A COMPARATIVE ANALYSIS OF BAYESIAN NONPARAMETRIC VARIATIONAL INFERENCE ALGORITHMS FOR SPEECH RECOGNITION

A Thesis
Submitted to
the Temple University Graduate Board

In Partial Fulfillment
of the Requirements for the Degree
Master of Science in Electrical Engineering

by
John Steinberg
May, 2013

ABSTRACT

Nonparametric Bayesian models have become increasingly popular in speech recognition tasks such as language and acoustic modeling due to their ability to discover underlying structure in an iterative manner. These methods do not require a priori assumptions about the structure of the data, such as the number of mixture components, and can learn this structure directly. Dirichlet process mixtures (DPMs) are a widely used nonparametric Bayesian method which can be used as priors to determine an optimal number of mixture components and their respective weights in a Gaussian mixture model (GMM). Because DPMs potentially require an infinite number of parameters, inference algorithms are needed to make posterior calculations tractable. The focus of this work is an evaluation of three of these Bayesian variational inference algorithms which have only recently become computationally viable: Accelerated Variational Dirichlet Process Mixtures (AVDPM), Collapsed Variational Stick Breaking (CVSB), and Collapsed Dirichlet Priors (CDP).

To eliminate other effects on performance such as language models, a phoneme classification task is chosen to more clearly assess the viability of these algorithms for acoustic modeling. Evaluations were conducted on the CALLHOME English and Mandarin corpora, consisting of two languages that, from a human perspective, are phonologically very different. It is shown in this work that these inference algorithms yield error rates comparable to a baseline Gaussian mixture model (GMM) but with a factor of up to 20 fewer mixture components. AVDPM is shown to be the most attractive choice because it delivers the most compact models and is computationally efficient, enabling its application to big data problems.

ACKNOWLEDGEMENTS

There are many people to whom I am deeply indebted and I never would have gotten here without them. First, I would like to thank my wonderful wife Amanda whose unyielding support encouraged me to return to grad school and pursue speech recognition. Both of our families have also been extremely supportive and have always helped me to succeed.

I would also like to thank my fellow graduate students at Temple who have reminded me that I'm still sane throughout this whole process. Of particular note is Amir Harati who has personally guided me through the field of speech recognition and has helped me more than almost anyone during my studies. Dr. Iyad Obeid also deserves special thanks for having me as his TA for the past 3 years which has allowed me to survive financially as a grad student. I've been extremely lucky to work for him and I would not have had such a smooth and successful teaching career at Temple without him.

Most importantly, I would like to thank my adviser Dr. Joe Picone. Without him I never would have found speech recognition, let alone begun a successful career in it. No other person would have given me, an undergraduate that had studied physics and Chinese 5 years earlier, a chance to reenter graduate school in a field I had no experience in. I have never met anyone so passionate about students' educations or about their successes both during and after their academic careers. At times I thought his high standards would drive me insane, but I have learned more from him than anyone during my academic and professional life and he has truly been an inspiration to me. Thank you.

This research was also supported in part by the National Science Foundation through Major Research Instrumentation Grant No. CNS-09-58854. I would also like to thank the Linguistic Data Consortium (LDC) for awarding a data scholarship to this project and providing the lexicon and transcripts for CALLHOME Mandarin.

TABLE OF CONTENTS

ABSTRACT .... **II**

ACKNOWLEDGEMENTS .... **III**

LIST OF FIGURES .... **VI**

LIST OF TABLES .... **VIII**

CHAPTER 1 INTRODUCTION **1**

1.1 Speech Recognition Overview 3

1.2 Previous Work 5
1.2.1 Common Paradigms for Acoustic Models 5
1.2.2 Applications of Nonparametric Models 10

1.3 Thesis Overview 11

CHAPTER 2 NONPARAMETRIC BAYESIAN MODELS AND VARIATIONAL INFERENCE **13**

2.1 Dirichlet Distributions and Processes 14
2.1.1 Dirichlet Distributions 14
2.1.2 Dirichlet Processes 18

2.2 Variational Inference 21
2.2.1 Accelerated Variational Dirichlet Process Mixtures (AVDPM) 22
2.2.2 Collapsed Variational Methods 23

CHAPTER 3 DATA AND EXPERIMENTS **26**

3.1 A Comparison of English and Mandarin 26

3.2 Data 29
3.2.1 TIMIT 29
3.2.2 The LDC's CALLHOME Corpora 30

3.3 Data Preparation 31
3.3.1 TIMIT 32
3.3.2 CALLHOME English (CH-E) 34
3.3.3 CALLHOME Mandarin (CH-M) 36

3.4 Baseline Algorithm Implementations 36

3.5 Variational Inference Algorithm Implementations 38

CHAPTER 4 RESULTS AND DISCUSSION 40

4.1 Baseline Algorithm Tuning 40

4.2 Variational Inference Algorithm Tuning 45

4.3 Evaluation Set Performance 46

4.4 Computational Complexity Analysis 48

CHAPTER 5 CONCLUSIONS AND FUTURE WORK 52

REFERENCES CITED 55

LIST OF FIGURES

Figure 1 – A diagram that shows some of the fields where nonparametric Bayesian methods have been applied. 2

Figure 2 – A typical speech recognition system first converts input audio data into feature vectors. The probability that these vectors were generated by the acoustic model is combined with a language model score to generate the probability of word labels. The search engine essentially implements Bayes Rule by finding the most probable word sequence. 3

Figure 3 – An example of a neural network architecture with *60* neurons in the hidden layer used in this work. The input *40x3* feature matrix is converted to a 1*20x1* feature vector and the output layer represents the *42* possible phoneme labels. 7

Figure 4 – A diagram depicting the Polya urn method (Frigyik et.al., 2010). 17

Figure 5 – A diagram depicting the stick breaking representation of a Dirichlet process. 18

Figure 6 – A diagram showing how a Dirichlet distribution initially characterized by a uniform distribution can be decimated infinitely many times to yield discrete values. 19

Figure 7 – An example showing several equivalent sentences using different word orders (Lee, 2006). 28

Figure 8 – The average misclassification error is shown for *10* iterations of the single hidden layer neural network model for TIMIT (left) and CH-E and CH-M (right). The near optimal number of neurons was determined to be 150 for acoustic modeling tasks. 41

Figure 9 – The average misclassification error is shown for 5 iterations of the random forest algorithm for TIMIT (left) and CH-E and CH-M (right). An operating point of 150 trees was selected due to computational considerations. 42

Figure 10 – The misclassification error for KNN as a function of *K*, the number of nearest neighbors, is shown for TIMIT (left) and CH-E and CH-M (right). The near optimal operating point selected for acoustic modeling was *K = 20*. 43

Figure 11 – The misclassification error for 10 iterations of a GMM is shown as a function of the number of mixture components. Results for TIMIT are shown on the left, and results for CH-E and CH-M are shown on the right. Though there is measurable degradation in performance when using automatically-generated alignments, the trends are consistent. Further, there are significant differences between the optimal settings for TIMIT and CALLHOME. 44

Figure 12 – The misclassification error rates are shown as the initial depth of the KD tree is varied for TIMIT (left), CH-E, and CH-M (right). 46

Figure 13 – The misclassification error rates for CVSB and CDP are shown as the truncation level is varied for TIMIT (left), CH-E, and CH-M (right). 47

Figure 14 – The average amount of CPU training time is shown for 10 iterations of AVDPM (left), CVSB (right) and CDP (right) on TIMIT as a function of the KD tree depth and truncation level. 49

Figure 15 – The average amounts of CPU training time vs. the number of training samples across 200 iterations at optimal operating points on the TMIIT Corpus are shown. Trends are extrapolated to show that AVDPM would outperform CVSB and CDP on much larger corpora such as Fisher. 50

LIST OF TABLES

Table 1 – A description of the different tones in Mandarin Chinese. Words in parentheses indicate examples of English words that share similar sounds with Mandarin tones. 27

Table 2 – A description of the speakers and utterances found in the TIMIT Corpus. 29

Table 3 – A description of the TIMIT training, validation, and evaluation (Core Test) sets used for this work. 30

Table 4 – A chart of TIMIT's 61 phone labels after being folded into 48 classes for training. 33

Table 5 – A table showing the amount of CH-E data before and after filtering out segments of overlapping speakers.. 35

Table 6 – A table showing the amount of CALLHOME Mandarin data before and after filtering out segments of overlapping speakers... 36

Table 7 – A comparison of misclassification error and number of mixture components for the evaluation sets of the TIMIT, CH-E, and CH-M corpora using automatically generated alignments. 48

CHAPTER 1

INTRODUCTION

For the past three decades, parametric statistical models have dominated acoustic modeling in speech recognition. More specifically, hidden Markov models (HMMs) trained using Mel frequency cepstral coefficients (MFCCs) have proven to yield reasonable error rates at a relatively low computational complexity. However, these models make a priori assumptions about the structure of the data, e.g. the number of mixture components in a Gaussian mixture model (GMM), and typically use the same structure for each label's model. A priori assumptions about the structure of the data are largely presumptuous, especially for complex data such as speech. It would be safer to assume that the underlying structure is unknown and that this structure can vary for each phoneme model. For this reason, researchers have increasingly explored nonparametric Bayesian methods that can automatically find this structure directly from the data. There is a wide array of applications of these methods including regression, density estimation, and survival analysis (Quintana & Muller, 2004), some of which are shown in Figure 1.

The focus of this work is acoustic modeling in speech recognition, so Dirichlet distributions, and by extension Dirichlet processes (DP), are investigated. These models act as conjugate priors for multinomial distributions, i.e. they can be used as a prior for the number of mixture components and their corresponding weights in a Gaussian model. These models are therefore commonly used for density estimation tasks such as the acoustic modeling problem explored here.

Dirichlet process mixtures (DPMs) refer to a model where a DP is used to find the structure of a GMM. DPMs have many advantages over standard GMM-based systems. Rigidly setting the structure of a GMM requires that, given a set of training data, a system's structure is chosen by tuning a set of validation data. If new training data is provided the system would

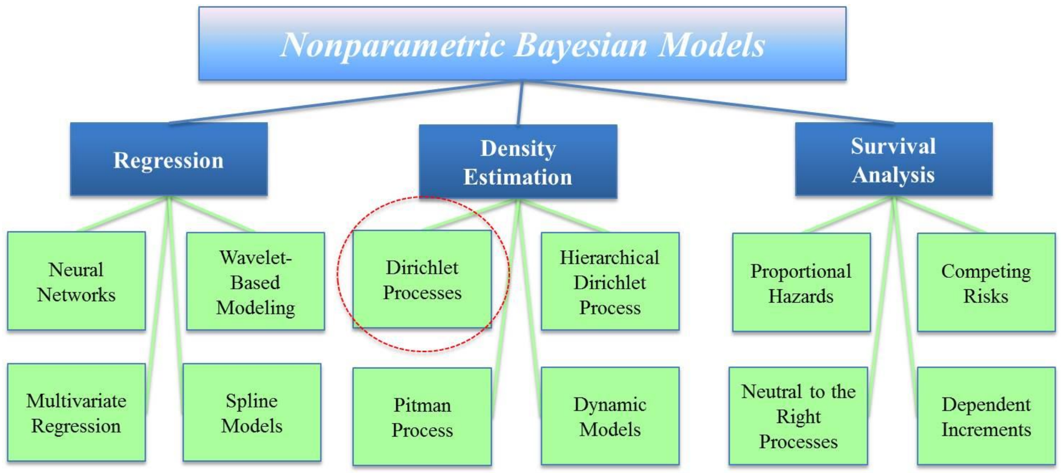


Figure 1 – A diagram that shows some of the fields where nonparametric Bayesian methods have been applied.

require retuning. DPMs, however, can automatically find the underlying structure and grow as new data is introduced.

Parametric methods can also be interpreted as models that capture general acoustic features by averaging across a set number of distributions. A DPM's potentially infinite number of distributions can simultaneously capture more unique acoustic traits of individual speakers along with the more general acoustic features used for phone identification (Harati et al., 2012). This high degree of complexity makes sampling from these distributions intractable, however, so approximations are made using inference algorithms. The focus of this work is an evaluation of three of these Bayesian variational inference algorithms which have only recently become computationally viable: Accelerated Variational Dirichlet Process Mixtures (AVDPM), Collapsed Variational Stick Breaking (CVSB), and Collapsed Dirichlet Priors (CDP).

Rather than conducting a complete speech recognition experiment where classification is affected by several interdependent factors (e.g. language modeling and search), a simple phone recognition task is chosen to more clearly assess the efficacy of these algorithms for future applications in acoustic modeling. Furthermore, this evaluation is conducted using the CALLHOME English (Canavan et al., 1997) and Mandarin (Canavan & Zipperlen, 1996) corpora, two languages that, from a human perspective, are phonologically very different. This serves two purposes: (1) artifacts from either language that influence classification will be identified; (2) if no such artifacts exist, then these algorithms will have strongly supported their use for future multi-language recognition tasks.

This chapter will focus first on a simple introduction to speech recognition followed by some of the statistical methods commonly used in speech recognition tasks. This includes generative models (e.g. HMMs), discriminative models, neural networks, and exemplar based approaches. Finally a brief overview of how nonparametric models have been integrated in speech research is also provided.

1.1 Speech Recognition Overview

Although this work does not focus directly on improving the overall performance of a speech recognition system it is worth describing the general process, shown in Figure 2, since it will be employed in our experimental design. The acoustic front end converts input audio data

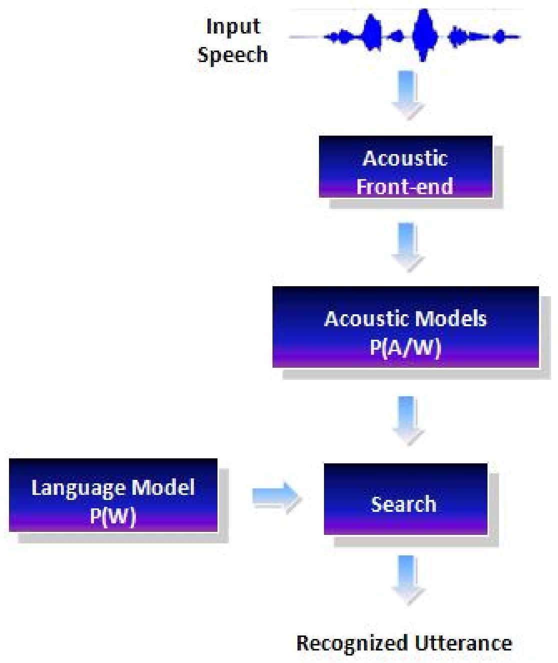


Figure 2 – A typical speech recognition system first converts input audio data into feature vectors. The probability that these vectors were generated by the acoustic model is combined with a language model score to generate the probability of word labels. The search engine essentially implements Bayes Rule by finding the most probable word sequence.

into feature vectors. Mel frequency cepstral coefficients (MFCCs) (Mermelstein & Davis, 1980), very commonly used features (Young, 1996), are used in this work. The new feature vectors are then passed to the next stage of the speech recognizer where the probability that the acoustic models generated this data is computed. A wide variety of training methods can be used in this stage to train these acoustic models, and some of these are addressed in the following sections. These methods can include parametric or nonparametric methods using generative, discriminative, or exemplar-based models. The general training process typically begins by first training simple monophone models which in turn are used as a basis to train more complex triphone models.

A language model is also trained independently of the acoustic models. *N*-grams, a simple and very common type of language model (Kneser & Ney, 1995), determine the probability of a word conditioned on the *N* previous words from the utterance, $P(W_k \mid W_{k-1}, W_{k-2}, \ldots, W_{k-N})$. The language model's score is combined with the acoustic model's score to generate a maximum likelihood score.

A search algorithm is used to implement Bayes Rule by maximizing the probability of a word sequence given the data (Jelinek, 1997). The space of potential hypotheses is large, and hence the search algorithm must efficiently search through this large space to find the most probable word sequence. The most popular approach for search is based on Viterbi beam search (Lee, 1989) but this topic is beyond the scope of this thesis. Either word lattices or one-best scores are typically used to determine the most likely output utterance(s). These newly created labels are compared to reference labels and a WER is determined for the system following an industry-standard scoring algorithm provided by NIST (NIST, 2010).

Although the focus of this work is not to evaluate a complete speech recognition system's performance, we do train an acoustic model for portions of this work. A monophone acoustic model (Lee, 1989) is trained using an HMM with *16* Gaussian mixture components per state to generate a Viterbi-based time alignment (Viterbi, 1967). Features corresponding to individual phonemes can then be extracted and used for this phoneme classification task.

