Neighborhood Analysis Methods in Acoustic Modeling for Automatic Speech Recognition

by

Natasha Singh-Miller

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2010

Author .....
Department of Electrical Engineering and Computer Science
August 6, 2010

Certified by .....
Michael J. Collins
Associate Professor
Thesis Supervisor

Accepted by .....
T. P. Orlando
Chairman, Department Committee on Graduate Students

Neighborhood Analysis Methods in Acoustic Modeling for Automatic Speech Recognition

by

Natasha Singh-Miller

Submitted to the Department of Electrical Engineering and Computer Science
on August 6, 2010, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

This thesis investigates the problem of using nearest-neighbor based non-parametric methods for performing multi-class class-conditional probability estimation. The methods developed are applied to the problem of acoustic modeling for speech recognition.

Neighborhood components analysis (NCA) (Goldberger et al. [2005]) serves as the departure point for this study. NCA is a non-parametric method that can be seen as providing two things: (1) low-dimensional linear projections of the feature space that allow nearest-neighbor algorithms to perform well, and (2) nearest-neighbor based class-conditional probability estimates.

First, NCA is used to perform dimensionality reduction on acoustic vectors, a commonly addressed problem in speech recognition. NCA is shown to perform competitively with another commonly employed dimensionality reduction technique in speech known as heteroscedastic linear discriminant analysis (HLDA) (Kumar [1997]).

Second, a nearest neighbor-based model related to NCA is created to provide a class-conditional estimate that is sensitive to the possible underlying relationship between the acoustic-phonetic labels. An embedding of the labels is learned that can be used to estimate the similarity or confusability between labels. This embedding is related to the concept of error-correcting output codes (ECOC) and therefore the proposed model is referred to as *NCA-ECOC*. The estimates provided by this method along with nearest neighbor information is shown to provide improvements in speech recognition performance (2.5% relative reduction in word error rate).

Third, a model for calculating class-conditional probability estimates is proposed that generalizes GMM, NCA, and kernel density approaches. This model, called locally-adaptive neighborhood components analysis, *LA-NCA*, learns different low-dimensional projections for different parts of the space. The models exploits the fact that in different parts of the space different directions may be important for discrimination between the classes. This model is computationally intensive and prone to over-fitting, so methods for sub-selecting neighbors used for providing the class-conditional estimates are explored. The estimates provided by LA-NCA are shown

to give significant gains in speech recognition performance (7-8% relative reduction in word error rate) as well as phonetic classification.

Thesis Supervisor: Michael J. Collins
Title: Associate Professor

Acknowledgments

I would like first to thank my advisor Michael Collins who taught me how to be a researcher. He provided insight when I needed it and always knew the answers to my technical questions. I especially appreciate the freedom to explore (and fail) he gave me while carefully directing my research towards fruitful endeavors. Finally, he provided me with unfliching support when I chose to start a family, for which I will always be grateful.

There are numerous other researchers who deserve thanks. Tommi and Jim both provided me with excellent feedback during the course of my research and on the thesis itself. My group taught me a lot over the years about various topics in speech and language and were excellent for bouncing around ideas. T. J. Hazen taught me a lot about the speech recognizer and Hung-An Chang helped me develop and understand various ideas for acoustic modeling. To all these people and many others (too numerous to list) whose discussions, support, and ideas helped me shape my own, I am sincerely grateful.

No one supported me more during my graduate work than my husband, Nick, who took care of me and our family when we needed it and whose continued interest in my work and my ideas were was always inspiring. My son Neel provided a much needed sense of urgency to actually finish and submit my thesis by giving me additional responsibilities, so I'm grateful to him as well. I'm thankful to my parents for supporting me throughout my many years at MIT and always trying to take responsibilities off my shoulders so that I could study and explore unhindered. My sisters and brother-in-law also always supported me and were proud of me no matter what which helped me through this process. I would also like to thank numerous friends who made life fun and kept me connected to the outside world throughout these many years.

Last, but not least, I'd like to thank MIT where I studied for my Bachelor's, Master's, and Ph.D. This has been an amazing place to live and study, and I've met some of the best friends of my life here. IHTFP–in every possible sense.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This thesis explores the use of discriminative non-parametric methods for performing acoustic modeling in speech recognition. Specifically, the supervised training problem of providing conditional probability estimates for multi-class data is considered. Given an input set of training samples $\{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_N, y_N)\}$, the goal is to provide estimates of $p(y|\mathbf{x})$ for a new test sample $\mathbf{x}$. The inputs $\mathbf{x}$ are D dimensional real-valued vectors, $\mathbf{x} \in \mathbb{R}^D$, and the labels y are drawn from some finite label set, $y \in \mathcal{Y}$. The *class-conditional estimates*, $p(y|\mathbf{x})$, are used to discriminate amongst the possible labelings of the test point $\mathbf{x}$.

Certain properties of the data can make it difficult to calculate good estimates of $p(y|\mathbf{x})$. First, the dimensionality of the inputs, D, may be large, which could lead to a sparse distribution of training samples in the space $\mathbb{R}^D$. Second, the size of the label set $|\mathcal{Y}|$ may be large, and there may only be a small number of samples for some labels. Third, there may be some underlying structure to the label space $\mathcal{Y}$, therefore each label may provide some information about other possible labels. Fourth, the measurements $\mathbf{x}$ may be insufficient for distinguishing between classes and fifth, the labeling of the training samples may be noisy.

In this thesis, *nearest neighbor* techniques are developed and applied to the problem of providing the conditional estimate $p(y|\mathbf{x})$. *Neighborhood components analysis (NCA)* (Goldberger et al. [2005]) serves as the inspiration and departure point for the techniques presented. The basic idea of nearest neighbor techniques is to use

the labels of the closest training samples to classify a test sample. As well as being simple and intuitive to apply, nearest neighbor techniques are chosen to solve the class-conditional estimation problem for the following reasons:

(1) They allow us to avoid making potentially incorrect parametric assumptions about the data

(2) While generally more computationally and memory-intensive than parametric methods, recent increases in available resources as well as improvements in algorithms for nearest neighbor search (Andoni and Indyk [2006, 2008], Arya et al. [2002, 1998], Indyk and Motwani [1998], Kushilevitz et al. [1998], Mount [2006]) make them more feasible to study

(3) While generally more data-intensive than parametric methods, recent increases in the amount of available training data may allow them to perform well

The primary application for motivating and evaluating the proposed methods is acoustic modeling for speech recognition, which is both a difficult and interesting task. Acoustic modeling is a component of speech recognition where the inputs $\mathbf{x}$ represent raw acoustic measurements while the labels y can represent an acoustic category such as a phone. Data for this task often presents many of the difficulties previously discussed. NCA and speech recognition are introduced below, followed by an outline of this thesis and its main contributions.

Note that while the methods presented here have been developed with acoustic modeling in mind, they are general and should be applicable to other tasks such as some that arise in computer vision and pattern classification.

1.1 Neighborhood Components Analysis (NCA)

NCA (Goldberger et al. [2005]) is introduced briefly here and described in greater detail in Chapter 2. The method makes use of the idea of stochastic or "soft" nearest neighbors when labeling a test point $\mathbf{x}$ using a training set of points. This means

that each training point j is assigned a weight that determines its influence over the labeling of the test point. This weight decays as the distance between $\mathbf{x}$ and $\mathbf{x}_j$ increases.

$$p(y|\mathbf{x}) = \frac{\sum_{j=1,y_i=y}^{N} e^{-dist(\mathbf{x}_j,\mathbf{x})}}{\sum_{j=1}^{N} e^{-dist(\mathbf{x}_j,\mathbf{x})}}$$

The function $dist(\mathbf{x}, \mathbf{z})$ is a Mahalanobis distance parameterized by a matrix $\mathbf{A}$.

$$dist(\mathbf{x}, \mathbf{z}) = (\mathbf{Ax} - \mathbf{Az})^T(\mathbf{Ax} - \mathbf{Az})$$

The matrix $\mathbf{A}$ can be learned so that the nearest neighbor estimate of $p(y|\mathbf{x})$ is well calibrated. Additionally, $\mathbf{A}$ may be of size $d \times D$, with $d \leq D$, allowing us to easily learn a low-dimensional representation of the initial inputs. NCA learns the parameters of $\mathbf{A}$ using gradient methods to optimize leave-one-out performance over the training set.

NCA provides a simple and elegant framework for learning embeddings of the features vectors in a low-dimensional space. Furthermore, the use of soft nearest-neighbors allows for the definition of optimization criteria that are learnable using gradient methods. The use of stochastic nearest-neighbors also provides a non-parametric estimate of $p(y|\mathbf{x})$ that allows us to avoid making parametric assumptions about the structure of the data. The two main ideas of NCA (1) using stochastic nearest neighbors, and (2) learning low-dimensional embeddings or Mahalanobis distance metrics are exploited in this thesis to develop new models for nearest-neighbor based class-conditional estimation.

1.2 Speech Recognition

Speech recognition has proven an invaluable part of human-computer interaction and is commonly used in many fields such as medicine, banking, customer service, and intelligence gathering. The demand for speech recognition systems is high because it has the potential to help process large amounts of spoken information that would

be difficult or expensive for humans to process alone. Research in speech recognition has also given rise to methods that have proven valuable in other areas of artificial intelligence research.

Given a raw acoustic signal, or waveform, an automatic speech recognizer generates a transcript of the words that most likely gave rise to the signal. However, the accuracy of these systems in a large vocabulary multi-user setting, such as transcribing spontaneous un-constrained speech, is still suboptimal, or worse than human performance. Most recognizers incorporate several sources of information to effectively deal with noise and uncertainty in the raw acoustic waveform. Uncertainty can arise in many ways as described in *Spoken Language Processing* (Huang et al. [2001]).

> *Acoustic models* include the representation of knowledge about acoustics, phonetics, microphone and environment variability, gender and dialect differences among speakers, etc. *Language models* refer to a system's knowledge of what constitutes a possible word, what words are likely to co-occur, and in what sequence. The semantics and functions related to an operation a user may wish to perform may also be necessary for the language model. Many uncertainties exist in these areas, associated with speaker characteristics, speech style and rate, recognition of basic speech segments, possible words, likely words, unknown words, grammatical variation, noise interference, non-native accents and confidence scoring of results. A successful speech recognition system must contend with all of these uncertainties. But that is only the beginning. The acoustic uncertainties of the different accents and speaking styles of individual speakers are compounded by the lexical and grammatical complexity and variations of spoken language, which are all represented in the language model.

Amongst the many components required to build an effective speech recognizer, this thesis focuses on methods for scoring acoustic models. The proposed non-parametric models are more flexible than parametric models that have been traditionally used. Specifically, given a representation of the raw acoustic waveform at a point in time, this work focuses on methods for providing class-conditional probability estimates that a specific acoustic-phonetic class gave rise to that signal. Proposed non-parametric methods are compared and conciliated with baseline parametric methods to obtain improved recognition performance. Three problems related to acoustic

modeling are explored and introduced in detail later in this thesis: [1] dimensionality reduction of the acoustic feature space, [2] modeling of underlying relationships between acoustic-phonetic labels, and [3] adapting the feature representation to different parts of the feature space.

1.3 Outline

There are three main pieces of work presented in this thesis.

First, the non-parametric technique known as neighborhood components analysis, *NCA*, for reducing the dimension of the input data is explored (Goldberger et al. [2005]). Reducing the dimension of the input data is a commonly addressed problem in speech recognition as dimensionality reduction can significantly improve results when fitting a Gaussian mixture model (GMM) to the samples of a particular class, i.e. learning the estimate for $p(\mathbf{x}|y)$. The nearest-neighbor based technique, NCA, is used to linearly project $\mathbf{x}$ to a lower dimensional representation. NCA is shown to perform competitively with another commonly employed dimensionality reduction technique in speech known as heteroscedastic linear discriminant analysis (HLDA) (Kumar [1997]).

Second, a nearest neighbor-based model related to NCA is created to provide an estimate for $p(y|\mathbf{x})$ that is sensitive to the possible underlying relationship between the labels, y. An embedding of the labels is learned to provide an estimate of the similarity or confusability between them. This embedding is related to the concept of error-correcting output codes (ECOC) and therefore the proposed model is referred to as *NCA-ECOC*. The estimates provided by this method along with nearest neighbor information is shown to provide improvements in speech recognition performance.

Third, a model for calculating $p(y|\mathbf{x})$ is proposed that generalizes GMM, NCA, and kernel density approaches. This model, called locally-adaptive neighborhood components analysis, *LA-NCA*, learns different low-dimensional projections for different parts of the space. The model exploits the fact that in different parts of the space different directions may be important for discrimination between the classes.

This model is computationally intensive and prone to over-fitting, so methods for sub-selecting neighbors used for providing the estimates $p(y|\mathbf{x})$ are explored. The estimates provided by this method are shown to give significant improvements in speech recognition performance.

These three pieces of work constitute the three main contributions of this thesis:

(1) NCA is shown to perform competitively with HLDA on acoustic modeling tasks

(2) NCA-ECOC is developed to learn and model the underlying relationship between acoustic labels and is shown to provide significant improvements in acoustic modeling tasks (on academic lecture task a 2.5% relative improvement in word-error-rate)

(3) LA-NCA is developed to provide nearest neighbor estimates that adapt to different parts of the input space and is shown to provide significant improvements in acoustic modeling tasks (on academic lecture task a 7-8% relative improvement in word-error-rate)

This thesis is outlined as follows. First in Chapter 2, the NCA model is described and in Chapter 3, background information on speech recognition and acoustic modeling as well as dimensionality reduction is presented. In Chapter 4, NCA is applied to dimensionality reduction for acoustic modeling. In Chapter 5, the NCA-ECOC model is developed and experiments using this model and nearest neighbor information are presented. In Chapter 6, the LA-NCA model is developed and experiments using this model along with generatively and discriminatively trained GMMs are performed. In Chapter 7, conclusions drawn from the thesis and several possible avenues for future work are discussed.

Chapter 2

Background: Neighborhood Components Analysis

NCA was introduced by (Goldberger et al. [2005]); the details of the method are described here for completeness. NCA learns a linear projection of vectors into a space that optimizes a criterion related to the leave-one-out accuracy of a nearest neighbor classifier on the training set. The method can be thought of as learning a Mahalanobis distance metric or as providing nearest-neighbor based class-conditional probability estimates.

2.1 Training NCA

Formally, NCA takes as input a training set $\{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_N, y_N)\}$ where $\mathbf{x}_i \in \mathbb{R}^D$ and $y_i \in \mathcal{Y}$. For example, in acoustic modeling, $\mathbf{x}_i$ may consist of concatenated vectors of MFCC measurements and y_i may indicate the class of phonemic event described by the vector, such as `/oy/`. The method outputs a learned projection matrix $\mathbf{A}$ of size $d \times D$ where d may be less that D. By selecting $d < D$, a dimensionality reducing linear projection matrix is learned. This matrix projects the training vectors $\mathbf{x}_i$ into a d dimensional representation $\mathbf{a}_i$.

$$\mathbf{a}_i = \mathbf{A}\mathbf{x}_i \tag{2.1}$$

This projection matrix $\mathbf{A}$ also defines a Mahalanobis distance metric that can be used by a nearest neighbor classifier in the projected space.

$$\begin{aligned} dist(\mathbf{x}_i, \mathbf{x}_j) &= (\mathbf{A}\mathbf{x}_i - \mathbf{A}\mathbf{x}_j)^T(\mathbf{A}\mathbf{x}_i - \mathbf{A}\mathbf{x}_j) \\ &= (\mathbf{x}_i - \mathbf{x}_j)^T(\mathbf{A}^T\mathbf{A})(\mathbf{x}_i - \mathbf{x}_j) \end{aligned} \tag{2.2}$$

The goal of the method is to learn a projection $\mathbf{A}$ that maximizes the accuracy of a nearest neighbor classifier when used in a leave-one-out setting on the training set. In order to define a differentiable optimization criterion, the method makes use of "soft-neighbor" assignments instead of directly using the k nearest neighbors. Specifically, each point j in the training set has a weight $\alpha_j(i)$ which determines its probability of assigning its label to a point i. This weight decays exponentially as the distance, parameterized by $\mathbf{A}$, between points i and j increases.

$$\alpha_j(i) = e^{-||\mathbf{A}\mathbf{x}_i - \mathbf{A}\mathbf{x}_j||^2} \quad \text{if } i \neq j, \quad 0 \text{ if } i = j \tag{2.3}$$

A quantity p_{ij} can then be defined which indicates that probability of point i receiving its label from point j.

$$p_{ij} = \frac{\alpha_j(i)}{\sum_{k=1}^{N} \alpha_k(i)}, \quad p_{ii} = 0 \tag{2.4}$$

p_{ij} is computed simply by normalizing the weights $\alpha_j(i)$ over the entire training set. Note that $p_{ii} = 0$, indicating that a training point is not used to label itself, and hence a leave-one-out model of training performance can easily be computed from this quantity.

One possible optimization criterion is to maximize the expected number of points correctly classified in a leave-one-out setting over the training set. This optimization criterion can be defined using the above soft-neighbor assignments. First a quantity $p(y|i)$ is defined that denotes the probability of a point i being assigned the class label

y by summing over all training points j whose label is y.

$$p(y|i) = \sum_{j=1;y_j=y}^{N} p_{ij} \tag{2.5}$$

The optimization criterion $f(\mathbf{A})$ can then be defined simply as the sum of the probabilities of classifying each point correctly.

$$f(\mathbf{A}) = \sum_{i=1}^{N} p(y_i|i) \tag{2.6}$$

Gradient methods can be used to optimize the leave-one-out criterion $f(\mathbf{A})$. The following gradient rule can be derived from Equation 2.6. (Note that $\mathbf{x}_{ij}$ is shorthand for $\mathbf{x}_i - \mathbf{x}_j$.)

$$\frac{\partial f}{\partial \mathbf{A}} = 2\mathbf{A} \sum_{u=1}^{N} \left(p(y_i|i) \sum_{k=1}^{N} p_{ik} \mathbf{x}_{ik} \mathbf{x}_{ik}^T - \sum_{j=1,y_j=y_i}^{N} p_{ij} \mathbf{x}_{ij} \mathbf{x}_{ij}^T \right) \tag{2.7}$$

The function $f(\mathbf{A})$ can be optimized using a number of gradient methods, such as stochastic gradient ascent, or conjugate gradient ascent. Note that the function $f(\mathbf{A})$ is not convex, so care needs to be taken when initializing the matrix $\mathbf{A}$ in order to avoid sub-optimal solutions. Specifically, multiple random initializations of the matrix $\mathbf{A}$ could be used to select the optimal solution. Additionally, as the magnitude of $\mathbf{A}$ increases, the effective number of nearest neighbors considered when labeling a sample decreases. Initializing $\mathbf{A}$ to have a large magnitude starts the optimization at a point where few neighbors are used to label a sample, whereas initializing $\mathbf{A}$ to have a small magnitude starts the optimization at a point where many neighbors are used to label a sample. Because the optimization is non-convex, this variation could lead to different solutions.

2.2 Computational Performance

The calculation of the above gradient can be computationally quite expensive. Calculating the soft-neighbor probabilities $\alpha_j(i)$ alone requires $O(N^2 d)$ calculations. We can minimize the amount of computation when calculating the gradient by re-arranging the terms of the gradient as follows:

$$\frac{\partial f}{\partial \mathbf{A}} = 2 \sum_{i=1}^{N} \left(p(y_i|i) \sum_{k=1}^{N} p_{ik} (\mathbf{A}\mathbf{x}_{ik}) \mathbf{x}_{ik}^T - \sum_{j=1, y_j = y_i}^{N} p_{ij} (\mathbf{A}\mathbf{x}_{ij}) \mathbf{x}_{ij}^T \right)$$

Calculating the full gradient takes $O(N^2 dD)$, and therefore the running time of the NCA algorithm is $O(LN^2 dD)$ where L is the number of gradient steps taken.

In many cases the projections $\mathbf{a}_i = \mathbf{A}\mathbf{x}_i$ can be precomputed before each gradient calculation. When scoring new examples, only the projected version of the training set and the projection matrix need to be stored, which can greatly improve the space efficiency of a nearest neighbor classifier when $d < D$.

Many of the soft-neighbor probabilities, p_{ij} will be very close to zero, since they drop off exponentially as the distance between points i and j increases. This justifies the truncation of the gradient calculation for those samples. Two strategies are explored in the literature for performing this truncation (Goldberger et al. [2005], Weinberger and Tesauro [2007]). One strategy is to retain only the top m neighbors j of i, or those with the largest values of $\alpha_j(i)$ and use only these for computing p_{ij}. All other members k of the training set have values of p_{ik} set to 0. The other strategy is to truncate based on some threshold, ϵ, for the value of p_{ij}. If a sample j has an associated probability p_{ij} less than ϵ, it is ignored during the gradient calculation. In the experiments presented here, the first strategy is employed but in the literature both strategies have been shown to be effective (Weinberger and Tesauro [2007]).

The full gradient can be computed in parallel across several machines, which can reduce the time needed to train the model. In the experiments presented here, NCA is trained using conjugate gradient ascent using parallelization.

2.3 Scoring Test Samples

The NCA method can be interpreted two ways: (1) as learning a Mahalanobis distance metric or linear projection, or (2) as learning class conditional probability estimates.

The learned distance metric or projection matrix $\mathbf{A}$ can be used to project test samples $\mathbf{x}$ to a new representation $\mathbf{a} = \mathbf{A}\mathbf{x}$. This new representation can then be used by other scoring algorithms. For instance, a model like a Gaussian mixture model (GMM) can be trained for each class where the training and test points are projected into the new space using $\mathbf{A}$. The GMMs can then provide scores for $p(\mathbf{a}|y)$ as described in Section 3.4. The two main benefits of using the projected points to train the GMMs are (1) the reduction in parameters achieved by the lower dimensional representation and (2) the separation achieved between the classes which can allow for better discrimination. Experiments using GMMs trained in this way are presented in Chapter 4.

NCA can also be used in two ways for performing multiclass classification:

- **kNN Classification:** Test samples can be classified using the k nearest neighbor method (kNN) where the projected training set is used to label the projected test point.

- **NCA Classification:** Class-conditional estimates for $p(y|\mathbf{x})$ can be calculated directly using the NCA model as follows.

$$\alpha_j(\mathbf{x}) = e^{-||\mathbf{A}\mathbf{x}-\mathbf{A}\mathbf{x}_j||^2} \tag{2.8}$$

$$p(y|\mathbf{x}) = \frac{\sum_{j=1,y_j=y}^{N} \alpha_j(\mathbf{x})}{\sum_{j=1}^{N} \alpha_j(\mathbf{x})} \tag{2.9}$$

  These estimates can be used to classify the new sample $\mathbf{x}$ by choosing

$$\hat{y} = \arg\max_y p(y|\mathbf{x}).$$

These two classification strategies will be compared in experiments presented in

Chapter 4. The class-conditional estimates $p(y|\mathbf{x})$ can also be used directly for tasks like acoustic modeling, and experiments with this idea are described in Chapter 5.

