and f independently, and can be calculated in n^2 time by processing each α in ascending order, and each β in descending order in a fashion similar to that of the forward-backward algorithm. Finally, for any span, we define the outside heuristic as the minimum of the two independent look-ahead probabilities over each language

$$O^*(a_{s,t,u,v}) = \min(\alpha_e(s) * \beta_e(t), \alpha_f(u) * \beta_f(v)). \tag{5.5}$$

Taking a look again at the example in 5.1 (b), it can be seen that the relative probability difference between the highest probability span "les/the" and the spans "années/1960s" and "60/1960s" decreases, allowing for tighter beam pruning without losing these good hypotheses. In contrast, the relative probability of "les/1960s" remains low as it is in conflict with a high-probability alignment, allowing it to be discarded.

5.3 Substring Prior Probabilities

While the Bayesian ITG framework uses the previously mentioned phrase distribution P_t during search, it also allows for definition of a phrase pair prior probability $P_{base}(\boldsymbol{e}_s^t, \boldsymbol{f}_u^v)$ through the base measure. This can help efficiently seed the search process with a bias towards phrase pairs that satisfy certain properties.

As noted in Section 4.3.1 this can be achieved by using the IBM Model 1 probabilities, which can be efficiently calculated using the dynamic programming algorithm described by (Brown et al., 1993). However, for reasons previously stated, these methods are less satisfactory when performing character-based alignment, as the amount of information contained in a character does not allow for proper alignment.

Instead, this work proposes a method for using raw substring co-occurrence statistics to bias alignments towards substrings that often co-occur in the entire training corpus. This is similar to the method of (Cromieres, 2006), but instead of using these co-occurrence statistics as a heuristic alignment criterion, they are incorporated as a prior probability in a statistical model that can take into account mutual exclusivity of overlapping substrings in a sentence.

We define this prior probability using three counts over substrings $c_{\boldsymbol{e}}$, $c_{\boldsymbol{f}}$, and $c_{\langle \boldsymbol{e},\boldsymbol{f} \rangle}$. $c_{\boldsymbol{e}}$ and $c_{\boldsymbol{f}}$ count the total number of sentences in which the substrings $\boldsymbol{e}$ and $\boldsymbol{f}$ occur, respectively. $c_{\langle \boldsymbol{e},\boldsymbol{f} \rangle}$ is a count of the total number of sentences in which the substring $\boldsymbol{e}$ occurs on the target side, and $\boldsymbol{f}$ occurs on the source side. We can perform the calculation of these statistics using enhanced suffix arrays, a data structure that can efficiently calculate all substrings in a corpus (Abouelhoda et al., 2004).[2]

While suffix arrays allow for efficient calculation of these statistics, storing all co-occurrence counts $c_{\langle \boldsymbol{e},\boldsymbol{f} \rangle}$ is an unrealistic memory burden for larger corpora. In order to reduce the amount of memory used, each count is discounted fixed value d, which is set to 5. This has a dual effect of reducing the amount of memory needed to hold co-occurrence counts by removing values for which $c_{\langle \boldsymbol{e},\boldsymbol{f} \rangle} < d$, as well as helping to prevent over-fitting the training data. In addition, we can heuristically prune values for which the conditional probabilities $P(\boldsymbol{e}|\boldsymbol{f})$ or $P(\boldsymbol{f}|\boldsymbol{e})$ are less than some fixed value, which is set to 0.1 for the reported experiments.

Preliminary experiments designed to determine how to combine $c_{\boldsymbol{e}}$, $c_{\boldsymbol{f}}$, and $c_{\langle \boldsymbol{e},\boldsymbol{f} \rangle}$ into prior probabilities tested a number of methods proposed by previous research including plain co-occurrence counts, the Dice coefficient, and Chi-squared statistics (Cromieres, 2006), as well as a new method of defining substring pair probabilities to be proportional to bi-directional con-

[2]Using the open-source implementation esaxx `http://code.google.com/p/esaxx/`

ditional probabilities

$$P_{cooc}(\boldsymbol{e}, \boldsymbol{f}) = P_{cooc}(\boldsymbol{e}|\boldsymbol{f})P_{cooc}(\boldsymbol{f}|\boldsymbol{e})/Z \tag{5.6}$$

$$= \left(\frac{c_{\langle \boldsymbol{e},\boldsymbol{f}\rangle} - d}{c_{\boldsymbol{f}} - d}\right)\left(\frac{c_{\langle \boldsymbol{e},\boldsymbol{f}\rangle} - d}{c_{\boldsymbol{e}} - d}\right)/Z \tag{5.7}$$

for all substring pairs where $c_{\langle \boldsymbol{e},\boldsymbol{f}\rangle} > d$ and where Z is a normalization term equal to

$$Z = \sum_{\{\boldsymbol{e},\boldsymbol{f};c_{\langle \boldsymbol{e},\boldsymbol{f}\rangle}>d\}} P_{cooc}(\boldsymbol{e}|\boldsymbol{f})P_{cooc}(\boldsymbol{f}|\boldsymbol{e}). \tag{5.8}$$

The motivation for combining the probabilities in this fashion is similar to that of the base measure in Equation (4.6), finding highly reliable alignments that are supported by both models. The preliminary experiments showed that the bi-directional conditional probability method gave significantly better results than all other methods, so this method will be adopted for the remainder of the experiments.

It should be noted that as we are using discounting, many substring pairs will be given zero probability according to P_{cooc}. As the prior is only supposed to bias the model towards good solutions and not explicitly rule out any possibilities, we can instead linearly interpolate the co-occurrence probability with the one-to-many Model 1 probability, which will give at least some probability mass to all substring pairs

$$P_{base}(\boldsymbol{e}, \boldsymbol{f}) = \lambda P_{cooc}(\boldsymbol{e}, \boldsymbol{f}) + (1 - \lambda)P_{m1}(\boldsymbol{e}, \boldsymbol{f}). \tag{5.9}$$

In order to find an appropriate value, we put a Beta prior ($\alpha = 1$, $\beta = 1$) on the interpolation coefficient λ and learn it during training.

5.4 Experiments

This section describes experiments over a variety of language pairs designed to test the effectiveness of character-based translation.

5.4.1 Experimental Setup

Evaluation was performed on a combination of four languages with English, using freely available data. The first three languages, French-English, German-English, and Finnish-English, used data from EuroParl (Koehn, 2005), with development and test sets designated for the 2005 ACL shared

Table 5.1: The number of words in each corpus for TM and LM training, tuning, and testing.

	de-en	fi-en	fr-en	ja-en
TM (en)	2.80M	3.10M	2.77M	2.13M
TM (other)	2.56M	2.23M	3.05M	2.34M
LM (en)	16.0M	15.5M	13.8M	11.5M
LM (other)	15.3M	11.3M	15.6M	11.9M
Tune (en)	58.7k	58.7k	58.7k	30.8k
Tune (other)	55.1k	42.0k	67.3k	34.4k
Test (en)	58.0k	58.0k	58.0k	26.6k
Test (other)	54.3k	41.4k	66.2k	28.5k

task on machine translation.[3] Experiments were also performed with Japanese-English Wikipedia articles from the Kyoto Free Translation Task[4] using the designated training and tuning sets, and reporting results on the test set. These languages were chosen as they have a variety of interesting characteristics. French has some level of inflection, but among the test languages has the strongest one-to-one correspondence with English, and is generally considered easy to translate. German has many compound words, which must be broken apart to translate properly into English. Finnish is an agglutinative language with extremely rich morphology, resulting in long words and the largest vocabulary of the languages in EuroParl. Japanese does not have any clear word boundaries, and uses logographic characters, which contain more information than phonetic characters.