1.2 Previous Work

The problem posed by acoustic modeling has been studied for several decades (Viterbi, 1967; Rabiner, 1989; Halberstadt & Glass, 1998). Consequently, a multitude of potential solutions have been proposed ranging from exemplar models such as K nearest neighbors (Cover & Hart, 1967), which create predictions using a distance between individual samples, to generative models (Baum & Petrie, 1966) such as HMMs that attempt to fit distributions to training data and make predictions on maximum likelihood, to discriminative models (McCulloch & Pitts, 1943) such as neural networks that calculate posteriors directly. The following subsections outline some of the details of these algorithms and some of the benefits and disadvantages they offer for acoustic modeling. Finally, a brief overview of how nonparametric methods have been integrated in speech recognition and why they can be useful in acoustic modeling is provided.

1.2.1 Common Paradigms for Acoustic Models

A wide variety of statistical modeling methods have been proposed to improve speech recognition performance. HMMs, the most commonly used type of generative models (Baum & Petrie, 1966), were introduced to speech research in the 1980s and were widely acknowledged for their ability to model temporally changing data such as speech. These generative graphical models utilize GMMs to represent the probability density function (pdf) of a feature vector x_t at a given time t:

$$p(x_t \mid \Lambda_i) = \sum_{k=1}^{K} \varpi_{ik} N(x_t; \mu_{ik}, \Sigma_{ik}) \tag{1}$$

where $\Lambda = \{\Lambda_i\} = \{\pi_i, a_{ij}, \omega_{ik}, \mu_{ik}, \Sigma_{ik}\}$ are the HMM parameters, ϖ_{ik} represents the weights of K mixtures for cluster i such that $\sum_k \varpi_{ik} = 1$, and μ_{ik} and Σ_{ik} represent the means and covariances of the mixtures. With a collection of feature vectors $X = \{x_t\}_{t=1}^{T}$, the likelihood of the data is then:

$$p(X \mid \Lambda) = \sum_{S=\{s_t\}} \left[\pi_{s_1}\, p(x_1 \mid \Lambda_{s_1}) \prod_{t=2}^{T} a_{t-1,\, s_t}\, p(x_t \mid \Lambda_{s_t}) \right] \tag{2}$$

where $S = \{s_t\}$ is the set of state labels, π is the state probability such that $\sum_i \pi_i = 1$, and a represents the state transition probability such that $\sum_i a_{ij} = 1$. Equation (2) is trained using the expectation maximization (EM) algorithm (Dempster, 1977). In each iteration a new set of HMM parameters are generated such that:

$$\Lambda^{(k+1)} = \underset{\Lambda}{\arg\max} \sum_{S} p(S \mid X, \Lambda^{(k)}) \log(X, S \mid \Lambda) . \tag{3}$$

Finally, with an acoustic score from the HMMs and a language model as a prior, final classification is done using Bayes Rule:

$$P(W \mid X; \Lambda) = \frac{P(W) P(X; \Lambda)}{\sum_W P(W) P(X; \Lambda)} . \tag{4}$$

Given labels, generative models like HMMs attempt to fit distributions to the data. Unlabeled data is then passed to the system and the output hypothesis is selected by choosing the most probable hypothesis generated from the trained distributions. This is not necessarily optimal since theoretically an infinitely large set of data is required to correctly model a distribution. At one extreme, huge corpora can make training systems computationally impractical, while at the other extreme, there may be insufficient amounts of training data leading to poorly estimated model parameters.

Furthermore, system performance is often evaluated by measuring a misclassification error rate. Discriminative models attempt to do this by directly minimizing this error rate. One popular approach is to maximize the log of the posterior probability, *log P(W|X)*, i.e. the probability of a word (or phone) given the data. One very common discriminative model is the neural network (NN) (McCulloch & Pitts, 1943) shown in Figure 3. These models are defined by a three-level architecture, i.e. input, hidden, and output layers, such that features are mapped to subsequent layers using a non-linear transfer function. A very common choice is the sigmoid transfer function given by:

$$y_j = \frac{1}{1+e^{-x_j}} \qquad x_j = b_j + \sum_i y_i w_{ij} \tag{5}$$

where y is the layer following x, y_j is the new state in layer y, b_j is the bias coefficient and w_{ij} are the weights for connections from layer i to layer j. It is important that these weights are initially given small random values to break symmetry and prevent identical gradients among different states (i.e. preventing all weights within a layer from being equal). A cost function, such as minimizing mean squared error, is used to train the model using a back propagation algorithm (Werbos, 1974; Rumelhart et al., 1986).

This architecture makes neural networks ideally suited to modeling nonlinear speech data that is generated by a system containing a small number of degrees of freedom, i.e. the human vocal tract. These neural network models still require an extremely large number of parameters, especially when modeling triphones, often referred to as context-dependent (CD) phones. Tens of thousands of these units are required to model the English language, and languages such as

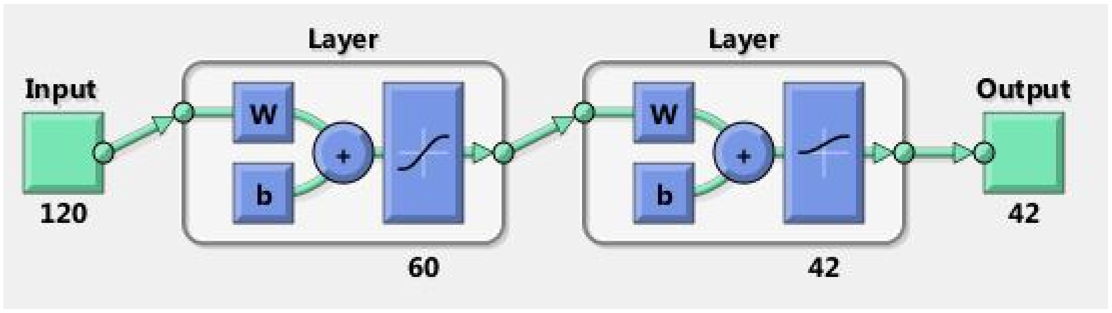


Figure 3 – An example of a neural network architecture with *60* neurons in the hidden layer used in this work. The input *40x3* feature matrix is converted to a *120x1* feature vector and the output layer represents the *42* possible phoneme labels.

Chinese can easily require over one hundred thousand such units. Neural networks used for such applications often require an output layer of several thousand nodes.

It was not until recent advances in hardware and training algorithms that manageable computation time was achieved (Hinton et al., 2012). New research has shown that deep belief networks (DBNs), consisting of multiple hidden layers, can avoid the endemic problem of overfitting and are able to outperform GMM-based HMM systems (Hinton et al., 2012; Larochelle et al., 2007). In this work, a simple neural network consisting of a single hidden layer is investigated rather than a DBN since the classification task being posed here is much simpler than that required for state of the art speech recognition.

Many other discriminative models exist. Saon & Chen (2012) provide an excellent overview of some of these methods. These include techniques which incorporate objective functions that minimize classification error (MCE), minimize word or sentence error rates using maximum mutual information (MMI), or minimize phone error rates (MPE). All of these have shown to produce results that are significantly better than maximum likelihood (ML) approaches (Juang et al., 1997; Bahl et al., 1986; Povey & Woodland, 2002). However, while the previously discussed models estimate continuous distributions, these discriminative methods minimize discrete classification errors and require the use of smoothing techniques before gradient descent can be used to converge on an optimal solution.

Another interesting approach is the ensemble technique proposed by Leo Breiman (2001) known as a random forest (RF). This method builds multiple classification trees which vote on the output label for a given test sample. If training samples consist of M variables, then m variables (such that $m << M$) are randomly selected to determine the best way to split a tree's current node. Each tree in the forest is grown to the largest possible depth without pruning. Once trained, each tree selects an hypothesis and the mode is chosen as the final output. This technique is very robust to overfitting (Breiman, 2001) and is also able to identify the importance of training features based on the voting success of each tree, i.e. individual trees whose voting records match true output labels indicate better partitions across features.

All of the generative and discriminative methods mentioned in the above paragraphs build models by generalizing training data and are known as global data techniques (Sainath et al., 2012). On the opposite end of the spectrum are exemplar-based models which aim to classify data based on just a few examples or templates. While global data models can suffer if the amount of training data is limited, exemplar-based techniques can still yield high performance (Belkin & Niyogi, 2003). These systems typically follow three stages (Sainath et al., 2012): exemplar modeling, instance modeling, and decoding. In the first stage, the search space is reduced and the best exemplars for a given label are determined. In the next phase, a given test vector is compared to the exemplar and weights are assigned to each of them. Finally, an acoustic score is generated that ultimately determines the most probable utterance.

K nearest neighbors (KNN) (Cover & Hart, 1967) is a good example of an exemplar-based model in which a distance metric is used to determine the *K* closest exemplars to a test vector. There are several methods to improve the computational efficiency of KNN (Samet, 2008) but these are beyond the scope of this paper. In this work, a slightly simpler method is used. Rather than building an exemplar model, the Euclidean distance between each test point and all other training samples is calculated. This distance is measured for *K* neighbors and the mode of the closest labels is chosen as the prediction. If there are multiple modes, the label with the shortest average distance is chosen. Again, this simple KNN model was selected to serve as a baseline comparison for the variational inference algorithms.

It is worth mentioning that for many practical purposes, hybrid systems have been developed that incorporate acoustic training phases that utilize multiple algorithms such as HMMs and NNs (Heigold et al., 2012). A DBN-HMM system, for example, resulted in a relative reduction of *33%* in WER from the standard GMM-based HMM system's performance on the SWITCHBOARD Corpus (i.e. WER dropped from *27.4%* to *18.5%*) (Seide, Li, & Yu, 2011).

1.2.2 Applications of Nonparametric Models

The application of nonparametric Bayesian models to speech research has become increasingly popular although mostly in language modeling and speaker adaptation. In this section, some of these applications are described.

N-grams are very simple and commonly used methods for language modeling where *P(W)* from (4) is determined by a conditional probability of a word given the previous *N-1* words. This is a fast and efficient technique but is only able to capture local lexical information since *N* has to be manually set. *N* is typically a relatively small number due to data sparseness constraints. To help mitigate the sparsity problems, there have been several smoothing techniques have been developed such as Kneser-Ney language models (KNLM) which help eliminate cases where various *N*-grams don't exist within the training data (S. F. Chen & Goodman, 1999). More recently, nonparametric Bayesian techniques such as the Pitman-Yor language model (PYLM) have expanded KNLMs and allow *N*-gram parameters to grow as new data is introduced to the system (Renals, 2010). This helps to greatly reduce the risk of over-fitting or under-fitting test data. Other techniques such as the latent Dirichlet allocation language model (LDALM) (Tam & Schultz, 2005), have been used to capture non-local lexical information. Rather than model a word's probability strictly from its context (i.e. *N-1* previous words), LDALMs map the word history into a topic class and base the probability of a word on the word history and the semantic content.

Nonparametric methods like DPMs have also been recently applied to other speech recognition tasks such as acoustic unit detection and speaker adaptation (Harati et al., 2012). Speaker adaptation is used to complement acoustic training by adapting speaker independent (SI) models that are trained on large amounts of data. Speaker adaptation systems use parametric models to map SI models to higher-performing models by adjusting a parametric mapping on small amounts of adaptation data. Dirichlet processes can find the structure of this mapping directly from the adaptation data. Harati et al. (2012) have shown that DPM-based speaker adaptation reduced WER by *10%* from the more common maximum likelihood linear regression (MLLR) method.

1.3 Thesis Overview

A nonparametric model's ability to discover the underlying structure of data makes it extremely attractive for speech recognition. The remainder of this thesis focuses on describing why Dirichlet processes in particular are ideally suited for this task and how the performance of nonparametric approaches compares to more standard parametric models such as NNs, RFs , KNNs, and GMMs. Furthermore, since inference algorithms are required to make posterior calculations of nonparametric models tractable, three variational methods will be evaluated: AVDPM, CVSB and CDP.

The goal of this work is to select the best variational inference algorithm for speech recognition from these three algorithms. This evaluation is based on a static phoneme classification task using English and Mandarin corpora which will identify if the algorithms are prone to any language specific artifacts. Final evaluation is based on average misclassification error rate and also on computational complexity. It is hypothesized that the three variational inference algorithms will yield comparable error rates but the incorporation of the KD tree will allow AVPDM to train on larger data sets significantly faster than CVSB or CDP. Furthermore, one of the major aims of this research is to move towards a complete nonparametric Bayesian speech recognition system and the conclusions found in this work will dictate which variational method will be implemented in said system.

The remainder of this thesis is organized into the following sections. In CHAPTER 2 an in-depth description of Dirichlet process mixture models is provided followed by explanations of the three variational inference algorithms. Experimental set-ups including data preparations are discussed in CHAPTER 3 along with a discussion of the major language differences between

English and Mandarin. CHAPTER 4 provides a detailed analysis of the results found from these experiments. Finally, CHAPTER 5 offers conclusions and proposes some viable directions for future research.

CHAPTER 2

NONPARAMETRIC BAYESIAN MODELS AND VARIATIONAL INFERENCE

Parameterized models have been widely popular for their efficiency, ease of use, and reasonable performance in various clustering and classification problems. They are of particular use when the underlying structure of the data is either previously known or can be easily approximated. However, this results in a significant limitation in application since clusters in real data often vary in shape and may follow their own individual distributions (Mallapragada, et. al., 2010). Thus parametric methods can sometimes fail to produce successful models. Bayesian nonparametric models on the other hand are able to independently infer the underlying structure of data by determining an optimal number of clusters or latent variables during fitting. Moreover, the complexity and accuracy of the model evolves as more data is added to the system and retuning is unnecessary. One example of a nonparametric model is the Dirichlet process (DP) (Ferguson, 1973). A DP is a distribution over distributions that theoretically requires an infinite number of parameters.

The purpose of a statistical model in a typical application of Bayes Rule to classification is to determine the posterior probability of an event given a model's parameters. For example, a speech recognition system attempts to train an acoustic model with parameters, θ, given a set of data such that word error rate is minimized during testing. However, because nonparametric models theoretically require an infinite number of parameters, it is often impossible to manipulate such distributions directly. Instead, inference algorithms are used to generate approximations. In the next few sections Dirichlet distributions and processes will be introduced along with the three variational inference algorithms that are the subject of this thesis.

2.1 Dirichlet Distributions and Processes

Typical parameterized models make the assumption that the underlying structure of the data is known or easily approximated. For acoustic models this implies that the number of mixture components for a GMM is known and fixed for all phoneme models. This is largely presumptuous and it would be far more reasonable to assume each model is unique and that the number of mixture components is not known a priori. The problem of finding that a sample came from component *i*, where $i = 1, 2, 3, \ldots, k$ is best characterized by a multinomial distribution. This however, requires some sort of prior knowledge about the likelihood that each component exists.

Bayes Rule states that the probability that sample x came from mixture i, $p(\theta_i|x)$, is proportional to the likelihood, $p(x|\theta_i)$, multiplied by the prior, $p(\theta_i)$. Dirichlet distributions, and by extension Dirichlet processes, serve as this prior (in the case of multinomial distributions) and thus, despite the multitudes of other nonparametric models, make them particularly attractive for acoustic modeling. In the following sections a basic overview of the definitions and properties of Dirichlet distributions and processes are put forth which follow the explanations found in Frigyik et. al. (2010).