Chapter 3

Background: Speech Recognition

This chapter contains background information on speech recognition that is useful in understanding the role of this thesis within the broader field. The speech recognition model used within the SUMMIT system is formally described. Additionally common methods for performing linear dimensionality reduction are reviewed. Gaussian mixture models (GMMs) are also defined and common methods for training and using them for acoustic model scoring are discussed. Finally the datasets used throughout the thesis are detailed.

3.1 Speech Recognition and the SUMMIT System

This section provides a general description of the speech recognition model of the SUMMIT recognizer, which is used for all recognition experiments described in this thesis. See (Livescu [1999]) for a general overview of the model and its components as well as (Glass [2003], Glass et al. [2004]) for additional details about the recognizer.

The goal of automatic speech recognition can be described formally as follows: Given a sequence of acoustic feature vectors $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T$ that describe a raw acoustic waveform, identify the most likely sequence of words $\mathbf{w} = w_1, w_2, ..., w_M$ that gave rise to that waveform.

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} p(\mathbf{w}|\mathbf{x}_1, \mathbf{x}_2, ...\mathbf{x}_T) \tag{3.1}$$

Typically, an automatic speech recognizer will model several different possible pronunciations $\mathbf{u}$ for a word sequence $\mathbf{w}$ which leads to the following maximization.

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \sum_{\mathbf{u}} p(\mathbf{w}, \mathbf{u}|\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T) \tag{3.2}$$

For computational reasons the summation is often replaced with maximization over $\mathbf{u}$ as well. This identifies the single best word sequence $\mathbf{w}^*$ and pronunciation sequence $\mathbf{u}^*$ in a Viterbi decoding approach (Bahl et al. [1983]). Using Bayes rules and the fact that $p(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T)$ is constant we can derive the following.

$$\begin{aligned} p(\mathbf{w}, \mathbf{u}|\mathbf{x}_1, \mathbf{x}_2, ...\mathbf{x}_T) = & \quad \frac{p(\mathbf{w})p(\mathbf{u}|\mathbf{w})p(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T|\mathbf{w}, \mathbf{u})}{p(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T)} \\ \sim & \quad p(\mathbf{w})p(\mathbf{u}|\mathbf{w})p(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T|\mathbf{w}, \mathbf{u}) \end{aligned} \tag{3.3}$$

$$\mathbf{w}^*, \mathbf{u}^* = \arg\max_{\mathbf{w},\mathbf{u}} \ (p(\mathbf{w})p(\mathbf{u}|\mathbf{w})p(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T|\mathbf{w}, \mathbf{u})) \tag{3.4}$$

The first part of the product, $p(\mathbf{w})$, is known as the *language model* and is an estimate of the likelihood that the word sequence $\mathbf{w}$ would arise in the language. The second part of the product, $p(\mathbf{u}|\mathbf{w})$, is the pronunciation or *lexical model* and computes the likelihood that the word sequence $\mathbf{w}$ is composed of acoustic-phonetic subunits (such as phones, group of phones, noises or silences) identified by $\mathbf{u}$. Finally, $p(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T|\mathbf{w}, \mathbf{u})$ is the *acoustic model* that computes the likelihood $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T$ were generated by a specific choice of $\mathbf{w}$ and $\mathbf{u}$. This work will focus on the acoustic modeling portion of automatic speech recognition.

The SUMMIT recognizer used for the experiments described in this thesis operates similarly to the model described above, but varies in one important way (Glass [2003], Glass et al. [2004], Livescu [1999]). SUMMIT uses landmark modelling to segment the acoustic waveform. Typically recognizers use a frame-based approach and measure the acoustic vectors $\mathbf{x}_i$ at specific time intervals (every 10 ms for instance). However landmark-based models first segment the acoustic waveform by laying "landmarks" at points of interest (often points of large acoustic change) and

then hypothesizing segmentations of the waveform where a segment spans the region between two landmarks. Then the acoustic vectors $\mathbf{x}_i$ measured at the landmarks of a specific segmentation, $\mathbf{s}$, are used to compute the overall score of a word sequence hypothesis. The landmark based model expands on the decoding scheme of Equation 3.4 with the inclusion of a *segment model*, $p(\mathbf{s}|\mathbf{w},\mathbf{u})$. The SUMMIT model looks like the following:

$$p(\mathbf{w},\mathbf{u},\mathbf{s}|\mathbf{x}_1,\mathbf{x}_2,....\mathbf{x}_T) \sim \quad p(\mathbf{w})p(\mathbf{u}|\mathbf{w})p(\mathbf{s}|\mathbf{w},\mathbf{u})p(\mathbf{x}_1,\mathbf{x}_2,...,\mathbf{x}_T|\mathbf{w},\mathbf{u},\mathbf{s}) \tag{3.5}$$

$$\mathbf{w}^*,\mathbf{u}^*,\mathbf{s}^* = \quad \arg\max_{\mathbf{w},\mathbf{u},\mathbf{s}} \; (p(\mathbf{w})p(\mathbf{u}|\mathbf{w})p(\mathbf{s}|\mathbf{w},\mathbf{u})p(\mathbf{x}_1,\mathbf{x}_2,...,\mathbf{x}_T|\mathbf{w},\mathbf{u},\mathbf{s})) \tag{3.6}$$

The landmark-based recognizer can improve computational efficiency as there are often far fewer landmarks than frames. The methods presented in this thesis are trained and tested using the SUMMIT recognizer but could be applied within frame-based recognizers as well.

3.2 Acoustic Modeling

In order to calculate the likelihood $p(\mathbf{x}_1,\mathbf{x}_2,...,\mathbf{x}_T|\mathbf{w},\mathbf{u},\mathbf{s})$, the acoustic model will assume that each vector $\mathbf{x}_i$ is generated independently of one another. This is a strong assumption since each acoustic vector is part of a structured sequence probably spoken by a single speaker and clearly not independently drawn. However, in practice, the other components of the decoding process, such as the language model or speaker adaptation will compensate for this assumption. Additionally, each $\mathbf{x}_i$ is also assumed to depend only on the acoustic-phonetic class, y_i at the pertinent time in $\mathbf{u}$ and $\mathbf{w}$. Therefore the acoustic model can be broken up as follows.

$$p(\mathbf{x}_1,\mathbf{x}_2,...,\mathbf{x}_T|\mathbf{w},\mathbf{u},\mathbf{s}) = \prod_{i=1}^{T} p(\mathbf{x}_i|y_i) \tag{3.7}$$

This work focuses on the calculation of $p(\mathbf{x}|y)$ or the related quantity $p(y|\mathbf{x})$ and both quantities will be referred to as acoustic models as we proceed. While the quantity

$p(\mathbf{x}|y)$ is employed within the recognizer, learning models for estimating $p(y|\mathbf{x})$ can also be useful in a discriminative sense. When we learn estimates of $p(y|\mathbf{x})$, these can be easily converted to a estimate of $p(\mathbf{x}|y)$ using Bayes Rule.

$$p(\mathbf{x}|y) = \frac{p(y|\mathbf{x})p(\mathbf{x})}{p(y)}$$

Typically $p(y)$ can be estimated from the training set of acoustic samples and $p(\mathbf{x})$ can be ignored because it is equal for all labels and does not effect the output of the decoding process. In our experiments $p(y)$ is estimated using the relative proporitions of the labels in the training set. Note that this is a heuristic estimate, but often works well in practice.

The acoustic modeling problem is an instance of the supervised class-conditional probability estimation problem laid out in Chapter 1. The acoustic input vectors $\mathbf{x}$ are D dimensional real-valued vectors, $\mathbf{x} \in \mathbb{R}^D$. The labels y are drawn from a pre-defined acoustic phonetic label space $\mathcal{Y}$. In general, the dimensionality D can be high, in the hundreds, and the size of the label space $|\mathcal{Y}|$ can be large, in the thousands. This makes learning the estimates for $p(\mathbf{x}|y)$ difficult due to problems of over-fitting. Also, it is unknown what the correct hypothesis class of distributions $p(\mathbf{x}|y)$ should be, however a mixture of Gaussians is often used. The training set of samples $\{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_N, y_N)\}$ used to learn the estimates $p(\mathbf{x}|y)$ is usually large, in the tens of millions or larger. While N is not large enough to eliminate over-fitting problems, it is large enough to cause computational issues. Finally the labels $y \in \mathcal{Y}$ often share some underlying relationships and are not truly independent. They may share acoustic properties that make them confusable. For instance `/m/` and `/n/` are both nasals and can be difficult to distinguish. Therefore we can see that acoustic modeling presents many of the difficulties discussed in Chapter 1.

The non-parametric methods developed in this thesis can help eliminate some of the assumptions of typically employed parametric methods, such as mixtures of Gaussians, though they can be computationally expensive and data intensive. However, with recent increases in computational power and the availability of increasing

amounts of data, these methods are becoming more feasible.

3.3 Dimensionality Reduction for Acoustic Modeling

Dimensionality reduction is commonly applied to the problem of acoustic modeling. Many techniques for changing the basis of the feature space and reducing the number of feature dimensions exist. Linear methods such as PCA, LDA, and HLDA as well as non-linear methods such as Kernel PCA, and neural networks have been applied successfully to acoustic modeling. Here, the commonly used techniques of PCA, class-based PCA, LDA, and HLDA are described in detail. Many other techniques are reviewed in Chapter 4.

3.3.1 Principal Components Analysis

Principal components analysis (PCA) is a simple and elegant method for performing a linear rotation of the feature space (Pearson [1901]). Given a training set of inputs $\{\mathbf{x}_1, ..., \mathbf{x}_N\}$ where $\mathbf{x}_i \in \mathbb{R}^D$, the method computes an orthonormal matrix $\mathbf{A}$ which is $D \times D$ to create a new representation $\mathbf{z}_i = \mathbf{A}\mathbf{x}_i$. Briefly, the method can be described formally as follows. Let $\mathbf{X}$ be a D by N matrix where each column is one of the training samples with the sample mean of the data, μ, subtracted out, $\mathbf{x}_i - \mu$. Let $\mathbf{Z} = \mathbf{A}\mathbf{X}$. Calculate $\mathbf{A}$ such that $\mathbf{Z}\mathbf{Z}^T$ is diagonalized. The method ensures that in the projected space the dimensions of the data are not correlated. To perform dimensionality reduction, one can look at the variance in $\mathbf{Z}\mathbf{Z}^T$. The dimensions with high variances are usually most important. For a more detailed description of PCA see (Shlens [2005], Smith [2002]).

The method is simple and elegant to apply, but suffers from some serious drawbacks when applied to acoustic modeling. First, the method works by computing sufficient statistics, or in other words looks only at the mean and variance of the data samples. This works well for exponential distributions (such as a single Gaussian),

but does not work well for other types of distributions. Second, the data arises from multiple separate classes making it very unlikely that all the data can be modelled using a single exponential distribution. Third, no attempt is made to learn a rotation that allows for discrimination between classes.

While PCA is in many ways inadequate for the acoustic modeling task, it is still widely applied as a first step towards refining the feature space. Additionally, the orthogonality assumption makes this rotation well-suited for the use of GMMs whose covariances are restricted to be diagonal. Such GMMs are often used in acoustic modeling since they possess far fewer parameters and are much more resistant to over-fitting and are faster to compute.

3.3.2 Class-Based Principal Components Analysis

The SUMMIT recognizer makes use of a variation of PCA known as *class-based PCA*. The basic idea is to pool covariance matrices that are calculated for the samples from each class alone. Therefore the covariance relationships between samples of separate classes are not modelled. Formally, let $\mathbf{Z_k} = \mathbf{A}\mathbf{X_k}$ where $\mathbf{X_k}$ now contains only samples from class k. Each column of $\mathbf{X_k}$ is a sample from class k with mean of samples from class k, μ_k subtracted out. Calculate $\mathbf{A}$ such that $\sum_{k \in \mathcal{Y}} \frac{N_k}{N} \mathbf{Z_k} \mathbf{Z_k}^T$ is diagonalized. Here N_k is the number of samples of class k and $\mathcal{Y}$ is the set of all possible class labels. This works better for acoustic modeling than PCA since the samples from each class can be located far apart in the feature space without interacting with the samples from another class. Essentially this is better at modelling data that arises from multiple classes with different means but the same covariance structure. However, this is still too strong an assumption for the acoustic modeling problem. Also this method also makes no attempt to learn a projection of the points that aids in discrimination between the classes.

3.3.3 Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) (Fisher [1936]) learns a linear projection of the input space like the previous methods, but also attempts to separate the distributions of points for each class. For a discussion of detailed discussion of LDA see (Duda and Hart [1973]). A brief description of the method follows. Let the following quantities be defined where N_y is the number of samples from class y.

$$\mathbf{V} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}_i - \mu)(\mathbf{x}_i - \mu)^T$$

$$\mathbf{W_y} = \frac{1}{N_y}\sum_{i=1, y_i=y}^{N}(\mathbf{x}_i - \mu_y)(\mathbf{x}_i - \mu_y)^T$$

$$\mathbf{W} = \frac{1}{N}\sum_{y\in\mathcal{Y}} N_y \mathbf{W_y}$$

$\mathbf{V}$ is the total normalized sum of squares and products (SSQP), while $\mathbf{W_y}$ is the per class normalized SSQP, and $\mathbf{W}$ is the pooled per class normalized SSQP. To obtain the projection matrix $\mathbf{A}$, whose rows may be restricted to $d < D$, maximize the following quantity:

$$\mathbf{A} = \arg\max_{\mathbf{A}} \frac{|\mathbf{AVA}^T|}{|\mathbf{AWA}^T|}$$

This solution to this equation can be found by taking the d eigenvectors of $\mathbf{W}^{-1}\mathbf{V}$ with the largest associated eigenvalues. LDA thus tries to separate the data in the projected space while minimizing the separation of within-class samples.

LDA works well under the assumption that each class of data is generated by an exponential distribution with a shared covariance structure and separate means. This is a large improvement over the previously described methods since an attempt is made to separate the classes, however, the assumption that data for each class is generated by a single exponential with shared covariance structure is still very strong.

3.3.4 Heteroscedastic Linear Discriminant Analysis

Heteroscedastic linear discriminant analysis (HLDA) generalizes LDA by modeling

each class with a separate covariance structure. See (Kumar [1997]) for the details of HLDA, which is more complex than the previous methods. This method is motivated by the observation that most of the information important for discrimination is retained by the projection matrix $\mathbf{A}$ and the 'rejected' subspace of dimension $D-d$ contains little information. Specifically in the rejected subspace, the means and covariance structure for each class are equal and is therefore common to each class. Let $\mathbf{A}_{(neg)}$ denote the rejected subspace. The method optimizes the log-likelihood of the data under the linear transformation $\mathbf{A}$. The optimization can be performed by solving the following equation:

$$\hat{\mathbf{A}}, \hat{\mathbf{A}}_{(neg)} = \arg \max_{\mathbf{A}, \mathbf{A}_{(neg)}} \{-\frac{N}{2} \log |\mathbf{A}_{(neg)} \mathbf{V} \mathbf{A}_{(neg)}^T| - \sum_{y \in \mathcal{Y}} \frac{N_y}{2} \log |(\mathbf{A} \mathbf{W_j} \mathbf{A})| + N \log |\mathbf{A}_{(t)}|\}$$

$\mathbf{A}_{(t)}$ denotes the full rank matrix whose first d rows are $\mathbf{A}$ and whose last rows are $\mathbf{A}_{(neg)}$. This method works well for acoustic modeling tasks in practice, but still makes the assumption that each class can be modelled with a single Gaussian distribution. Nearest neighbor methods used in the next chapter are more flexible since they do not make this assumption.

3.4 Gaussian Mixture Models

Gaussian mixture models (GMMs) are commonly used in acoustic modeling and are used throughout the experiments in this thesis. A single multivariate Gaussian distribution is parameterized as follows:

$$p(\mathbf{x}) = N(\mathbf{x}; \mu, \Sigma) = \frac{1}{(2\pi)^{D/2} |\Sigma|^{1/2}} \exp\{-\frac{1}{2}(\mathbf{x}-\mu)^T \Sigma^{-1} (\mathbf{x}-\mu)\}$$

where μ is the mean of the distribution and Σ is the covariance structure. μ is a D dimensional real-valued vector, $\mu \in \mathbb{R}^D$, and Σ is a $D \times D$ matrix. A GMM model is a simple extension of the above model.

$$p(\mathbf{x}) = \sum_{m=1}^{M} \lambda_m \mathcal{N}(\mathbf{x}; \mu_m, \Sigma_m)$$

M indicates the number of Gaussian mixtures which each have their own mean μ_m and covariance structure Σ_m. The mixture weights λ_m are constrained to lie between 0 and 1, $0 \leq \lambda_m \leq 1$. Additionally the sum of the mixture weights is 1, $\sum_{m=1}^{M} \lambda_m = 1$. Gaussian mixture models are flexible and therefore a powerful tool when modeling distributions whose underlying structure is unknown as occurs often in acoustic modeling. However the number of mixtures to use M must be chosen carefully as a number that is to high can lead to over-fitting and a number that is too low may not represent the data well. Additionally different acoustic classes may be modelled best with different numbers of mixture components. Gaussian mixtures models are typically trained using the EM algorithm, the details of which can be found in (Huang et al. [2001]).

3.5 Speech Datasets

Two speech datasets are used for experimentation in this thesis. The TIMIT dataset is used primarily for performing multi-class classification experiments. The academic lecture data set is used for speech recognition experiments. Both sets are described in detail below.

3.5.1 TIMIT

The TIMIT phone classification task (Lamel et al. [1986]) is a popular task for which many results have been reported. The standard NIST training set, development set, and core test set were used in these experiments. The properties of these sets are summarized in Table 3.1. The samples are labeled with 61 different phonetic labels which are typically collapsed down to 39 possible labels for classification (see Table 3.2). As is standard, glottal stops are ignored in both training and testing for this task.

Set	# of speakers	# of utterances	# of tokens	# of hours
Train	462	3696	140225	3.14
Development	50	400	15056	0.34
Core Test	24	192	7215	0.16

Table 3.1: Properties of data sets used for TIMIT experiments.

0	/iy/	1	/ih/,/ix/	2	/eh/
3	/ae/	4	/ax/,/ah/,/ax-h/	5	/uw/,/ux/
6	/uh/	7	/ao/,/aa/	8	/ey/
9	/ay/	10	/oy/	11	/aw/
12	/ow/	13	/er/,/axr/	14	/l/,/el/
15	/r/	16	/w/	17	/y/
18	/m/,/em/	19	/n/,/en/,/nx/	20	/ng/,/eng/
21	/v/	22	/f/	23	/dh/
24	/th/	25	/z/	26	/s/
27	/sh/,/zh/	28	/jh/	29	/ch/
30	/b/	31	/p/	32	/d/
33	/dx/	34	/t/	35	/g/
36	/k/	37	/hh/,/hv/		
38	/bcl/,/pcl/,/dcl/,/tcl/,/gcl/,/kcl/,/q/,/epi/,/pau/,/h#/				

Table 3.2: Phones belonging to each class of TIMIT data.

In the experiments presented in this thesis the segmental feature measurements described by (Halberstadt and Glass [1997]) are used. Eight different segmental feature sets are described. Each feature set includes MFCC or PLP measurements as well as a log-duration measurement. The feature measurements are summarized in Table 3.3 reproduced from (Chang and Glass [2007]). The MFCC or PLP coefficients are consolidated via a temporal basis function (that extended 30ms beyond the segment boundaries) of either averages or cosine transforms as indicated in the table.

The features referred to as the "S4" features are chosen for classification experiments as they have performed best in past work (Chang and Glass [2007]).

3.5.2 Academic Lecture Recognition Task

Throughout this thesis experiments are conducted on academic lecture data (Glass et al. [2004], Park et al. [2004]). The data consists of 121 hours of training data, 8

	Dimensions	Window Size (ms)	Spectral Representation	Temporal Basis
S1	61	10	12MFCC	5 avg
S2	61	30	12MFCC	5 avg
S3	61	10	12MFCC	5 cos
S4	61	30	12MFCC	5 cos
S5	64	10	9MFCC	7 cos
S6	61	30	15MFCC	4 cos
S7	61	20	12PLPCC	5 avg
S8	61	20	12PLPCC	5 cos

Table 3.3: Summary of features used for TIMIT experiments. Reproduced from (Chang and Glass [2007]).

hours of development data, and 6 hours of test data. The data is drawn from the MIT World and MIT OpenCourseWare collection of lectures (ocw, mit). The audio data is recorded with omni-directional microphones usually within a classroom, which can add noise such as background talking and coughing to the signal. The data is drawn from multiple speakers, and the lectures cover various topics. For a detailed analysis of the data characteristics see (Glass et al. [2004]). Because the lecture data consists of spontaneous speech, it includes disfluencies such as filled pauses, false starts, and partial words.

The recognition experiments conducted for this thesis make use of a topic-independent language model with 37.4K unique words (Glass et al. [2007], Hazen and McDermott [2007]). Topic-dependent language models, which adapt to the subject matter of the lecture considered have been shown to provide improvements over a topic-independent language model (Hazen and McDermott [2007]). However, the experiments conducted are primarily to study various strategies for acoustic modelling and a topic-independent language model is more straightforward to apply. It should be noted, however, that more sophisticated language modelling techniques could provide further improvements in performance on this task Hsu and Glass [2008]. For more detail on the setup of the SUMMIT recognizer for the lecture recognition task see (Chang [2008]).

The recognition experiments conducted here also use speaker-independent acoustic

Feature Vector	b_{-4}	b_{-3}	b_{-2}	b_{-1}	b_{+1}	b_{+2}	b_{+3}	b_{+4}
Start Time (ms)	-75	-35	-15	-5	0	5	15	35
End Time (ms)	-35	-15	-5	0	5	15	35	75

Table 3.4: Telescoped time intervals used to construct acoustic vector. Eight individual feature vectors are concatenated to produced a single 112-dimensional vector.

models. While speaker-adaptive acoustic models can provide dramatic improvements, it is useful to evaluate the technique in a speaker-independent setting for simplicity. Future work may consider ways to adopt the proposed non-parametric acoustic modeling techniques within a speaker-adaptive setting.