With regards to data preparation, the EuroParl data was pre-tokenized, so the experiments simply used the tokenized data as-is for the training and evaluation of all models. For word-based translation in the Kyoto task, training was performed using the provided tokenization scripts. For character-based translation, no tokenization was performed, using the original text for both training and decoding. For both tasks, all sentences for which both source and target were 100 characters or less were selected as training data, the total size of which is shown in Table 5.1. In character-based translation, white spaces between words were treated as any other character and not given any special treatment. Evaluation was performed on tokenized and lower-cased data.

For alignment, GIZA++ was used as an implementation of one-to-many

[3] `http://www.statmt.org/wpt05/mt-shared-task/`.

[4] `http://www.phontron.com/kftt/`.

alignment and pialign was used as an implementation of the ITG models[5] modified with the proposed improvements. For GIZA++, the default settings were used for word-based alignment, but the HMM model was used for character-based alignment to allow for alignment of longer sentences. For pialign, default settings were used except for character-based ITG alignment, which used a probability beam of 10^{-4} instead 10^{-10}. Decoding was performed with the Moses decoder,[6] with the default settings except for the stack size, which was set to 1000 instead of 200. Minimum error rate training was performed to maximize word-based BLEU score for all systems.[7] For language models, word-based translation used a word 5-gram model, and character-based translation used a character 12-gram model, both smoothed using interpolated Kneser-Ney smoothing.

5.4.2 Quantitative Evaluation

This section presents a quantitative analysis of the translation results for each of the proposed methods. As previous research has shown that it is more difficult to translate into morphologically rich languages than into English (Koehn, 2005), experiments are performed to test the accuracy translating in both directions for all language pairs. Translation quality is evaluated using BLEU score (Papineni et al., 2002), both on the word and character level, as well as METEOR (Denkowski and Lavie, 2011) on the word level.

Table 5.2 shows the results of the evaluation. It can be seen that in general, character-based translation with all of the proposed alignment improvements greatly exceeds character-based translation using the IBM models, confirming the hypothesis that substring-based information is necessary for accurate alignments. In general, the accuracy of character-based translation is comparable or slightly inferior to that of word-based translation. The evaluation of character-based BLEU shows that character-based translation is superior, comparable, or inferior based on the language pair, word-based METEOR shows that character-based translation is comparable or inferior, and word-based BLEU shows that character-based translation is inferior.

For translation into English, character-based translation achieves higher relative accuracy compared to word-based translation on Japanese and Finnish

[5] `http://phontron.com/pialign/`

[6] `http://statmt.org/moses/`

[7] This setup was chosen to minimize the effect of tuning criterion on the comparison between the baseline and the proposed system, although it does indicate that we must have access to tokenized data for the development set.

Table 5.2: Translation results in word-based BLEU (wBLEU), character-based BLEU (cBLEU), and METEOR for the GIZA++ and ITG models for word and character-based translation, with bold numbers indicating a statistically insignificant difference from the best system according to the bootstrap resampling method at $p = 0.05$.

	de-en			en-de		
	wBLEU	cBLEU	METEOR	wBLEU	cBLEU	METEOR
GIZA-word	**24.58**	64.28	30.43	**17.94**	62.71	**37.88**
ITG-word	23.87	**64.89**	**30.71**	17.47	**63.18**	**37.79**
GIZA-char	08.05	45.01	15.35	06.17	41.04	19.90
ITG-char	21.79	64.47	30.12	15.35	61.95	35.45

	fi-en			en-fi		
	wBLEU	cBLEU	METEOR	wBLEU	cBLEU	METEOR
GIZA-word	20.41	60.01	27.89	**13.22**	58.50	**27.03**
ITG-word	**20.83**	61.04	28.46	**13.12**	**59.27**	**27.09**
GIZA-char	06.91	41.62	14.39	04.58	35.09	11.76
ITG-char	18.38	**62.44**	**28.94**	12.14	**59.02**	25.31

	fr-en			en-fr		
	wBLEU	cBLEU	METEOR	wBLEU	cBLEU	METEOR
GIZA-word	**30.23**	**68.79**	34.20	**32.19**	69.20	**52.39**
ITG-word	**29.92**	68.64	**34.29**	31.66	**69.61**	51.98
GIZA-char	11.05	48.23	17.80	10.31	42.84	25.06
ITG-char	26.70	66.76	32.47	27.74	67.44	48.56

	ja-en			en-ja		
	wBLEU	cBLEU	METEOR	wBLEU	cBLEU	METEOR
GIZA-word	**17.95**	56.47	**24.70**	**20.79**	27.01	**38.41**
ITG-word	17.14	56.60	**24.89**	**20.26**	**28.34**	**38.34**
GIZA-char	09.46	49.02	18.34	01.48	00.72	06.67
ITG-char	15.84	**58.41**	24.58	17.90	**28.46**	35.71

Table 5.3: METEOR scores for alignment with and without look-ahead and co-occurrence priors.

	fi-en	en-fi	ja-en	en-ja
ITG +cooc +look	**28.94**	**25.31**	**24.58**	**35.71**
ITG +cooc -look	28.51	24.24	**24.32**	**35.74**
ITG -cooc +look	**28.65**	24.49	**24.36**	35.05
ITG -cooc -look	27.45	23.30	23.57	34.50

input, followed by German, and finally French. This is notable in that it confirms the fact that character-based translation is performing well on languages that have long words or ambiguous boundaries, and less well on language pairs with relatively strong one-to-one correspondence between words.

5.4.3 Effect of Alignment Method

This section compares the translation accuracy for character-based translation using the ITG model with and without the proposed improvements of substring co-occurrence priors and look-ahead parsing as described in Sections 5.2 and 5.3.

METEOR scores for experiments translating Japanese and Finnish are shown in Table 5.3. It can be seen that the co-occurrence prior probability gives gains in all cases, indicating that the using substring statistics over the whole corpus are providing effective prior knowledge to the ITG aligner. The introduced look-ahead probabilities improve accuracy significantly when substring co-occurrence counts are not used, and slightly when co-occurrence counts are used. More importantly, they allow for more aggressive beam pruning, increasing sampling speed from 1.3 sent/s to 2.5 sent/s on the Finnish task, and 6.8 sent/s to 11.6 sent/s on the Japanese task.

5.4.4 Qualitative Evaluation

This section presents the results of a subjective evaluation of Japanese-English and Finnish-English translations. In the evaluation, two raters evaluated 100 sentences each, assigning an adequacy score of 0-5 based on how well the translation conveys the information contained in the reference translation. The raters were asked to rate on shorter sentences of 8-16 English words to ease rating and interpretation. The results of this evaluation are in Table 5.4. It can be seen that the results are comparable, with no significant

Table 5.4: A human adequacy evaluation of word and character-based machine translation (0-5 scale).

	fi-en	ja-en
ITG-word	2.851	2.085
ITG-char	2.826	2.154

Table 5.5: The major gains of character-based translation.

Unknown (13/26)	Ref:	directive on equality
	Word:	tasa-arvodirektiivi
	Char:	equality directive
Hyphen (5/26)	Ref:	yoshiwara-juku station
	Word:	yoshiwara no eki
	Char:	yoshiwara-juku station
Uncommon (5/26)	Ref:	world health organisation
	Word:	world health
	Char:	world health organisation

difference in average scores for either language pair.

A breakdown of the sentences for which character-based translation was given a score of at least two points more than word-based is shown in Table 5.5. It can be seen that character-based translation is, in fact, properly handling a number of sparsity phenomena. On the other hand, word-based translation was generally stronger with reordering and lexical choice of more common words.

5.4.5 Phrases Used in Translation

This section presents the results of an analysis of the phrases used in the translation of 50 sentences by the systems created using word and character-

Table 5.6: The number of phrases that were the same, different but of equal quality, or subjectively better translations in one of the two models.

	fi-en	ja-en
Same phrase	220	215
Equal quality	209	217
ITG-char better	67	96
ITG-word better	35	69

based ITG alignment for the Finnish-English and Japanese-English tasks. First, Table 5.6 shows the number of phrases where the phrase used by one of the two systems was subjectively better than the phrase used by the other system. It can be seen that there are a greater number of accurate translations at the phrase level for the character-based system than for the word based system across both languages.