2.1.1 Dirichlet Distributions

A Dirichlet distribution in its most basic form is a distribution over probability mass functions (pmfs) (often referred to as a discrete probability distribution in engineering). Mathematically this can be represented by the following. First, let $Q = | Q_1, Q_2, \ldots, Q_k |$ be a random pmf such that $Q_i \geq 0$ and $\sum_{i=1}^{k} Q_i = 1$. Let q_i be a realization from $Q \sim Dir(\alpha)$ where α represents an inverse variance. Furthermore, we can set the Dirichlet distribution's hyperparameter, $\alpha = | \alpha_1, \alpha_2, \ldots, \alpha_k |$, such that $\alpha_i > 0$ and $\alpha_0 = \sum_{i=1}^{k} \alpha_i$. A Dirichlet distribution is then given by:

$$f\left(q;\alpha\right)=\frac{\Gamma\left(\alpha_0\right)}{\prod_{i=1}^{k}\Gamma\left(\alpha_i\right)}\prod_{i=1}^{k}q_i^{\alpha_i-1}. \tag{6}$$

For the case where *k = 2,* if *Q = (X, 1-X)*, (6) reduces to a Beta distribution:

$$f\left(x;\alpha,\beta\right)=\frac{\Gamma\left(\alpha+\beta\right)}{\Gamma\left(\alpha\right)\Gamma\left(\beta\right)}x^{\alpha-1}\left(1-x\right)^{\beta-1} \ . \tag{7}$$

A real world example of this is offered by Frigyik et. al. (2010). In this application a bag full of one hundred 6-sided dice is given. A roll of each die therefore has the same set of possible outcomes as any other die (1, 2, 3, 4, 5, or 6) but might have slightly different probabilities of achieving them. This can be modeled with a Dirichlet distribution, *Dir(α)*, consisting of pmfs $q^{(i)}$, where *i = 1, 2, ..., L* and *L=100* (dice in this example). Further, suppose that n_i samples are drawn from the i^{th} pmf. A large α_i indicates that the i^{th} dice is rolled more often and consequently has a greater effect on the overall distribution. This concept can be extended to the language modeling problem in speech recognition by replacing each die with an utterance and letting the observed rolls represent observed words (from a finite vocabulary). This can be written as follows:

$$Dir(\alpha)\begin{array}{l}\xrightarrow{iid}q^{(1)}\xrightarrow{iid}x_{1,1},x_{1,2},\ldots,x_{1,n_1}\triangleq x_1,\\ \xrightarrow{iid}q^{(2)}\xrightarrow{iid}x_{2,1},x_{2,2},\ldots,x_{2,n_2}\triangleq x_2,\\ \qquad\qquad\vdots\\ \xrightarrow{iid}q^{(L)}\xrightarrow{iid}x_{L,1},x_{L,2},\ldots,x_{L,n_L}\triangleq x_L.\end{array}$$

The probability of a possible outcome *x* is then given by:

$$p\left(x\,|\,\alpha\right)=\prod_{i=1}^{L}p(x_i\,|\,\alpha) \tag{8}$$

where the probability that *x* is drawn from the i^{th} pmf is:

$$\begin{aligned}p\left(x_i\,|\,\alpha\right)&=\int p\left(x_i,q^{(i)}\,|\alpha\right)dq^{(i)}\\&=\int p\left(x_i\,|q^{(i)},\alpha\right)p\left(q^{(i)}\,|\alpha\right)dq^{(i)}\\&=\int p\left(x_i\,|q^{(i)}\right)p\left(q^{(i)}\,|\alpha\right)dq^{(i)}\ .\end{aligned} \tag{9}$$

Finally, it is shown that $p(x_i\,|\,q^{(i)})$ is given by the multinomial distribution (Frigyik et al., 2010):

$$p\left(x_i \mid q^{(i)}\right) = \frac{n_i!}{\prod_{i=1}^{k} n_{ij}!} \prod_{j=1}^{k} \left(q_j^{(i)}\right)^{n_{ij}} \tag{10}$$

where $p(q^{(i)} \mid \alpha)$ follows a Dirichlet distribution and is given by:

$$p\left(q^{(i)} \mid \alpha\right) = \frac{\Gamma(a_0)}{\prod_{i=1}^{k} \Gamma(a_j)} \prod_{j=1}^{k} \left(q_j^{(i)}\right)^{\alpha_j - 1} . \tag{11}$$

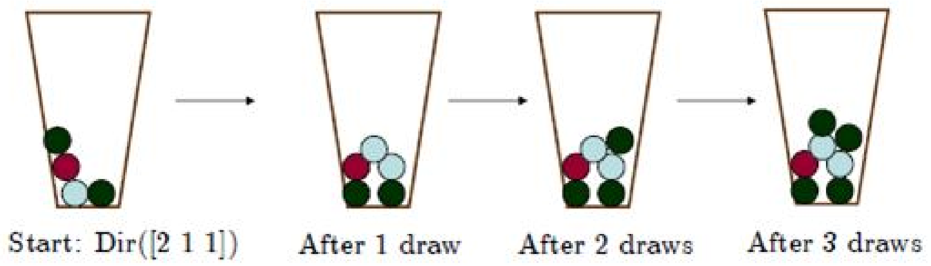
When (10) and (11) are combined, (9) becomes:

$$p(x_i \mid \alpha) = \frac{n_i!}{\prod_{j=1}^{k} n_{ij}!} \frac{\Gamma(a_0)}{\Gamma\left(\sum_{j=1}^{k}\left(n_{ij} + \alpha_j\right)\right)} \prod_{j=1}^{k} \frac{\Gamma(n_{ij} + \alpha_j)}{\Gamma(a_j)} . \tag{12}$$

This can be used in (8) to find the likelihood of the data. To find the optimal *α* that maximizes the likelihood, the log of (12) is maximized using an optimization technique such as the EM algorithm (Baum & Petrie, 1966).

A few of the more common representations of a Dirichlet distribution include the Polya urn and stick-breaking methods (Blackwell & MacQueen, 1973; Sethuraman, 1994). In the Polya urn method, α_i balls of color *I* (where *I = 1, 2… k*) are placed in an urn. Each iteration consists of drawing a ball randomly from the urn and then replacing it along with an additional ball of the same color. After placing a set number of balls in the urn, the proportions of balls of each color *I* is found. As mentioned in the previous chapter, this can easily be applied to language modeling. The balls of different colors are replaced with cards that have a word written on them. Every time a word is drawn we replace it in the urn along with another card with the same word. After several iterations the probability of each word *I* can be found as the proportion of the total number of cards. These steps are outlined by Frigyik et al. (2010) in Figure 4.

The stick breaking method depicts a Dirichlet distribution as a stick of length one broken into *k* different pieces. For an example, assume $k = 3$. A temporary variable, $\{v_i\}$, is used and initially generated from a Beta distribution, $v_1=q_1=Beta(\alpha_1, \alpha_2+ \alpha_3)$.. This represents the first break in the stick and therefore the remaining length of the stick is given by $1-v_1$. Next,



"**Step 1:** Set a counter n=1. Draw $X_1 \sim \alpha/\alpha_0$. (Note that α/α_0 is a non-negative vector whose entries sum to 1 so it is a pmf.

Step 2: Update the counter to n+1. Draw $X_{n+1} \mid X_1, X_2, \ldots, X_n \sim \alpha_n/\alpha_{n0}$ where $\alpha_n = \alpha + \sum_{i=1}^{n} \delta_{X_i}$ and α_{n0} is the sum of the entries α_n. Repeat this step an infinite number of times. "

Figure 4 – A diagram depicting the Polya urn method (Frigyik et.al., 2010).

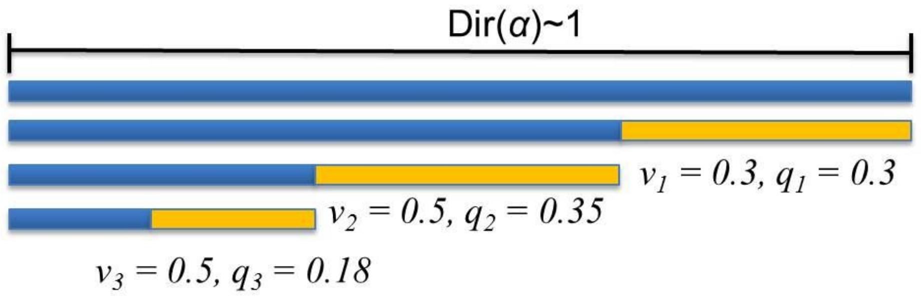
$v_2 = \left(\frac{Q_2}{1-Q_1} \mid Q_1 \right)$ is generated from *Beta (α_2, α_2)* and set $q_2 = (1-v_1)\, v_2$. For the case of $k=3$ the final vector is then $q_i = [v_1, (1-v_1)v_2, (1 - v_1) - (1-v_1)v_2]$. Frigyik et al. (2010) summarizes these steps and generalizes them for any value of *k* in Figure 5. The stick breaking representation is particularly useful for acoustic modeling as one can imagine the initial, whole stick as a single Gaussian model. Additional sticks, or mixture components, are broken off until additional breaks no longer minimize a cost function and eventually the optimal structure of the model is found.

Two interesting features of a Dirichlet distribution are the agglomerative and decimative properties (Teh, 2007). The agglomerative property indicates that two stick breaks can be collapsed into one piece if the two respective concentration parameters are also added together:

$$(q_1 + q_2, q_3, \ldots, q_k) \sim Dir(\alpha_1 + \alpha_2, \alpha_3, \ldots, \alpha_k)\ . \tag{13}$$

The decimative property, on the other hand describes that a given stick break can be further broken such that:

$$\begin{gathered}(q_{11}, q_{12}, q_2, \ldots, q_k) \sim Dir(\alpha_1\beta_1, \alpha_1\beta_2, \alpha_2, \ldots, \alpha_k) \\ \beta_1 + \beta_2 = 1\ .\end{gathered} \tag{14}$$



"Step 1: Simulate $v_1 \sim Beta\left(\alpha_1, \sum_{i=2}^{k} \alpha_i\right)$ and set $q_1 = v_1$. This is the first piece of the stick. The remaining piece has length $1 - v_1$.

Step 2: For $2 \leq j \leq k-1$, if $j-1$ pieces with lengths, $v_1, v_2, \ldots, v_{j-1}$, have been broken off, the length of the remaining stick is $\prod_{i=1}^{j-1}(1-v_i)$. We simulate $v_j \sim Beta\left(\alpha_j, \sum_{i=j+1}^{k} \alpha_i\right)$ and set $q_j = v_j \prod_{i=1}^{j-1}(1-v_i)$. The length of the remaining part of the stick is:

$$\prod_{i=1}^{j-1}(1-v_i) - v_j \prod_{i=1}^{j-1}(1-v_i) = \prod_{i=1}^{j}(1-v_i).$$

Step 3: The length of the remaining piece is q_k."

Figure 5 – A diagram depicting the stick breaking representation of a Dirichlet process.

2.1.2 Dirichlet Processes

A Dirichlet process is a stochastic process parameterized using a base measure, *H*, which serves as the center of the nonparametric distribution (essentially the mean) (Antoniak, 1974), and a concentration, α (where $\alpha > 0$). Unlike the Dirichlet distribution's α_i, where $i = 1, 2 \ldots k$ and whose values are discrete, Dirichlet processes are parameterized by a continuous function across the sample space, $\alpha(\chi)$. A Dirichlet process can be viewed as a Dirichlet distribution that has been decimated infinitely many times. Thus, drawing from a Dirichlet process yields a discrete random distribution. This can be seen in Figure 6 where initially $Dir(\alpha) \sim 1$ and then it is decimated, or split, infinitely many times to generate discrete values. Notice that each split is done in such a way that the overall density remains unchanged.

One way to interpret a Dirichlet process is to compare it to a dartboard (Frigyik et al., 2010). If we assume the dartboard is the infinite sample space, and a realization from the

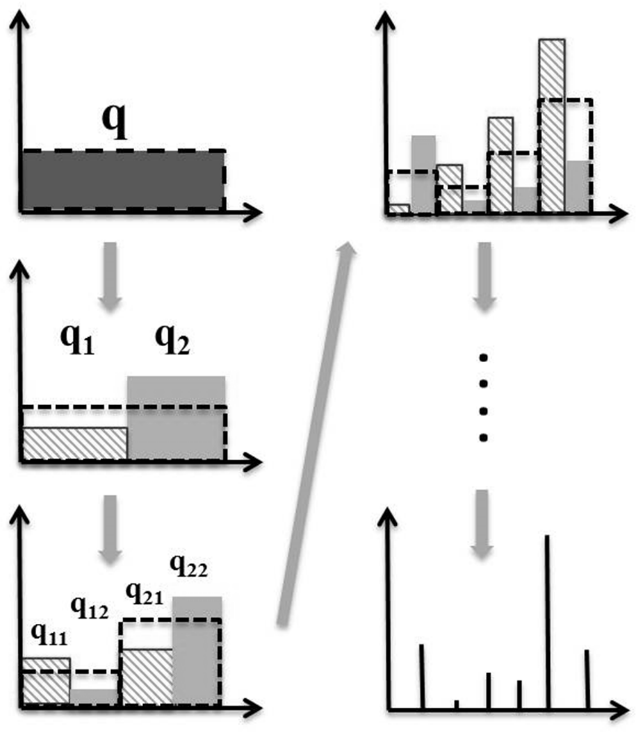


Figure 6 – A diagram showing how a Dirichlet distribution initially characterized by a uniform distribution can be decimated infinitely many times to yield discrete values.

Dirichlet process is a distribution characterized by an infinite set of darts of various lengths, then the length of each dart represents the weight given to that distribution. These weights are constrained such that the sum of all weights must be equal to one. More formally:

$$P(B) = \sum_{k=1}^{\infty} p_k \delta_{yk}(B) \tag{15}$$

where B represents a set of the infinite sample space, p_k represents the darts' weights, $\delta_{yk}(B)$ indicates the location of the k^{th} dart ($\delta_{yk} = 1$ if $y_k \epsilon B$ and $\delta_{yk} = 0$ otherwise), and $\sum_{k=1}^{\infty} p_k = 1$.

Another nice example is given by Frigyik et. al. (2010) where a Dirichlet Process is compared to polling a group of people several times for their favorite color. Depending on someone's mood, each person may give a different answer depending on the day. We can therefore treat each person as a separate pmf and model the probability of a given color being their favorite. The colors they can choose are not specified and could be anything from red to sky blue to sea-foam green. Thus, an infinite sample space, i.e. colors, is modeled over another infinite sample space, people. If the Dirichlet process is not split infinitely many times, however, it can be modeled as a Dirichlet distribution, i.e. limit the selection of colors. In the example of modeling people's favorite colors, the range of all (infinite) possible responses can be categorized into *M* distinct choices.

To generate samples from a Dirichlet process, the same procedures for a Dirichlet distribution are used with a few small changes. The first, described above, is the Polya urn method (also known as the Chinese restaurant process). The major difference from the above method is that there are now an infinite number of ball colors and the urn is initially empty. Or, in the case of the language model example mentioned above, the original set of words is unknown and there are infinitely many possibilities. Frigyik et. al. (2010) describes the steps for generating samples below by first setting $n=1$ and then:

"**Step 1:** Pick a new color with probability distribution $\alpha/\alpha(\chi)$ from the set of infinite ball colors. Paint a new ball that color and add it to the urn.

Step 2: With probability $\frac{n}{n+\alpha(\chi)}$ pick a ball out of the urn, put it back with another ball of the same color, and repeat Step 2. With probability $\frac{\alpha(\chi)}{n+\alpha(\chi)}$, go to Step 1."

Thus a random sequence of colors, or words, $(X_1, X_2, \ldots)$ is drawn from the *set* $(y_1, y_2, \ldots, y_k, \ldots, y_\infty)$. Frigyik et al. (2010) continues to explain that after the first *n* draws there are m_k occurrences of *K* different colors, $(y_1, y_2, \ldots y_k)$ then the next observation is:

$$X_{n+1} \mid X_1, \ldots, X_n \sim \sum_{k=1}^{n} \frac{m_k}{\alpha(\chi)+n} \delta_{y_k} + \frac{1}{\alpha(\chi)+n} \alpha \ . \tag{16}$$

To generate samples from a Dirichlet process using the stick-breaking method, the distributions $\{p_k, y_k\}$ must be characterized. To simplify matters though, $\{\theta_k, y_k\}$ will be used instead (measure theory can be used to prove the relationship between $\{p_k, y_k\}$ and $\{\theta_k, y_k\}$ but is beyond the scope of this work). The following steps are followed:

Step 1: Let $p_1 = \theta_1$. Thus the stick (originally of length 1), now has a length of $1- \theta_1$.

Step 2: Break off a fraction of the remaining stick, θ_2. Now, $p_2 = \theta_2(1\text{-}\theta_1)$ and the length of the remaining stick is $(1\text{-}\theta_1)\ (1\text{-}\theta_2)$. If this is repeated k times, then the remaining stick's length is given by $\prod_{i=1}^{k-1}(1-\theta_i)$ and $p_k = \theta_k \prod_{i=1}^{k-1}(1-\theta_i)$.

Step 3: Finally the probability distribution can be found by using (15).

2.2 Variational Inference

The process of making clustering or classification predictions using complex multivariate distributions is often intractable. Instead inference algorithms are used to analyze samples from distributions in order to generate an approximation which in turn can be used to make the necessary prediction. Markov chain Monte Carlo (MCMC) methods such as Gibbs sampling are extremely popular for their low mathematical complexity (Neal, 1991; Paisley, 2010; Rasmussen, 2000). These methods approximate complex posteriors by sampling latent variables from a Markov chain that represents the distribution of interest (Blei & Jordan, 2005). Unfortunately, converging to optimal posterior approximations is often slow and these methods can become intractable for big data problems such as speech recognition (Paisley, 2012; Blei & Jordan, 2005). Consequently, newer variational inference algorithms have been introduced which do not require sampling latent variables but still yield comparable results with more reasonable computation times.