A forced alignment of the training set is performed with a GMM acoustic model trained using ML estimation: the acoustic frames in the training set are then labeled with their most probable label given 1) the model; 2) the acoustic input; and 3) the manually-transcribed training sentence for the relevant example. This provides us with a time alignments of words and phonetics events with the raw acoustic signal. Because the labels, y_i, of the training samples, $\mathbf{x}_i$, are attained using forced alignments, it is possible that they will be noisy, which might need to taken into account when learning acoustic models for this data.

There are 73 internal acoustic-phonetic labels used by the recognizer:

```
-, _, _b1, _b2, _b3, _b4, _c1, _c2, _c3, _c4, _l1, _l2, _l3, _l4,
_n1, _n2, _n3, _n4, _n5, _n6, aa, ae, ah, ah_fp, ao, aw, ax, axr,
ay, b, bcl, ch, d, dcl, dh, dx, eh, el, em, en, epi, er, ey, f,
g, gcl, hh, ih, iy, jh, k, kcl, l, m, n, ng, ow, oy, p, pcl, r,
s, sh, t, tcl, th, uh, uw, v, w, y, z, zh
```

The first twenty labels are for silence, pauses, background, coughing, laughing, and other noise; the remainder are used to model phonetic events. The label `<>` is also used to mark the ends of an utterance. In additional to the internal labels, there are diphone transition labels that model the transition from one of these classes to another (e.g. `/f/ -> /ae/` or `/s/ -> /t/`). Many of the possible transitions never occur in the lexicon and need not be modeled. Other transitions occur rarely and are therefore clustered with similar transitions. In the end, 1871 possible classes are

retained, some consisting of multiple transition labels grouped together.

Each acoustic sample is represented using a 112-dimensional feature vector, consisting of the concatenation of eight 14-dimensional feature vectors. Each of these vectors contain 14 MFCC measurements taken at eight telescoped time intervals around the point of the acoustic sample. A summary of the time bounds of these feature vectors is included in Table 3.4. A Hamming window size of 25.6ms and frame rate of 5ms is used to compute 14 MFCC measurements which are then averaged within each telescoped interval. In total about 11.5 million training samples are available from the 121 hours of training data.

Chapter 4

Dimensionality Reduction for Acoustic Modelling Using Neighborhood Components Analysis

A fundamental problem in speech recognition has been learning low-dimensional projections of relatively high-dimensional acoustic vectors (Kumar [1997]). The goal is typically to find a low-dimensional projection that retains only the information most useful to discriminating between acoustic-phonetic classes. Lowering the dimensionality of the acoustic vectors in this way provides two important benefits. First, reducing the dimensionality of the acoustic data improves the time and space efficiency of the acoustic model. Second, a low-dimensional representation reduces the number of parameters that must be trained for the acoustic model and therefore has the potential to reduce the risk of over-training. Note that the problem of learning low-dimensional projections will also be referred to as learning low-dimensional linear embeddings throughout this thesis. [1]

[1] Much of the work presented in this chapter has been previously published in (Singh-Miller et al. [2007]).

The neighborhood components analysis method (NCA) (Goldberger et al. [2005]) is one approach to learning low-dimensional projections. NCA is a non-parametric method that learns a low-dimensional linear projection of the feature space that optimizes the performance of a nearest neighbor classifier. As we will see, using a projection such as NCA can greatly improve the performance of nearest neighbor classifiers when discriminating amongst acoustic-phonetic classes.

While there are a number of approaches to the dimensionality reduction problem, one of the most commonly used in acoustic modeling is heteroscedastic discriminant analysis (HLDA) (Kumar [1997]) (see Section 3.3.4 for a more thorough discussion of HLDA). Comparisons between NCA and HLDA for the acoustic modeling problem will be drawn in this chapter. Low-dimensional representations of acoustic vectors learned from both methods are used within a conventional speech recognizer to train Gaussian mixture models (GMMs) for each acoustic-phonetic class. These GMMs provide scores $p(\mathbf{x}|y)$ that form the acoustic model component during the speech recognition process.

While NCA and HLDA can both be used to perform the same task, the criteria each optimizes is quite different. NCA and HLDA both make use of training sets consisting of acoustic vectors and their associated class labels in order to learn projections that will be effective at separating classes in the projected space. However, HLDA makes stronger assumptions about the distribution of samples in each class than NCA; specifically, HLDA assumes that each class of acoustic vectors have a normal distribution. Because NCA optimizes for a nearest neighbor classifier, the method makes weaker assumptions about the shape of the distribution in each class, making it a closer match to the use of mixtures of Gaussians which are eventually employed in modeling these distributions in the acoustic model.

The NCA method is presented, along with discussion of specific implementation issues. The method is extended by introducing regularization which provides small improvements in performance. Our end goal is to use these projections to lower word error rate (WER) in a large vocabulary speech recognition task. Academic lecture data (Glass et al. [2004], Park et al. [2004]) is used to train and test our approach.

In our experiments, we compare NCA, class-based principal components analysis (PCA), and HLDA and show that NCA outperforms both other methods, showing a 2.7% absolute improvement in WER over a class-based PCA projection, and a 0.7% absolute (1.9% relative) improvement over HLDA.

4.1 Neighborhood Components Analysis

NCA was introduced by (Goldberger et al. [2005]); the details of the method are described in Chapter 2. In this section, we apply NCA to some example data and consider the addition of regularization to the model.

4.1.1 2-D Acoustic Modelling Examples

An example of learning a two dimensional projection of the phonemes `/s/`, `/sh/`, `/z/`, and `/zh/` is shown in Figure 4-1. Five hundred training samples from each class were attained from the lecture data task described in Section 3.5.2. The initial dimension of the data is $D = 112$ and they are projected down to $d = 2$ dimensions for visualization. The NCA matrix $\mathbf{A}$ is initialized randomly as depicted in Figure 4-1. Another example including the six vowels `/aa/`, `/ae/`, `/ey/`, `/iy/`, `/ow/`, and `/uw/` was similarly trained and is shown in Figure 4-2. The visualization shows that NCA when randomly initialized learns a projection of the original space where it attempts to separate training points of different classes.

4.1.2 Regularization

Regularization can be introduced into the NCA optimization criterion in order to alleviate a few possible problems with the method. Regularization can help counteract over-fitting effects we might see with the training data. The other problem we seek to address with regularization is specifically related to the definition of the soft-neighbor assignments used by the method. Because soft-neighbor assignments p_{ij} decay very rapidly with distance, as the magnitude of $\mathbf{A}$ increases the effective number of nearest

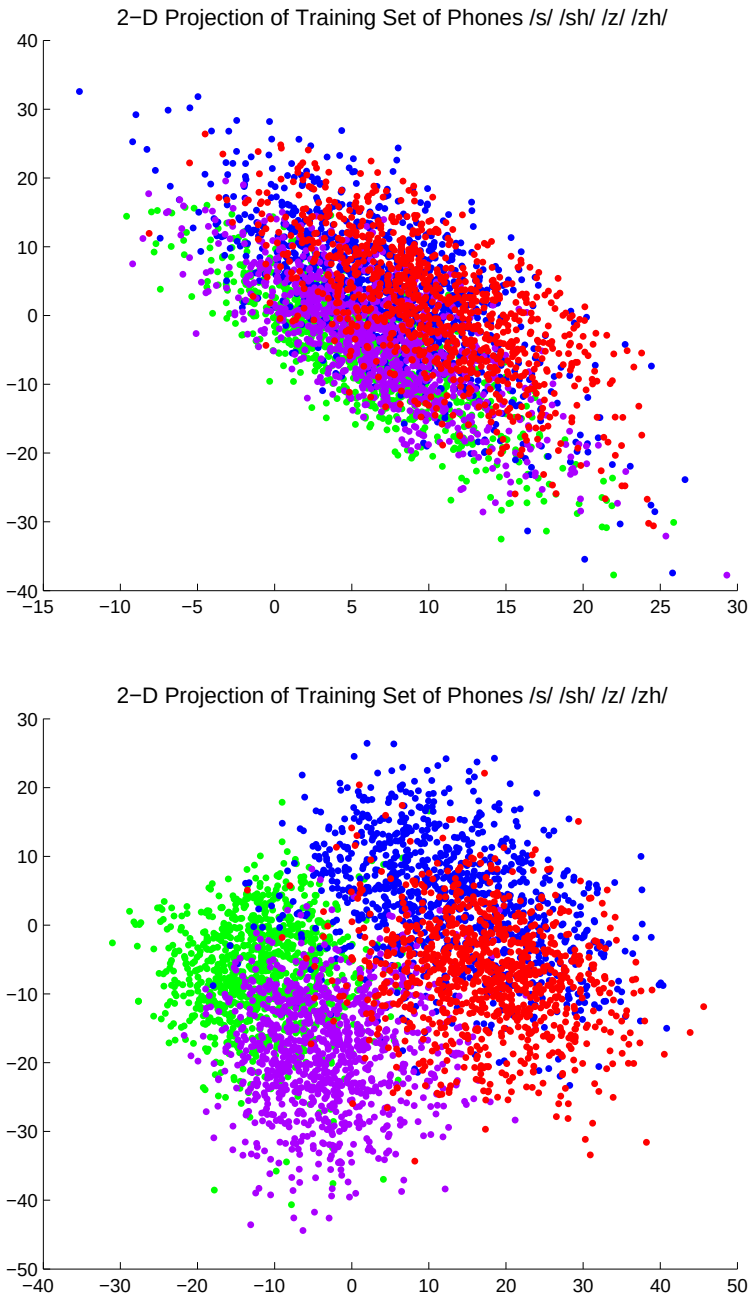

Figure 4-1: Initial 2-D random projection and 2-D projection learned using NCA of the four phones /s/ /z/ /sh/ and /zh/.

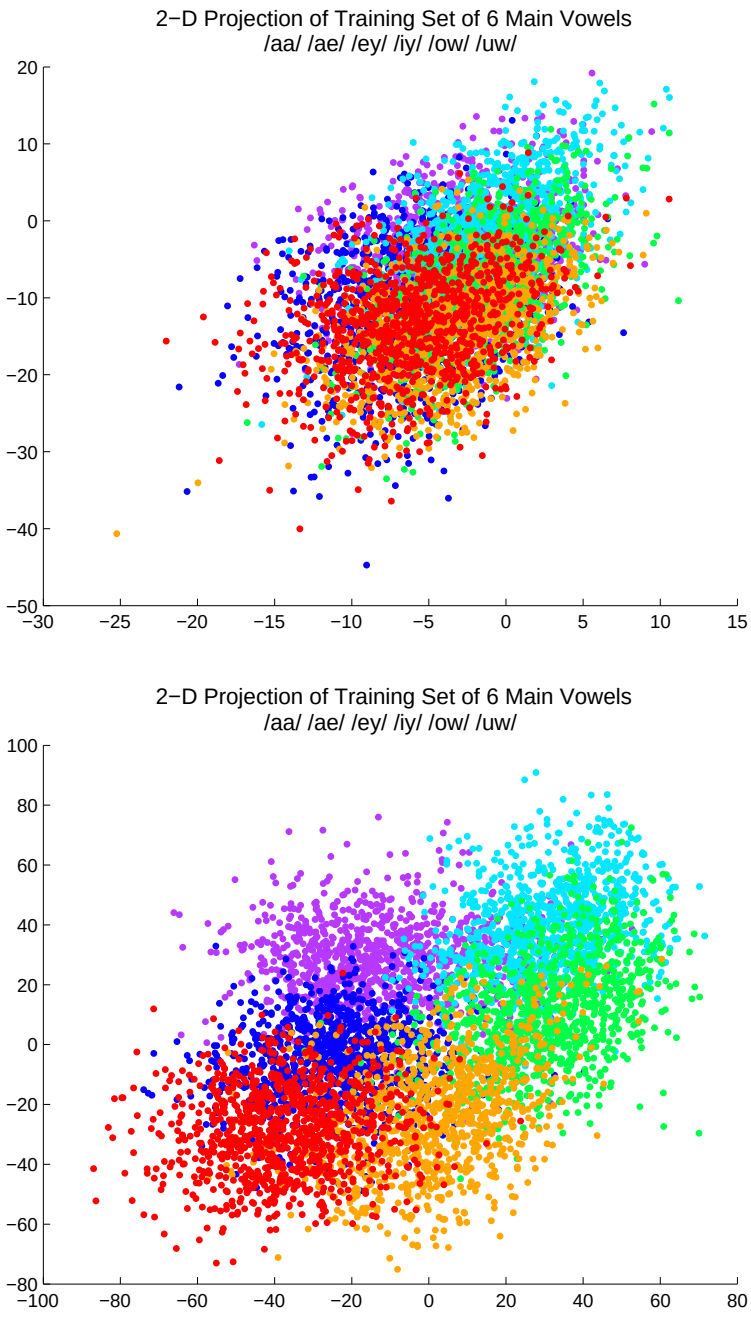

Figure 4-2: Initial 2-D random projection and 2-D projection learned using NCA of the six vowels /aa/ /ae/ /ey/ /iy/ /ow/ /uw/.

neighbors influencing the labeling of a point decreases. If the magnitude of $\mathbf{A}$ grows sufficiently large, the method might simply consider just the one closest neighbor, which could lead to a quite suboptimal projection of the data with poor generalization properties. Therefore the following regularized version of the optimization function is introduced where C is a constant chosen by optimizing the classification performance of the learned metric over a development set.

$$f_{reg}(\mathbf{A}) = \sum_i p(y_i|i) - C \sum_{j,k} A_{j,k}^2 \tag{4.1}$$

$A_{j,k}$ indicates the element at the jth row and kth column of matrix $\mathbf{A}$. The associated gradient rule for the criterion $f_{reg}(\mathbf{A})$ is the following.

$$\frac{\partial f_{reg}}{\partial \mathbf{A}} = 2\mathbf{A} \sum_{u=1}^{N} \left(p(y_i|i) \sum_{k=1}^{N} p_{ik} \mathbf{x}_{ik} \mathbf{x}_{ik}^T - \sum_{j=1, y_j=y_i}^{N} p_{ij} \mathbf{x}_{ij} \mathbf{x}_{ij}^T \right) - \frac{C}{2} \mathbf{A} \tag{4.2}$$

4.2 Speech Recognition Experiments

NCA and HLDA are compared on a large-vocabulary speech recognition task. The task is described in detail in Section 3.5.2

4.2.1 Dimensionality Reduction on Lecture Data

NCA and HLDA are both used to learn low-dimensional projections of the training samples. The HLDA projections are learned using the code provided by (Kumar [1997]) for HLDA using full covariance matrices. The NCA projections are learned using conjugate gradient ascent parallelized across several machines.

The low-dimensional representations of the training points are used to train a conventional maximum-likelihood GMM. This GMM forms the acoustic model component of the SUMMIT recognizer (Glass [2003]) for the recognition experiments.

To train both NCA and HLDA, data from 53 context-independent internal phone classes are used. This is done to limit the computational complexity of training. Also, these classes contain the phonetic structures we want to focus on separating. Only a

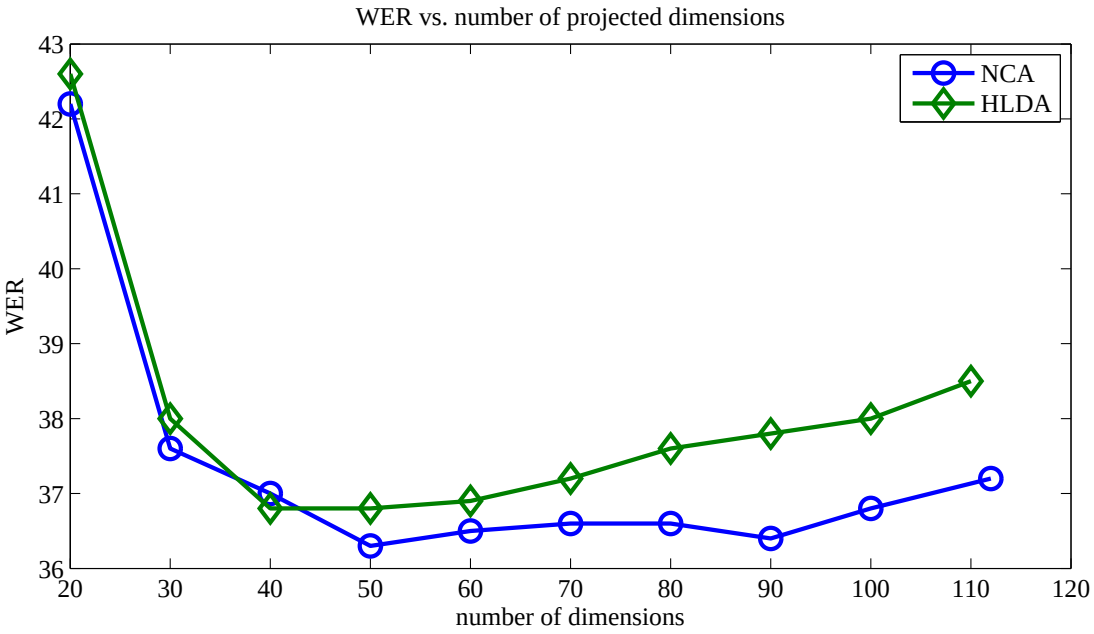

Figure 4-3: Word error rate (WER) of HLDA and NCA on a test set for several different dimensionalities.

small portion of this data, 500 samples from each of 53 phonetic classes, is used to train a projection using the NCA method. These samples are randomly selected and come from a number of different speakers. To train the HLDA projection all of the training data across the 53 phonetic classes, around 4 million samples, is used.

Because the optimization function for NCA is non-convex, care needs to be taken when initializing $\mathbf{A}$. In these experiments, the matrixes for both the NCA and regularized NCA projections were initialized randomly.

The end goal is to use these projections to reduce speech recognition word error rates (WER). Because the recognizer employs mixtures of Gaussians with diagonal covariance matrices, and models 1871 classes instead of 53 classes used to train the projections, a class-based PCA transform is applied after learning NCA and HLDA. This class-based whitening transform is performed by applying PCA to a covariance matrix obtained by pooling the covariances of the individual phonetic classes. The WER achieved by both the unregularized version of NCA and HLDA for a number of dimensions in the projected space is shown in Figure 4.2.1. As the number of dimensions is increased, initially large improvements in WER are achieved for both

Dimensionality Reduction Method	WER
PCA	38.8
HLDA, all training data	36.8
HLDA, 500 samples per class	37.1
NCA, 500 samples per class	36.3
NCA (regularized), 500 samples per class	36.1

Table 4.1: Word error rate of recognizer using PCA, HLDA, NCA, and regularized NCA to learn 50 dimensional projections.

methods, but these level-off quickly at around 40 dimensions. The minimal WER achieved by NCA occurs at 50 dimensions. As the number of dimensions increases beyond 90, the performance of the recognizer begins to deteriorate, indicating an over-training effect. A similar trend is seen with the HLDA method, with optimal performance achieved at 40 and 50 dimensional projections.

Table 4.1 records the performance of the recognizer using class-based PCA, HLDA, NCA, and the regularized version of NCA for 50-dimensional projections. Regularized NCA achieves a large improvement over classed-PCA alone of 2.7%, and a significant improvement over HLDA as well of 0.7% (1.9% relative improvement). Regularized NCA also slightly outperforms NCA, with the regularized method achieving an improvement of 0.2% over the baseline NCA method.

Experiments were conducted by increasing the number of data points per class used to train NCA to 1000 and 5000. These experiments led to negligible differences in speech recognition WER, suggesting that the NCA method can be well trained using relatively little data.

4.2.2 Discussion

Looking at the classification accuracy achieved in a kNN setting using both HLDA and regularized NCA can help identify some of the differences between the two methods. The accuracy of a kNN classifier is calculated on a set of held-out training samples (i.e. a set of 500 samples per class not used to train the NCA or HLDA projections). A 50-dimensional projection is learned for both regularized NCA and

HLDA and both are trained with 500 samples from each of the 53 internal phonetic classes. These are the same training points used to label the held out samples. The classification performance of the internal phonetic classes ordered by reduction in error rate achieved using regularized NCA are shown in Table 4.2. NCA achieves large improvements in classification accuracy across almost all the phonetic classes.

Another fact to note is that while the kNN performance of NCA and HLDA are very different, the difference in recognition performance of the two methods in terms of WER is not as large. This suggests that large increases in kNN performance does not necessarily mean large improvements in WER. This may indicate that a mismatch occurs between the NCA and GMM framework, where the GMM is unable to exploit the same information in the data as NCA. This may also be because simple increases in classification accuracy of phones does not necessarily lead to fewer word errors since word errors depend on a multitude of other factors. Later in this thesis the kNN framework is more directly applied to scoring the acoustic models.

One of the most striking aspects of the results is how little data NCA and indeed HLDA need to achieve competitive performance, only 500 samples from 53 classes. Because NCA seems to be able to perform well with a relatively small number of samples, one potential application of this method would be in speaker adaptation where hopefully with just a small number of samples the method could quickly adapt to a specific speaker.

4.2.3 An Alternative Log-Likelihood Criterion

The optimization criterion from Equation 2.6 for NCA optimizes the number of training points classified correctly. For some applications, including acoustic modeling, optimizing the class-conditional log-likelihood of the training samples may be more desirable.

$$f_{log}(\mathbf{A}) = \sum_{i=1}^{N} \log p(y_i|i) \tag{4.3}$$

Optimizing this criterion discourages the assignment of very low probability to the correct class of any training sample and hopefully generalizes similarly to test samples.

Phone	NCA Acc.	HLDA Acc.	% Red. in Err.
/epi/	88.60	62.40	69.68
/em/	79.00	56.60	51.61
/ah_fp/	72.80	48.80	46.88
/g/	77.80	58.80	46.12
/b/	80.40	66.40	41.67
/pcl/	63.40	39.80	39.20
/zh/	78.00	65.20	36.78
/p/	67.80	50.00	35.60
/jh/	55.00	34.20	31.61
/uw/	56.60	37.20	30.89
/oy/	51.80	30.60	30.55
/sh/	67.00	52.60	30.38
/w/	72.20	60.40	29.80
/ey/	62.40	46.60	29.59
/tcl/	49.60	28.80	29.21
/y/	67.20	54.40	28.07
/ay/	48.00	28.40	27.37
/uh/	58.20	42.60	27.18
/axr/	56.00	40.20	26.42
/v/	53.60	37.00	26.35
/en/	55.60	40.60	25.25
/d/	46.40	29.00	24.51
/z/	56.40	43.00	23.51
/f/	63.80	52.80	23.31
/ch/	51.00	37.00	22.22
/ng/	59.40	48.00	21.92
/k/	62.20	52.40	20.59
/dx/	49.60	36.60	20.50
/th/	52.40	40.60	19.87
/dh/	55.80	45.80	18.45
/t/	53.40	43.40	17.67
/ae/	37.40	24.40	17.20
/aw/	39.40	27.60	16.30
/hh/	54.20	45.40	16.12
/dcl/	39.20	27.80	15.79
/bcl/	50.00	41.80	14.09
/ao/	45.20	36.40	13.84
/iy/	55.40	48.40	13.57
/gcl/	42.00	33.00	13.43
/er/	43.40	35.20	12.65
/el/	66.80	62.00	12.63
/l/	25.40	14.80	12.44
/ow/	31.80	22.80	11.66
/r/	40.40	33.00	11.04
/ih/	32.60	25.40	9.65
/eh/	31.80	24.60	9.55
/aa/	30.00	23.40	8.62
/ah/	27.20	21.20	7.61
/n/	39.00	34.20	7.29
/ax/	25.60	20.60	6.30
/m/	31.60	27.60	5.52
/kcl/	45.20	43.20	3.57
/s/	57.20	57.40	-0.47

Table 4.2: Accuracy of a kNN classifier on a test set of acoustic vectors with their associated phonetic labels. Vectors are first projected into a 50-dimensional space using HLDA or regularized NCA trained on a training set of 500 points per class.