In order to examine the types of phrases where one of the two systems is more accurate than the other, Table 5.7 and Table 5.8 provide more detailed break-downs by type of the mistranslated phrases used by each of the models for Finnish-English and Japanese-English translation respectively. It can be seen that character-based translation naturally handles a number of phenomena due to unknown words that are not handled by word-based systems, such as those requiring transliteration, decompounding, and division of morphological components. It should also be noted that this process is not perfect, there are a number of cases where character-based translation splits or transliterates words that would be more accurately translated as a whole, although the total number of correctly translated compounds and inflected words is more than twice the number of incorrectly translated ones.

With regards to Finnish-English, it is interesting to note that character-based translation also succeeded in discovering a number of inflectional suffixes that have a clear grammatical function in the language (Karlsson, 1999). Examples of the most common subword units used in translation are shown in Table 5.9. It can be seen that all but one have a clear grammatical function in Finnish. The only exception "s" is used in the transliteration of unknown words, as well as part of some morphological paradigms (similarly to "e"). This demonstrates that despite using no sort of explicit morphological knowledge, character-based translation is able to handle, to some extent, the more common morphological paradigms in morphologically rich languages.

One significant area for improvement in the character-based model is that it has a tendency to create alignments of actual content words on the source side to the white space character on the target side, effectively deleting content words. While deleted words are a problem in the word-based model as well, the problem is more prevalent in the character-based model, so it will be worth examining the possibilities of giving space characters a special status in the translation model in the future.

Finally, it is important to note that the character-based model helps with not only unknown words, but also words that do exist in the training corpus, but are mis-aligned by the word-based model because they are rare, or do not have a consistent translation. In fact, this was the single most

Table 5.7: Examples of phrases where one system performed better for Finnish-English translation.

ITG-char Better

Freq.	Type	Source (Case)	Reference	ITG-char	ITG-word
19	Misalignment	itsenäisille	independent (pos./all.)	independent for	economic reform
18	Conjugation	perustuslaillisempi	constitution (comp.)	constitution more	perustuslaillisempi
12	Word Deletion	kuuluisi	belong (cond.)	would include	would
12	Compounding	yleismietintöä	the general report	the general report ,	yleismietintöä
10	Word Insertion	myös	also	also	will also be
8	Lexical Choice	pelkojen	fears/emotions	fears	emotions

ITG-word Better

Freq.	Type	Source (Case)	Reference	ITG-char	ITG-word
19	Word Deleted	itsellemme	ourselves (all.)	*space*	ourselves
15	Word Inserted	haluan	i would like to	i would like to make	i would like to
6	Lexical Choice	jo	in/already	already	in
5	Misalignment	on vastattava	is answer	must be	is answer
4	False Conjugation	vertailuanalyysiä	benchmarking	comparative analysis	benchmarking

Table 5.8: Examples of phrases where one system performed better for Japanese-English translation.

ITG-char Better

Freq.	Type	Source	Reference	ITG-char	ITG-word
38	Transliteration	希玄	kigen	kigen	希玄
19	Word Insertion	半年間	six months	half year	six months between
19	Word Deletion	病のため	due to illness	due to his illness	illness
17	Misalignment	を求めて	seeking	seeking	the
2	Lexical Choice	俗	lay/commonly called	lay	commonly called
2	Compound	顔洗い	face washing	face washing	顔洗い

ITG-word Better

Freq.	Type	Source	Reference	ITG-char	ITG-word
28	Word Deletion	も用いた。	was also used .	.	was also used .
11	Word Insertion	招請	invited	invited cont	invited
11	False Translitertaion	無常	vanity	mujo	vanity
10	Misalignment	書かれた	written	the	written
5	Partial Transliteration	大佛	osaragi	os	osaragi
2	Lexical Choice	で	in/and	and	in

Table 5.9: Examples of common Finnish subword phrases along with their grammatical function.

String	Freq.	Grammatical Function
n	564	genitive (“of ∼”)
a	467	partitive (“some ∼”)
i	307	plural, non-nominative (“∼ s”)
t	241	plural, nominative (“∼ s”)
sta	235	elative (“out of ∼”)
e	156	similar to “e” in “play**e**d”
lle	134	allative (“onto ∼”)
s	133	-
ä	121	partitive (“some ∼”)
in	114	plural, genitive (“of ∼ s”)
ssa	94	inessive (“in ∼”)

common error category for Finnish-English, and a significant portion of the Japanese-English errors. This indicates that simply applying character-based methods to process unknown words will not be sufficient to overcome the sparsity issues of the word-based model.

5.5 Conclusion

This chapter demonstrated that character-based translation is able to act as a unified framework for handling a number of difficult problems in translation: morphology, compound words, transliteration, and word segmentation. It also presented two advances to many-to-many alignment methods that allow them to be run on much longer sentences.

One of the major challenges for the future is the development of efficient decoding methods for character-based translation models. As shown in the analysis of phrase quality in the system, the character-based model is able to produce better translations on the phrase level, but nevertheless achieves results that are approximately equal to those of the word-based systems. One of the major reasons for this gap is that the word-based model tends to be better at reordering, as it is able to treat whole words as single units, which gives it more freedom in reordering. Given more effective and efficient decoding methods, it is likely that we will be able to further close this gap in reordering quality, resulting in a clear advantage of the character-based models over word-based models.

Chapter 6

Conclusion

This thesis has concerned itself with the fundamental problem of learning lexical units for practical tasks. The results presented here give new tools and new motivation for further studies of the role of the lexicon in language processing systems.

The new tools come in the form of models for lexical learning and the inference algorithms that are used to learn these models. For speech recognition, this is a model of statistical word segmentation, with the accompanying finite-state representation and inference algorithm for use on continuous speech. For machine translation, this is a hierarchical model of many-to-many alignment with its accompanying inference algorithm using efficient search over inversion transduction grammar (ITG) parses, which can also be applied to learning lexical units.

The new motivation comes from experimental results demonstrating that fully unsupervised lexical learning is not only possible, but also useful when measured by extrinsic measures over the tasks at hand. In particular, the works here have shown that unsupervised learning has the potential to provide one solution to the data bottleneck and the problem of choosing segmentation standards appropriate for the task at hand.

With regards to the data bottleneck, the work on speech recognition showed that language model learning is possible without text, removing the necessity to prepare text resources for language model learning. The work on machine translation demonstrated that it is possible to perform translation using only characters, allowing the handling of unsegmented or morphologically rich languages without preparing manually segmented or morphologically analyzed data.

In addition, for both machine translation and speech recognition, experiments demonstrate that unsupervised learning is able to find units that are

useful for practical applications, often matching or surpassing those used by supervised systems. This is due to the fact that unsupervised methods are able to automatically adjust the length of the unit used, using longer units consisting of multiple words when there is enough data to allow for the robust estimation of the corresponding statistics, and shorter units consisting of characters or subwords for less common words that suffer from problems of data sparsity. It is also significant that lexical learning in such a manner affords new insights into the basic units that we are using for language processing systems such as speech recognition and machine translation, and these insights can potentially be integrated into more traditional supervised lexical processing systems.

6.1 Future Work

Each chapter of this thesis has provided a brief look at extensions to the individual techniques presented therein. This section wraps up the thesis with a look at the broad, over-reaching challenges that still remain in the area.

6.1.1 Use of Prosodic or Textual Boundaries

All of the works presented here concern themselves with learning lexicons from strings of symbols, either phonemes in speech, or characters in text. However, speech is not simply a string of phonemes, and while text may generally be considered as strings of characters, all characters do not necessarily merit the same treatment.