The problem variational inference solves is simply described by Eisner (2011): the calculations to generate predictions using a posterior distribution $p(y|x)$, where y represent outputs and x is an input, is intractable. The solution is to use a simpler distribution that makes more independence assumptions, $q(y)$, to approximate $p(y|x)$. This can be handled as an optimization problem, i.e. minimizing an objective function, where an optimal q is found from a set of distributions $Q=\{q_1, q_2, \ldots, q_m)$. This inherently requires more complex computation but is still much faster than MCMC methods.

In the next few subsections, the three variational inference algorithms used in this work are described. Each is fairly similar. All have been shown to produce results comparable to Gibbs sampling but are significantly faster (Harati et al., 2012).

2.2.1 Accelerated Variational Dirichlet Process Mixtures (AVDPM)

Mean-field techniques (Blei & Jordan, 2006) are commonly used in variational inference. Mean-field approximation makes assumptions about a joint distribution's dependencies. For example, a joint distribution *p(a,b,c,...)*, could be approximated as (Eisner, 2011):

$$p(a,b,c,\ldots) = p(a,b) * p(b,c) * p(a,c) * \ldots \tag{17}$$

The marginal log probability is found by either minimizing the Kullback-Leibler (KL) divergence or by maximizing the lower bound *L(q)* (Harati et al., 2012):

$$\begin{aligned} \ln P(x) &= L(q) + KL(q \parallel p) \\ L(q) &= \int q(y) \ln \left\{ \frac{P(x,y)}{q(y)} \right\} dy \\ KL(q \parallel p) &= -\int q(y) \ln \left\{ \frac{P(y \mid x)}{q(y)} \right\} dy. \end{aligned} \tag{18}$$

The EM algorithm is used for convergence to create the best set of T variational distributions (i.e. approximations), $Q = \{q_1(y), q_2(y), \ldots, q_T(y)\}$, of the posterior, $p(y|x)$.

Following Kurihara et. al. (2006), a DPM makes the following assumptions:

1. There are an infinite number of components (i.e. distributions), $H = \{\eta_i\}_{i=1}^{\infty}$, that are taken independently from a prior with hyperparameter λ, $p_\eta(\eta_i/\lambda)$.
2. $V = \{v_i\}_{i=1}^{\infty}$ represents an infinite number of stick lengths that are drawn from another prior with hyperparameters α, $p_v(v_i/\alpha)$. These represent mixing weights $\{\pi_i\}_{i=1}^{\infty}$ for the mixtures $\pi_i(V) = v_i \prod_{j=1}^{i-1}(1-v_i)$ for $i = 1, \ldots, \infty$.
3. An observation model $p_x(x|\eta)$ is used to generate a data point from distribution η.
4. For dataset $X = \{x_n\}_{n=1}^{N}$, each x_n is generated from $p_x(x_n|\eta_k)$ by assigning a component label $z_n = k \in \{1, \ldots, \infty\}$ where $p(z_n=k \mid V) = \pi_k(V)$.
5. The DPM has a set of latent variables, $W = \{H, V, Z\}$ and hyperparameters $\theta = \{\lambda, \alpha\}$.
6. Class prediction relies on finding $p(W|X, \theta)$ which is intractable so a new set of variational distributions q_{AVDPM} is defined as

$$q_{AVDPM} = \prod_{i=1}^{L}\left[q_{v_i}\left(v_i;\phi_i^v\right)q_{\eta_i}\left(\eta_i;\phi_i^\eta\right)\right]\prod_{n=1}^{N}q_{z_n}(z_n)\ . \tag{19}$$

In contrast to (Blei & Jordan, 2006), who truncate (19) with the stipulations that $L=T$, $q_{v_i}(v_T=1)=1$, and $q_{z_n}(z_n>T)=0$, AVDPMs allow L to approach infinity but set any variational distributions equal to their priors for any $i > T$.

Furthermore, binary trees known as KD trees (Bentley, 1975), are used as a preprocessing step to split feature vectors along hyperplanes. These trees consist of a root node that contains all data points and child nodes that contain subsets of their parent's data points. Every training sample in a given leaf node shares the same variational distribution $q_{z_n}(z_n)$, so the incorporation of the KD tree reduces the amount of computation from *O(N)* to *O(A)*, where *A* is the number of leaf nodes in the tree. According to Kurihara (2006) $q_{z_n}(z_n)$, which was originally proportional to $E_{q_{\eta_i}}[\eta_i]^T x_n$ without the KD tree, becomes proportional to $E_{q_{\eta_i}}[\eta_i]^T \langle x_n\rangle$ where $\langle x_n\rangle$ is the average of all data in node *A*. Nodes are then split such that (18) is minimized. This ensures that nodes are theoretically split correctly. The number of times a parent node is split can be adjusted which allows the user to control the tradeoff between computational resources and model accuracy (Kurihara et al., 2006). This is extremely beneficial for speech recognition tasks where large training corpora are both common and necessary and gives AVDPM a huge advantage over the other inference algorithms.

2.2.2 Collapsed Variational Methods

A truncated stick-breaking model is used in CVSB where the joint density is given by:

$$P\left(X,z,v,\eta\right)=\left[\prod_{n=1}^{N}p\left(x_n\mid\eta_{z_n}\right)p\left(z_n\mid\pi(v)\right)\right]\left[\prod_{t=1}^{T}p(\eta_t)B(v_t;1,\alpha)\right] \tag{20}$$

where *B(*v_t*; 1, α)* is a Beta distribution for *v*. It is important to note that if cluster labels are changed in this model, the probability of (20) will also be altered. CDP, on the other hand, does allow for interchangeable cluster labels by limiting the model to K clusters and by setting a symmetric prior D on the mixing weights, $\pi \sim D\left(\pi, \frac{\alpha}{K}, \ldots, \frac{\alpha}{K}\right)$. The equivalent joint density for CDP is then:

$$P(X, z, \pi, \eta) = \left[\prod_{n=1}^{N} p\left(x_n \mid \eta_{z_n}\right) p\left(z_n \mid \pi\right)\right]\left[\prod_{t=1}^{K} p(\eta_t)\right] D\left(\pi; \frac{\alpha}{K}, \ldots, \frac{\alpha}{K}\right) \tag{21}$$

In either CVSB or CDP, the mixing weights can be marginalized out to produce the collapsed density (Kurihara, Welling, & Teh, 2007):

$$P(X, z, \eta) = \left[\prod_{n=1}^{N} p\left(x_n \mid \eta_{z_n}\right)\right] p(z) \left[\prod_{t=1}^{K} p(\eta_t)\right] \tag{22}$$

where the distributions over cluster labels *p(z)* are different for the CVSB and CDP models due to the interchangeability (or lack thereof) of cluster labels. Finally, the lower bound used in variational inference is given by:

$$\mathcal{L}(X) \geq \sum_{z} \int_{d\theta} Q(z) Q(\theta) \log \frac{P(X, z, \theta)}{Q(z) Q(\theta)} \tag{23}$$

where θ represents $\{\eta, v\}$, $\{\eta, \pi\}$, or $\{\eta\}$ depending on whether (20), (21), or (22) is used. The corresponding variational distributions are given by:

$$q_{CVSB}(z, \eta, v) = \left[\prod_{n}^{N} q(z_n)\right]\left[\prod_{t=1}^{T} q(\eta_t) q(v_t)\right] \tag{24}$$

$$q_{CDP}(z, \eta, \pi) = \left[\prod_{n}^{N} q(z_n)\right]\left[\prod_{k=1}^{T} q(\eta_k)\right] q(\pi) \tag{25}$$

where $q(v_t)$ and $q(\pi)$ are marginalized out of (24) and (25) respectively if using the collapsed density shown in (22). Equations (24) and (25) only differ by the replacement of $q(v)$ by $q(\pi)$. The i^{th} stick break, v_i, represents the fraction of the remaining stick length and is modeled with a Beta distribution while π_i is the actual mixture weight (i.e. the fraction of the original, whole stick).

Since the length of each stick break is held constant for CDP, the effect from the stick lengths can be removed from the product in (24) and replaced by $q(\pi)$.

Although these three inference algorithms are very similar, there are a few key differences worth highlighting. First, unlike CVSB and CDP, AVDPM incorporates KD trees as a preprocessing step. This splits feature vectors across hyperplanes that theoretically should increase the rate of convergence during the optimization of the KL divergence (or lower bounds mentioned in the previous sections). Secondly, AVDPM allows for a potentially infinite number of stick breaks such that any split after T breaks are tied to their priors rather than estimated. Conversely CVSB and CDP both have rigid truncation levels where only a finite number of stick breaks are allowed. CVSB imposes no constraints on the lengths of each break while CDP uses a symmetric prior. This forces the length of each stick break to be equal, i.e. the mixture component weights are equal, and, in essence, reduces the problem of the Dirichlet process to a Dirichlet distribution.

CHAPTER 3

DATA AND EXPERIMENTS

One of the overarching goals of this work is to move towards developing a speech recognition system that is robust to variations in the acoustic channel and can be adapted for use with multiple languages. This chapter will highlight some of the key differences between Mandarin Chinese and English, explain why the TIMIT and CALLHOME corpora were selected for this work, and describe some of the relevant characteristics of these corpora. Following this is a detailed explanation of the experimental setup including a description of the baseline algorithms used as well as the variational inference algorithms discussed in Chapter 2.

3.1 A Comparison of English and Mandarin

With the advent of China's rapid economic development over the past few decades, Mandarin has become a language of growing interest in the speech recognition community. A study has shown that the world contains approximately 350 million native English speakers compared to Mandarin's one billion (Chen et al., 1994). Moreover, there are at least as many English language learners in China as there are native speakers in the world (Chien et al., 1995). Although these statistics are somewhat dated and the populations have since grown dramatically, they highlight the increasingly pressing demand for high performance Mandarin speech recognition systems.

Unfortunately, there is often a large disparity between speech recognition performance for English and Mandarin datasets. This is particularly apparent for conversational telephone speech (CTS) data sets. One study has shown that two comparable CTS corpora yielded word error rates (WERs) of *42.7%* for Mandarin and *17.5%* for English (Schwartz et al., 2004). In the following paragraphs major differences between the two languages will be discussed to highlight some of the key difficulties in Mandarin speech recognition.

While words in English are created using a phonetic alphabet, Chinese words consist of one or more syllables represented by Chinese characters. Approximately 8,000 characters compose as many as 200,000 of the most common words in Mandarin. Furthermore, unlike English, whose words are segmented, i.e. separated by a space, words in Mandarin are often not delimited. Because of this, it is up to the user to determine which characters belong to a given word. Although this is not always the case, it can greatly affect our ability to train certain elements of a speech recognition system, such as the language model.

Each Chinese character is a monosyllabic morpheme and is assigned a specific tone. While tone generally represents a speaker's emotion in English, Mandarin tones specify word meaning. Two characters with the same phonetic syllable but with different tones represent two different words. Four distinct tones and one neutral tone exist in Mandarin Chinese as shown in Table 1.

Although each character has a set tone associated with it, Mandarin is highly susceptible to effects from coarticulation and thus a character's tone can change depending on the surrounding context. One common example of this is if a word consists of two consecutive characters that have the third tone in which case the first character's tone is changed to the second tone. Another such example occurs when the character used to negate meaning, "不" (phonetically written as"bu4" where “4” represents the character’s tone), precedes another

Table 1 – A description of the different tones in Mandarin Chinese. Words in parentheses indicate examples of English words that share similar sounds with Mandarin tones.

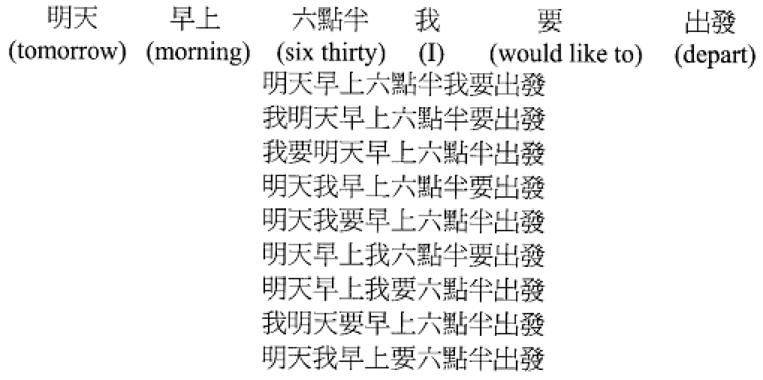
Tone Number	Description
1	High, constant pitch (“Do” i.e. Do Re Mi Fa So La Ti Do)
2	Rising inflection (“Huh?”)
3	Lower but slightly rising pitch (“Ugh”)
4	Rapidly descending pitch (“No!”)
5	Neutral - Short and without inflection

character with the fourth tone. In such an example the tone of "不" is changed to the second tone.

Furthermore, Mandarin has just over 400 unique syllables ignoring tone (or about 1300 with tones) compared to English's 10,000 syllables (Gu, et al., 2006). Consequently, Mandarin has an extremely large number of homophones compared to English. This creates the need for a more developed language model during the decoding phase to be able to discern characters correctly.

Creating a strong language model is difficult since Mandarin has an extremely flexible grammatical structure. Long phrases are often interchangeable with shortened versions consisting of only one or two characters. For example, the phrase for Beijing University, "北京大学" (Bei3 jing1 da4 xue2) is often abbreviated to "北大" (bei3 da4). Another, more significant example is shown in Figure 7 (Lee, 2006). This depicts an example of how word order can easily be altered without affecting the overall sentence meaning. This can reduce the efficacy of *N*-gram type language models and therefore require incorporating more advanced techniques such as neural networks or random forests (Oparin et al., 2010).

The possible lack of a segmented lexicon, the need to model tones as well as phonetic sounds, the high number of homophones, and the incredibly flexible grammatical structure of Mandarin make it extremely difficult for Mandarin speech recognizers to perform as well as their



明天	早上	六點半	我	要	出發
(tomorrow)	(morning)	(six thirty)	(I)	(would like to)	(depart)

明天早上六點半我要出發
我明天早上六點半要出發
我要明天早上六點半出發
明天我早上六點半要出發
明天我要早上六點半出發
明天早上我六點半要出發
明天早上我要六點半出發
我明天要早上六點半出發
明天我早上要六點半出發

Figure 7 – An example showing several equivalent sentences using different word orders (Lee, 2006).

English counterparts. Many of these challenges directly affect a typical speech recognizer's decoding process, specifically the predictive power of the language model. By including at least one contrasting language in our evaluations, we are reducing the probability that our conclusions drawn are specific to artifacts present in any one language.

3.2 Data

Three corpora were utilized in this work – TIMIT, CH-E, and CH-M – to investigate the performance of the inference algorithms. This section describes some of the key characteristics of each corpus, lexicon information, and the partitioning of the data into training, development, and evaluation data sets.

3.2.1 TIMIT

Although this work will focus more on results collected from the CALLHOME corpora, it is important to ensure that the experimental setup is reasonable. To accomplish this task, the well-calibrated TIMIT Corpus (Lamel et al., 1986) was selected. This dataset was originally created by a collaboration between Texas Instruments (TI), the Massachusetts Institute of Technology (MIT), and the National Institute of Standards and Technology (NIST). The data consists of read speech for three sentence types as shown in Table 2. The dialect sentence type is used to highlight variations in speakers' dialects while the compact and diverse types highlight phonetically comprehensive data. As is commonly practiced (Lee & Hon, 1988; Halberstadt & Glass, 1998; Ager et al., 2009), the 1260 dialect sentences are ignored for classification. The

Table 2 – A description of the speakers and utterances found in the TIMIT Corpus.

Sentence Type	# of Sentences	# of Speakers	Total Sentences	# Sent. / Speaker
Dialect (SA)	2	630	1260	2
Compact (SX)	450	7	3150	5
Diverse (SI)	1890	1	1890	3
Total	2342	NA	6300	10

Table 3 – A description of the TIMIT training, validation, and evaluation (Core Test) sets used for this work.

Set	# of Sentences	# of Utterances	# of Hours	# of Phonemes
Train	462	3696	3.14	140,225
Development	50	400	0.34	15,064
Core Test	24	192	0.16	7,215

corpus also comes with a manually generated phoneme alignment that is used for a baseline comparison.

The data is divided into training, validation, and evaluation sets using the common partitioning shown in Table 3 (Halberstadt & Glass, 1998; Ager et al., 2009). It should be noted that for this study *3* utterances are missing from the training set due to corrupted files during the acquisition of the data so there are slightly fewer examples than noted above. The training set in this work actually contains *140,078* phoneme tokens (note that the number of tokens in each set listed above excludes the glottal stop, "q", as will be mentioned in the following sections).

3.2.2 The LDC's CALLHOME Corpora

A major goal of this project is to compare and contrast performance on phoneme classification for both non-tonal and tonal languages. For such a comparison, it is necessary to use data for both languages that was collected from comparable recording environments and contain similar content. For this reason, the CALLHOME English (CH-E) and CALLHOME Mandarin (CH-M) corpora were selected. Although conversational telephone speech is not normally used for phoneme recognition due to noisy backgrounds and overlapping speech, this choice of data will provide unique insight into the robustness of DPM classification methods to data from more difficult recording environments and from more casual spoken language.