This can be important for acoustic modeling where assigning near zero probability to the correct class of an acoustic sample can have detrimental effects on the word error rate of surrounding words as well.

The gradient associated with this optimization criterion is the following:

$$\frac{\partial f_{log}}{\partial \mathbf{A}} = 2\mathbf{A}\sum_{i=1}^{N}\left(\sum_{k=1}^{N} p_{ik}\mathbf{x}_{ik}\mathbf{x}_{ik}^T - \frac{1}{p(y_i|i)}\sum_{j=1,y_j=y_i}^{N} p_{ij}\mathbf{x}_{ij}\mathbf{x}_{ij}^T\right) \tag{4.4}$$

In (Goldberger et al. [2005]), it is reported that both $f(\mathbf{A})$ and $f_{log}(\mathbf{A})$ deliver similar performance. For the speech recognition task, the findings are similar, with this method achieving 36.5% WER on the test set. This criterion is further developed in the other chapters of this thesis.

4.3 Experiments with TIMIT

Phonetic classification experiments on the TIMIT (et al [1993]) corpus were performed. As is commonly done with this phonetic classification task, the 61 phonetic class labels provided for the training and test data are collapsed down to 39 classes (Halberstadt and Glass [1997]). There have been a number of approaches to this task including the use of Gaussian mixture models (Halberstadt and Glass [1997], Rifkin et al. [2007], Sha and Saul [2007a,b], Chang and Glass [2007]), conditional random fields (CRFs) (Gunawardana et al. [2005]), and support vector machines (SVMs) (Clarkson and Moreno [1999]). Recent work has focused on the use of large-margin GMMs (Sha and Saul [2007a,b], Chang and Glass [2007]) with the best published results achieved using hierarchical large-margin GMMs (Chang and Glass [2007]). For a more detailed description of the TIMIT data see Section 3.5.1.

Two sets of experiments are performed in order to determine the effectiveness of NCA alone as well as in a hierarchical setting with the results presented in Table 4.3. The initial feature representation for the training and test samples is a 61-dimensional feature representation described previously in (Halberstadt and Glass [1997], Chang and Glass [2007]). These features include MFCC measurements as well

Model	Development Error (%)	Test Error (%)
Baseline	26.5	27.2
NCA	21.1	21.9
Hierarchical NCA	**20.1**	**21.6**

Table 4.3: Results of k-nearest neighbor classification experiments on the TIMIT development and test corpus. Results are calculated with k=15. The baseline classification results make use of the initial feature representation. The NCA results are computed using a 61-dimensional NCA projection. The hierarchical NCA setup makes use of a separate projection matrix trained for each node in the hierarchy.

Level 1	Level 2 (Phonetic classes)
s_1	/aa/, /ae/, /ah/, /ao/, /aw/, /ax/, /axh/, /axr/, /ay/, /eh/, /el/, /er/, /ey/, /ih/, /ix/, /iy/, /l/, /ow/, /oy/, /r/, /uh/, /uw/, /ux/, /w/, /y/
s_2	/em/, /en/, /eng/, /m/, /n/, /ng/, /nx/
s_3	/ch/, /jh/
s_4	/s/, /sh/, /z/, /zh/
s_5	/b/, /bcl/, /d/, /dcl/, /dh/, /dx/, /epi/, /f/, /g/, /gcl/, /h#/, /hh/, /hv/, /k/, /kcl/, /p/, /pau/, /pcl/, /q/, /t/, /tcl/, /th/, /v/

Table 4.4: 2-level Hierarchical decomposition of TIMIT phone labels.

as a log duration measurement. The performance a kNN classifier with $k = 15$ on this baseline representation is shown in Table 4.3. The results are quite poor, at 26.5% on the development set and 27.2% on the test set.

The 61-dimensional NCA rotation of the feature space achieves a significant improvement in classification error over the baseline model. A reduction in error of 5.4% on the development set and 5.3% on the test set are achieved. This provides concrete evidence of the ability of NCA to define a distance metric that works well for nearest neighbor classification.

Additionally a two-level hierarchical classifier is constructed with the phonetic class labels grouped together as described in Table 4.4. The labels are generally grouped by vowels and semi-vowels, nasals, affricates, sibilants, and stops and fricatives. The super-classes of the hierarchy are referred to as s_i. Classification is per-

formed by making a hard decision using a kNN classifier amongst these five groupings $(s_1, ..., s_5)$, and then using a kNN classifier amongst the sub-classes to determine the final label assigned to the test point. k equals 15 for both levels of the hierarchy and was optimized on the development set. A separate 61-dimensional NCA rotation is learned for each of these decisions; in total six separate NCA rotation matrices are learned. Performing classification in this manner leads to an error rate of 20.1% on the development set and 21.6% on the test set. This is a significant improvement over the basic NCA model on the development set with a small improvement in performance over the test set.

The hierarchical grouping of the phones was manually chosen, and there are many other ways to hierarchically decompose these classes. However, it is clear that a hierarchical model can significantly improve classification performance which corroborates findings of other recent publications (Chang and Glass [2007]). The hierarchical NCA model learns a separate NCA distance metric for each node in the hierarchy. In this way a separate distance metric is learned for different partitions of the labels (as for the top node), and for different subsections of the input space (as for the leaf nodes). These preliminary experiments provide evidence that adaptive distance metrics for different parts of the space can give significant improvements; this idea is further developed in Chapter 6.

4.4 Experiments with MNIST Digits

While the primary motivation for the methods developed in this thesis have been acoustic modeling, the methods can also be applied to other tasks. The MNIST handwritten digit task seeks to classify handwritten samples of the ten digits 0-9. The data consists of 60,000 labeled samples, approximate 6,000 for each digit, where the images are 24×24 pixels with 8-bit grayscale labels. Therefore each image forms a 784-dimensional input vector. Ten thousand test samples are also available.[2]

Classification experiments are conducted on this data set. kNN classifiers and

[2]The MNIST data set is available at: http://cs.nyu.edu/r̃oweis/data.html

	Baseline, $D = 784$	NCA, $d = 2$	NCA, $d = 3$	NCA, $d = 5$	NCA, $d = 10$	NCA, $d = 20$
Optimal k	4	100	100	15	4	2
Test Error	2.8	27.9	17.2	8.9	4.1	2.3

Table 4.5: Error rate of a **kNN classifier** on MNIST digits test data. The baseline model is just the original 784-dimensional data. The NCA models are trained for various values of d. The optimal values for k selected on held-out samples are also noted.

	Baseline, $D = 784$	NCA, $d = 2$	NCA, $d = 3$	NCA, $d = 5$	NCA, $d = 10$	NCA, $d = 20$
Optimal k	1	200	200	25	5	8
Test Error	3.1	27.9	17.1	9.1	4.3	2.6

Table 4.6: Error rate of a **NCA classifier** on MNIST digits test data. The baseline model is just the original 784-dimensional data. The NCA models are trained for various values of d. The optimal values for k selected on held-out samples are also noted.

NCA classifiers as described in Section 2.3 are tested with many values of k, the number of nearest neighbors used to label a test point. Results for the kNN classifier are reported in Table 4.5 and results for the NCA classifier are reported in Table 4.6. The optimal value for k is determined on held-out training samples. The classification performance improves as the dimensionality of the NCA projection increases, with projections of $d = 20$ achieving the lowest error rates. Also, the optimal performance of a kNN classifier is sometimes better than the performance of the NCA classifier and vice versa. However, generally both methods deliver similar optimal performance.

In Figure 4-4, performance of both kNN and NCA classifiers are depicted on held-out training samples. We can see that the NCA method for classification delivers more stable classification accuracy across various values of k, which might make it more attractive when k can not be carefully chosen.

Figure 4-5 shows a plot of the test points projected using a two-dimensional NCA rotation. Beneath the plot, each row of the projection matrix is portrayed as a 24 $\times$ 24 image. Each element of the row is displayed at its corresponding pixel location with red indicating a strongly positive value and blue indicating a strongly negative

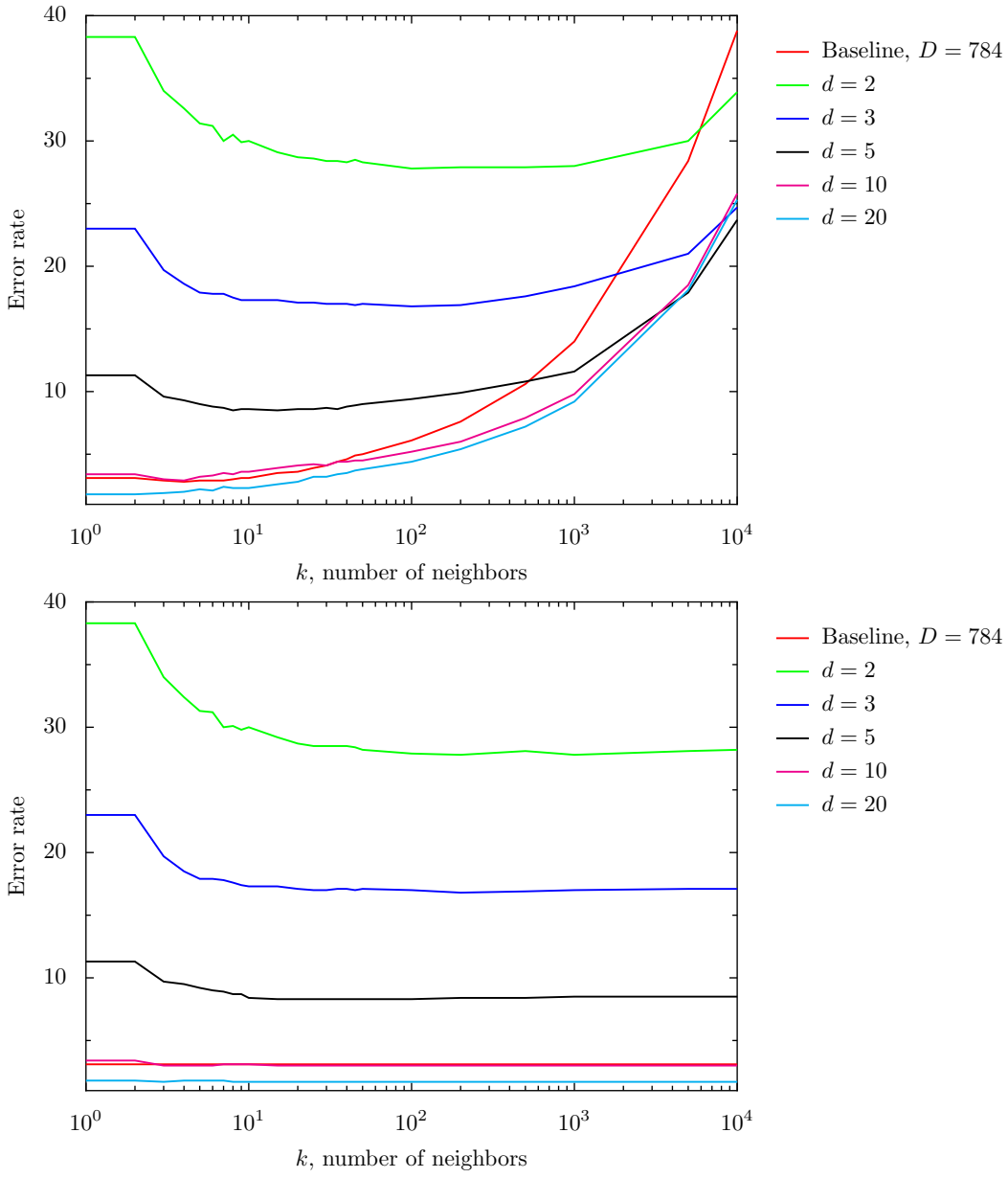

Figure 4-4: Performance on held-out development data from MNIST training set for various values of k. The baseline model makes use of the original images, and different low-dimensional NCA projections are also used. The top plot indicates the performance of a kNN classifier and the bottom plot shows an NCA classifier.

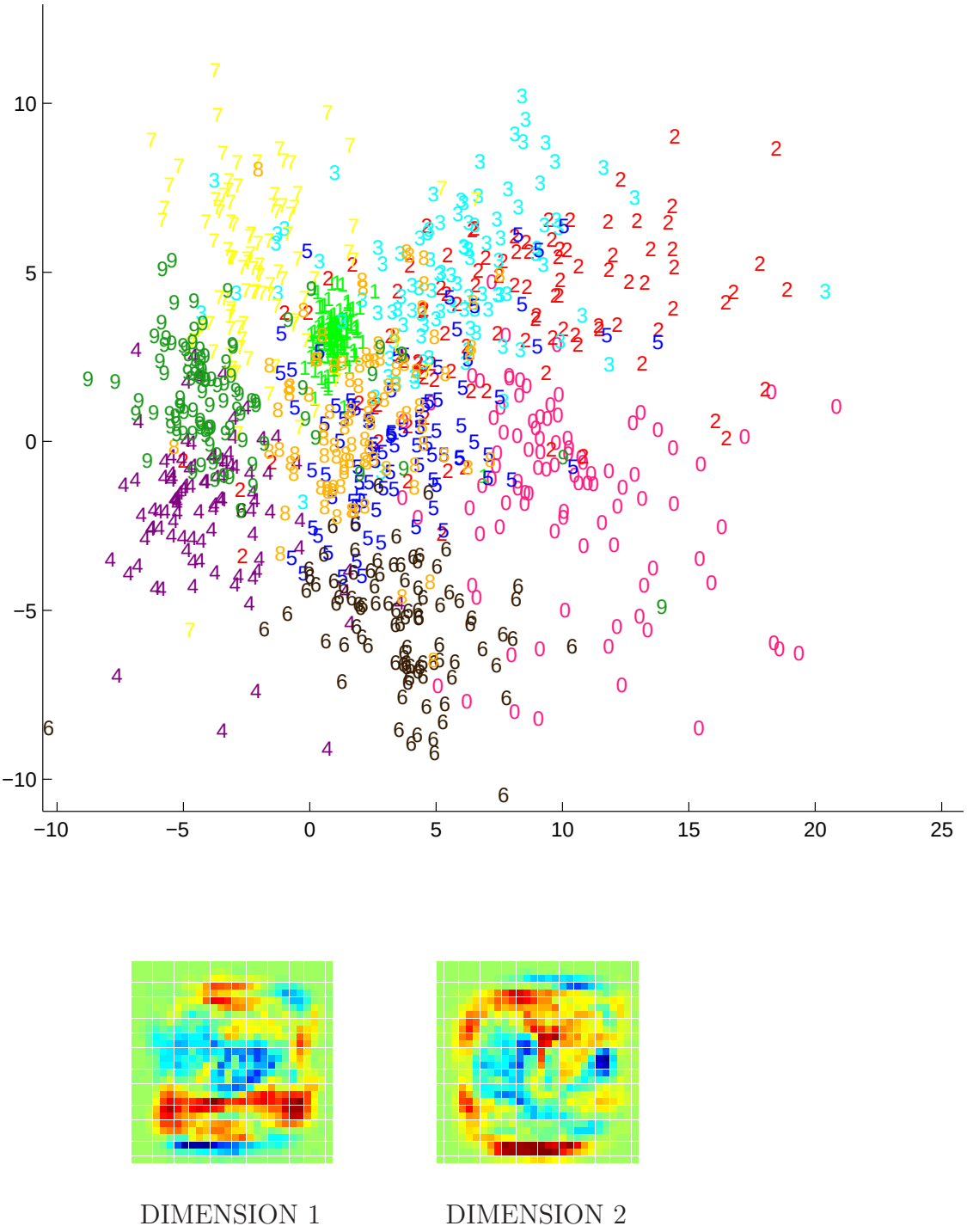

Figure 4-5: A plot of 10% of MNIST digit test points projected using a two-dimensional NCA rotation. Beneath the plot, each row of the projection matrix is portrayed as a 24×24 image. Each element of the row is displayed at its corresponding pixel location with red indicating a strongly positive value and blue indicating a strongly negative value.

value. It is clear that each dimension separates the points based on different regions of pixels. These two learned dimensional projections do a reasonable job of separating the ten classes of data so they can be visualized in the 2-D plot. This task will be used in later Chapters to help visualize the results of the proposed methods.

4.5 Related Work

There are numerous alternatives to NCA for performing dimensionality reduction.

Many of these alternatives make strong assumptions about the underlying structure of the data that NCA does not. Some such methods already described include PCA (Pearson [1901]), LDA (Duda and Hart [1973], Fisher [1936]), and HLDA (Kumar [1997]). Relevant components analysis (RCA) is another method that learns a low-dimensional linear projection of the data points (Bar-Hillel et al. [2003]). However, it also assumes the points are drawn from Gaussian distributions. In (Koren and Carmel [2004]) a generalized family of LDA-like methods are proposed that are more robust than LDA, especially in terms of outliers.

There are also many more flexible methods that make use of nearest neighbor structure in order to learn a Mahalanobis distance metric. One method called variable-kernel similarity metric (VSM) learns a metric very similar to NCA, but constrains the projection matrix to be diagonal and optimizes an objective function equal to the squared difference between the true label and the leave-one-out predicted label (Lowe [1995]). Another method for achieving large margin nearest neighbor classification is introduced by (Weinberger et al. [2006]). A Mahalanobis distance metric is learned by setting target nearest neighbors for classifying each training sample and then learning the metric to achieve a margin (with slack variables) to separate those target neighbors from other neighbors of a different classes. This method has a convex optimization function but depends on the selection of the initial desired neighbors to work well. Also dimensionality reduction is performed by first using PCA to reduce the number of dimensions and then learning their distance metric. In (Torresani and Lee [2007]) a method is proposed for directly learning the low-dimensional represen-

tation without first using PCA, and they demonstrate better results. In (Xing et al. [2003]), they propose a method for learning a distance metric that minimizes the distance between all pairs of points in the same class. Such a method will not work well if data for each class is multimodal, or has clusters that are separated. In (He and Niyogi [2003]), the authors propose a method known as locality preserving projection (LPP) for calculating a low-dimensional linear projection. The method uses a nearest neighbor based adjacency graph where the edges between the neighbors may have some weights. Nearby points in the original space are kept close in the learned low-dimensional space. This method is not specifically discriminative. In (Sugiyama [2007]), they propose a method called local Fisher discriminant analysis FLDA that generalizes LDA to include nearest neighbor information in a manner similar to LPP. This method is better able to cope with multimodal class distributions. However, the weights between neighbors is again predefined for both LPP and FLDA. In (Globerson and Roweis [2006]), the author proposes a method called maximally collapsing metric learning (MCML) that learns a Mahalanobis distance metric that attempts to project all points from a single class to one point while maximizing the distance to points of other classes. This method also may not work well for multimodal data within a single class. In (Peltonen et al. [2007]), a method is presented for learning an embedding in a semi-supervised setting with the assumption that each class is represented by a mixture of Gaussians. This method may be very useful to apply to acoustic modeling when using unlabeled data to supplement the training set.

There are many non-linear dimensionality reduction techniques. One interesting technique is locally linear embedding (LLE) introduced by (Roweis and Saul [2000]). They present a very intuitive algorithm for embedding points in a lower dimensional space. Essentially each point is first constructed from points within its neighborhood where each point gets a weight w_{ij} which indicates how much j contributes to the reconstruction of i. Then the points are all mapped to a lower dimensional space with the idea that the reconstruction weights and points in the new space should still result in a good reconstruction. It is unclear, however, how test points could be projected into the lower dimensional space and no specific attempt at discrimi-

nation is made. A method is described for non-linear dimensionality reduction that preserves local information in (Belkin and Niyogi [2003]). Salakhutdinov et al propose a method for learning a non-linear embedding that is discriminative and also based on neighborhood structure (Salakhutdinov and Hinton [2007]). In (Hinton and Salakhutdinov [2006]), the authors describe a method for constructing a neural network that allows for non-linear compression of data to a low-dimensionality as well as a reconstruction of the data from the learned low-dimensional code. The method is efficient and provides good results. Non-linear embeddings can be quite effective but can be computationally more expensive than linear embeddings and analysis of the resulting dimensions can be difficult whereas with linear embeddings the resulting dimensions are simply linear combinations of the original features and therefore easier to interpret.

There are alternatives to HLDA that have been proposed and applied specifically to acoustic modeling problems that also learn discriminative projections. In (Zhang and Matsoukas [2005]), an alternative to HLDA is presented that optimizes a minimum phoneme error (MPE) criterion, that is more closely related to the end goal of speech recognition performance. This type of discriminative projection can also be used for speaker adaptation (Wang and Woodland [2004]). HLDA has also been effectively applied to speaker adaptation (Matsoukas and Schwartz [2003], Saon et al. [2001]). For a single speaker, HLDA will most likely perform better than in the speaker-independent projections we learn here because multiple speakers can introduce a high amount of variability in the data. In (Povey et al. [2005]), the authors introduce a method called fMPE; a linear projection of a large non-linear expansion of the feature space is performed with improved results.

4.6 Lessons

In this chapter, NCA was shown to deliver significant improvements in speech recognition word error rates over PCA and HLDA. However, NCA has drawbacks, including increased computational cost and non-convex optimization. Parallelizing the NCA

optimization can help with computational time during training, and the results show that NCA can sometimes be well trained with a relatively small amount of data. Though NCA has a non-convex optimization, with appropriate care to the initialization and optimization of the projection, good results are easy to attain. A regularized version of NCA was introduced with slightly improved results.

NCA provided better kNN classification results than HLDA, but the improvement in speech recognition performance is more tempered. This is probably due to the other factors affecting speech recognition, such as the language and pronunciation models. Also the projections trained by NCA and HLDA were used to train GMM models for attaining acoustic model scores, whose scores are very different from the classification decisions provided by kNN.