In addition to its phonemic content, speech is also rich in social and contextual signals that may be able to help learning which lexical units to use in recognition in a more robust and generalizable fashion. For example, studies on human language acquisition have shown that while children are heavily influenced by the statistical regularities in the phoneme stream that are used in this work (Saffran et al., 1996), changes in tone, stress, and other prosodic factors also play an important role in lexical learning (Jusczyk et al., 1999).

With regards to learning lexical units from text, while the methods presented here were agnostic to explicit word boundaries delimited by spaces, these spaces certainly provide clues about the boundaries of meaningful units that can be exploited in the translation and alignment processes. In addition, it has been shown that for languages such as Chinese, which do

not contain specific boundaries, punctuation can provide a strong indicator in their place (Li and Sun, 2009).

While this thesis has shown that these indicators are not explicitly necessary to achieve some degree of lexical learning, a promising future direction is to explore the middle ground between affording them blind trust and ignoring them completely.

6.1.2 Learning Morphological Patterns

This thesis has handled the problem of lexical acquisition as a task of separating the character string into appropriate word units. However, for most languages in the world there are significant morphological regularities. Going back to the example of the word "uninspiring" provided in the introduction, we can see that the word stem "inspir" has possible conjugations "e," "ing," and "ed," among others. This common property is shared by any number of other words such as "invite," "relate," and "abide."

As mentioned in Section 1.2, the automatic discovery of such morphological patterns has received some treatment in the fields of psychology and computational linguistics. There has also been some research on learning the segmentation of morphemes across languages (Snyder and Barzilay, 2008; Naradowsky and Toutanova, 2011), and even some work on learning the splitting of compound words for machine translation using morphological operations such as deletion and replacement (Garera and Yarowsky, 2008; Macherey et al., 2011), but there is still no systematic treatment of more complex operations such as learning conjugation paradigms for verbs. This is also true for language modeling for speech recognition, with some effort being made to segment text in morphologically rich languages with both supervised (Afify et al., 2006) and unsupervised methods (Hirsimäki et al., 2006), but little to cluster or utilize patterns to improve segmentation results.

6.1.3 Learning on Large Scale

One final challenge for the future is the scaling of learning to larger data, which has been shown repeatedly to provide superior results with no change in algorithms (Halevy et al., 2009). While each of the chapters did speak to speed improvements in their current methods, the overall process of blocked Gibbs sampling is still relatively slow compared to other simpler methods (Li and Sun, 2009; Zhikov et al., 2010).

Fortunately, with the proliferation of computing clusters, parallel processing has allowed for large-scale learning, even for more complicated struc-

tured models such as the ones presented in this thesis (McDonald et al., 2010). While Gibbs sampling is not trivially parallelizable, there are a number of reports that parallelizing learning can afford great increases in speed with little to no sacrifice in accuracy (Newman et al., 2007; Asuncion et al., 2008). Using these techniques to scale up to web scale data will likely afford more robust, more accurate acquisition of a wider variety of lexical phenomena, allowing for improvements in system accuracy and a more complete insight into the nature of lexical knowledge that must be used therein.

Bibliography

Steven Abney and Steven Bird. 2010. The human language project: Building a universal corpus of the world's languages. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Mohamed I. Abouelhoda, Stefan Kurtz, and Enno Ohlebusch. 2004. Replacing suffix trees with enhanced suffix arrays. *Journal of Discrete Algorithms*, 2(1).

Mohamed Afify, Ruhi Sarikaya, Hong-Kwang Jeff Kuo, Laurent Besacier, and Yuqing Gao. 2006. On the use of morphological analysis for dialectal arabic speech recognition. In *Ninth International Conference on Spoken Language Processing*.

Yuya Akita and Tatsuya Kawahara. 2010. Statistical transformation of language and pronunciation models for spontaneous speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(6).

Yaser Al-Onaizan and Kevin Knight. 2002. Translating named entities using monolingual and bilingual resources. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Arthur Asuncion, Padhraic Smyth, and Max Welling. 2008. Asynchronous distributed learning of topic models. In *Proceedings of the 22nd Annual Conference on Neural Information Processing Systems (NIPS)*.

Ming-Hong Bai, Keh-Jiann Chen, and Jason S. Chang. 2008. Improving word alignment by adjusting Chinese word segmentation. In *Proceedings of the 3rd International Joint Conference on Natural Language Processing (IJCNLP)*.

Thomas Bayes and Richard Price. 1763. An essay towards solving a problem in the doctrine of chances. *Philosophical Transactions*, 53:370.

Issam Bazzi and James Glass. 2001. Learning units for domain-independent out-of-vocabulary word modelling. In *Proceedings of the 7th European Conference on Speech Communication and Technology (EuroSpeech)*.

Matthew J. Beal. 2003. *Variational algorithms for approximate Bayesian inference*. Ph.D. thesis, University College London.

Kenneth R. Beesley. 1996. Arabic finite-state morphological analysis and generation. In *Proceedings of the 21th International Conference on Computational Linguistics (COLING)*.

Frédéric Bimbot, Roberto Pieraccini, Esther Levin, and Bishnu Atal. 1995. Variable-length sequence modeling: Multigrams. *Signal Processing Letters, IEEE*, 2(6).

Christopher M. Bishop. 2006. *Pattern recognition and machine learning*. Springer New York.

Phil Blunsom and Trevor Cohn. 2010. Inducing synchronous grammars with slice sampling. In *Proceedings of the Human Language Technology: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Phil Blunsom, Trevor Cohn, Chris Dyer, and Miles Osborne. 2009. A Gibbs sampler for phrasal synchronous grammar induction. In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Ondřej Bojar. 2007. English-to-Czech factored machine translation. In *Proceedings of the 2nd Workshop on Statistical Machine Translation (WMT)*.

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

Peter F. Brown, Vincent J.Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19.

Ralf D. Brown. 2002. Corpus-driven splitting of compound words. In *Proc. TMI*.

Chris Callison-Burch, Philipp Koehn, Christof Monz, Kay Peterson, Mark Przybocki, and Omar F. Zaidan. 2010. Findings of the 2010 joint workshop on statistical machine translation and metrics for machine translation. In *Proceedings of the Joint 5th Workshop on Statistical Machine Translation (WMT) and MetricsMATR*.

Marine Carpuat and Dekai Wu. 2005. Word sense disambiguation vs. statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*.

Chao-Huang Chang and Cheng-Dur Chen. 1993. Hmm-based part-of-speech tagging for Chinese corpora. In *Proceedings of the Workshop on Very Large Corpora*, pages 40–47.

Pi-Chuan Chang, Michel Galley, and Christopher D. Manning. 2008. Optimizing Chinese word segmentation for machine translation performance. In *Proceedings of the 3rd Workshop on Statistical Machine Translation (WMT)*.

Stanley F. Chen and Joshua Goodman. 1996. An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Colin Cherry and Dekang Lin. 2007. Inversion transduction grammar for joint phrasal translation modeling.

David Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2).

Noam Chomsky. 1956. Three models for the description of language. *IRE Transactions on Information Theory*, 2(3).

Tagyoung Chung and Daniel Gildea. 2009. Unsupervised tokenization for machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Shay B. Cohen, David M. Blei, and Noah A. Smith. 2010. Variational inference for adaptor grammars. In *Proceedings of the Human Language Technology: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Simon Corston-Oliver and Michael Gamon. 2004. Normalizing German and English inflectional morphology to improve statistical word alignment. *Machine Translation: From Real Users to Research*.

Mathias Creutz and Krista Lagus. 2007. Unsupervised models for morpheme segmentation and morphology learning. *ACM Transactions on Speech and Language Processing*, 4(1):3.

Fabien Cromieres. 2006. Sub-sentential alignment using substring co-occurrence counts. In *Proc. COLING/ACL 2006 Student Research Workshop*.