CH-E (Canavan et al., 1997) consists of *120* unscripted transoceanic telephone calls that last up to *30* minutes. Overall, *200* calls were recorded, *80* of which were assigned to the training set, *20* of which were assigned to the development set and *20* of which were assigned to the evaluation set. The remaining *80* calls were held by the LDC for future speech recognition benchmark tests. The audio was sampled at a rate of *8* kHz using a *2*-channel µlaw format and encoded using Cambridge's SHORTEN format (Robinson, 1994). Ten-minute segments from calls from each of the training and development sets were chosen to be transcribed while five-minute segments were selected from calls from the evaluation set.

CH-M was recorded under identical circumstances as CH-E and consists of unscripted conversations that last up to *30* minutes. All calls originated in North America and were made to people overseas. Again, *200* calls were recorded and transcribed but *80* were reserved for future benchmarks. The remaining calls are divided into training, development, and evaluation sets consisting of *80*, *20*, and *20* calls respectively.

3.3 Data Preparation

This section describes the steps taken to prepare the data for our classification experiments followed by detailed explanations of each individual experiment. At this point it should be noted that the experiments discussed below include both phone classification and recognition problems. Phone classification is defined as the use of manually aligned phone segmentations. Phone recognition refers to the use of automatically generated phone alignments. While the TIMIT Corpus includes manual phone segmentations, the CH-E and CH-M corpora require that we perform recognition since they do not include reference segmentations. However, recognition is also performed on TIMIT to better illustrate the comparison between well calibrated data and the conversational telephone speech contained in the CALLHOME corpora. The general procedure used for these experiments is outlined below:

1. Convert each utterance to MFCC features.
2. Format dictionaries and transcripts to ensure all words are included.
3. Remove features that pertain to simultaneous speakers or only noise.
4. Train an acoustic model using an HMM with 16 Gaussian mixtures.
5. Use a Viterbi alignment to generate phoneme alignments.
6. Partition MFCC features from each utterance to pertain to individual instances of phonemes.

7. Use 3-4-3 averaging across all frames pertaining to each phoneme instance. Now feature vectors are all of equal length despite different durations and are phoneme based instead of frame based.
8. Use classification algorithms to generate misclassification error on the phoneme-based feature vectors.

The following sections describe this process in more detail for each of the corpora.

3.3.1 TIMIT

As mentioned earlier, the TIMIT Corpus was selected for having an abundance of published results for phone classification and recognition. While the lack of reference segmentations limit the use of the CALLHOME corpora to phone recognition experiments, the TIMIT Corpus is complemented with phone alignments that allow for a clear comparison of the experimental setup to other baseline classification systems.

The manual phoneme alignments contain *61* labels that, as generally practiced (Lee & Hon, 1989; Halberstadt & Glass, 1998; Ager et al., 2009), are reduced to *48* labels for training. These classes are shown in Table 4 below (note that glottal stops, "*q*", are simply removed). Final classification occurs after these *48* classes are further reduced to *39* labels by merging the following classes: {*sil*, *cl*, *vcl*, *epi*}, {*el*, *l*}, {*en*, *n*}, {*sh*, *zh*}, {*ao*, *aa*}, {*ih*, *ix*}, {*ah*, *ax*}.

The raw audio was converted into *13* MFCCs and their first and second derivatives using the Hidden Markov Model Toolkit (HTK) ("HTK", 2009) (generating the typical *39* feature format) with frame and window durations of *10* ms and *25* ms respectively. Each instance of a phoneme was represented as a concatenated feature vector formed by combining all feature vectors associated with that instance of the phoneme (using time alignment information). The dimension of this concatenated vector is $n_i \, x \, 39$, where n_i is the duration in frames of the phoneme instance. This process was completed using both the manual alignments provided with the TIMIT Corpus and also with alignments generated automatically.

For the automatic alignments the *61* phoneme labels were collapsed into *40* classes. These *40* labels were identical to the collapsed set from the manual alignments with three

Table 4 – A chart of TIMIT's 61 phone labels after being folded into 48 classes for training.

Phone	Folded	Phone	Folded	Phone	Folded	Phone	Folded
iy	--	l	--	g	--	epi	--
ih	--	el	--	p	--	uw	ux
eh	--	r	--	t	--	er	axr
ae	--	y	--	k	--	m	em
ix	--	w	--	z	--	n	nx
ax	--	en	--	zh	--	ng	eng
ah	--	ch	--	v	--	hh	hv
uh	--	jh	--	f	--	cl	pcl, tcl, kcl
ao	--	dh	--	th	--	vcl	bcl, dcl, gcl
aa	--	b	--	s	--	sil	#h, pau
ey	--	d	--	sh	--		
ay	--	dx	--				
oy	--						
aw	--						
ow	--						

exceptions: all instances of *epi*, *cl*, and *vcl* were completely removed (and do not exist in the dictionary), *dx* was ignored since it does not exist in the dictionary, and finally {*aa*, *ao*} and {*sh*, *zh*} remained unmerged since they are prevalent in the dictionary. These changes were an artifact of the lexicon provided with the TIMIT Corpus (Ma, 2010). Finally, automatic alignments were obtained using an HMM-based speech recognition system with monophone acoustic models. Each state in these acoustic models used a GMM with *16* mixture components. This model was passed to a Viterbi-based time alignment algorithm to extract the start and stop times of individual phones from the audio data.

Because it is difficult to classify phones that have different durations, every instance of every phone was broken into a *3x39* vector of features using a 3-4-3 averaging technique. This involved first converting a phone's start and stop time to a corresponding frame number, and then averaging the features from the first *30%* of the frames, the middle *40%*, and the final *30%*. Each phone's frame duration was added as an additional feature such that our final feature vector had a dimension of *3x40*. In this way, we formatted the data such that every instance of each phone had a consistent number of features for classification.

3.3.2 CALLHOME English (CH-E)

Each CH-E call was recorded as a sphere audio file and compressed using SHORTEN (Robinson, 1994). These files were uncompressed into a two-channel μ-law format using *w_decode*. Because each phone call encapsulates so much data (up to thirty minutes of conversation), the original audio files were divided into smaller audio clips corresponding to single utterances. This was accomplished using the time stamps from the CH-E transcripts. Start and stop times were extracted in order to parse each call and to determine the number of speakers within a given utterance's time frame. Every clip was given a new utterance ID (in the form of originalFileName_clip#). In several instances multiple speakers talked simultaneously which made phone recognition extremely difficult without filtering to separate the audio channels. Since there was plenty of data, we chose to avoid this additional step and instead selected only audio from single speakers during the training process. All other utterances from the newly created audio clips that had overlapping speakers were discarded. The amounts of data before and after this step are shown in Table 5.

The original transcripts from the LDC were also reformatted. This is necessary to perform the forced alignment to obtain phone locations. The following actions were taken for this process:

1. Time stamps and speaker ID's are removed from the original transcripts.
2. Markers for proper names, non-lexemes (i.e. "uh", "um", "er", etc.), and utterance comments from transcribers are removed.
3. In some cases, certain audio files are distorted such that transcribers were forced to guess at the actual words. Rather than attempting to correctly identify these instances, they were simply replaced by a generic word, *"{garbled}"*, whose pronunciation is given by a garbage phone and treated as noise.
4. When a speaker makes a noise such as a laugh, cough, sneeze, sigh, etc., it is replaced by a generic word, *"{sound}"*, whose pronunciation is given by a garbage phone and treated as noise.
5. If a speaker voices a partial, unfinished word it is replaced by the generic word *"{partial}"*, whose pronunciation is given by a garbage phone and treated as noise.
6. Markers are removed from a word spoken from a language other than English. Many of these words already exist in the dictionary (i.e. "hola", "c'est la vie", etc.) and do not require special attention.
7. Any mispronounced words marked in the original files are transcribed as the intended word instead.
8. Any utterances that contain only distortions, partial words, or sounds are removed.

Table 5 – A table showing the amount of CH-E data before and after filtering out segments of overlapping speakers.

Set	Data Before Filtering (min)	Data After Filtering (min)
Training	767	308
Development	185	69
Evaluation	94	43

By following these conventions we avoided emphasizing uninformative noise and focused on speech instead. With these formatting steps complete, utterances and their new IDs were saved in a .dot format (i.e. *'utterance (utteranceID)'*).

The CMU7 dictionary ("The CMU Pronouncing Dictionary," 2008) was used as the primary lexicon for this task. It was augmented with a few common words (typically proper nouns). A list was compiled of every unique word found in the transcripts and compared to those in the augmented dictionary. Any words that were not found in the dictionary were added and given a pronunciation using a garbage phone (i.e. equivalent to noise). Approximately *500* words consisting primarily of uncommon proper nouns and idiosyncratic words were assigned the garbage phone out of a total of the total *8,545* words in the compiled dictionary. These modifications to the lexicon are available at *http://www.isip.piconepress.com/projects/dpm_inference*. Including the garbage phone, *sil*, and *sp*, this corpus contains a total of *42* unique phones.

Once the lexicon was compiled and transcripts reformatted, all of the newly created audio clips were converted into MFCC features following the same procedure outlined for the TIMIT Corpus in the previous section. However, only audio clips that contain no speaker overlap were selected for feature extraction and training. Features for each phone instance were again obtained using the automatic phoneme alignment generated from HTK and averaged in a 3-4-3 manner to ensure that each sample had the same number of features for classification.

3.3.3 CALLHOME Mandarin (CH-M)

The LDC's CALLHOME Mandarin (CH-M) Corpus was recorded and processed in an identical fashion to its English counterpart. The lengthy recordings were again divided into smaller audio clips corresponding to single utterances using the time stamps from the CH-M transcripts and given a new utterance ID (in the form of originalFileName_clip#). Again, any audio that contained simultaneous speech from multiple speakers was discarded. Finally, transcripts were reformatted following the same eight steps outlined above for CH-E. The amount of data before and after removing simultaneous speech is shown in Table 6.

The Mandarin lexicon was augmented since the transcripts contained some English words that were not originally present in the dictionary. Any phonemes pertaining to vowel sounds from these English words were given the neutral tone for their pronunciations. Furthermore, popular names, idiosyncratic words and phrases, and any other missing Mandarin words were given pronunciations by a native Mandarin speaker following the same phoneme labeling methodology used in the original CH-M lexicon. Any unidentifiable words in the transcripts were assigned pronunciations given by a garbage phone. These modifications to the lexicon are available at *http://www.isip.piconepress.com/projects/dpm_inference*. The corpus in total contains *92* phoneme labels. The CH-M audio data was then converted to features using the same approach described above for TIMIT and CH-E.

3.4 Baseline Algorithm Implementations

Most published work that utilizes the CALLHOME corpora focuses on a wide variety of

Table 6 – A table showing the amount of CALLHOME Mandarin data before and after filtering out segments of overlapping speakers.

Set	Data Before Filtering (min)	Data After Filtering (min)
Training	512	251
Development	127	67
Evaluation	69	31

tasks but does not directly involve phone recognition as presented in this work. Consequently, to ensure the validity of these baseline experiments, we also investigated algorithm performance on TIMIT. In this section, the general setups of the baseline experiments are described. All programming was completed in MATLAB in order to take advantage of its wide variety of built-in functions. It is worth noting here that for all algorithms the means and covariances of the training data were used to normalize the training, validation, and evaluation sets for each corpus such that they have zero mean and a standard deviation of one.

The neural network algorithm was run using a single hidden layer as can be seen in the example architecture shown in Figure 3. The network consisted of an input layer of *120* features (corresponding to an unrolled version of the averaged MFCC features mentioned above), a hidden layer with a varying number of neurons, and an output layer whose size corresponded to the number of possible phonemes. MATLAB's built-in "newff" function (Mathworks, 2013a) was used for this model. This neural network implementation used tangent sigmoid transfer functions between each layer and resilient back propagation to train the network (Bishop, 1995). The stopping criterion was set by *1000* maximum epochs or *20* consecutive epochs that failed to improve performance. For each network, the algorithm was run for *10* iterations and error rates were found for the number of neurons in the hidden layer ranging from *20* to *300*.

The random forest algorithm was implemented with MATLAB's built-in "TreeBagger" (Mathworks, 2013b) function which simply required inputting the number of trees used in the ensemble. This parameter was varied to determine an optimal error rate for this algorithm. MATLAB's implementation of the algorithm was computationally expensive (i.e. storing large ensembles requires very large amounts of memory) so error rates were averaged across only *5* iterations. Error rates were found as the number of trees in the ensemble were varied from *10* to *150*.

In *K*-nearest neighbors, MATLAB's built-in "knnsearch" function (Mathworks, 2013c) was used to calculate the Euclidean distance for up to *K=99* nearest neighbors. Output predictions were made for a given phoneme segment by selecting the mode of the labels in the *K* nearest neighbors. If multiple modes existed, the label with the closest average distance was selected for the output prediction.

The most important of the baseline algorithms used in this work was the GMM. Many HMM-based speech recognition systems are based on GMMs. Most GMM-based acoustic models specify a set number of mixture components to be used for every phoneme model. Therefore as a baseline experiment, the same method was applied here and error rates were found for different numbers of mixture components. A GMM was constructed using MATLAB's built-in "gmdistribution.fit" (Mathworks, 2013d) function which takes the number of components as an argument along with a few other parameters that specify initialization and covariance structure. To prevent errors created by ill-conditioned covariance matrices (which is not uncommon when using features that include first and second order derivatives), the covariance matrix was limited to be diagonal and a small value (1E-15) was added to the diagonal to ensure that it was positive-definite. The model parameters were then optimized using the EM algorithm. Output labels were generated by finding the model that yielded the maximum likelihood. Finally, the best error rates were taken from ten random initializations as the number of mixture components was swept from *1* to *64* for TIMIT and *16* to *192* for CH-E and CH-M.

3.5 Variational Inference Algorithm Implementations

As mentioned in Chapter 2, the three variational inference algorithms used in this work make use of the stick-breaking representation of a DP. This implies that each algorithm initially begins by assigning one mixture component to a given model and estimating the means and covariances of the variational distributions shown in (19), (24), or (25). Pieces of the stick, or in this case new mixture components, are broken off if doing so minimizes the cost function shown in (18). In these experiments, the means and covariances of the new components were re-estimated and this process continued until an optimal number of mixture components were found.

Thus, the models grew automatically to better fit the data. This process was repeated for each phoneme label and finally classification was conducted using maximum likelihood. Again, this is very different from the standard GMM where mixture components are fixed for each phoneme label.

The number of discovered mixture components and the amount of CPU training time used by each algorithm were also recorded to better assess their viability in acoustic modeling using MATLAB’s built in profiler and tic and toc functions (Mathworks, 2013e) Each algorithm was randomly initialized ten times and average error rates, number of mixture components, and CPU training times were recorded.

All data was normalized in the same method mentioned above for the baseline algorithms prior to training. The variational inference algorithms were implemented in MATLAB and were provided by Kurihara (2010). For all three algorithms, the default parameters were selected and set using the functions provided in the software package: mkopts_avdp, mkopts_csb, and mkopts_cdp for AVDPM, CVSB, and CDP respectively. Each algorithm had one additional parameter that was tuned. For AVDPM, this was the initial depth of the KD tree which can be used to control the balance between computational complexity and accuracy. CVSB and CDP, on the other hand, were investigated using a wide array of truncation levels that served as the maximum number of possible mixture components (i.e. stick breaks).

CHAPTER 4

RESULTS AND DISCUSSION

Chapter 4 focuses on the results found from the experiments using the baseline and variational inference algorithms described in Chapter 3. Optimal parameters for each system were found by tuning the system to minimize classification error (the proportion of phonemes labeled incorrectly to the total number of phonemes) on validation sets. We refer to this as the validation error. Since the depth of the KD tree affects the balance between computational resources and accuracy, and the truncation level for CVSB and CDP serves as a maximum number of mixture components, the computational complexity was investigated as these parameters were varied. This was evaluated by measuring the CPU time required to train models using each variational inference algorithm as the depth of the KD tree or truncation levels were changed. Optimal operating points were selected from these tuning experiments for each variational algorithm and the final misclassification error was tabulated (as shown in Table 7).

4.1 Baseline Algorithm Tuning

Having prepared the data following the steps outlined in Chapter 3, feature vectors and their corresponding labels were processed using a traditional pattern recognition paradigm in which each algorithm was trained on a designated set of data and then optimized on a held-out set of data. Sets of validation data for each corpus were used to select the optimal parameters.