In the subsequent chapters, the neighborhood analysis framework introduced here will be applied and expanded to better address the acoustic modeling problem. The nearest neighbor-based estimates will also be applied directly in the recognizer.

Chapter 5

Learning Label Embeddings for Modeling Structure in the Label Space (NCA-ECOC)

In this chapter, the problem of providing class-conditional probability estimates $p(y|\mathbf{x})$ is investigated under conditions where the number of possible labels y is large and the labels share some underlying structure. To solve this problem, a method is proposed for using label embeddings, similar to error-correcting output codes (ECOCs), to model the relationship between labels. A label embedding is learned within the neighborhood analysis framework developed in Chapter 2 and Chapter 4. The learned label embedding and nearest neighbor information are used to provide class-conditional probability estimates. These estimates are applied directly to the problem of acoustic modeling for speech recognition to demonstrate significant improvements in terms of word error rate (WER) on the academic lecture recognition task over a baseline GMM model.[1]

[1]Much of the work presented in this chapter has been previously published in (Singh-Miller and Collins [2009]).

5.1 Introduction

For some applications the training set may have some of the following properties: (1) the size of the label space $|\mathcal{Y}|$ is large, (2) some labels have relatively few training samples, (3) there tends to be a lot of overlap between the labels, and (4) the labels assigned to training samples may be noisy. Estimating $p(y|\mathbf{x})$ based on the neighborhood of x under these conditions can be difficult. For example, consider a test point $\mathbf{x}$ whose neighborhood contains many samples with the phonetic labels `/s/` and `/ae/` and a few with the transition label `/s/->/ae/` (indicating a transition from the phone `/s/` to `/ae/`). The NCA model would give a high probability to the first two phonetic classes and a low probability to the transition class. Though, intuitively, one might think that the presence of neighbors from both sides of the transition should bolster the probability of the transition label being the correct choice. From this example, we can see that the label y of a test point's neighbor may provide evidence for improving the class-conditional estimate for other classes as well. This occurs because in the acoustic modeling problem, as in many problems, there is some underlying structure or relationship between separate labels in the label space.

In order to discover the structure between the labels and use it to improve class-conditional probability estimates, the similarity between pairs of labels is modeled explicitly. To estimate these similarity values, a label embedding approach is proposed. In this approach each label y is represented by an L dimensional real-valued vector $\mathbf{M}_y \in \mathbb{R}^L$. This vector is called the *prototype vector* or *output code* representing the class y and is related to the idea of error-correcting output codes (Allwein et al. [2000], Crammer and Singer [2000], Dietterich and Bakiri [1995], Klautau et al. [2003a]). The similarity between two classes y and z can be modeled using the output codes $\mathbf{M}_y$ and $\mathbf{M}_z$ for the two classes.

The output codes are learned within a neighborhood analysis framework related to the NCA framework from the previous chapter. The prototype vectors $\mathbf{M}_y$ are used to calculate $p(y|\mathbf{x})$, and are learned by optimizing a leave-one-out estimate of the conditional log-likelihood (CLL) over the training samples. The end result is a

method that embeds labels y into $\mathbb{R}^L$ in a way that can significantly improve class-conditional probability estimates and reveal some aspects of the underlying structure of the label space.

The application focused on is acoustic modeling for speech recognition, where each input $\mathbf{x} \in \mathbb{R}^D$ is a vector of measured acoustic features, and each label $y \in \mathcal{Y}$ is an acoustic-phonetic label. As is common in speech recognition applications, the size of the label space $\mathcal{Y}$ is large (for the academic lecture task there are 1871 possible labels), and there is significant structure within the labels: many acoustic-phonetic labels are highly correlated or confusable, and many share underlying phonological features. Experiments are conducted measuring both conditional log-likelihood of test data, and word error rates when the method is incorporated within a full speech recognizer. In both settings the experiments show significant improvements for the NCA-ECOC method over both baseline nearest neighbor methods (e.g., the NCA method of (Goldberger et al. [2005])), as well as Gaussian mixture models (GMMs), as conventionally used in speech recognition systems.

While the experiments are on speech recognition, the method should be relevant to other domains which involve large multi-class problems with structured labels—for example problems in natural language processing, or in computer vision (e.g., see (Torralba et al. [June 2008]) for a recent use of neighborhood components analysis (NCA) (Goldberger et al. [2005]) within an object-recognition task with a very large number of object labels). Note also that the approach is relatively efficient: the model is trained on around 11 million training examples.

5.2 Sources of Structure Between Acoustic-Phonetic Labels

There are several possible sources of structure underlying the acoustic-phonetic label set used in the academic lecture recognition task.

The labels are either internal phonetic labels, representing a single phonetic or

acoustic category (e.g. `/ow/`), or transition labels that represent the transition from one of these categories to another (e.g. `/g/->/ow/`. It is clear then the transition labels may be strongly related to the internal label for both its left and right hand sides.

The phones modeled with the labels also are known to share certain underlying phonological properties. These properties include manner of articulation, place of articulation, voicing, rounding, front-back, etc. Multiple phones may share the same value of these underlying features. For a more detailed discussion of the acoustic-phonetic properties of these classes see (Stevens [2000]).

Another possible relation between the labels may arise because of frequent co-occurrence of labels. This may happen if two labels occur next to each other in a common word for instance. Frequent co-occurrence between two labels may lead to more frequent confusions between the two or an expectation of seeing one when the other occurs.

Furthermore, evidence has been presented in speech recognition literature to indicate that phonetic boundaries and the identity of the phone itself can be difficult to pinpoint in a speech stream. In the work of (Fosler-Lussier et al. [1999]), the phenomenon of "feature spreading" and "cue-trading" for the phonetic realization of segments are discussed. Feature spreading occurs when "segments are deleted entirely in production, though their influence is often manifest in the phonetic properties of the segmental neighbors" (Fosler-Lussier et al. [1999]). This phenomenon makes it difficult to separate phonetic segments and indeed to correctly classify them. Cue-trading occurs when "phonetic realizations often occur in place of canonical acoustic patterns...[where] there is no evidence for very predictable words (e.g. more *of* that" (Fosler-Lussier et al. [1999])). Both these phenomenon are evidence against the "beads-on-a-string" model of speech, or the assumption that each segment's acoustic characteristics depend only on the phonetic identity of the segment. Clearly the identity of the surrounding phones can also have an impact on the acoustic properties of a segment. Trying to model the underlying structure of the chosen label space can help weaken the "beads-on-a-string" assumption.

5.3 Baseline Models

There are two baseline models that make use of neighborhood information to estimate $p(y|\mathbf{x})$ that will be considered. The first is the NCA model from Chapter 4, and the second is a model based on the kNN classification algorithm. All the models described in this chapter will be learned using a training set of samples $\{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_N, y_N)\}$, where $\mathbf{x} \in \mathbb{R}^D$ is a feature vector representing some input, and y is a label drawn from a set of possible labels $\mathcal{Y}$.

5.3.1 NCA Model

The NCA model is optimized using the log-likelihood criterion from Equation 4.3. This criterion optimizes the conditional log-likelihood of the training points:

$$\sum_{i=1}^{N} \log p(y_i|\mathbf{x}_i)$$

Optimizing the log-likelihood rather than simple classification error is performed because the estimates will be applied within a larger system, in this case a speech recognizer, where the probabilities will be propagated throughout the recognition model; hence it is important for the model to provide well-calibrated class-conditional probability estimates.

The NCA model learns a projection matrix $\mathbf{A}$ of size $d \times D$ with $d < D$. This provides a new representation $\mathbf{a} = \mathbf{A}\mathbf{x}$ for the original samples $\mathbf{x}$ where $\mathbf{a}$ is now a d dimensional real-valued vector. For the duration of this chapter, the projection matrix $\mathbf{A}$ will be fixed and the representation $\mathbf{a}$ will be used to train the other models. Therefore the training set will now be $\{(\mathbf{a}_1, y_1), ..., (\mathbf{a}_N, y_N)\}$. Now, the NCA method for providing the class-conditional probability estimates for a test point $\mathbf{a}$ can be described using the following equations:

$$\alpha_j(\mathbf{a}) = e^{-||\mathbf{a}-\mathbf{a}_j||^2} \tag{5.1}$$

$$p_{nca}(y|\mathbf{a}) = \frac{\sum_{j=1,y_j=y}^{N} \alpha_j(\mathbf{a})}{\sum_{j=1}^{N} \alpha_j(\mathbf{a})} \tag{5.2}$$

In NCA, for any test point $\mathbf{a}$, the distribution $\alpha(j|\mathbf{a})$ decreases rapidly as the distance between $\mathbf{a}$ and $\mathbf{a}_j$ increases. Because the matrix $\mathbf{A}$ is trained using the log-likelihood criterion, the resulting NCA estimates and the representation $\mathbf{a}$ are well-calibrated in terms of using nearest neighbors to estimate $p(y|\mathbf{a})$ through Eq. 5.2. A first baseline method for our problem is therefore to directly use the estimates defined by Eq. 5.2.

We will, however, see that this baseline method performs poorly at providing estimates of $p(y|\mathbf{a})$ within the speech recognition application. Importantly, the model fails to exploit the underlying structure or correlations within the label space. For example, consider a test point that has many neighbors with the phonetic label `/s/`. This should be evidence that closely related phones, `/sh/` for instance, should also get a relatively high probability under the model, but the model is unable to capture this effect.

5.3.2 kNN Model

A second baseline model for estimating $p(y|\mathbf{a})$ is based on the kNN algorithm. This baseline uses the k nearest neighbors directly by estimating $p_k(y|\mathbf{a})$ to be the number of k nearest neighbors from the training set with label y divided by the total number of neighbors considered, k.

$$p_k(y|\mathbf{a}) = \frac{\#\text{ of } k\text{-nearest neighbors of } \mathbf{a} \text{ in training set with label } y}{k} \tag{5.3}$$

Here the choice of k is crucial. A small k will be very sensitive to noise and necessarily lead to many classes receiving a probability of zero, which is undesirable for our application. On the other hand, if k is too large, samples from far outside the neighborhood of $\mathbf{a}$ will influence $p_k(y|\mathbf{a})$, and the estimate will start to resemble the prior distribution over the labels, $p(y)$. A baseline method, p_{nn} that interpolates estimates

from several different values of k can mitigate these detrimental effects.

$$p_{nn}(y|\mathbf{a};\bar{\lambda}) = \sum_{k\in\mathcal{K}} \lambda_k p_k(y|\mathbf{a}) \tag{5.4}$$

This distribution is an interpolation of many distributions calculated for different values of k drawn from some pre-defined set $\mathcal{K}$. The parameters λ_k define the mixing proportions for the individual distributions $p_k(y|\mathbf{a})$, and together are denoted by $\bar{\lambda}$. For all k, $0 \leq \lambda_k \leq 1$ and, $\sum_{k\in\mathcal{K}} \lambda_k = 1$. Methods for estimating these parameters are discussed in the experiments section of this chapter.

This baseline will be useful within the final speech recognition model, but again suffers from the fact that it does not model the underlying structure of the label space.

5.4 Error-Correcting Output Codes for Class Conditional Probability Estimation

A new model, called NCA-ECOC is proposed that uses error correcting output codes to explicitly represent and learn the underlying structure of the label space $\mathcal{Y}$. Each label y is represented by a prototype vector $\mathbf{M}_y \in \mathbb{R}^L$ in this model. To measure the similarity between class y and class z, the inner product of their corresponding prototype vectors will be used.

$$similarity(y, z) = \langle \mathbf{M}_y, \mathbf{M}_z \rangle \tag{5.5}$$

This similarity measure is symmetric since $similarity(y, z) = similarity(z, y)$, though it is possible to use an asymmetric measure. Note that other measures of similarity, such as Euclidean distance, could also be used here.

The vectors $\mathbf{M}_y$ will be learned automatically using an NCA-style paradigm, effectively learning an embedding of the labels in $\mathbb{R}^L$. In this section first the structure of the model is described, and then a method for training the parameters of the model

is presented (i.e., learning the prototype vectors $\mathbf{M}_y$).

5.4.1 NCA-ECOC Model

The NCA-ECOC model makes direct use of the learned output codes $\mathbf{M}_y$ when providing the class-conditional probability estimates $p(y|\mathbf{a})$. When considering a test sample $\mathbf{a}$, first weights $\alpha_j(\mathbf{a})$ are assigned to points $\mathbf{a}_j$ from the training set through the NCA definition in Eq. 5.1. Let $\mathbf{M}$ be a matrix that contains all the prototype vectors $\mathbf{M}_y$ as its rows. A vector $H(\mathbf{a};\mathbf{M})$ can then be constructed that uses the weights $\alpha_j(\mathbf{a})$ and the true labels of the training samples to calculate the expected value of the output code representing $\mathbf{a}$.

$$H(\mathbf{a};\mathbf{M}) = \frac{\sum_{j=1}^{N} \alpha_j(\mathbf{a})\mathbf{M}_{y_j}}{\sum_{j=1}^{N} \alpha_j(\mathbf{a})} \tag{5.6}$$

This vector is essentially a weighted average of the prototype vectors for each label, where the weights consist of the NCA estimate for the probability of the label. Therefore the above equation is equivalent to the following.

$$H(\mathbf{a};\mathbf{M}) = \sum_{y\in\mathcal{Y}} p_{nca}(y|\mathbf{a})\mathbf{M}_y \tag{5.7}$$

The *representative output code*, $H(\mathbf{a};\mathbf{M})$, can be used to directly provide an estimate of $p(y|\mathbf{a})$. Using the definition of similarity from Equation 5.5, the new class conditional probability estimate is the following:

$$p_{ecoc}(y|\mathbf{a};\mathbf{M}) = \frac{e^{\langle \mathbf{M}_y, H(\mathbf{a};\mathbf{M})\rangle}}{\sum_{y'\in\mathcal{Y}} e^{\langle \mathbf{M}_{y'}, H(\mathbf{a};\mathbf{M})\rangle}} \tag{5.8}$$

The similarity between the representative output code $H(\mathbf{a};\mathbf{M})$ and the output code for a label $\mathbf{M}_y$ can be either positive or negative, and exponentiating this value ensures a positive probability distribution. The denominator of the above equation simply normalizes the estimate over the contributions of all possible labels. This distribution assigns most of the probability for a sample vector $\mathbf{a}$ to classes whose prototype vectors

have a large inner product with $H(\mathbf{a}; \mathbf{M})$. All labels receive a non-zero weight under $p_{ecoc}(y|\mathbf{a}; \mathbf{M})$.

5.4.2 Properties of the NCA-ECOC Model

There are some interesting special cases of estimates that can arise from the NCA-ECOC model.

- If a sample $\mathbf{a}$ had a representative output code $H(\mathbf{a}; \mathbf{M})$ that was equal to 0 for all dimensions, the estimate of $p_{ecoc}(y|\mathbf{a})$ would be equivalent to $\frac{1}{|\mathcal{K}|}$, a uniform prior over the labels.

- It is possible for the NCA estimate of a label to be very high, say $p_{nca}(y|\mathbf{a}) = 1$, but have y not be the top candidate selected by the NCA-ECOC model. This can happen if $similarity(y, y) < similarity(y, z)$ for some other label z which may occur if $\langle \mathbf{M}_y, \mathbf{M}_y \rangle < \langle \mathbf{M}_y, \mathbf{M}_z \rangle$.

- Even if the estimate for a class under the NCA model is very small or equal to 0, the estimate under the NCA-ECOC model will be non-zero.

- If the matrix of output codes was the identity matrix, where all output codes have the length $L = |\mathcal{K}|$, the classification decisions of the NCA-ECOC and NCA models would be the same, but the class-conditional estimates would be different.

The probability estimate for p_{ecoc} can be decomposed as follows:

$$\begin{aligned} p_{ecoc}(y|\mathbf{a}; \mathbf{M}) =& \frac{e^{\langle \mathbf{M}_y, H(\mathbf{a};\mathbf{M}) \rangle}}{\sum_{y' \in \mathcal{Y}} e^{\langle \mathbf{M}_{y'}, H(\mathbf{a};\mathbf{M}) \rangle}} \\ =& \frac{e^{\sum_{z \in \mathcal{Y}} p_{nca}(z|\mathbf{a}) \langle \mathbf{M}_y, \mathbf{M}_z \rangle}}{\sum_{y' \in \mathcal{Y}} e^{\sum_{z \in \mathcal{Y}} p_{nca}(z|\mathbf{a}) \langle \mathbf{M}_{y'}, \mathbf{M}_z \rangle}} \\ =& \frac{\prod_{z \in \mathcal{Y}} e^{p_{nca}(z|\mathbf{a}) \langle \mathbf{M}_y, \mathbf{M}_z \rangle}}{\sum_{y' \in \mathcal{Y}} \prod_{z \in \mathcal{Y}} e^{p_{nca}(z|\mathbf{a}) \langle \mathbf{M}_{y'}, \mathbf{M}_z \rangle}} \end{aligned}$$

$$p_{ecoc}(y|\mathbf{a}; \mathbf{M}) \propto \prod_{z \in \mathcal{Y}} e^{p_{nca}(z|\mathbf{a}) \langle \mathbf{M}_y, \mathbf{M}_z \rangle} \tag{5.9}$$

This decomposition of the estimate makes it easier to see how the estimate varies with the inner product between the output codes for two labels. It is clear from Equation 5.9 that when computing $p_{ecoc}(y|\mathbf{a})$, the similarity between y and every possible label z is used. There are three possible effects each label z can have on the expression in Equation 5.9: (1) if $\langle \mathbf{M}_y, \mathbf{M}_z \rangle$ is 0, then there is no net effect, (2) if $\langle \mathbf{M}_y, \mathbf{M}_z \rangle$ is greater than 0, then the net effect is positive, and (3) if $\langle \mathbf{M}_y, \mathbf{M}_z \rangle$ is less than 0, then the net effect is negative. Intuitively, $p_{ecoc}(y|\mathbf{a})$ is high if labels that are similar to y receive a high probability under p_{nca}, and $p_{ecoc}(y|\mathbf{a})$ is low if labels that are dissimilar to y receive a high probability under p_{nca}.

5.5 NCA-ECOC Optimization Criterion

As in NCA, the NCA-ECOC method uses a leave-one-out optimization criterion, which is particularly convenient within nearest-neighbor approaches. The optimization problem will be to maximize the conditional log-likelihood function.

$$f_{ecoc}(\mathbf{M}) = \sum_{i=1}^{N} \log p_{ecoc}^{(loo)}(y_i|\mathbf{a}_i; \mathbf{M}) \tag{5.10}$$

Here $p_{ecoc}^{(loo)}(y_i|\mathbf{a}_i; \mathbf{M})$ is a leave-one-out estimate of the probability of label y_i given the input $\mathbf{a}_i$, assuming an ECOC matrix $\mathbf{M}$. This criterion is related to the classification performance of the training data and also discourages the assignment of very low probability to the correct class, both of which are important for the acoustic modeling application.

The leave-one-out estimates for the training set are given through the following definitions. The weights $\alpha_j(i)$ and the probabilities p_{ij} are defined as in NCA except the projection matrix $\mathbf{A}$ is now fixed:

$$\alpha_j(i) = e^{-||\mathbf{a}_i - \mathbf{a}_j||^2} \text{ if } i \neq j \text{ and 0 otherwise} \tag{5.11}$$

$$p_{ij} = \frac{\alpha_j(i)}{\sum_{m=1}^{N} \alpha_m(i)} \tag{5.12}$$

Again note that $p_{ii} = 0$ for all i, indicating that each point contributes nothing to the labeling of itself. The representative output code is then defined as:

$$H(i;\mathbf{M}) = \sum_{j=1}^{N} p_{ij}\mathbf{M}_{y_j} \tag{5.13}$$

This equation is equivalent to the following equation, which can help reduce the complexity of the optimization.

$$H(i;\mathbf{M}) = \sum_{y\in\mathcal{Y}} p_{nca}(y|i)\mathbf{M}_{y} \tag{5.14}$$

where

$$p_{nca}(y|i) = \sum_{j=1,y_j=y}^{N} p_{ij} \tag{5.15}$$

The NCA-ECOC leave-one-out class conditional estimates for the training samples are then:

$$p_{ecoc}(y|i;\mathbf{M}) = \frac{e^{\langle \mathbf{M}_y, H(i;\mathbf{M})\rangle}}{\sum_{y'\in\mathcal{Y}} e^{\langle \mathbf{M}_{y'}, H(i;\mathbf{M})\rangle}} \tag{5.16}$$

The optimization of the training criterion can be performed using gradient methods similar to those used to train the NCA criterion.

5.6 Learning the NCA-ECOC Model Parameters

The gradient derived from the optimization criterion $f_{ecoc}(\mathbf{M})$ is the following:

$$\begin{aligned}
\frac{\partial f_{ecoc}(\mathbf{M})}{\partial \mathbf{M}_z} &= \nabla(z) - \nabla'(z) \\
\nabla(z) &= \sum_{i=1}^{N}\sum_{j=1}^{N}[p_{ij}(\delta_{z,y_i}\mathbf{M}_{y_j} + \delta_{y_j,z}\mathbf{M}_{y_i})] \\
\nabla'(z) &= \sum_{i=1}^{N}\sum_{y'\in\mathcal{Y}} p_{ecoc}(y'|i;\mathbf{M})\left(\sum_{j=1}^{N}[p_{ij}(\delta_{z,y'}\mathbf{M}_{y_j} + \delta_{y_j,z}\mathbf{M}_{y'})]\right)
\end{aligned} \tag{5.17}$$

Here δ is the Kronecker delta; $\delta_{a,b} = 1$ if $a = b$ and $\delta_{a,b} = 0$ if $a \neq b$. To improve the computational efficiency, this gradient can be rewritten.

$$\begin{aligned} \nabla(z) &= \sum_{i=1}^{N} \sum_{y \in \mathcal{Y}} [p_{nca}(y|i)(\delta_{z,y_i}\mathbf{M}_y + \delta_{y,z}\mathbf{M}_{y_i})] \\ \nabla'(z) &= \sum_{i=1}^{N} \sum_{y' \in \mathcal{Y}} p_{ecoc}(y'|i;\mathbf{M}) \left(\sum_{y \in \mathcal{Y}}^{N} [p_{nca}(y|i)(\delta_{z,y'}\mathbf{M}_y + \delta_{y,z}\mathbf{M}_{y'})] \right) \end{aligned} \tag{5.18}$$

This gradient can be calculated most efficiently if it is possible to precompute and store the values of $p_{nca}(y|i)$ for all training samples i and classes y. In Figure 5-1 an algorithm for computing the gradient is presented. The computation of the gradient can be used for each iteration of an optimization algorithm such as conjugate gradient ascent. Stochastic alternatives for optimizing $f_{ecoc}(\mathbf{M})$ could also easily be implemented. Note that this optimization is non-convex and it is possible to converge to a local optimum. In the experiments described here the prototype vectors were initialized randomly for each class by selecting each parameter of each vector randomly from $[-\sigma, \sigma]$ where σ was selected to be $\sigma = 0.01$.