Carl de Marcken. 1995. The unsupervised acquisition of a lexicon from continuous speech. Technical report, Massachusetts Institute of Technology, Cambridge, MA, USA.

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

Sabine Deligne and Frédéric Bimbot. 1997. Inference of variable-length linguistic and acoustic units by multigrams. *Speech Communication*, 23(3).

John DeNero and Dan Klein. 2010. Discriminative modeling of extraction sets for machine translation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*.

John DeNero, Dan Gillick, James Zhang, and Dan Klein. 2006. Why generative phrase models underperform surface heuristics. In *Proceedings of the 1st Workshop on Statistical Machine Translation (WMT)*.

John DeNero, Alex Bouchard-Côté, and Dan Klein. 2008. Sampling alignment structure under a Bayesian translation model. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Michael Denkowski and Alon Lavie. 2011. Meteor 1.3: Automatic Metric for Reliable Optimization and Evaluation of Machine Translation Systems. In *Proceedings of the 6th Workshop on Statistical Machine Translation (WMT)*.

Markus Dreyer and Jason Eisner. 2011. Discovering morphological paradigms from plain text using a dirichlet process mixture model. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Joris Driesen and Hugo Van Hamme. 2008. Improving the Multigram Algorithm by using Lattices as Input. In *Proceedings of the 9th Annual Conference of the International Speech Communication Association (InterSpeech)*.

Greg Durrett and Dan Klein. 2011. An empirical investigation of discounting in cross-domain language models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Peter D. Eimas, Einar R. Siqueland, Ppeter Jusczyk, and James Vigorito. 1971. Speech perception in infants. *Science*, 171(3968).

Thomas S. Ferguson. 1973. A bayesian analysis of some nonparametric problems. *The annals of statistics*, pages 209–230.

Andrew Finch and Eiichiro Sumita. 2007. Phrase-based machine transliteration. In *Proc. TCAST*.

Daniel Fink. 1997. A compendium of conjugate priors. Technical report, Montana State University.

Jonathan G. Fiscus. 1997. A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (ROVER). In *Proceedings of the 1997 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*.

Atsushi Fujii, Masao Utiyama, Mikio Yamamoto, and Takehito Utsuro. 2008. Overview of the patent translation task at the NTCIR-7 workshop. In *Proceedings of the 7th NTCIR Workshop Meeting on Evaluation of Information Access Technologies*.

Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. 2006. Scalable inference and training of context-rich syntactic translation models. In *Proceedings of the 44th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Nikesh Garera and David Yarowsky. 2008. Translating compounds by learning component gloss translation models via multiple languages. In *Proceedings of the 3rd International Joint Conference on Natural Language Processing (IJCNLP)*.

Andrew Gelman. 1995. *Bayesian data analysis*. CRC press.

Stuart Geman and Donald Geman. 1984. Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6).

James R. Glass. 1988. *Finding acoustic regularities in speech: Application to phonetic recognition*. Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA, USA.

Sharon Goldwater and Thomas Griffiths. 2007. A fully bayesian approach to unsupervised part-of-speech tagging. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Sharon Goldwater and David McClosky. 2005. Improving statistical MT through morphological analysis. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*.

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

Joshua T. Goodman. 2001. A bit of progress in language modeling. *Computer Speech & Language*, 15(4).

Allen L. Gorin, Dijana Petrovska-Delacretaz, Giuseppe Riccardi, and Jeremy H. Wright. 1999. Learning spoken language without transcriptions. In *Proceedings of the 1999 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*.

Aria Haghighi, John Blitzer, John DeNero, and Dan Klein. 2009. Better word alignments with supervised itg models. In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Alon Halevy, Peter Norvig, and Fernando Pereira. 2009. The unreasonable effectiveness of data. *Intelligent Systems, IEEE*, 24(2):8–12.

Zellig S. Harris. 1954. Distributional structure. *Word*.

W. Keith Hastings. 1970. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1).

Teemu Hirsimäki, Mathias Creutz, Vesa Siivola, Mikko Kurimo, Sami Virpioja, and Janne Pylkkönen. 2006. Unlimited vocabulary speech recognition with morph language models applied to Finnish. *Computer Speech & Language*, 20(4).

Koji Hukushima and Koji Nemoto. 1996. Exchange Monte Carlo method and application to spin glass simulations. *Journal of the Physical Society of Japan*, 65(6):1604–1608.

Hemant Ishwaran and Lancelot F. James. 2001. Gibbs sampling methods for stick-breaking priors. *Journal of the American Statistical Association*, 96(453):161–173.

Naoto Iwahashi. 2003. Language acquisition through a human-robot interface by combining speech, visual, and behavioral information. *Information Sciences*, 156(1-2).

Aren Jansen, Kenneth Church, and Hynek Hermansky. 2010. Towards spoken term discovery at scale with zero resources. In *Proceedings of the 11th Annual Conference of the International Speech Communication Association (InterSpeech)*.

Claus S. Jensen, Uffe Kjærulff, and Augustine Kong. 1995. Blocking Gibbs sampling in very large probabilistic expert systems. *International Journal of Human Computer Studies*, 42(6).

J. Howard Johnson, Joel Martin, George Foster, and Roland Kuhn. 2007a. Improving translation quality by discarding most of the phrasetable. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Mark Johnson, Thomas Griffiths, and Sharon Goldwater. 2007b. Bayesian inference for PCFGs via Markov chain Monte Carlo. In *Proceedings of the Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Mark Johnson, Thomas L. Griffiths, and Sharon Goldwater. 2007c. Adaptor grammars: A framework for specifying compositional nonparametric Bayesian models. *Advances in Neural Information Processing Systems*, 19.

Peter W. Jusczyk, Derek M. Houston, and Mary Newsome. 1999. The beginnings of word segmentation in english-learning infants. *Cognitive Psychology*, 39(3-4):159–207.

Fred Karlsson. 1999. *Finnish: An Essential Grammar*. Psychology Press.

Dan Klein and Christopher D. Manning. 2003. A* parsing: fast exact viterbi parse selection. In *Proceedings of the Human Language Technology Conference (HLT-NAACL)*.

Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*.

Kevin Knight and Jonathan Graehl. 1998. Machine transliteration. *Computational Linguistics*, 24(4).

Philipp Koehn et al. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Phillip Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the Human Language Technology Conference (HLT-NAACL)*.

Philipp Koehn, Amittai Axelrod, Alexandra Birch Mayne, Chris Callison-Burch, Miles Osborne, and David Talbot. 2005. Edinburgh system description for the 2005 IWSLT speech translation evaluation. In *Proceedings of the International Workshop on Spoken Language Translation (IWSLT)*.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *MT Summit*.

Grzegorz Kondrak, Daniel Marcu, and Kevin Knight. 2003. Cognates can improve statistical translation models. In *Proceedings of the Human Language Technology Conference (HLT-NAACL)*.

Kimmo Koskenniemi. 1984. A general computational model for word-form recognition and production. In *Proceedings of the 10th International Conference on Computational Linguistics (COLING)*, pages 178–181. Association for Computational Linguistics.

Canasai Kruengkrai, Kiyotaka Uchimoto, Jun'ichi Kazama, Yiou Wang, Kentaro Torisawa, and Hitoshi Isahara. 2009. An error-driven word-character hybrid model for joint Chinese word segmentation and POS tagging. In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Taku Kudo, Kaoru Yamamoto, and Yuji Matsumoto. 2004. Applying conditional random fields to Japanese morphological analysis. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Sadao Kurohashi, Toshihisa Nakamura, Yuji Matsumoto, and Makoto Nagao. 1994. Improvements of Japanese morphological analyzer JUMAN. In *Proceedings of The International Workshop on Sharable Natural Language*.

Lori Lamel, Jean-Luc Gauvain, and Gilles Adda. 2002. Lightly supervised and unsupervised acoustic model training. *Computer Speech and Language*.