We first optimized the neural network, random forest, *K* nearest neighbor, and GMM algorithms to establish solid baselines for performance. In Figure 8, we show neural network performance as a function of the number of neurons used in the network. It was found that this model performed best with *140, 170* and *130* neurons in the hidden layer for TIMIT, CH-E, and CH-M respectively. It can be seen that *150* neurons in the hidden layer is a good general operating point for these speech recognition experiments since it results in less than *0.25%*

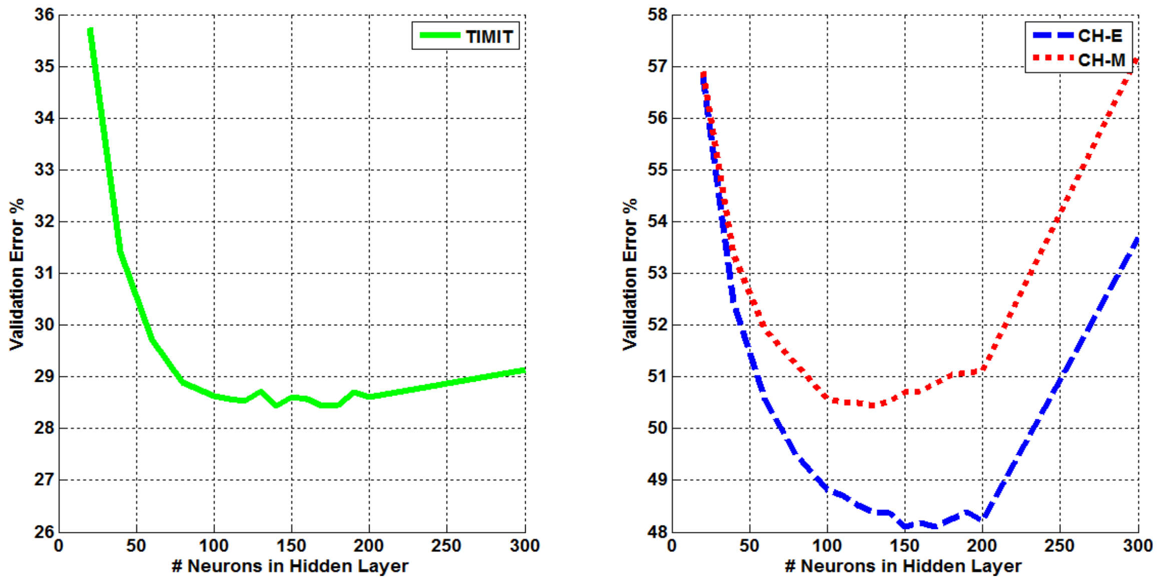


Figure 8 – The average misclassification error is shown for *10* iterations of the single hidden layer neural network model for TIMIT (left) and CH-E and CH-M (right). The optimal number of neurons was determined to be 150 for acoustic modeling tasks.

degradation in performance for any of the corpora, and delivers good performance across the range of recording environments and languages. Furthermore, the noisier conversational telephone speech of CH-E and CH-M display a greater susceptibility to overfitting than TIMIT. This is not unexpected since all three corpora have comparable amounts of data but CH-E and CH-M are recorded on a much noisier acoustic channel. Thus the models are partially being fitted to more noise which degrades misclassification errors.

Next, the random forest baseline system was optimized with respect to the number of trees, as shown in Figure 9. The error rate decreased monotonically with the number of trees used. Although there are more efficient implementations of this algorithm, the approach used here, based on MATLAB's TreeBagger function, is computationally expensive. Configurations larger than *150* trees were beyond the computational capabilities of our computing infrastructure, both in terms of CPU time and disk space. Hence, we selected *150* trees as an operating point for these experiments. However, the trends shown in Figure 9 indicate that with additional resources, random forests could potentially yield the best error rates. Thus an optimal operating point for

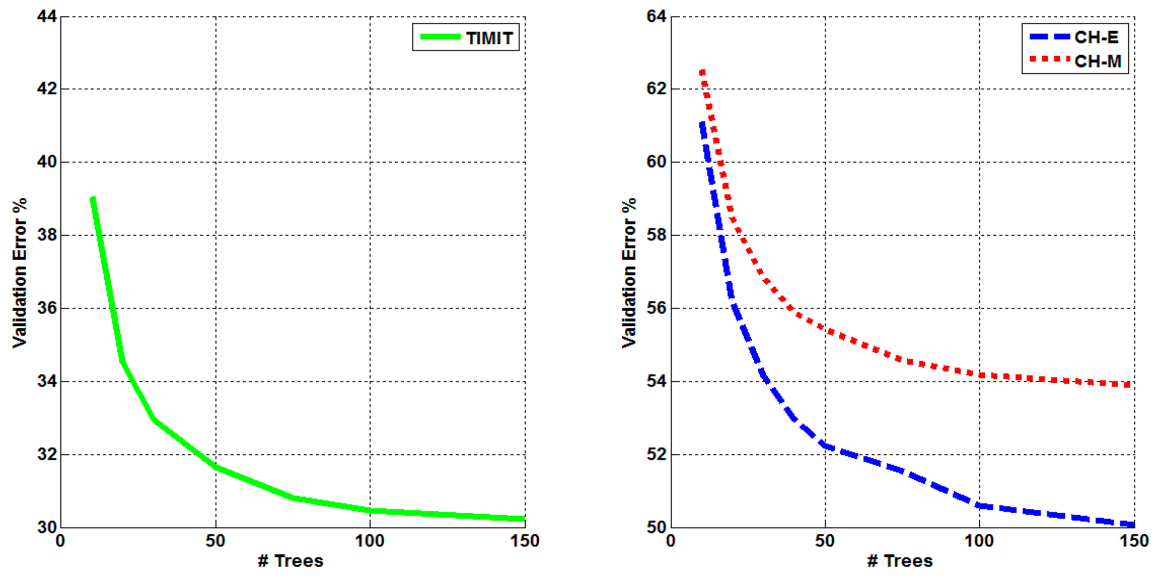


Figure 9 – The average misclassification error is shown for 5 iterations of the random forest algorithm for TIMIT (left) and CH-E and CH-M (right). An operating point of 150 trees was selected due to computational considerations.

this algorithm depends heavily on the implementation used, the amount of computational resources available, and the importance of good generalization (avoiding over-fitting).

The *K* Nearest neighbors algorithm showed significant variation in performance as a function of *K*. Results are summarized in Figure 10. Performance was optimal for *K=15* on TIMIT, *K=27* for CH-E, and *K=28* for CH-M respectively. This again indicates that simpler models (i.e. fewer neighbors for KNN) can be used for studio recorded read speech than noisier conversational telephone speech. From these results it is shown that selecting *K=20* for other speech recognition experiments would yield near optimal error rates since this results in less than *0.5%* absolute degradation in error rate for any of the corpora used in this work.

GMMs were examined closely since they are commonly used in speech recognition (i.e. they are used to model states in HMMs). Moreover, since the variational inference algorithms are used to find the optimal number of mixture components and their respective parameters, a GMM serves as an important comparison point for nonparametric Bayesian models. Furthermore, while neural networks, random forests, and *K* nearest neighbors have operating points that are fairly

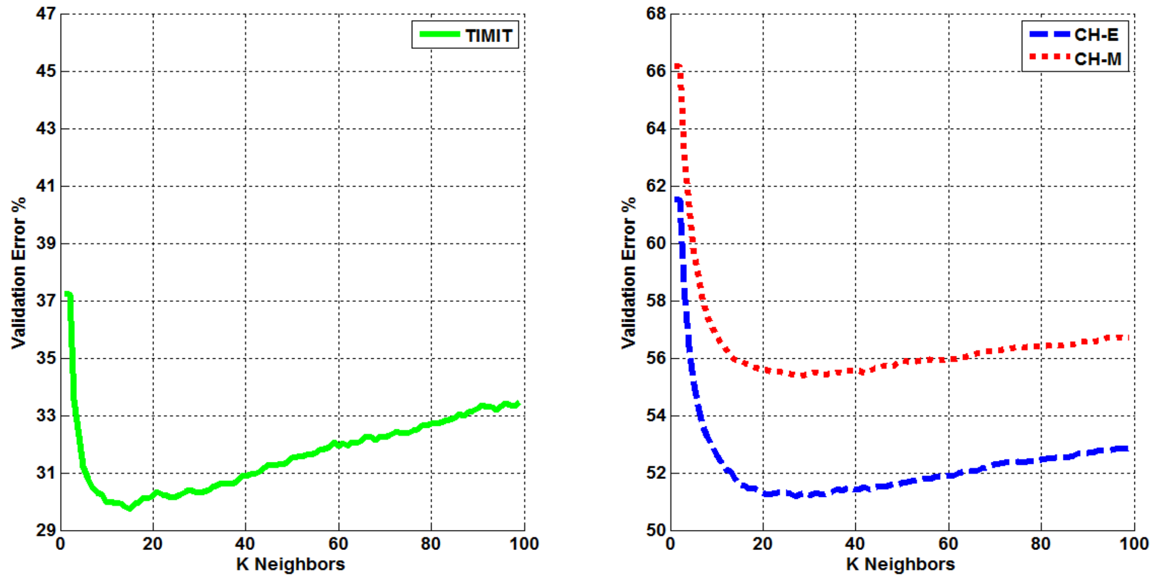


Figure 10 – The misclassification error for KNN as a function of *K*, the number of nearest neighbors, is shown for TIMIT (left) and CH-E and CH-M (right). The optimal operating point selected for acoustic modeling was *K = 20*.

similar for each corpus, the optimal performance of a GMM varies significantly across the three corpora. Results are summarized in Figure 11. We see that the optimal number of mixtures ranges from *8* for TIMIT to *64* for CH-M and *128* for CH-E. This emphasizes the importance of finding optimal numbers of mixture components for each phoneme label rather than assigning a set number for every class.

To confirm that the experimental setup yielded reasonable performance, features extracted using TIMIT's manual phoneme alignments were first used. It was found that this approach led to error rates between *30%-40%* that were very similar to the range of values found in other published work (Ager et al., 2009). However, as explained previously, this operating point is not a fair comparison point for the other corpora because we do not have manual segmentations for the data. Hence, in order to do a fair comparison across corpora, features were extracted from automatically generated segmentations. Performance for both the manual and automatic segmentations are shown in Figure 11.

The optimal number of mixture components for TIMIT was found to be *4* for the manual

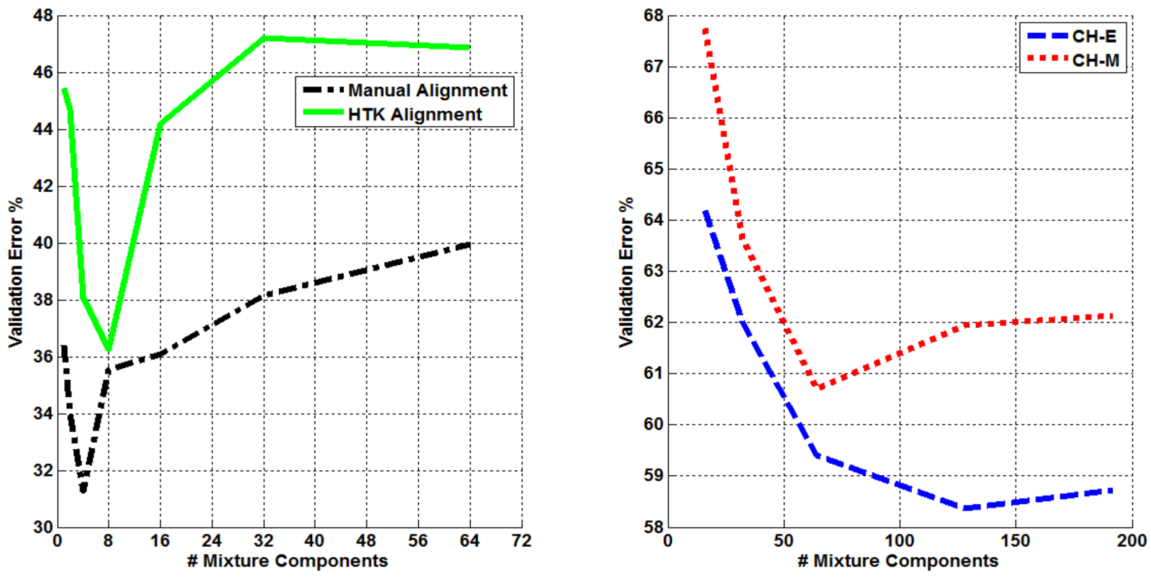


Figure 11 – The misclassification error for 10 iterations of a GMM is shown as a function of the number of mixture components. Results for TIMIT are shown on the left, and results for CH-E and CH-M are shown on the right. Though there is measurable degradation in performance when using automatically-generated alignments, the trends are consistent. Further, there are significant differences between the optimal settings for TIMIT and CALLHOME.

phoneme alignments, generating an error rate of *31.56%*. The automatically generated alignments produced an optimum at *8* mixture components, generating an error rate of *38.02%*. This indicates that despite some degradation in performance, the general trends remain unchanged. This suggests that the automatically generated alignments were suitable for this work. We have no alternative for large speech corpora since manual segmentations are not available. We often use a high performance speech recognition system operating in forced alignment mode to generate high quality segmentations for oracle-type studies that need segmentation information.

As expected, the clean studio recorded data of TIMIT yielded significantly better results than the noisy, conversational telephone speech of the CH-E and CH-M corpora. The gap in performance between CH-E and CH-M was never greater than *7.08%* for any baseline algorithm (i.e. KNN) and a minimum disparity of *4.24*% was found using GMMs. This shows that although producing worse error rates than other baseline algorithms, GMMs are better suited to generalize across different languages. Furthermore, the discrepancies in error rates can easily be attributed to the number of phoneme labels (*42* in CH-E and *92* in CH-M). Since both corpora have a similar amount of training, validation, and evaluation data, each phoneme model in CH-M is essentially trained on less than half the data of each phoneme model in CH-E. This in large part indicates that the general disparity in overall speech recognition performance is most likely due to a lack of accepted transcription segmentation or the flexible grammatical structure of Mandarin, not the acoustic models. The flexibility in sentence structure indicates that word order is more interchangeable than a language such as English, and that makes an *N*-gram language model significantly less effective. This is further exacerbated by the high number of homophones that would make it extremely difficult to determine more likely word orders.

4.2 Variational Inference Algorithm Tuning

The three variational inference algorithms were evaluated on the same three corpora. To tune these algorithms the initial depth of the KD tree was varied for AVDPM and the truncation level was varied for CVSB and CDP. These results are shown in Figure 12 and Figure 13. For AVDPM, a KD tree depth of *6* is optimum. For CVSB and CDP, a truncation level of *4* is optimum for TIMIT and *6* is optimum for the CALLHOME corpora.

For AVDPM, CH-M displayed the expected trend of improved error rates as the initial depth of the KD tree was increased. However, TIMIT and CH-E displayed anomalous increases in error rates. CVSB and CDP also showed strange behavior at a truncation level of *2* for CH-E and CH-M. To further investigate these phenomena "ideal" data was simulated from four bivariate GMMs and used to evaluate the algorithms. The results did not indicate any spike in error at low truncation levels and mirrored the trend found for TIMIT. This led to a hypothesis that these algorithms have difficulty finding good covariance matrices for noisier data that can cause degradations in performance.

Upon closer inspection it was found that when training on CH-E, some of the phones' mixture components were so sparsely populated that the covariance could not be directly calculated. Furthermore, using 3-4-3 averaging created feature vectors that were highly correlated, making direct covariance calculation difficult. Although these variational inference algorithms estimate covariances, they can produce bad approximations in such cases. To confirm that the spike in error was caused by this, all mixture components of a given phoneme were assigned a shared covariance matrix. As one would expect, general performance worsened but the spikes in error were removed, indicating that they were caused by poor estimations due to sparse mixture component populations. There are many techniques to avoid these problems, such as selecting only mixture components that are sufficiently populated or by independently calculating the covariance matrices using other methods, but this is beyond the scope of this work.