5.6.1 Selecting the Length of the Prototype Vectors, L

For the acoustic modeling experiments on the academic lecture data set, the length of the prototype vectors is selected experimentally. A set of approximately 115,000 held-out labeled samples (DevSet1) are used to select L. The average conditional log-likelihood achieved by different values of L are listed in Table 5.1. The prototype vectors are randomly initialized for each experiment as described above. The performance of the method improves initially as the size of L increases, but the objective levels off around $L = 40$. Performance deteriorates slightly as the value of L increases, which may be due to over-fitting.

Inputs:
Training samples $\{(\mathbf{a}_1, y_1), ..., (\mathbf{a}_N, y_N)\}$. Length of output codes L. Initialization parameter σ.

Initialization:

1. Initialize the output codes $\mathbf{M}_y$ for all $y \in \mathcal{Y}$ by randomly selecting $\mathbf{M}_y$ from $[-\sigma, \sigma]^L$.
2. Precompute and store $p_{nca}(y|i)$:

for $i = 1 : N$
  for $j = 1 : N$
    $\alpha_j(i) \leftarrow e^{-||\mathbf{a}_i - \mathbf{a}_j||^2}$
  $\alpha_i(i) \leftarrow 0$
  $t_{nca}(i) \leftarrow \sum_{j=1}^{N} \alpha_j(i)$
  for $y \in \mathcal{Y}$
    $p_{nca}(y|i) \leftarrow \frac{\sum_{j=1, y_j = y}^{N} \alpha_j(i)}{t_{nca}(i)}$

Algorithm for calculating gradient:

for all $y \in \mathcal{Y}$, $df(y) \leftarrow [0]^L$

for $i = 1 : N$

1. Calculate $p_{ecoc}(y|i)$:

  $H(i; \mathbf{M}) \leftarrow \sum_{y \in \mathcal{Y}} p_{nca}(y|i) \mathbf{M}_y$
  $t_{ecoc}(i) \leftarrow \sum_{y \in \mathcal{Y}} e^{\langle \mathbf{M}_y, H(i;\mathbf{M}) \rangle}$
  for $y \in \mathcal{Y}$
    $p_{ecoc}(y|i) \leftarrow \frac{e^{\langle \mathbf{M}_y, H(i;\mathbf{M}) \rangle}}{t_{ecoc}(i)}$

2. Calculate each training sample's contribution to the gradient:

  for $y \in \mathcal{Y}$
    $df(y) \leftarrow df(y) + p_{nca}(y|i) \mathbf{M}_{y_i}$
    $df(y_i) \leftarrow df(y_i) + p_{nca}(y|i) \mathbf{M}_y$
  for $y' \in \mathcal{Y}$
    for $y \in \mathcal{Y}$
      $df(y) \leftarrow df(y) - p_{ecoc}(y'|i) p_{nca}(y|i) \mathbf{M}_{y'}$
      $df(y') \leftarrow df(y') - p_{ecoc}(y'|i) p_{nca}(y|i) \mathbf{M}_y$

Output:
The gradient $df(y)$ for each label y.

Figure 5-1: The algorithm for calculating the gradient of NCA-ECOC.

L	average CLL
2	-4.388
10	-2.748
20	-2.580
30	-2.454
40	-2.432
50	-2.470
60	-2.481

Table 5.1: Average conditional log-likelihood achieved by NCA-ECOC model over DevSet1 for different values of L

5.7 Experiments on Log-Likelihood

In this section experiments on the academic lecture data described in Section 3.5.2 are conducted to determine the effectiveness of several methods in terms of their ability to provide good class-conditional probability estimates. The 112 dimensional data is initially projected down to 50 dimensions using the NCA projection learned in Chapter 3. There are about 11.5 million training samples, and 1871 distinct class labels. Additionally there are two sets of held out labeled samples, each containing about 115,000 points called DevSet1 and DevSet2.

In total six models are compared in terms of their conditional log-likelihood performance on DevSet1.

NCA Model (p_{nca})

The estimates $p_{nca}(y|\mathbf{a})$ are calculated using the equations defined in Section 5.3.1.

kNN Model (p_{nn})

The baseline model, p_{nn}, makes use of estimates $p_k(y|\mathbf{a})$ as defined in section 5.3.2. The set $\mathcal{K}$ is a set of integers representing different values for k, the number of nearest neighbors used to evaluate p_k. In these experiments,

$$\mathcal{K} = \{5, 10, 20, 30, 50, 100, 250, 500, 1000\}.$$

Additionally, we assume c functions over the labels, $p_1(y)$, ..., $p_c(y)$. These can be thought of as different prior functions over the labels to help smooth the nearest-neighbor based estimate and ensure the $p_{nn}(y|\mathbf{a})$ is not equal to 0 for any label y. The model is then defined as

$$p_{nn}(y|\mathbf{a};\bar{\lambda}) = \sum_{k\in\mathcal{K}} \lambda_k p_k(y|\mathbf{a}) + \sum_{j=1}^{c} \lambda_j^0 p_j(y) \tag{5.19}$$

where $\lambda_k \geq 0, \forall k \in \mathcal{K}$, $\lambda_j^0 \geq 0$ for $j = 1, ..., c$, and $\sum_{k\in\mathcal{K}} \lambda_k + \sum_{j=1}^{c} \lambda_j^0 = 1$. The $\bar{\lambda}$ values are estimated using the EM algorithm on a validation set of examples (DevSet2).

Three functions over the labels are used in these experiments. Each of the three distributions is based on the prior probabilities, $p(y)$, of the 1871 acoustic phonetic classes. The set of labels, $\mathcal{Y}$, is divided into three disjoint categories.

- $\mathcal{Y}^{(1)}$ includes labels involving *internal* phonetic events (e.g. `/ay/`)
- $\mathcal{Y}^{(2)}$ includes labels involving the *transition* from one phonetic event to another (e.g. `/ow/->/ch/`)
- $\mathcal{Y}^{(3)}$ includes labels involving only *non-phonetic* events like noise and silence

A distribution $p^{(1)}(y)$ for the first category is then defined as follows. Similar distributions are defined for the other two categories.

$$p^{(1)}(y) = \frac{\begin{cases} p(y), & \text{if } y \in \mathcal{Y}^{(1)} \\ 0, & \text{otherwise} \end{cases}}{\sum_{y'\in\mathcal{Y}^{(1)}} p(y')} \tag{5.20}$$

The prior function over the labels was partitioned in this way because each of the three categories may behave quite differently. In particular, members of the second category tend to have fewer examples and perhaps should receive a higher backoff weight λ. Indeed this effect is seen when tuning the $\bar{\lambda}$ parameters, with category 2 receiving the highest weight, followed by category 1, and category 3 receiving the lowest weight (probably because it contains few classes with many samples and hence doesn't require much smoothing).

NCA-ECOC Model (p_{ecoc})

The estimates $p_{ecoc}(y|\mathbf{a})$ are calculated using the equations defined in Section 5.4.1.

Combining NCA-ECOC and kNN (p_{mix})

The information in the p_{nn} and p_{ecoc} models are combined using interpolation into a fourth model p_{mix}. This model makes use of nearest neighbor information, the label structure information provided by the NCA-ECOC model, as well as the distributions over the labels described above. The model takes the following form:

$$p_{mix}(y|\mathbf{a};\bar{\lambda}) = \sum_{k\in\mathcal{K}} \lambda_k p_k(y|\mathbf{a}) + \sum_{j=1}^{c} \lambda_j^0 p_j(y) + \lambda_{ecoc} p_{ecoc}(y|\mathbf{a};\mathbf{M}) \tag{5.21}$$

The values of $\bar{\lambda}$ here have similar constraints as before; where $\lambda_k \geq 0, \forall k \in \mathcal{K}$, $\lambda_j^0 \geq 0$ for $j = 1, ..., c$, $\lambda_{ecoc} \geq 0$, and $\sum_{k\in\mathcal{K}} \lambda_k + \sum_{j=1}^{c} \lambda_j^0 + \lambda_{ecoc} = 1$. The $\bar{\lambda}$ values are estimated using the EM algorithm on a validation set of examples (DevSet2).

GMM model (p_{gmm})

A GMM model, as conventionally used in speech recognition systems, is also used to compute class-conditional probability estimates. The GMM is trained with the SUMMIT system (Glass [2003]) and is the same as the baseline GMM model used in Chapter 3. The GMM defines a generative model $p_{gmm}(\mathbf{a}|y)$; a conditional model can be derived as follows:

$$p_{gmm}(y|\mathbf{a}) = \frac{p_{gmm}(\mathbf{a}|y)^r p(y)}{\sum_{y'\in\mathcal{Y}} p_{gmm}(\mathbf{a}|y')^r p(y')} \tag{5.22}$$

The parameter r is a smoothing parameter and is selected experimentally to achieve maximum conditional log-likelihood on DevSet2. This parameter is useful because the GMM is trained for a maximum likelihood criterion and may be poorly calibrated for calculating conditional log-likelihoods. In these experiments r was selected to be 0.5 which indicates that indeed a large amount of smoothing is necessary to achieve good performance on conditional log-likelihood for the GMM model. $p(y)$ refers to

the prior over the labels calculated directly from their relative proportions in the training set.

Combining ECOC, GMM, and kNN (p_{full})

A final interpolated model combines nearest neighbor information, the label structure information provided by the NCA-ECOC model, the GMM estimates based on the global structure of each class, as well as the distributions over the labels described above. The model is called p_{full}:

$$p_{full}(y|\mathbf{a};\bar{\lambda}) = \sum_{k\in\mathcal{K}} \lambda_k p_k(y|\mathbf{a}) + \sum_{j=1}^{d} \lambda_j^0 p_j(y) + \lambda_{ecoc} p_{ecoc}(y|\mathbf{a};\mathbf{M}) + \lambda_{gmm} p_{gmm}(y|\mathbf{a}) \tag{5.23}$$

The values of $\bar{\lambda}$ are constrained as follows; $\lambda_k \geq 0, \forall k \in \mathcal{K}$, $\lambda_j^0 \geq 0$ for $j = 1, ..., c$, $\lambda_{ecoc} \geq 0$,$\lambda_{gmm} \geq 0$, and $\sum_{k\in\mathcal{K}} \lambda_k + \sum_{j=1}^{c} \lambda_j^0 + \lambda_{ecoc} + \lambda_{gmm} = 1$. Again, the $\bar{\lambda}$ values are estimated using the EM algorithm on a validation set of examples (DevSet2).

5.7.1 Perplexity Results

Table 5.2 contains the average conditional log-likelihood achieved on a set of acoustic samples (DevSet1) by each of the six proposed models. The perplexity achieved by the models is also noted, where the perplexity is simply e^{-x}, where x is the average CLL of the samples. These results show that p_{ecoc} clearly outperforms these two baseline nearest neighbor models, p_{nn} and p_{nca}. The NCA model performs the worst, probably because this model contains no smoothing. The interpolated model p_{mix} outperforms each of the previous three models and comes close to p_{gmm} in performance, with p_{gmm} giving slightly improved results. Results for p_{full} are shown in the final row in the table. This interpolated model gives a clear improvement over both the GMM and NCA-ECOC models alone. Thus the NCA-ECOC model, combined with additional nearest-neighbor information, can give a clear improvement over state-of-the-art GMMs on this task.

It is interesting to note that all these models still perform quite poorly in terms

Model	Average CLL on DevSet 1	Perplexity
P_{nca}	-2.657	14.25
P_{nn}	-2.535	12.61
P_{ecoc}	-2.432	11.38
P_{mix}	-2.337	10.35
P_{gmm}	-2.299	9.96
P_{full}	-2.165	8.71

Table 5.2: Average conditional log-likelihood (CLL) of p_{nca}, p_{nn}, p_{ecoc}, p_{mix}, p_{gmm} and p_{full} on DevSet1. The corresponding perplexity values are indicated as well where the perplexity is defined as e^{-x} given that x is the average CLL.

of conditional log-likelihood or perplexity. The perplexity for the best model is 8.71, showing that there is a long way to go towards improving these estimates.

5.8 Recognition Experiments

In this section experiments are conducted with a model that integrates the NCA-ECOC estimates within a full speech recognition system. The parameters $\bar{\lambda}$ are learned using both DevSet1 and DevSet2 for $p_{mix}(y|\mathbf{a})$. The conditional probability estimates cannot be used directly, and instead an estimate for $p(\mathbf{a}|y)$ needs to be derived for use by the recognizer. This can be done by using an estimate for $p(\mathbf{a}|y)$ proportional to $\frac{p(y|\mathbf{a})}{p(y)}$. This is motivated by Bayes rule and has been used previously by (Zavaliagkos et al. [1994]).

In these experiments the following two methods for calculating the acoustic model are considered.

- Baseline Model: $\beta_1 \log p_{gmm}(\mathbf{a}|y)$
- Augmented Model: $\beta_2 \log \left(\frac{\gamma p_{gmm}(y|\mathbf{a}) + (1-\gamma) p_{mix}(y|\mathbf{a})}{p(y)} \right)$

The baseline method is just a GMM model with the commonly used acoustic model scaling parameter β_1. The augmented model combines p_{gmm} linearly with p_{mix} using parameter γ and the log of the combination is scaled by parameter β_2. The parameters β_1, β_2, γ are selected using the downhill simplex algorithm by optimizing

Acoustic Model	WER (Development Set)	WER (Test Set)
Baseline Model	36.3	35.4
Augmented Model	**35.2**	**34.5**
Alternative Augmented Model	35.5	34.7

Table 5.3: WER of recognizer for different acoustic models on the development and test set.

WER over the development set (Press et al. [2007]). The development set consists of eight hours of data including six speakers and the test set consists of eight hours of data including five speakers. The mixing parameters within p_{mix} are determined using the EM algorithm to optimize perplexity performance over held-out samples as described in the previous section. Results for both methods on the development set and test set are presented in Table 5.3.

The augmented model outperforms the baseline GMM model. This indicates that the nearest neighbor information along with the ECOC embedding, can significantly improve the acoustic model. Overall, an absolute reduction of 1.1% in WER on the development set and 0.9% on the test set are achieved using the augmented acoustic model. These results are significant with $p < 0.001$ using the sign test calculated at the utterance level.

The following alternative augmented model was also tested:

- Alternative Augmented Model: $\beta_3 \log p_{gmm}(\mathbf{a}|y) + \beta_4 \log \left(\frac{p_{mix}(y|\mathbf{a})}{p(y)} \right)$

This method combines the GMM model with p_{mix} in log-space using scaling parameters β_3 and β_4. These parameters are also optimized for WER over the development set. This method outperforms the baseline model but performs slightly worse than the augmented model. Results for recognition experiments using this model are included in Table 5.3.

5.9 Discussion

5.9.1 Plot of a low-dimensional embedding

In order to get a sense of what is learned by the output codes of the NCA-ECOC model a plot of 2-D output codes can be constructed. Figure 5-2 shows a plot of the output codes learned when $L = 2$. The output codes are learned for 1871 classes, but only 73 internal acoustic-phonetic classes are shown in the plot for clarity. In the plot, classes of similar acoustic-phonetic category are shown in the same color and shape.

We can see that items of similar acoustic categories are often grouped closely together. For example, the vowels are close to each other in the bottom left quadrant, while the stop-closures are grouped together in the top right, the affricates in the top left, and the nasals in the bottom right. The fricatives are a little more spread out but usually grouped close to another fricative that shares some underlying phonological feature such as `/sh/` and `/zh/` which are both palatal and `/f/` and `/th/` which are both unvoiced.

Other properties of the data also emerge in the plot of the output codes. For example `/b/` is placed close to some of the vowels because `/b/` often precedes these vowels and can be difficult to recognize. Therefore a test sample whose neighborhood contains examples of these vowels is seen as evidence supporting the classification of the example as a `/b/`.

Additionally, the magnitude of the output code, or their distance from the origin, varies. Often this is a sign of their relative frequencies within the training set. For example the vectors for `/sh/` and `/zh/` are close together, but the magnitude of the more infrequent phone `/zh/` is smaller.

Figure 5-3 depicts the output codes for classes that represent transition or diphone labels. Classes in which `/z/` is the first phone is shown in the top plot and classes in which `/z/` is the second phone are displayed in the bottom plot. These plots show that the transition labels can often cluster near one or another of the corresponding internal phone labels. This effect may be stronger depending on the position of the

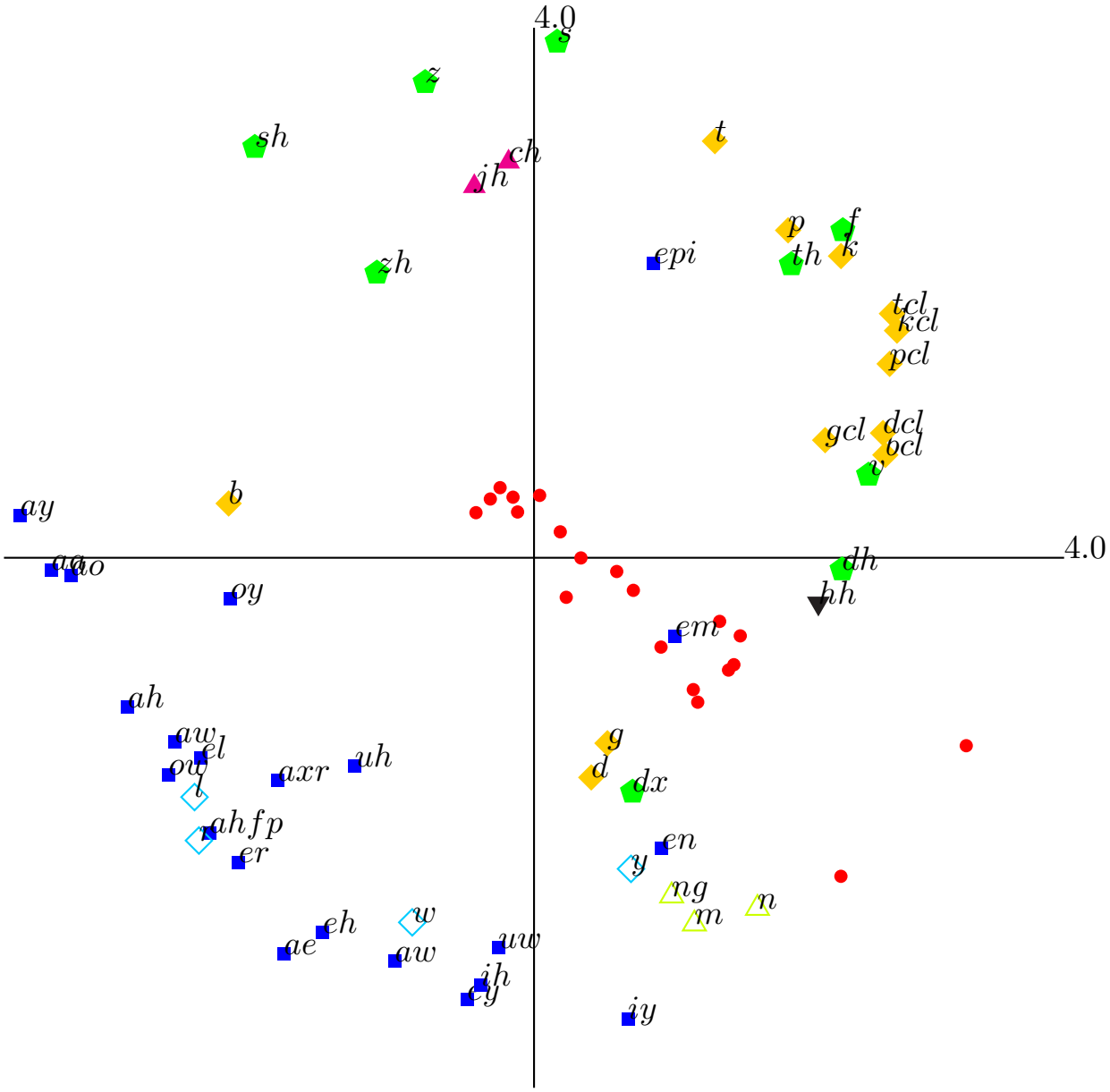

Figure 5-2: Plot of 2-dimensional output codes corresponding to 73 acoustic phonetic classes. The red circles indicate noise and silence classes. The phonetic classes are divided as follows: vowels, semivowels, nasals, stops and stop closures, fricatives, affricates, and the aspirant `/hh/`.