Young-Suk Lee. 2004. Morphological analysis for statistical machine translation. In *Proceedings of the Human Language Technology Conference / North American Chapter of the Association for Computational Linguistics (NAACL) Meeting (HLT/NAACL)*.

Zhongguo Li and Maosong Sun. 2009. Punctuation as implicit annotations for chinese word segmentation. *Computational Linguistics*, 35(4):505–512.

Percy Liang, Ben Taskar, and Dan Klein. 2006. Alignment by agreement. In *Proceedings of the Human Language Technology Conference - North American Chapter of the Association for Computational Linguistics (NAACL) Annual Meeting (HLT-NAACL)*.

Percy Liang, Michael I. Jordan, and Dan Klein. 2010. Type-based MCMC. In *Proceedings of the Human Language Technology: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Klaus Macherey, Andrew Dai, David Talbot, Ashok Popat, and Franz Och. 2011. Language-independent compound splitting with morphological operations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL)*.

David J.C. Mackay and Linda C. Bauman Petoy. 1995. A hierarchical Dirichlet language model. *Natural Language Engineering*, 1.

Christopher D. Manning and Hinrich Schütze. 1999. *Foundations of statistical natural language processing*, volume 59. MIT Press.

Daniel Marcu and William Wong. 2002. A phrase-based, joint probability model for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Ryan McDonald, Keith Hall, and Gideon Mann. 2010. Distributed training strategies for the structured perceptron. In *Proceedings of the Human Language Technology: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

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

Daichi Mochihashi, Takeshi Yamada, and Naonori Ueda. 2009. Bayesian unsupervised word segmentation with nested Pitman-Yor modeling. In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Mehryar Mohri, Fernando Pereira, and Michael Riley. 2008. Speech recognition with weighted finite-state transducers. *Handbook on speech processing and speech communication, Part E: Speech recognition*.

Robert C. Moore and Chris Quirk. 2007. An iteratively-trained segmentation-free phrase translation model for statistical machine translation. In *Proceedings of the 2nd Workshop on Statistical Machine Translation (WMT)*.

Masaaki Nagata. 1994. A stochastic Japanese morphological analyzer using a forward-DP backward-A* N-best search algorithm. In *Proceedings of the 15th International Conference on Computational Linguistics (COLING)*.

Tetsuji Nakagawa. 2004. Chinese and Japanese word segmentation using word-level and character-level information. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING)*.

Jason Naradowsky and Sharon Goldwater. 2009. Improving morphology induction by learning spelling rules. In *UCAI 2009, Proceedings of the 21st, International Joint Conference on Artificial Intelligence*, pages 11–17.

Jason Naradowsky and Kristina Toutanova. 2011. Unsupervised bilingual morpheme segmentation and alignment with context-rich hidden semi-markov models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Graham Neubig and Shinsuke Mori. 2010. Word-based partial annotation for efficient corpus construction. In *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC)*, pages 1495–1498, Valetta, Malta, May.

Graham Neubig, Yosuke Nakata, and Shinsuke Mori. 2011. Pointwise prediction for robust, adaptable Japanese morphological analysis. In *Pro-*

ceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL), pages 529–533, Portland, USA, June.

David Newman, Arthur Asuncion, Padhraic Smyth, and Max Welling. 2007. Distributed inference for latent dirichlet allocation. *Proceedings of the 21st Annual Conference on Neural Information Processing Systems (NIPS)*, 20(1081-1088).

David Newman, Arthur Asuncion, Padhraic Smyth, and Max Welling. 2009. Distributed algorithms for topic models. *Journal of Machine Learning Research*, 10.

Hwee Tou Ng, Chung Yong Lim, and Shou King Foo. 1999. A case study on inter-annotator agreement for word sense disambiguation. In *Proc. of SIGLEX99: Standardizing Lexical Resources*.

ThuyLinh Nguyen, Stephan Vogel, and Noah A. Smith. 2010. Nonparametric word segmentation for machine translation. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING)*.

Sonja Nießen and Hermann Ney. 2000. Improving SMT quality with morpho-syntactic analysis. In *Proceedings of the 18th International Conference on Computational Linguistics (COLING)*.

Franz Josef Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Franz Josef Och, Christoph Tillmann, and Hermann Ney. 1999. Improved alignment models for statistical machine translation. In *Proceedings of the 4th Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Franz Josef Och. 1999. An efficient method for determining bilingual word classes. In *Proceedings of the 9th European Chapter of the Association for Computational Linguistics (EACL)*.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL)*.

Hideki Ogura, Hanae Koiso, Yumi Fujiike, Sayaka Miyauchi, Hikaru Konishi, and Yutaka Hara. 2011. Balanced corpus of contemporary written Japanese, morphological information standard (4th edition, in

Japanese). Technical report, National Institute for the Japanese Language and Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING)*.

Alex S. Park and James R. Glass. 2008. Unsupervised pattern discovery in speech. *IEEE Transactions on Audio Speech and Language Processing*, 16(1).

Rebecca Passonneau, Nizar Habash, and Owen Rambow. 2006. Inter-annotator agreement on a multilingual semantic annotation task. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC)*.

Lisa Pearl, Sharon Goldwater, and Mark Steyvers. 2010. How ideal are we? incorporating human limitations into Bayesian models of word segmentation. In *Proceedings of the 34th annual Boston University Conference on Child Language Development*, Somerville, MA. Cascadilla Press.

Fuchun Peng, Fangfang Feng, and Andrew McCallum. 2004. Chinese segmentation and new word detection using conditional random fields. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING)*.

Jim Pitman and Marc Yor. 1997. The two-parameter Poisson-Dirichlet distribution derived from a stable subordinator. *The Annals of Probability*, 25(2).

Jim Pitman. 1995. Exchangeable and partially exchangeable random partitions. *Probability Theory and Related Fields*, 102(2):145–158.

Hoifung Poon, Colin Cherry, and Kristina Toutanova. 2009. Unsupervised morphological segmentation with log-linear models. In *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL) - Human Language Technology (NAACL HLT)*.

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

Carl E. Rasmussen. 2004. Gaussian processes in machine learning. *Advanced Lectures on Machine Learning*, pages 63–71.

Deb K. Roy and Alex P. Pentland. 2002. Learning words from sights and sounds: A computational model. *Cognitive Science*, 26(1).

Daniel M. Roy and Yee Whye Teh. 2009. The mondrian process. In *Proceedings of the 23rd Annual Conference on Neural Information Processing Systems (NIPS)*.

Markus Saers, Joakim Nivre, and Dekai Wu. 2009. Learning stochastic bracketing inversion transduction grammars with a cubic time biparsing algorithm. In *Proceedings of the The 11th International Workshop on Parsing Technologies (IWPT)*.

Jenny R. Saffran, Richard N. Aslin, and Elissa L. Newport. 1996. Statistical learning by eight-month-old infants. *Science*, 274(5294).

Manabu Sassano. 2002. An empirical study of active learning with support vector machines for Japanese word segmentation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Patrick Schone and Daniel Jurafsky. 2001. Knowledge-free induction of inflectional morphologies. In *Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Tanja Schultz and Alex Waibel. 2001. Language-independent and language-adaptive acoustic modeling for speech recognition. *Speech Communication*, 35(1).

Steven L. Scott. 2002. Bayesian methods for hidden Markov models: Recursive computing in the 21st century. *Journal of the American Statistical Association*, 97(457).

Benjamin Snyder and Regina Barzilay. 2008. Unsupervised multilingual learning for morphological segmentation. *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Virach Sornlertlamvanich, Chumpol Mokarat, and Hitoshi Isahara. 2008. Thai-lao machine translation based on phoneme transfer. In *Proceedings of the 14th Annual Meeting of the Association for Natural Language Processing (NLP)*.