4.3 Evaluation Set Performance

A summary of results for all the algorithms studied in this thesis is shown in Table 7. With the exception of GMM, the baseline algorithms (NN, RF and KNN) yielded error rates that outperform the DPM-based approaches (AVDPM, CVSB, and CDP). NNs, for example, gave an error rate of *31%* on TIMIT, while the DPM-based approaches delivered error rates in the range

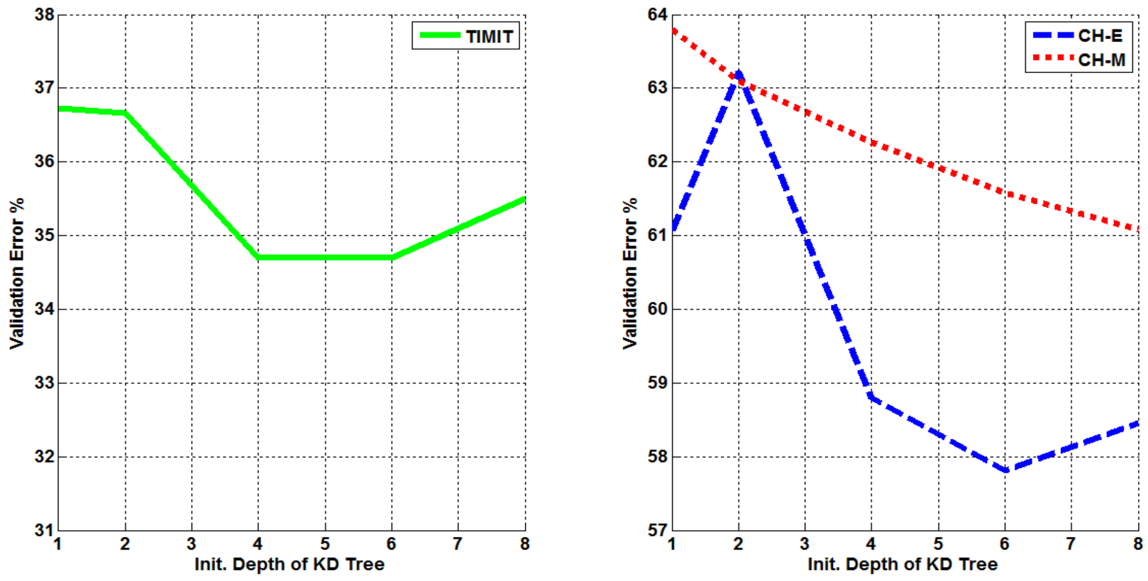
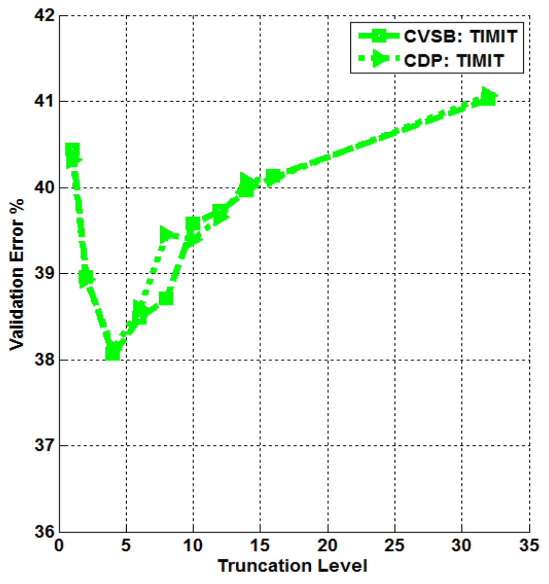


Figure 12 – The misclassification error rates are shown as the initial depth of the KD tree is varied for TIMIT (left), CH-E, and CH-M (right).



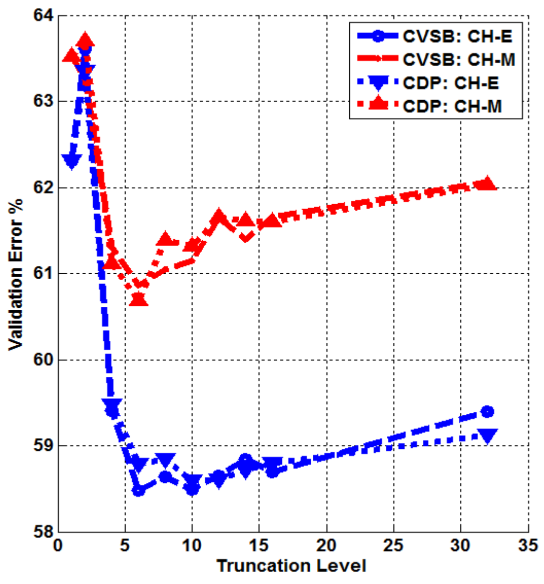


Figure 13 – The misclassification error rates for CVSB and CDP are shown as the truncation level is varied for TIMIT (left), CH-E, and CH-M (right).

of *37%* to *40%*. AVDPM and GMM delivered comparable performance, while CVSB and CDP were slightly worse on TIMIT and comparable on CALLHOME.

The baseline algorithms (NN, RF and KNN) are known to work well for static classification problems but can be less suitable for data that dynamically changes. GMMs are of particular interest since they are an integral part of an HMM. GMMs are known to perform well on time-varying data like speech when integrated into an HMM framework. Similarly, DPMs and the inference algorithms used in this work can be extended to hierarchical Dirichlet process HMMs (HDP-HMMs) (Harati & Picone, 2012; Harati & Picone, 2013) that use nonparametric Bayesian methods to automatically discover model structure. HDP-HMMs are currently limited to using MCMC methods and have not yet incorporated any of the variational inference algorithms proposed here (Harati et al., 2012). From Table 7 we conclude that the performance of AVDPM, CVSB and CDP are comparable when averaged across a wide range of conditions.

One major distinction between the GMM and DPM models is the significant decrease in the number of mixture components found by the DPM models. This is shown in the "Notes" column of Table 7. AVDPM, for example, found approximately *5* mixture components per phoneme label compared to GMM's *128*, a decrease of more than *2500%*. Again, this is due to the nature of the stick breaking representation of the DPM. Instead of arbitrarily selecting the number of mixture components for every phone class as employed by the parametric GMM approach in this work, new stick breaks (or mixture components) are broken off only if the cost function in (18) is minimized. In this way DPMs find the underlying structure of the data. Ad hoc strategies can be employed to make the number of mixture components in GMM adapt to the data, but these techniques are not integral to the algorithm.

4.4 Computational Complexity Analysis

Though Table 7 shows that the variational inference algorithms produce comparable error rates to the baseline GMM system, they do so using far fewer numbers of mixture components. This is significant since it supports our hypothesis that DPM approaches can control the complexity of a system and adapt to the complexity of the data. To apply these methods to speech recognition, though, it is important that they not only produce low error rates but also be

Table 7 – A comparison of misclassification error and number of mixture components for the evaluation sets of the TIMIT, CH-E, and CH-M corpora using automatically generated alignments.

Model	TIMIT		CH-E		CH-M	
	Error %	Notes	Error %	Notes	Error %	Notes
NN	30.54%	140 neurons in hidden layer	47.62%	170 neurons in hidden layer	52.92%	130 neurons in hidden layer
KNN	32.08%	K = 15	51.16%	K = 27	58.24%	K = 28
RF	32.04%	150 trees	48.95%	150 trees	55.72%	150 trees
GMM	38.02%	# Mixt. = 8	58.41%	# Mixt. = 128	62.65%	# Mixt. = 64
AVDPM	37.14%	Init. Depth = 4 Avg. # Mixt. = 4.63	57.82%	Init. Depth = 6 Avg. # Mixt. = 5.14	63.53%	Init. Depth = 8 Avg. # Mixt. = 5.01
CVSB	40.30%	Trunc. Level = 4 Avg. # Mixt. = 3.98	58.68%	Trunc. Level = 6 Avg. # Mixt. = 5.89	61.18%	Trunc. Level = 6 Avg. # Mixt. = 5.75
CDP	40.24%	Trunc. Level = 4 Avg. # Mixt. = 3.97	57.69%	Trunc. Level = 10 Avg. # Mixt. = 9.67	60.93%	Trunc. Level = 6 Avg. # Mixt. = 5.75

computationally efficient. For this reason the TIMIT Corpus was used to further investigate the computational complexity of each algorithm as a function of key algorithm parameters, i.e. initial KD tree depth or truncation levels, and also as a function of the amount of training data used.

AVDPM grows exponentially as the initial depth of the KD tree increases. Kurihara et al. (2006) states that the complexity of AVDPM is proportional to the number of outer nodes in the KD tree which is equivalent to $O(2^{depth})$ where a $depth=1$ represents a single split of the data into two outer nodes. The cost of building the KD tree is an additional $O(NlogN)$, though Kurihara et al. (2006) use optimization techniques to help mitigate this cost. Computational performance is summarized in the left plot in Figure 14. On first inspection AVDPM's performance seems problematic since large initial depths require significantly longer training times. Fortunately, Figure 12 indicates that relatively low initial depths can provide good performance: *initial depth* = *4* (TIMIT), *6* (CH-E), or *8* (CH-M). Additionally, if an initial depth of *4* is used for training the CH-E and CH-M corpora, there is only a *1.32%* and *1.14%* absolute degradation in error respectively. Since TIMIT and CALLHOME corpora are from very different recording

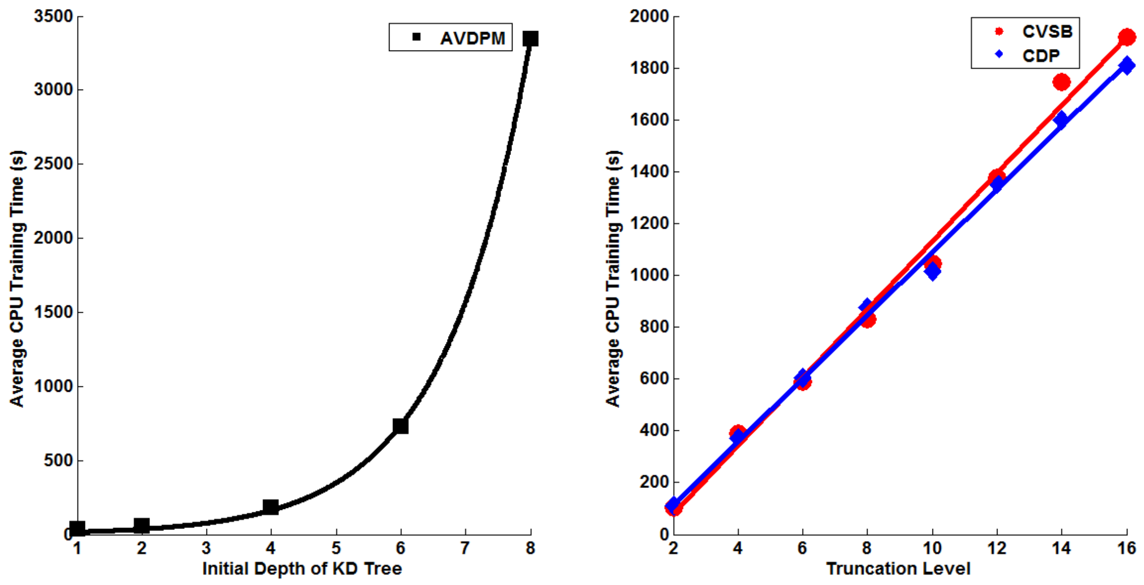


Figure 14 – The average amount of CPU training time is shown for 10 iterations of AVDPM (left), CVSB (right) and CDP (right) on TIMIT as a function of the KD tree depth and truncation level. AVDPM varies exponentially with the KD tree depth, while CVSB and CDP are linear with respect to the truncation level.

environments, and even different languages, it can be assumed that this is a good operating point for most acoustic model training regardless of the data.

The complexity of CVSB and CDP are both *O(TN)* (Kurihara et al. 2007), which can be seen in Figure 14, where the average training time grows linearly but rapidly as the truncation level is increased. This is reasonable since both algorithms generally use the maximum number of mixture components when the truncation level is low (e.g. the average number of mixture components found by CVSB on CH-E is *5.89* when the truncation level is *6*). Therefore, as the truncation level increases, so does the training time. Again, using an optimal truncation level for the TIMIT Corpus degraded error rates on CH-E and CH-M by less than *0.5%*.

The most significant finding in this work is shown in Figure 15 where the amount of average CPU training time for TIMIT's *40* phone classes across *200* iterations is found as the amount of training data is varied between *10,000* and *120,000* samples (the automatic phoneme alignment for TIMIT found a maximum of *~128k* phonemes). CDP and CVSB, as expected,

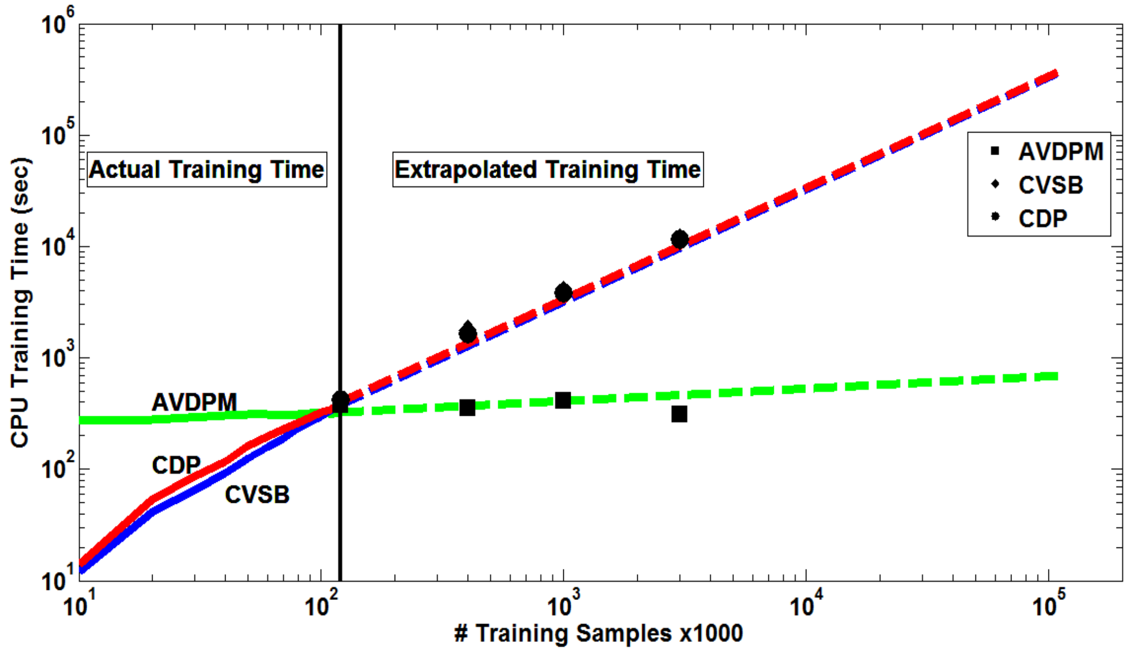


Figure 15 – The average amounts of CPU training time vs. the number of training samples across 200 iterations at optimal operating points on the TMIIT Corpus are shown. Trends are extrapolated to show that AVDPM would outperform CVSB and CDP on much larger corpora such as Fisher.

depict an almost identical linear increase as additional training data is used since they are both characterized by a complexity of *O(TN)*. AVDPM, however, requires significantly longer CPU times than CVSB and CDP for small amounts of data. This is due to the initial *O(NlogN)* cost of building the KD tree. This is a one-time cost though and it is apparent that once constructed the algorithm is able to train on larger amounts of data without significant computational cost.

For this reason trend lines were used to extrapolate the amount of time these algorithms would theoretically require to train a much larger corpus of data such as Fisher (Cieri et al., 2004) which is almost one thousand times larger than TIMIT ($\sim 10^8$ training samples). To verify this extrapolation, larger data sets were simulated by multiplying the number of samples from the TIMIT Corpus and then adding additional Gaussian noise. Due to limitations in computational resources a maximum corpus of *3* million samples was generated rather than something as large as the Fisher Corpus. These points are shown in Figure 15 and the extrapolated trend line was verified (though CVSB and CDP have such similar training times that the points are indistinguishable from each other). Thus, it is apparent in Figure 15 that although CVSB, CDP, and AVDPM require comparable training times for TIMIT, AVDPM is about 3 orders of magnitude faster for large corpora such as Fisher.

CHAPTER 5

CONCLUSIONS AND FUTURE WORK

One of the largest obstacles in the field of speech recognition is developing a system that is robust to significant variations in the acoustic channel and is able to generalize well for all data. The integration of nonparametric Bayesian methods in acoustic modeling is a step in this direction. The goal of this work was to introduce one such method, a Dirichlet process mixture (DPM), to a phone classification problem to show that these methods are computationally viable for big data problems such as speech recognition. However, due to the infinite number of parameters associated with nonparametric methods, the direct manipulation of target posteriors is intractable. Therefore inference algorithms are used to approximate these distributions and make calculations analytically solvable. Although there are many forms inference algorithms can take, three recent Bayesian variational inference algorithms – Accelerated Variational Dirichlet Process Mixtures (AVDPM), Collapsed Variational Stick Breaking (CVSB), and Collapsed Dirichlet Priors (CDP) – were assessed in this work.