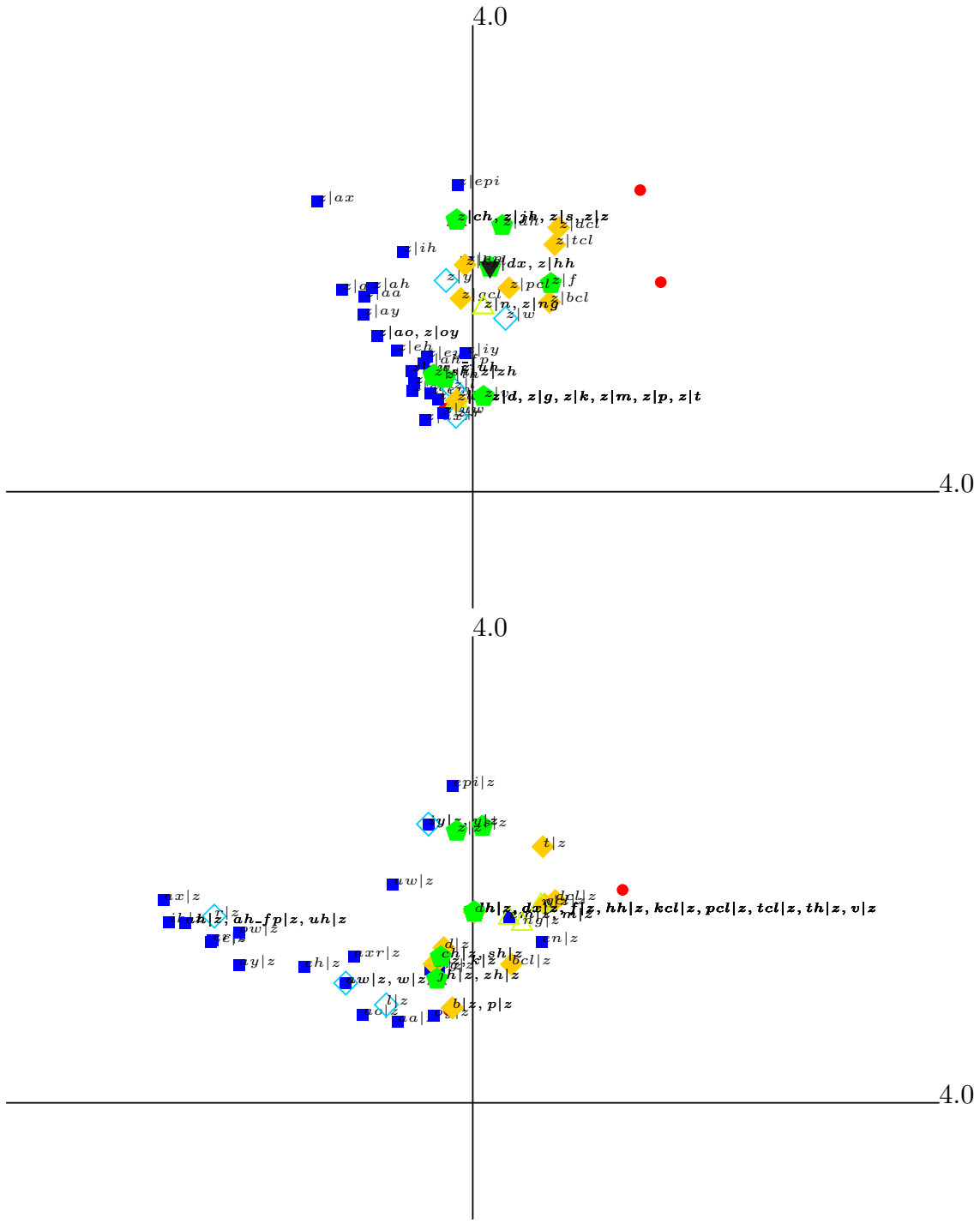

Figure 5-3: Plot of 2-dimensional output codes for two different subsets of the labels: (top) output codes for transition labels whose first phone is /z/ (bottom) output codes for transition labels whose second phone is /z/.

the phone as either the first or second element. When /z/ is the first of the two phones, the output codes are more tightly clustered than when /z/ is the second phone. These types of effects can be captured using the learned output codes and might be difficult to model using manually designed rules that indicate underlying phonological properties of the labels.

The plots in Figure 5-3 also suggest a potential difficulty with the NCA-ECOC model. Two labels a and b may each be similar to another label c. In some cases, it may be good to take this as evidence that a and b are also similar. For other cases, however a and b may in fact be dissimilar (e.g. /z/->/ey/ and /ey/->/z/ are similar to /z/ but dissimilar to each other). If the length of the output codes L is low, labels such as a and b are constrained to be similar, but as the length of L grows, such constraints are relaxed. Selecting a good value for L is therefore important to the performance of the method, as corroborated by the results in Table 5.1.

5.9.2 A Generalization of NCA-ECOC

The NCA-ECOC model presented in this chapter can be generalized to depend on the similarity measure. Let

$$s(y, z) = similarity(y, z) \tag{5.24}$$

As previously discussed the inner product of the output codes is only one way to measure similarity between labels. In the extreme case, $s(y, z)$ could be a parameter freely chosen from the reals and it need not be true that $s(y, z) = s(z, y)$. In this case the training estimate of $p(y|i)$ would be

$$p(y|i) = \frac{e^{\sum_{y' \in \mathcal{Y}} p_{nca}(y'|i) s(y,y')}}{\sum_{z \in \mathcal{Y}} e^{\sum_{y' \in \mathcal{Y}} p_{nca}(y'|i) s(z,y')}} \tag{5.25}$$

In the case of the academic lecture data this implies learning 3,500,641 free parameters and the model would be very under-constrained. The NCA-ECOC model constrains the values for $s(y, z)$ through the use of the limited length output codes. If each $s(y, z)$

value was unconstrained, regularization over the matrix of similarity values could be added to the training criterion to constrain the model. The generalized model also happens to be convex to estimate, whereas the NCA-ECOC model is not.

5.9.3 Co-learning NCA and NCA-ECOC

The ECOC embedding of the label space could also be co-learned with an embedding of the input vector space by extending the approach of NCA (Goldberger et al. [2005]). It would simply require the re-introduction of the projection matrix $\mathbf{A}$ in the weights $\alpha_j(\mathbf{x})$.

$$\alpha_j(\mathbf{x}) = e^{-||\mathbf{A}\mathbf{x}-\mathbf{A}\mathbf{x}_j||^2}$$

The representative output codes $H(\mathbf{x}; \mathbf{M}, \mathbf{A})$ and class-conditional probabilities $p_{ecoc}(y|\mathbf{x})$ would retain similar definitions.

The optimization criterion would now depend on both $\mathbf{A}$ and $\mathbf{M}$. Co-learning the two embeddings $\mathbf{M}$ and $\mathbf{A}$ could still be performed using gradient methods. The gradient rule for optimizing the output codes would not change. To optimize $\mathbf{A}$, there is the following gradient rule.

$$\begin{aligned}\frac{\partial f(\mathbf{A}, \mathbf{M})}{\partial \mathbf{A}} &= \sum_{i=1}^{N}\Big(\sum_{j=1, j\neq i}^{N} (\langle \mathbf{M}_{y_j}, \mathbf{M}_{y_i} \rangle \mathbf{U}_{ij}) - \Big(\sum_{y' \in \mathcal{Y}} p_{ecoc}(y'|i) \sum_{j=1, j\neq i}^{N} (\langle \mathbf{M}_{y_j}, \mathbf{M}_{y'} \rangle \mathbf{U}_{ij})\Big)\Big) \\ \mathbf{U}_{ij} &= -2\mathbf{A} p_{ij} \Big(\mathbf{x}_{ij}\mathbf{x}_{ij}^T - \sum_{m=1, m\neq i}^{N} p_{im}\mathbf{x}_{im}\mathbf{x}_{im}^T\Big)\end{aligned} \tag{5.26}$$

To train the two embeddings, one could alternate optimizing the ECOC and NCA parameters, or optimize both jointly.

5.10 Experiments with TIMIT

While the primary motivation for developing the NCA-ECOC method is to improve class-conditional probability estimates, it would be interesting to see if the method

DEVELOPMENT SET:

L	Error Rate
2	43.8
3	25.7
4	21.9
5	21.3
6	21.1
7	20.8
8	20.7
9	20.7
10	20.7

TEST SET:

L	Error Rate
2	44.5
3	26.5
4	23.0
5	22.2
6	21.8
7	21.7
8	21.4
9	21.5
10	21.5

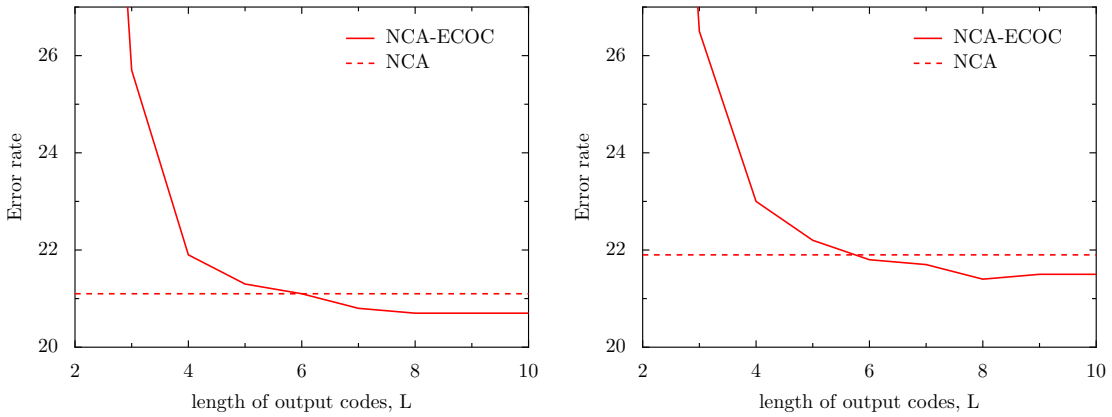

Figure 5-4: TIMIT classification results on the development and test set using NCA and NCA-ECOC models with multiple values of L, the length of the output codes.

can improve the error rate for classification tasks as well. The TIMIT task, like the lecture recognition tasks, has phonetic labels that have some underlying structure and therefore may be well-suited to the NCA-ECOC model. See Section 3.5.1 for a description of the TIMIT phone classification task.

Multiple NCA-ECOC models are trained and compared with the NCA model from Chapter 3 on this task. Figure 5-4 shows the performance of the NCA-ECOC model for multiple values of L, the length of the learned output codes. Results are shown for both the development and test sets. For both sets the NCA-ECOC model improves dramatically initially as the length L increases, and beyond $L = 6$, the performance of the model is better than the NCA model. The improvement is small but significant–0.4% on the development set and 0.5% on the test set.

Figure 5-5 shows the two dimensional label embedding learned for TIMIT. As

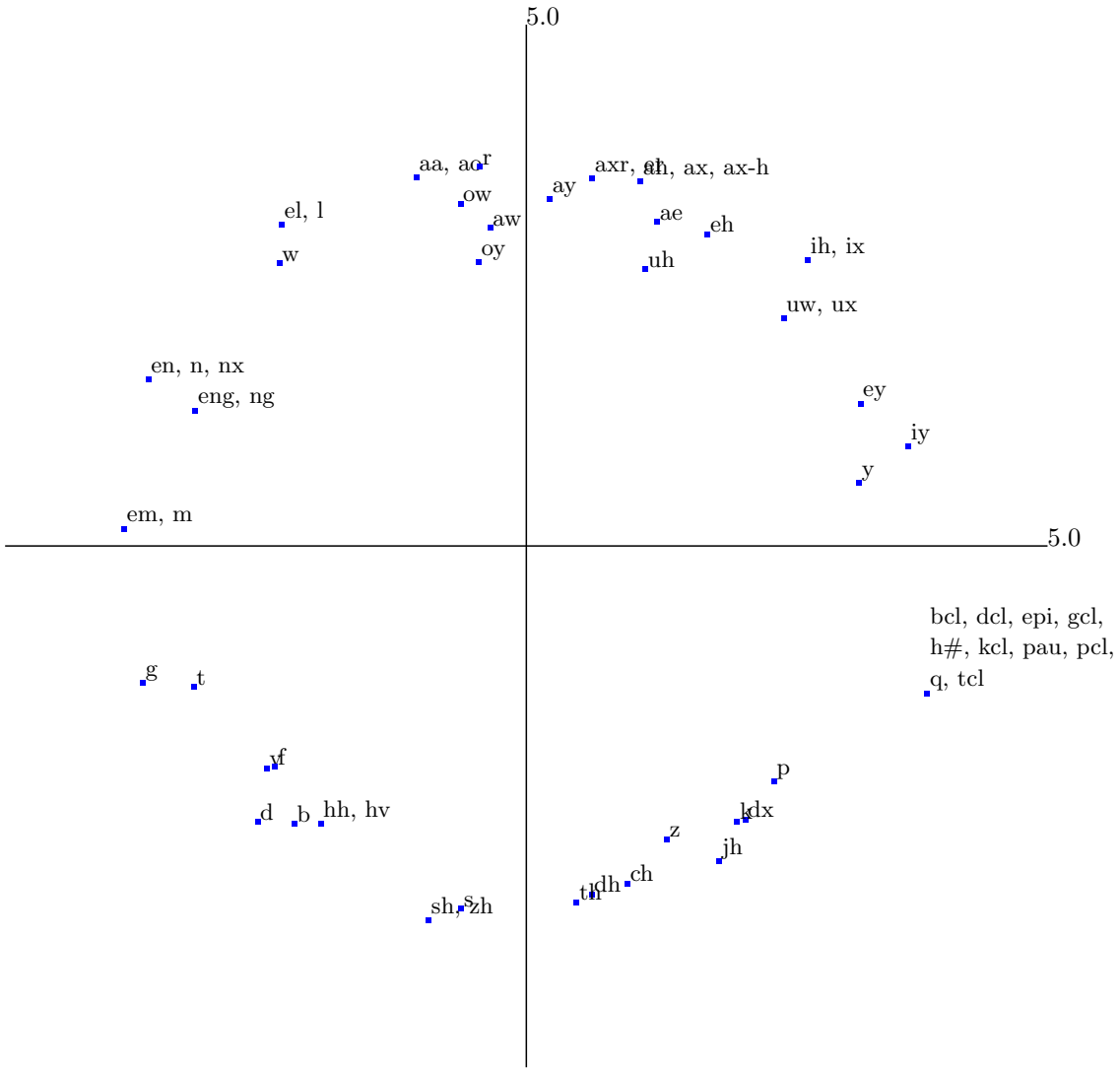

Figure 5-5: 2-D NCA-ECOC embedding of TIMIT labels.

with the lecture data many of the underlying phonetic properties of the labels are discernible from their distribution.

5.11 Related Work

This work is related to previous work on error-correcting output codes for multi-class problems. (Allwein et al. [2000], Crammer and Singer [2000], Dietterich and Bakiri [1995], Klautau et al. [2003a]) describe error-correcting output codes; more recently

(Crammer and Singer [2000, 2002], Klautau et al. [2003a], Pujol et al. [2006]) have described algorithms for learning ECOCs. Output codes are generally learned to combine the results of multiple binary classifiers to solve a multi-class classification problem. (Crammer and Singer [2000, 2002]) describe an algorithm for learning continuous valued codes (similar to the NCA-ECOC codes) for each class. This work differs from the NCA-ECOC model however as the output code for a test sample consists of the output of multiple classifiers, not a combination of the learned codes for different classes. Also the NCA-ECOC model learns the output codes within a nearest neighbor framework over the training samples.

In Section 5.2 phenomenon were described that make it difficult to segment the speech stream into separate phones and indeed to correctly label those phones. In work by (Livescu et al. [2003]) a method is presented for learning hidden feature models for each phonetic segment whereby each acoustic feature depends not only on the target phone, but directly on the corresponding acoustic feature of the surrounding segments. The authors of (Ostendorf [1999]) suggest learning similar hidden feature models for different subword units than the phoneme. Recent work (Frankel et al. [2007]) present a model for speech recognition that depends on articulatory features and not phonemes. Like many of these approaches, the NCA-ECOC model also softens the phonetic labeling decision. Unlike these approaches, however, the NCA-ECOC model makes use of information it learns about the structure of the label space and does not try to model any underlying phonetic features.

There are other approaches that make use of class-conditional estimates from multiple labels to leverage underlying structure in the acoustic label space. One such approach is proposed in work by (Hermansky et al. [2000]). In their approach, class-conditional estimates $p(y|\mathbf{x})$ from a neural network are recycled as features within the acoustic model when training a GMM. They found significant improvements using this approach. In the fMPE approach proposed by (Povey et al. [2005]), class-conditional estimates from Gaussian models are also included as features in the acoustic model. This approach also leverages cross class structure to improve the acoustic model estimates. Both these approaches differ significantly from the one presented in this

chapter because of their use of posterior estimates as features rather than using the posterior estimates to construct output codes that are employed directly in scoring the acoustic model.

5.12 Lessons

In this chapter, a model is proposed that makes use of error-correcting output codes to represent the underlying structure of the label space. This model is trained within a nearest neighbor framework similar to that used to train the NCA model. The NCA-ECOC model is shown to provide improvements to the class-conditional probability estimates for the academic lecture acoustic modeling task as well as improvements in word-error rates. Future work might investigate different definitions of similarity for the learned output codes and possible co training of the label space embedding with the input space embedding.

Chapter 6

Locally Adaptive Neighborhood Components Analysis (LA-NCA) for Acoustic Modeling

Neighborhood components analysis (NCA) Goldberger et al. [2005] is a simple but effective technique for improving the performance of nearest-neighbor classification or for class-conditional probability estimation by learning a single low-dimensional projection of the data. In this chapter a generalization of NCA called *Locally Adaptive Neighbourhood Components Analysis* (LA-NCA) is introduced. The key idea in LA-NCA is to allow each training point to have its own projection matrix or distance metric. The primary motivation for the generalization from NCA to LA-NCA is the acoustic modeling application, where the complexity of the data may warrant locally adaptive models. Experiments on the academic lecture recognition task show significant gains over both maximum-likelihood and MCE-trained Gaussian mixture models in terms of word-error-rate.

6.1 Introduction

NCA learns a low-dimensional linear projection of training and test data points, through optimization of a smooth approximation of the leave-one-out error of a

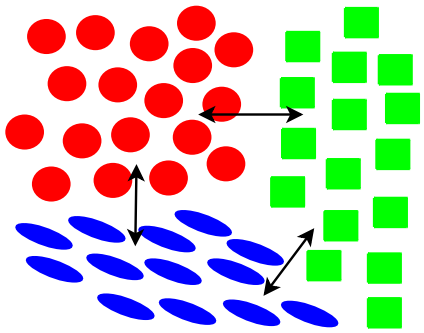

Figure 6-1: Example of three-class problem. The arrows indicate the direction most important for discrimination in their part of the space.

nearest-neighbour classifier on training examples. Equivalently, it can be viewed as a method that learns a globally-constant distance metric or covariance structure that is tailored towards nearest-neighbour classification or nearest-neighbour-based class-conditional estimation. In Chapter 4 we saw that use of projection learned by NCA can greatly improve the performance of nearest neighbor classifiers. However one of the limitations of the model is that it makes use of the same distance metric throughout the entire space, which is too simplistic for many datasets.

For many multi-class problems, in different parts of the feature space, different distance metrics or directions may be useful for discrimination amongst the classes. Consider the simple example in Figure 6-1. The three arrows indicate the directions most important for discrimination amongst the classes in different parts of the space. In other words, the distance measure should locally increase rapidly along the direction of the arrows. This ensures that points seek nearest neighbors along their side of the various boundaries. More complex data sets, such as those used for acoustic modeling, may benefit greatly by adapting the distance metric to each part of the input feature space.

In this chapter a generalization of NCA, LA-NCA, is introduced that uses locally adapted distance metrics to solve the multi-class classification and conditional estimation problems. The key idea in LA-NCA is to allow each training point to have its own projection matrix or distance metric. A leave-one-out training criterion that is closely related to the training criterion used in NCA is optimized. A parameter

estimation method for the model based on stochastic gradient updates is used to optimize the leave-one-out criterion.

The generalization from NCA to LA-NCA is relatively simple, but it leads to a radical increase in the number of parameters in the model. This raises concerns about computational efficiency and overfitting behavior. However with appropriate solutions to these problems, LA-NCA gives significant improvements in performance over NCA on the TIMIT phone classification task, and also gives impressive performance on the academic lectures recognition task.

In a first set of experiments, on the TIMIT phonetic classification problem, LA-NCA gives a 2.7% absolute reduction in error rate in comparison to NCA. A comparison to other methods on this data, including support vector machines (Clarkson and Moreno [1999]) and large-margin Gaussian mixture models (GMMs) (Sha and Saul [2007a]), shows competitive performance for LA-NCA, with only the hierachical large-margin GMMs of (Chang and Glass [2007]) giving higher accuracy on test examples.

In a second set of experiments, LA-NCA is scaled to the academic lecture recognition problem, where LA-NCA is used to provide acoustic-model probability estimates within the SUMMIT recognizer. This task is considerably larger than TIMIT, with 120 hours of training data drawn from multiple speakers. LA-NCA is compared with baseline models of maximum-likelihood trained GMMs (ML-GMMs), and GMMs trained using the minimum classification error (MCE) method developed by (McDermott and Hazen [2004], McDermott et al. [2007]) (MCE-GMMs). The findings are as follows:

- When interpolated with an ML-GMM, LA-NCA gives a 2.5% absolute (7.1% relative) reduction in word error rate (WER) over the ML-GMM (from 35.4% WER to 32.9% WER).

- When interpolated with an MCE-GMM, LA-NCA gives a 2.7% absolute (8.2% relative) reduction in WER over the MCE-GMM (from 33.1% WER to 30.4% WER).

It is particularly encouraging that the gains from LA-NCA and MCE training are additive; in combination, the two methods give a 5% absolute (14.1% relative) reduction in WER over ML-GMMs.

After introducing the LA-NCA model and training algorithm in section 6.2, and describing experiments in section 6.4, section 6.7 of this chapter discusses the relationship between LA-NCA and previous work. While LA-NCA is naturally derived as a generalization of NCA, the model is closely related to other discriminative parameter estimation methods for speech recognition, such as MMI training of mixtures of Gaussians (Valtchev et al. [1997], Woodland and Povey [2002]), or large-margin GMMs (Sha and Saul [2007a]).

6.2 The LA-NCA Model

This section describes the form of the LA-NCA model to provide estimates of $p(y|\mathbf{x})$ as well as an algorithm for learning the parameters of the model. Issues of computational efficiency and overtraining are also discussed.

6.2.1 Form of the Model

The goal of the LA-NCA model is to provide an estimate $p(y|\mathbf{x})$ of the conditional probability of a label $y \in \mathcal{Y}$ given a test input $\mathbf{x} \in \mathbb{R}^D$, where $\mathcal{Y}$ is a finite set. The method uses a training set $\{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_N, y_N)\}$ as did the NCA and NCA-ECOC models. When scoring a test point $\mathbf{x}$, each training sample j is assigned a weight $\alpha_j(\mathbf{x})$:

$$\alpha_j(\mathbf{x}) = e^{-||\mathbf{A}_j\mathbf{x}_j - \mathbf{A}_j\mathbf{x}||^2 + \beta_j} \tag{6.1}$$

This equation is similar to the definition from the NCA model (Equation 2.8), except the projection matrices $\mathbf{A}_j$ now vary with each training point and a bias parameter β_j is also introduced. Each matrix $\mathbf{A}_j$ is of dimension $d \times D$ where $d \leq D$. The value for $\alpha_j(\mathbf{x})$ depends on the distance between $\mathbf{x}$ and $\mathbf{x}_j$ under the Mahalanobis distance defined by the matrix $\mathbf{A}_j$ and decays rapidly as that distance increases. The

bias parameter β_j for each point j can be any real-valued number, and can indicate the relative importance of some training points over others. A large negative value for β_j indicates that the point is not useful for classifying its neighbors (such as an outlier point). A very large value for β_j can mean that point is very important for classification of its neighbors and might be seen at the boundaries or in relatively empty stretches of the input space where a point may have to look far to find its neighbors.

The conditional estimate for $p(y|\mathbf{x})$ for LA-NCA is then defined as follows using the definition for $\alpha_j(\mathbf{x})$ from Equation 6.1:

$$p(y|\mathbf{x}) = \frac{\sum_{j=1,y_j=y}^{N} \alpha_j(\mathbf{x})}{\sum_{j=1}^{N} \alpha_j(\mathbf{x})} \tag{6.2}$$

The numerator involves a sum over all training examples with the target label y. The denominator is a normalization constant calculated by summing over all training examples. This is the same definition as that of NCA except the α_j are calculated using the new LA-NCA definition.