Richard Sproat, Chilin Shih, William Gale, and Nancy Chang. 1996. A stochastic finite-state word-segmentation algorithm for chinese. *Computational Linguistics*, 22.

Michael Subotin. 2011. An exponential translation model for target language morphology. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Koichi Takeuchi and Yuji Matsumoto. 1995. Hmm parameter learning for Japanese morphological analyzer. In *Proc. of the 10th Pacific Asia Con-*

ference on Language, Information and Computation (PACLING), pages 163–172.

David Talbot and Miles Osborne. 2006. Modelling lexical redundancy for machine translation. In *Proceedings of the 44th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Deborah Tannen. 1982. *Spoken and Written Language: Exploring Orality and Literacy*. ABLEX.

Yee Whye Teh. 2006. A Bayesian interpretation of interpolated Kneser-Ney. Technical report, School of Computing, National Univ. of Singapore.

Louis ten Bosch and Bert Cranen. 2007. A computational model for unsupervised word discovery. In *Proceedings of the 8th Annual Conference of the International Speech Communication Association (InterSpeech)*.

Jörg Tiedemann. 2009. Character-based PSMT for closely related languages. In *Proceedings of the 13th Annual Conference of the European Association for Machine Translation*.

Dimitra Vergyri and Katrin Kirchhoff. 2004. Automatic diacritization of arabic for acoustic modeling in speech recognition. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING)*.

David Vilar, Jan-T. Peter, and Hermann Ney. 2007. Can we translate letters. In *Proceedings of the 2nd Workshop on Statistical Machine Translation (WMT)*.

Yiou Wang, Kiyotaka Uchimoto, Jun'ichi Kazama, Canasai Kruengkrai, and Kentaro Torisawa. 2010. Adapting Chinese word segmentation for machine translation based on short units. In *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC)*.

Yiou Wang, Jun'ichi Kazama, Yoshimasa Tsuruoka, Wenliang Chen, Yujie Zhang, and Kentaro Torisawa. 2011. Improving chinese word segmentation and pos tagging with semi-supervised methods using large auto-analyzed data. In *Proceedings of the 5th International Joint Conference on Natural Language Processing (IJCNLP)*.

Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3).

Fei Xia, Martha Palmer, Nianwen Xue, Mary Ellen Okurowski, John Kovarik, Fu-Dong Chiou, Shizhe Huang, Tony Kroch, and Mitch Marcus. 2000. Developing guidelines and ensuring consistency for chinese text annotation. In *Proceedings of the 2nd International Conference on Language Resources and Evaluation (LREC)*.

Jia Xu, Richard Zens, and Hermann Ney. 2004. Do we need Chinese word segmentation for statistical machine translation? In *Proceedings of the 3rd SIGHAN workshop on Chinese language processing*.

Jia Xu, Jianfeng Gao, Kristina Toutanova, and Hermann Ney. 2008. Bayesian semi-supervised chinese word segmentation for statistical machine translation. In *Proceedings of the 22th International Conference on Computational Linguistics (COLING)*.

Nianwen Xue and Libin Shen. 2003. Chinese word segmentation as LMR tagging. In *Proceedings of the 2nd SIGHAN workshop on Chinese language processing*.

David Yarowsky and Richard Wicentowski. 2000. Minimally supervised morphological analysis by multimodal alignment. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Kenji Yoshimura, Tooru Hitaka, and Sho Yoshida. 1983. Morphological analysis of non-marked-off Japanese sentences by the least bunsetsu's number method. *Transactions of Information Processing Society of Japan*, 24(1):40–46.

Chen Yu and Dana H. Ballard. 2004. A multimodal learning interface for grounding spoken language in sensory perceptions. *ACM Transactions on Applied Perception*, 1.

Hao Zhang and Daniel Gildea. 2005. Stochastic lexicalized inversion transduction grammar for alignment. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*.

Hao Zhang, Chris Quirk, Robert C. Moore, and Daniel Gildea. 2008a. Bayesian learning of non-compositional phrases with synchronous parsing. *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Ruiqiang Zhang, Keiji Yasuda, and Eiichiro Sumita. 2008b. Improved statistical machine translation by multiple Chinese word segmentation. In *Proceedings of the 3rd Workshop on Statistical Machine Translation (WMT)*.

Valentin Zhikov, Hiroya Takamura, and Manabu Okumura. 2010. An efficient algorithm for unsupervised word segmentation with branching entropy and MDL. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

George Kingsley Zipf. 1949. *Human behavior and the principle of least effort.* Addison-Wesley Press.

Appendix A

Commonly Used Symbols

- Calligraphic symbols ("$\mathcal{W}$", "$\mathcal{X}$") represent collections of input variables for the entirety of the training data.
- Upper-case symbols ("W", "X") represent collections of input variables for a single sentence or utterance, or all the parameters in a single model.
- Bold face lower-case symbols ("$\boldsymbol{w}$", "$\boldsymbol{x}$") are used to specify subsentential groupings (phrases) of variables.
- Regular face lower-case symbols ("w", "x") are used to represent a single variable.
- c: A variable specifying the count of a certain phenomenon. When the first subscript represents a collection, this indicates the collection of samples the count is performed over.
- d: Discount hyperparameters for the Pitman-Yor process.
- E: Target language words (or characters) for machine translation.
- F: Source language words (or characters) for machine translation.
- G: Parameters of a probability model in general, or word-based language model parameters for the specific case of word segmentation.
- H: Character-based spelling model parameters.
- s: The strength hyperparameter for the Pitman-Yor process.
- S: Sufficient statistics necessary for calculating probabilities in a model.

- t: A variable indicating the number of tables in the Chinese restaurant process representation.
- U: Acoustic features for speech recognition.
- W: Words.
- X: Observed samples in general, or in the specific case of word segmentation, characters or phonemes.
- Y: Word boundaries for word segmentation, or hidden variables in the input data for general explanation.
- Z: A normalization constant for a probability distribution.
- α: Concentration hyperparameters for the Dirichlet distribution.
- θ: A collection of probability parameters for a model.
- λ: Weights for features in a log-linear model.
- $\phi()$: A feature function.

Authored Works

Chapter 3

Refereed Works

[1] Graham Neubig, Masato Mimura, Shinsuke Mori, and Tatsuya Kawahara. Learning a language model from continuous speech. In *Proceedings of the 11th Annual Conference of the International Speech Communication Association (InterSpeech)*, pages 1495–1498, Makuhari, Japan, September 2010.

[2] Graham Neubig, Masato Mimura, Shinsuke Mori, and Tatsuya Kawahara. Bayesian learning of a language model from continuous speech. *IEICE Transactions on Information and Systems*, E95-D(2), pages 614–625, Februrary 2012.

Unrefereed Works

[1] Graham Neubig, Masato Mimura, Shinsuke Mori, and Tatsuya Kawahara. Learning a language model from continuous speech using Bayesian inference. In *Information Processing Society of Japan SIG Technical Report (SLP-82)*, Sendai, Japan, July 2010.

Chapter 4

Refereed Works

[1] Graham Neubig, Taro Watanabe, Eiichiro Sumita, Shinsuke Mori, and Tatsuya Kawahara. An unsupervised model for joint phrase alignment

and extraction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 632–641, Portland, USA, June 2011.

[2] Graham Neubig, Taro Watanabe, Eiichiro Sumita, Shinsuke Mori, and Tatsuya Kawahara. Joint phrase alignment and extraction for statistical machine translation. *Journal of Information Processing*, 2(20), April 2012 (To Appear).

Unrefereed Works

[1] Graham Neubig, Taro Watanabe, Eiichiro Sumita, Shinsuke Mori, and Tatsuya Kawahara. Phrase alignment for machine translation with a hierarchical model. In *Proceedings of the 17th Annual Meeting of the Association for Natural Language Processing (NLP)*, Toyohashi, Japan, March 2011.