Parametric models, such as the standard GMM used in many acoustic models, require significant tuning and make many rigid assumptions about the structure of the data, e.g. fixing the number of mixture components for all phone models. This is largely presumptuous, and it is far safer to assume that each phoneme has its own unique structure. In this work nonparametric methods in conjunction with AVDPM, CVSB, and CDP were used to discover the optimal number of mixture components and their corresponding weights for each individual phoneme. Finding this structure is best represented by a multinomial distribution but priors are also necessary to choose these values in statistically meaningful ways. Dirichlet distributions, and by extension DPMs, are the conjugate prior for the multinomial distribution and were therefore selected as the nonparametric method for this classification task.

It was shown in this work that all three variational inference algorithms produce comparable error rates to the standard GMM approach but with as much as *25* times fewer mixture components. While the optimal operating points for AVDPM, CVSB, and CDP are slightly different for each corpus, it was shown that using an initial depth of *4* of the KD tree for AVDPM and a truncation level of *6* for CVSB and CDP are very reasonable selections for future acoustic modeling tasks. These choices resulted in less than *1.32%* absolute degradation for AVDPM and less than *0.5%* absolute degradation in error for CVSB and CDP. Furthermore, the training time for CVSB and CDP are near identical and grow linearly as the number of training samples or truncation level is increased. AVDPM requires exponentially more time to train as the initial depth of the KD tree increases but is mitigated by the relatively low operating point found in this work. Moreover, although AVDPM requires a substantial amount of CPU time to build the KD tree, the amount of training data has a much less significant impact on the required training time. This makes AVDPM far superior to CVSB and CDP in the context of speech recognition where most corpora are orders of magnitude larger than the ones used in this work.

Nonparametric techniques such as the ones selected for this work have great potential in speech recognition. One obvious extension of these methods is the integration in HMM systems. In fact, Harati and Picone (2013) describe the use of hierarchical Dirichlet process HMM (HDP-HMM) where DPMs are used to not only discover the underlying structure of each state's distribution but to also model the structure of the HMM itself, i.e. transitions between a potentially infinite number of states. This work is currently limited to an MCMC-based block sampler which is significantly slower than the variational inference algorithms proposed here. Implementing variational methods with HDP-HMMs would be a huge step towards building a fully nonparametric speech recognition system that is computationally efficient.

Recently, DBN (Hinton et al., 2012) have also been gaining popularity and are another model where nonparametric techniques can be applied. In these systems large neural networks with multiple hidden layers are constructed which successfully model the nonlinear data such as speech. Similar to the use in HDP-HMMs, DPMs could be used to find the optimal number of hidden layers and also the number of neurons per layer. This would be a significant advance since the number of neurons in each layer is generally held constant since it would be overly tedious to tune this structure.

Overall, nonparametric methods appear to be a promising method for automatically capturing the underlying structure of data and can be applied to a wide variety of problems. With the use of new and computationally efficient inference algorithms such as AVDPM, these new methods can be easily extended to other algorithms and applications. It was shown in this work that AVDPM in particular can yield comparable performance to a baseline GMM system with far fewer parameters while remaining computationally viable for much larger corpora such as Fisher.

REFERENCES CITED

Antoniak, C. (1974). Mixtures of Dirichlet Process with Applications to Bayesian Nonparametric Problems. *The Annals of Statistics*, 2(7), 1152–1174.

Ager, M., Cvetkovic, Z., & Sollich, P. (2009). Robust phoneme classification: Exploiting the adaptability of acoustic waveform models. *Proceedings of the European Signal Processing Conference* (pp. 530–534). Glasgow, Scotland.

Baum, L. E., & Petrie, T. (1966). Statistical Inference for Probabilistic Functions of Finite State Markov Chains. *The Annals of Mathematical Statistics*, *37*(6), 1554–1563.

Bahl, L., Brown, P., De Souza, P., & Mercer, R. (1986). Maximum mutual information estimation of hidden Markov model parameters for speech recognition. *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing* (Vol. 11, pp. 49–52). Tokyo, Japan.

Belkin, M., & Niyogi, P. (2003). Using manifold structure for partially labeled classification. *Proceedings of Neural Information Processing Systems* (pp. 953–960). Lake Tahoe, Nevada, USA.

Bentley, J. L. (1975). Multidimensional binary search trees used for associative searching. *Communications of the ACM*, *18*(9), 509–517.

Bishop, C. (1995). *Neural Networks for Pattern Recognition*. Oxford, United Kingdom: Oxford Press.

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

Blackwell, D., & MacQueen, J. B. (1973). Ferguson Distributions Via Polya Urn Schemes. *The Annals of Statistics*, *1*(2), 353–355.

Breiman, L. (2001). Random Forests. *Machine Learning*, *45*(1), 5–32.

Canavan, A., & Zipperlen, G. (1996). *CALLHOME Mandarin Chinese Speech*. Philadelphia, Pennsylvania, USA. Retrieved from http://www.ldc.upenn.edu/Catalog/catalogEntry.jsp?catalogId=LDC96S34.

Canavan, A., Graff, D., & Zipperlen, G. (1997). *CALLHOME American English Speech*. Philadelphia, Pennsylvania, USA. Retrieved from http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC97S42.

Chen, K.-J., Tseng, C.-Y., Lyu, R., Chien, L.-F., Wang, H.-M., Shen, J.-L., Lin, S.-C., et al. (1994). Golden Mandarin(II) - an intelligent Mandarin dictation machine for Chinese character input with adaptation/learning functions. *Proceedings of the International Conference on Speech, Image Processing and Neural Networks* (pp. 155–159). Hong Kong, China.

Chen, S. F., & Goodman, J. (1999). An empirical study of smoothing techniques for language modeling. *Computer Speech & Language*, *13*(4), 359–393.

Chien, L.-F., Hwang, S.-H., Hsieh, Hu.-Y., Yang, R.-C., Bai, B.-R., Weng, J.-C., Yang, Y.-J., et al. (1995). Golden Mandarin (III) - a user-adaptive prosodic-segment-based Mandarin dictation machine for Chinese language with very large vocabulary. *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing* (Vol. 1, pp. 57–60). Detroit, Michigan, USA.

Cieri, C., Miller, D., & Walker, K. (2004). The Fisher Corpus: a Resource for the Next Generations of Speech-to-Text. *Proceedings of the International Conference on Language Resources and Evaluation* (pp. 69–71). Lisbon, Portugal.

The CMU Pronouncing Dictionary. (2008). *SPHINX Group, Carnegie Melon University*. Retrieved February 3, 2013, from http://www.speech.cs.cmu.edu/cgi-bin/cmudict.

Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, *13*(1), 21–27.

Davis, S., & Mermelstein, P. (1980). Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, *28*(4), 357–366.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, *39*(1), 1–38.

Eisner, J. (2011). High-Level Explanation of Variational Inference. Department of Computer Science, Johns Hopkins University. Retrieved February 11, 2012, from http://www.cs.jhu.edu/~jason/tutorials/variational.html.

Ferguson, T. S. (1973). A Bayesian Analysis of Some Nonparametric Problems. *The Annals of Statistics*, *1*(2), 209–230.

Frigyik, B., Kapila, A., & Gupta, M. (2010). Introduction to the Dirichlet Distribution and Related Processes. Department of Electrical Engineering, University of Washington, Seattle, Washington, USA. Retrieved from https://www.ee.washington.edu/techsite/papers/refer/UWEETR-2010-0006.html.

Gu, W., Hirose, K., & Fujisaki, H. (2006). Comparison of Perceived Prosodic Boundaries and Global Characteristics of Voice Fundamental Frequency Contours in Mandarin Speech. *Proceedings of the International Symposium on Chinese Spoken Language Processing* (pp. 31–42).

Halberstadt, A., & Glass, J. (1998). Heterogeneous acoustic measurements for phonetic classification. *Proceedings of the International Conference on Spoken Language Processing* (pp. 401–404). Sydney, Australia.

Harati, A., Picone, J., & Sobel, M. (2012). Applications of Dirichlet Process Mixtures to Speaker Adaptation. *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing* (pp. 4321–4324). Kyoto, Japan.

Harati, A., & Picone, J. (2013). Speech Acoustic Unit Segmentation Using Hierarchical Dirichlet Processes. Submitted to INTERSPEECH. Lyon, France. Retrieved from http://www.isip.piconepress.com/publications/unpublished/conferences/2013/interspeech/np b_seg/.

Heigold, G., Ney, H., Schluter, R., & Wiesler, S. (2012). Discriminative training for automatic speech recognition. *IEEE Signal Processing Magazine*, *29*(6), 60–71.

Hinton, G., Deng, L., Yu, D., Dahl, G., Mohammed, A., Jaitly, N., Senior, A., et al. (2012). Deep Neural Networks for Acoustic Modeling in Speech Recognition. *IEEE Signal Processing Magazine*, *29*(6), 83–97.

HTK. (2009). Machine Intelligence Laboratory, Department of Engineering, Cambridge University. Retrieved January 4, 2013, from http://htk.eng.cam.ac.uk/.

Juang, B.-H., Chou, W., & Lee, C.-H. (1997). Minimum classification error rate methods for speech recognition. *IEEE Transactions on Speech and Audio Processing*, *5*(3), 257–265.

Kurihara, K. (2010). Variational Dirichlet Process Gaussian Mixture Model. Google Inc., Mountain View, California, USA. Retrieved from https://sites.google.com/site/kenichikurihara/academic-software/variational-dirichlet-process-gaussian-mixture-model.

Kurihara, K. (2007). Collapsed variational Dirichlet process mixture models. *Proceedings of the International Joint Conference on Artificial Intelligence* (pp. 2796–2801). Hyderabad, India.

Kurihara, K., Welling, M., & Vlassis, N. (2006). Accelerated Variational Dirichlet Process Mixtures. In B. Scholkopf, J. Platt, & T. Hofmann (Eds.), *Advances in Neural Information Processing Systems* (1st ed., pp. 1–8). Boston, Massachusetts, USA: The MIT Press.

Larochelle, H., Erhan, D., Courville, A., Bergstra, J., & Bengio, Y. (2007). An empirical evaluation of deep architectures on problems with many factors of variation. *Proceedings of the International Conference on Machine Learning* (pp. 473–480). New York, New York, USA.

Lee, K.-F. (1989). *Automatic Speech Recognition: The Development of the SPHINX System* (1st ed., pp. 222). Media, New York, USA: Springer.

Lee, K.-F., & Hon, H.-W. (1989). Speaker-independent phone recognition using hidden Markov models. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, *37*(11), 1641–1648.

Lee, L.-S. (2006). Large Vocabulary Continuous Speech Recognition for Mandarin Chinese: Principles, Application Tasks and Prototype Examples. *Advances In Chinese Spoken Language Processing* (1st ed., pp. 125–152). Hackensack, NJ: World Scientific.

Ma, T. (2010). *Linear dynamic model for continuous speech recognition*. Department of Electrical and Computer Engineering, Mississippi State University. Retrieved from http://www.isip.piconepress.com/publications/phd_dissertations/2010/linear_dynamics/.

Mallapragada, P. K., Jin, R., & Jain, A. (2010). Non-parametric Mixture Models for Clustering. *Structural, Syntactic, and Statistical Pattern Recognition* (pp. 334–343). Springer Berlin Heidelberg.

Mathworks. (2013a). Neural Network Toolbox. Mathworks Documentation Center, The MathWorks, Inc., Natick, Massachusetts, USA. Retrieved from http://www.mathworks.com/help/nnet/index.html.

Mathworks. (2013b). TreeBagger. Mathworks Documentation Center, The MathWorks, Inc., Natick, Massachusetts, USA. USA. Retrieved from http://www.mathworks.com/help/stats/treebagger.html?searchHighlight=treebagger.

Mathworks. (2013c). Knnsearch. Mathworks Documentation Center, The MathWorks, Inc., Natick, Massachusetts, USA. Retrieved from http://www.mathworks.com/help/stats/knnsearch.html?searchHighlight=knnsearch.

Mathworks. (2013d). Gmdistribution. Mathworks Documentation Center, The MathWorks, Inc., Natick, Massachusetts, USA. Retrieved from http://www.mathworks.com/help/stats/gmdistribution.fit.html.

Mathworks. (2013e). Profiler. Mathworks Documentation Center, The MathWorks, Inc., Natick, Massachusetts, USA. Retrieved from http://www.mathworks.com/help/matlab/matlab_prog/analyzing-your-programs-performance.html.

McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, *5*(4), 115–133.

Neal, R. (1991). *Bayesian Mixture Modeling by Monte Carlo Simulation, Technical Report CRG-TR-90-7* (pp. 1–23). Toronto, Canada.

NIST. (2010).*Multimodal Information Group, National Institute of Standards and Technology*. The National Institute of Standards and Technology, U.S. Department of Commerce, Gaithersburg, MD, USA. Retrieved from http://www.itl.nist.gov/iad/mig//tools/.

Oparin, I., Lamel, L., & Gauvain, J.-L. (2010). Improving Mandarin Chinese STT system with Random Forests language models. *Proceedings of the International Symposium on Chinese Spoken Language Processing* (pp. 242–245). Orsay, France.

Paisley, J. (2012). Machine learning with Dirichlet and beta process priors: Theory and Applications. Department of Electrical and Computer Engineering, Duke University. Retrieved from http://dukespace.lib.duke.edu/dspace/bitstream/handle/10161/2458/D_Paisley_John_a_201005.pdf?sequence=1.

Picone, J., Steinberg, J. (2012). HTK Tutorials. Department of Electrical and Computer Engineering, College of Engineering, Temple University, Philadelphia, Pennsylvania, USA. Retrieved from http://www.isip.piconepress.com/projects/htk_tutorials/.

Povey, D., & Woodland, P. C. (2002). Minimum Phone Error and I-smoothing for improved discriminative training. *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing* (pp. 105–108). Orlando, Florida, USA.

Quintana, F. A., & Muller, P. (2004). Nonparametric Bayesian Data Analysis. *Statistical Science*, *19*(1), 95–110.

Rasmussen, C. E. (2000). The Infinite Gaussian Mixture Model. *Procceedings in Advances in Neural Information Processing Systems* (pp. 554–560). Denver, Colorado, USA.

Renals, S. (2010). Hierarchical Bayesian Language Models for Conversational Speech Recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, *18*(8), 1941–1954.

Robinson, T. (1994). *SHORTEN: Simple lossless and near lossless waveform compression*. Softsound, Autonomy Corporation, Cambridge, UK. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.114.856.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, *323*(6088), 533–536.

Sainath, T., Ramabhadran, B., Nahamoo, D., Kanevsky, D., Compernolle, D. Van, Demuynck, K., Gemmeke, J. F., et al. (2012). Exemplar-Based Processing for Speech Recognition. *IEEE Signal Processing Magazine*, *29*(6), 98–113.

Samet, H. (2008). Similarity searching: Indexing, nearest neighbor finding, dimensionality reduction, and embedding methods for applications in multimedia databases. *Proceedings of the International Conference on Pattern Recognition* (pp. 1–1). College Park, Maryland, USA.

Saon, G., & Chien, J.-T. (2012). Large-Vocabulary Continuous Speech Recognition Systems: A Look at Some Recent Advances. *IEEE Signal Processing Magazine, 29*(6), 18–33.

Schwartz, R., Colthurst, T., Duta, N., Gish, H., Iyer, R., Kao, C.-L., Liu, D., et al. (2004). Speech recognition in multiple languages and domains: the 2003 BBN/LIMSI EARS system. *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing* (Vol. 3, pp. 753–756). Cambridge, Massachusetts, USA.

Seide, F., Li, G., & Yu, D. (2011). Conversational Speech Transcription Using Context-Dependent Deep Neural Networks. *Proceedings of INTERSPEECH* (pp. 437–440). Florence, Italy.

Sethuraman, J. (1994). A constructive definition of Dirichlet priors. *Statistica Sinica*, *4*(2), 639–650.

Tam, Y.-C., & Schultz, T. (2005). Dynamic Language Model Adaptation Using Variational Bayes Inference. *Proceedings of INTERSPEECH* (pp. 5–8). Lisbon, Portugal.

Teh, Y.-W. (2007). Dirichlet process, tutorial and practical course. Department of Statistics, University of Oxford. Retrieved January 4, 2013, from www.stats.ox.ac.uk/~teh/teaching/npbayes/mlss2007.pdf.

Viterbi, A. (1967). Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, *13*(2), 260–269.

Werbos, P. (1974). *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*. Department of Applied Mathematics, Harvard University, Cambridge, Massachusetts, USA.

Xue, J., Che, L., & Zhao, Y. (2009). Semi-tied covariance matrices for acoustic models based on random forests of phonetic decision trees. *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing* (pp. 4469–4472). Taipei, Taiwan.

Young, S. (1996). A review of large-vocabulary continuous-speech recognition. *IEEE Signal Processing Magazine*, *13*(5), 45.