Chapter 5

Refereed Works

[1] Graham Neubig, Taro Watanabe, Shinsuke Mori, and Tatsuya Kawahara. Machine translation without words through substring alignment. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL)*, Jeju, Korea, July 2012 (In Submission).

Unrefereed Works

[1] Graham Neubig, Taro Watanabe, Shinsuke Mori, and Tatsuya Kawahara. Substring-based machine translation In *Proceedings of the 18th Annual Meeting of the Association for Natural Language Processing (NLP)*, Hiroshima, Japan, March 2012 (To Appear).

Other

Refereed Works

[1] Graham Neubig, Shinsuke Mori, and Tatsuya Kawahara. A WFST-based log-linear framework for speaking-style transformation. In *Proceedings of*

the 10th Annual Conference of the International Speech Communication Association (InterSpeech), pages 1495–1498, Brighton, UK, September 2009.

[2] Graham Neubig, Yuya Akita, Shinsuke Mori, and Tatsuya Kawahara. Improved statistical models for SMT-based speaking style transformation. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 5206–5209, Dallas, USA, March 2010.

[3] Graham Neubig and Shinsuke Mori. Word-based partial annotation for efficient corpus construction. In *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC)*, pages 1495–1498, Valetta, Malta, May 2010.

[4] Yuya Akita, Masato Mimura, Graham Neubig, and Tatsuya Kawahara. Semi-automated update of automatic transcription system for the Japanese national congress. In *11th Annual Conference of the International Speech Communication Association (InterSpeech 2010)*, pages 1495–1498, Makuhari, Japan, September 2010.

[5] Mijit Ablimit, Graham Neubig, Masato Mimura, Shinsuke Mori, Tatsuya Kawahara, and Askar Hamdulla. Uyghur morpheme-based language models and ASR. In *IEEE 10th International Conference on Signal Processing (ICSP)*, pages 581–584, Beijing, China, October 2010.

[6] Graham Neubig, Yosuke Nakata, and Shinsuke Mori. Pointwise prediction for robust, adaptable Japanese morphological analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 529–533, Portland, USA, June 2011.

[7] Shinsuke Mori, Tetsuro Sasada, and Graham Neubig. Language model estimation from a stochastically tagged corpus. *Journal of Natural Language Processing*, 18(2):71–87, June 2011.

[8] Shinsuke Mori and Graham Neubig. A pointwise approach to pronunciation estimation for a TTS front-end. In *12th Annual Conference of the International Speech Communication Association (InterSpeech 2011)*, pages 2181–2184, Florence, Italy, August 2011.

[9] Masao Utiyama, Graham Neubig, Takeshi Onishi, and Eiichiro Sumita. Searching translation memories for paraphrases. In *Proceedings of the Machine Translation Summit XIII*, pages 325–331, Xiamen, China, September 2011.

[10] Shinsuke Mori, Yosuke Nakata, Graham Neubig, and Tatsuya Kawahara. Morphological analysis with pointwise predictors. *Journal of Natural Language Processing*, 18(4):367–381, September 2011.

[11] Shinsuke Mori, Graham Neubig, and Yuta Tsuboi. A pointwise approach to automatic word segmentation. *Journal of Natural Language Processing*, 52(10):2944–2952, September 2011.

[12] Graham Neubig, Yuichiroh Matsubayashi, Masato Hagiwara, and Koji Murakami. Safety information mining - what can NLP do in a disaster -. In *Proceedings of the 5th International Joint Conference on Natural Language Processing (IJCNLP)*, pages 965–973, Chiang Mai, Thailand, November 2011.

[13] Daniel Flannery, Yusuke Miyao, Graham Neubig, and Shinsuke Mori. Training dependency parsers from partially annotated corpora. In *Proceedings of the 5th International Joint Conference on Natural Language Processing (IJCNLP)*, pages 776–784, Chiang Mai, Thailand, November 2011.

[14] Andrew Finch, Chooi ling Goh, Graham Neubig, and Eiichiro Sumita. The NICT translation system for IWSLT 2011. In *Proceedings of the International Workshop on Spoken Language Translation 2011*, San Francisco, USA, December 2011.

[15] Graham Neubig, Yuya Akita, Shinsuke Mori, and Tatsuya Kawahara. A monotonic statistical machine translation approach to speaking style transformation. *Computer Speech and Language*, 2012 (To Appear).

Unrefereed Works

[1] Graham Neubig, Shinsuke Mori, and Tatsuya Kawahara. Japanese character error correction using WFSTs. In *Proceedings of the 15th Annual Meeting of the Association for Natural Language Processing (NLP)*, pages 332–335, Tottori, Japan, March 2009.

[2] Graham Neubig, Shinsuke Mori, and Tatsuya Kawahara. Log-linear speaking-style transformation using weighted finite state transducers. In *Information Processing Society of Japan SIG Technical Report (SLP-77)*, Fukushima, Japan, July 2009.

[3] Graham Neubig, Yuya Akita, Shinsuke Mori, and Tatsuya Kawahara. Context-sensitive statistical models for speaking-style transformation. In *Information Processing Society of Japan SIG Technical Report (SLP-79)*, Tokyo, Japan, December 2009.

[4] Graham Neubig, Yosuke Nakata, and Shinsuke Mori. Domain adaptation of automatic word segmentation using pointwise prediction and active learning. In *Proceedings of the 16th Annual Meeting of the Association for Natural Language Processing (NLP)*, Tokyo, Japan, March 2010.

[5] Graham Neubig, Yuya Akita, Shinsuke Mori, and Tatsuya Kawahara. Speaking style transformation using statistical machine translation. In *Proceedings of the 16th Annual Meeting of the Association for Natural Language Processing (NLP)*, Tokyo, Japan, March 2010.

[6] Shinsuke Mori and Graham Neubig. Automatic improvement of the natural language processing accuracy through the use of kana-kanji conversion logs. In *Proceedings of the 16th Annual Meeting of the Association for Natural Language Processing (NLP)*, Tokyo, Japan, March 2010.

[7] Shinsuke Mori, Tetsuro Sasada, and Graham Neubig. Language model estimation from a stochastically tagged corpus. In *Information Processing Society of Japan SIG Technical Report (NL-196)*, pages 71–87, Tokyo, Japan, May 2010.

[8] Yosuke Nakata, Graham Neubig, Shinsuke Mori, and Tatsuya Kawahara. Morphological analysis with pointwise predictors. In *Information Processing Society of Japan SIG Technical Report (NL-198)*, Tokyo, Japan, September 2010.

[9] Yosuke Nakata, Graham Neubig, Shinsuke Mori, and Tatsuya Kawahara. Improving part-of-speech tagging by combining pointwise and sequence-based predictors. In *Information Processing Society of Japan SIG Technical Report (NL-200)*, Tokyo, Japan, January 2011.

[10] Yuya Akita, Masato Mimura, Graham Neubig, and Tatsuya Kawahara. Semi-automated update of automatic transcription system for the Japanese national congress. In *Information Processing Society of Japan SIG Technical Report (SLP-84)*, Tokyo, Japan, December 2010.

[11] Shinsuke Mori, Daniel Flannery, Yusuke Miyao, and Graham Neubig. Training MST parsers from partially annotated corpora. In *Information Processing Society of Japan SIG Technical Report (NL-201)*, Tokyo, Japan, May 2011.

[12] Graham Neubig and Shinsuke Mori. Efficient information filtering through active learning. In *Proceedings of the 18th Annual Meeting of the Association for Natural Language Processing (NLP)*, Hiroshima, Japan, March 2012 (To Appear).

[13] Daniel Flannery, Yusuke Miyao, Graham Neubig, and Shinsuke Mori. Word-based Japanese dependency parsing. In *Proceedings of the 18th Annual Meeting of the Association for Natural Language Processing (NLP)*, Hiroshima, Japan, March 2012 (To Appear).