NCA is a special case of the LA-NCA model, where $\beta_j = 0$ for all j, and the $\mathbf{A}_j$ matrices are constrained to be equal (i.e., for all j, $\mathbf{A}_j = \mathbf{A}$ for some matrix $\mathbf{A}$). Thus LA-NCA represents a radical increase in the number of parameters in the model, by allowing each point $\mathbf{x}_j$ to have its own projection matrix $\mathbf{A}_j$. Note that the goal of LA-NCA is to give improved estimates of $p(y|\mathbf{x})$. NCA can be interpreted as a method for improving estimates of $p(y|\mathbf{x})$, or improving the performance of a NN classifier; it can also be interpreted as a method for learning an embedding of points defined by $\mathbf{A}$ for visualization or other applications. In the latter case, a single projection matrix $\mathbf{A}$ may be preferable.

To compute a NCA estimate for a test point $O((N + D)d)$ parameters must be stored whereas for LA-NCA $O(NDd)$ parameters are stored, which is a dramatic increase. This may make the LA-NCA model prohibitively memory intensive to employ for tasks such as acoustic modeling in this naive form. Some methods for reducing the size of the model will be investigated in this chapter, but future research

may further investigate other ways to reduce the number of parameters used in the model.

6.2.2 The Training Criterion

As in NCA, it is natural to derive a leave-one-out criterion for training the LA-NCA model. For each training point $\mathbf{x}_i$, the leave-one-out estimate $p_{lanca}(y|i)$ is defined as follows:

$$\alpha_j(i) = e^{-||\mathbf{A}_j\mathbf{x}_j - \mathbf{A}_j\mathbf{x}_i||^2 + \beta_j} \text{ if } i \neq j,\ 0 \text{ otherwise;} \tag{6.3}$$

$$p_{lanca}(y|i) = \frac{\sum_{j=1, y_j=y}^{N} \alpha_j(i)}{\sum_{j=1}^{N} \alpha_j(i)} \tag{6.4}$$

Note that $\alpha_i(i) = 0$ indicates that a point can not be used to label itself. For convenience we will use $\Theta = \{(\mathbf{A}_i, \beta_i)\}_{i=1}^{N}$ to denote the full set of parameters in the model. Following Globerson and Roweis [2006], the training objective is the conditional log-likelihood of the training examples,

$$f(\Theta) = \sum_{i=1}^{N} \log p_{lanca}(y_i|i) \tag{6.5}$$

In the next section an algorithm based on stochastic gradient descent (SGD) is described for maximizing this criterion. Given the very large number of parameters in the model, overfitting is a clear concern; this issue is discussed in detail in section 6.2.4.

6.2.3 The Training Algorithm

This section describes an SGD algorithm for the optimization of the LA-NCA model. Differentiating $f(\Theta)$ gives the following partial gradients with respect to the individual projection matrices $\mathbf{A}_z$ and the bias parameters β_z. Note that $\mathbf{x}_{ij} = \mathbf{x}_i - \mathbf{x}_j$, and

$\delta(a, b) = 1$ if $a = b$, and 0 otherwise:

$$\frac{\partial f(\Theta)}{\partial \mathbf{A}_z} = \sum_{i=1, i\neq z}^{N} 2\mathbf{A}_z \mathbf{x}_{iz} \mathbf{x}_{iz}^T \left(\frac{\alpha_z(i)}{\sum_{j=1, j\neq i}^{N} \alpha_j(i)} \right) \left(1 - \frac{\delta(y_z, y_i)}{p(y_i|i)} \right) \tag{6.6}$$

$$\frac{\partial f(\Theta)}{\partial \beta_z} = \sum_{i=1, i\neq z}^{N} \left(\frac{\alpha_z(i)}{\sum_{j=1, j\neq i}^{N} \alpha_j(i)} \right) \left(\frac{\delta(y_z, y_i)}{p(y_i|i)} - 1 \right) \tag{6.7}$$

Figure 6-2 gives an SGD algorithm LeCun et al. [1998] based on these gradients. The stochastic algorithm is chosen here because it can converge with relatively few passes over the training data, which is important given the large size of the training set and relatively slow running time of the gradient computation. The algorithm has some important characteristics, as follows:

Sub-sampling The algorithm is defined for a slightly more general setting than that given in Equations 6.1 and 6.2, where a subset of the training examples, $S \subseteq \{1 \ldots N\}$, is used to define the model. The estimate for the conditional likelihood of a training sample under this scheme is redefined as follows:

$$p_{lanca}(y|\mathbf{x}) = \frac{\sum_{j\in S, y_j=y} \alpha_j(\mathbf{x})}{\sum_{j\in S} \alpha_j(\mathbf{x})} \tag{6.8}$$

The training estimates $p(y|i)$ and optimization criterion $f(\Theta)$ also change similarly.

$$p_{lanca}(y|i) = \frac{\sum_{j\in S, y_j=y} \alpha_j(i)}{\sum_{j\in S} \alpha_j(i)} \tag{6.9}$$

$$\begin{aligned} f(\Theta) &= \sum_{i=1}^{N} \log p(y_i|i) \\ &= \sum_{i=1}^{N} \log \frac{\sum_{j\in S, j\neq i, y_j=y_i} \alpha_j(i)}{\sum_{j\in S, j\neq i} \alpha_j(i)} \end{aligned} \tag{6.10}$$

Sub-sampling effectively ignores $\alpha_j(\mathbf{x})$ for any j that is not a member of S. This means that a subset of the training set S serves as the the *support points*, or neighbors used to label training and test samples. The original model is equivalent to

Inputs: Training samples (in randomized order), $\{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_N, y_N)\}$.
Learning rate parameter, η.
Number of iterations L.
Subset of training examples $S \subseteq \{1, ..., N\}$.
Truncation level m (where $1 \leq m \leq |S|$).

Algorithm:
Initialize $\mathbf{A}_j$ and β_j for all $j \in S$
$t \leftarrow 1$
for $iter = 1$ to L
  for $i = 1$ to N

1. $\alpha_j(i) \leftarrow \exp\{-||\mathbf{A}_j x_i - \mathbf{A}_j x_j||^2 + \beta_j\}$ for all $j \in S$
2. Set $S' \subseteq S$ to be the top m members of $S \setminus \{i\}$ with the highest values of $\alpha_j(i)$
3. $d_i \leftarrow \sum_{j \in S'} \alpha_j(i)$
4. $p(y_i|i) \leftarrow \frac{1}{d_i} \sum_{j \in S', y_j = y_i} \alpha_j(i)$
5. for $j \in S'$

$$\mathbf{x}_{ij} \leftarrow \mathbf{x}_i - \mathbf{x}_j$$
$$\mathbf{A}_j \leftarrow \mathbf{A}_j + \left(\frac{\eta}{1+t/N}\right) 2\mathbf{A}_j \mathbf{x}_{ij} \mathbf{x}_{ij}^T \left(\frac{\alpha_j(i)}{d_i}\right) \left(1 - \frac{\delta(y_j, y_i)}{p(y_i|i)}\right)$$
$$\beta_j \leftarrow \beta_j + \left(\frac{\eta}{1+t/N}\right) \left(\frac{\alpha_j(i)}{d_i}\right) \left(\frac{\delta(y_j, y_i)}{p(y_i|i)} - 1\right)$$

6. $t \leftarrow t + 1$

Output: β_j and $\mathbf{A}_j$ for all $j \in S$

Figure 6-2: The stochastic gradient algorithm used for parameter estimation of the LA-NCA model.

setting $S = \{1 \ldots N\}$. Note that while only a subset of the points are used to label the test and training points, the entire training set is still used to estimate the parameters associated with these support points. By sub-sampling the training set for a reduced set of support points the number of parameters that must be learned can be reduced as well. Intuitively sub-sampling should work well under conditions where the structure of the input space around multiple points in a neighborhood can be adequately modelled using a single support point with the associated projection and bias parameter able to compensate for the loss of nearby points.

In the experiments various scenarios are described, including the case where $S = \{1 \ldots N\}$, and also the case where S is a randomly sampled subset of the full training set. Defining S to be a subset of the training set increases the computational and space efficiency of training and testing and can also potentially reduce overfitting problems.

Truncation Motivated by computational concerns, a parameter m that governs the level of *truncation* is introduced in the model. In step 2 of the algorithm only a subset S' of the set S is retained, where $|S'| = m$. The set S' contains the m highest scoring members of S (sorted by $\alpha_j(i)$) for the training example i under consideration. Steps 3, 4 and 5 of the algorithm then use the set S' to approximate S, reducing each of these steps from $O(|S|)$ time to $O(m)$ time. This approximation is justified if the values for $\alpha_j(i)$ decrease relatively rapidly, which is often the case in practice. Note that this approximation only gives a constant factor speed-up, as step 1 takes $O(|S|kd)$ time, and steps 3, 4, and 5 combined take $O(|S'|kd)$ time. Nevertheless, this approximation leads to useful speed-ups in practice.

Gradient steps Each gradient step (step 5 of the algorithm) involves a pair of training examples, $i \in \{1 \ldots N\}$, and $j \in S'$. The point j is the sample which is being used to help determine the estimate of $p(y|i)$. Step 5 computes a "stochastic" approximation to the true gradient in the usual manner by extracting from Equations 6.6 and 6.7 the terms for $\mathbf{A}_j$ and β_j pertaining to the ith example. Following

previous work that uses SGD (e.g., see LeCun et al. [1998]), the learning rate decays slowly with the number of training examples t that have been visited over all iterations, with η being a constant that gives the initial learning rate.

$$\text{learning rate} = \frac{\eta}{1 + t/N}$$

Initialization In all the experiments described in this chapter, a random initialization for the matrices $\mathbf{A}_j$ is used. Each of the $(d \times D)$ components of each matrix $\mathbf{A}_j$ is drawn at random from the uniform distribution over $[-\sigma, +\sigma]$, where σ is selected by validation on development data (in the experiments here $\sigma \approx 0.05$). The β_j parameters are initialized to 0.

Runtime Each inner loop of the algorithm (steps 1 to 6) takes $O(|S|kd)$ time; it follows that each pass over the training data takes $O(N|S|kd)$ time. As in previous work on SGD methods, we have found that the model converges to a good solution in a relatively small number of iterations.

The optimization of $f(\Theta)$ is non-convex and it is therefore possible to get stuck in local optima during optimization. However, in practice good results were achieved despite this problem and multiple random initializations of the parameters resulted in similar performance on the TIMIT classification task.

6.2.4 Overfitting Issues

The proposed LA-NCA model has a very large number of parameters, and overfitting is a concern. To give an extreme example, consider the case where $S = \{1 \ldots N\}$. It is then possible to define a degenerate solution to the problem of maximizing $f(\Theta)$, where $p(y_i|i) = 1$ for all $i = 1 \ldots N$, but where the model will clearly not generalize to new examples. Define a function $g : \{1 \ldots N\} \rightarrow \{1 \ldots N\}$ such that $g(i) \neq i$ for all i, the mapping is one-to-one (i.e., $g(i) \neq g(j)$ for $i \neq j$), and for all i, $y_i = y_{g(i)}$. Thus the function g assigns a "target" point $g(i)$ to each point i, under the constraint that point $g(i)$ has the same label as i. It is always possible to find such a mapping, provided that

each label $y \in \mathcal{Y}$ has at least two data points in the training set. Then choose $\mathbf{A}_{g(i)}$ such that $||\mathbf{A}_{g(i)}x_{g(i)} - \mathbf{A}_{g(i)}x_i||^2 \ll ||\mathbf{A}_{g(i)}x_{g(i)} - \mathbf{A}_{g(i)}x_j||^2$ for all $j \neq i$, $j \neq g(i)$. In other words, $g(i)$ will only be used to label training point i and lend very little weight to the labeling of all other training points. This is always possible if the training points are in general position (i.e. no three points i, $g(i)$ and j are on the same line). As we scale the magnitude of $\mathbf{A}_{g(i)}$, then $||\mathbf{A}_{g(i)}x_{g(i)} - \mathbf{A}_{g(i)}x_i||^2 - ||\mathbf{A}_{g(i)}x_{g(i)} - \mathbf{A}_{g(i)}x_j||^2 \rightarrow -\infty$ for all i and $j \neq g(i)$, and it can be verified that $p(y_i|i) \rightarrow 1$ for all i.

Two main strategies are used to avoid overfitting. The first is to choose the set S in the algorithm in Figure 6-2 to be a strict subset of the full training set. While eliminating the degenerate solution discussed above, this method may still be prone to overfitting. The second, and more important, strategy is to use early stopping in the SGD algorithm (i.e., a small value for L), which was found to be effective in our experiments. Typically just a few iterations over the training set with carefully chosen values for σ and η. Future work may consider methods that regularize or constrain the projection matrices $\mathbf{A}_j$, as an alternative to early stopping.

There are several possible ways to regularize the model. One way would be to try to limit the magnitude of the parameters in the projection matrices. This would help avoid the degenerate solution suggested above. ($a_{i,r,s}$ denotes the element of the rth row and sth column of the matrix $\mathbf{A}_i$.)

$$f_{reg1}(\Theta) = f(\Theta) - k_1 \sum_{i=1}^{N} \sum_{r=1}^{d} \sum_{s=1}^{D} a_{i,r,s}^2$$

Another possibility is to encourage each training point i to draw its estimate of $p(y|i)$ from several points. In the degenerate case, each conditional probability is effectively estimated by a single point. To encourage more neighbors to contribute to the labeling of point i, the entropy of distribution over p_{ij} can be added to the optimization criterion.

$$f_{reg2}(\Theta) = f(\Theta) + k_2 \sum_{i=1}^{N} \sum_{j=1, j\neq i}^{N} p_{ij} \log \frac{1}{p_{ij}}$$

$$p_{ij} = \frac{\alpha_j(i)}{\sum_{k=1,k\neq i}^{N} \alpha_k(i)}, \quad p_{ii} = 0$$

While both these approaches can help combat overfitting, in experiments on TIMIT early stopping was found to be just as effective, without having to tune regularization parameters. Early stopping also implies less computation time training the parameters. The strategy of sub-sampling also implies improved computational efficiency which the regularized criteria do not. Therefore in the experiments reported in this chapter, sub-sampling and early stopping are the main methods used to limit overfitting effects.

6.3 An Illustration of the Approach

To illustrate the LA-NCA approach consider a simple example, where the task is to discriminative between three classes from the TIMIT data set (see section 3.5.1), `/s/`, `/z/`, and `/sh/+/zh/`. First an NCA model is trained with a projection down to $d = 2$ dimensions to allow us to visualize the data. In a second experiment, the 2-dimensional representation produced by NCA is used to train LA-NCA on the same data set with $d = 2$. Hence the LA-NCA model learns full rank 2×2 dimensional projection matrices for the data. Note that training LA-NCA on the original data, with $d = 2$, would perhaps be a preferable model, but it would not be possible to view the result in 2 dimensions. Each training point $\mathbf{x}_j$ has its own distance metric defined by $\mathbf{A}_j$ under the LA-NCA model.

Figure 6-3 shows the data under the NCA and LA-NCA representations. The distance metric for each point $\mathbf{x}_j$ is represented by an ellipse where each point on the ellipse is equidistant from $\mathbf{x}_j$. NCA leads to circular contours of this form since in the projected space the distance metric at each point can be represented by the identity matrix. The LA-NCA model leads to ellipses. Note also that outlier points generally have smaller ellipses, indicating that they contribute less to the labelling of the surrounding points. The classification performance of NCA on this small task

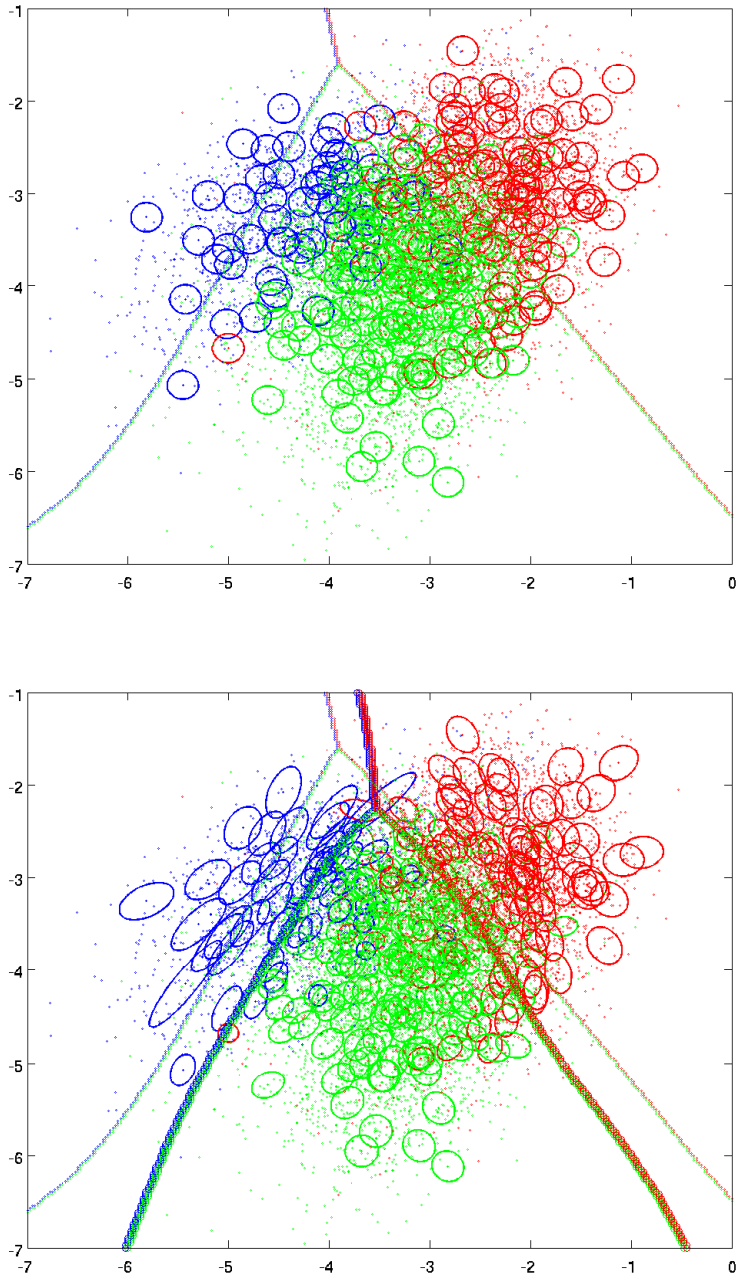

Figure 6-3: Results for a three class problem from TIMIT using NCA (top figure) and LA-NCA (bottom figure). Ellipses or circles centered on data points allow the covariance structure $\mathbf{A}_j$ to be visualized for a random sample of the points (about 2%). The top plot shows the resulting decision boundary under NCA; the bottom plot shows decision boundaries from both methods (LA-NCA in bold).

is 72.4% while the LA-NCA model is able to improve performance to 74.1%. Figure 6-3 also indicates the decision boundaries determined by each model. The decision boundaries are quite different, in particular the boundary between the green and blue classes. The density of the blue points is less than the density of the green points. The LA-NCA model compensates for this by learning distances metrics for the blue points that imply larger ellipses or larger weights when labeling a test sample than the green points. This is an example of how the freedom of each point to select its own distance metric allows the model to better estimate decision boundaries.

6.4 Experiments

Experiments were conducted on two different acoustic modeling tasks. The first is the TIMIT phone classification task, where the model is used strictly to perform multiclass classification. The second is a large-vocabulary speech recognition task. Here the model is used to provide conditional probability scores $p(y|\mathbf{x})$ that are used within the SUMMIT recognizer.

6.4.1 TIMIT Phone Classification

The TIMIT phone classification task (Lamel et al. [1986]) is a popular task for which many results have been reported and is described in detail in Section 3.5.1. A summary of previously reported results on this task is given in Table 6.1. Note that the approaches in this table use a variety of different feature representations as input to the learning algorithms.

For our experiments we use the 61-dimensional segmental feature measurements described in (Chang and Glass [2007], Halberstadt and Glass [1997]), referred to as the "S4" features. The S4 feature set often gives improved performance on the TIMIT task over other representations (e.g., for SVMs it results in a reduction in error from 22.4% to 19.5%; for large-margin GMMs it gives a reduction in error from 21.1% to 19.6%).

Table 6.2 gives results for several models using this representation. **Model 1** is

Model	Error (Test)
SVM (Clarkson and Moreno [1999])	22.4%
Hidden CRF (Gunawardana et al. [2005])	21.7%
Large Margin GMM (Sha and Saul [2007a])	21.1%
Hierarchical GMM (Halberstadt and Glass [1997])	21.0%
RLS2 (Rifkin et al. [2007])	20.9%
SVM w/ hybrid features (Yousafzai et al. [2010])	19.0%
Hierarchical LM GMMs (Chang and Glass [2007])	18.7%
Hierarchical LM GMMs w/committee classifiers (Chang and Glass [2007])	16.7%

Table 6.1: Previously reported results on the TIMIT phonetic classification task.

Model	Error (Dev)	Error (Test)
(1) NCA, w/knn classifier k=15	21.1 %	21.9%
(2) One projection per class, $\beta = 0$	19.7%	21.1%
Large Margin GMMs (Chang and Glass [2007])		19.6%
(3) Multi-class SVM	17.6%	19.5%
(4) One projection per point, $\beta = 0$	**17.4%**	19.2%
(5) One projection per point	17.7%	19.0%
Hierarchical Large Margin GMMs (Chang and Glass [2007])	18.1%	**18.7%**

Table 6.2: Results on the TIMIT phonetic classification task. The table shows systems which all make use of the same S4 feature representation, from (Chang and Glass [2007], Halberstadt and Glass [1997]).

an NCA model, combined with a k-nearest neighbour classifier; the value for k was set to 15 by validation on the development data.

Model 2 is a version of LA-NCA where each class $y \in \mathcal{Y}$ has its own projection matrix A_y; thus this model has many fewer parameters than the full LA-NCA model. Thirty-nine total projection matrices are learned with $d = 50$. This model tests whether using the same distance metric or covariance structure for each point within a class provides enough local adaption of the distance metrics to perform well on this task.

Model 3 is a reimplementation (using LIBSVM Chang and Lin [2001]) of the SVM approach described in (Clarkson and Moreno [1999]) using RBF kernels, where

the kernel width and the slack constant C are optimized over the development set.

Model 4 is a version of the LA-NCA model with $\beta_j = 0$ for all j (hence no bias parameters are learned), whereas **Model 5** is the full LA-NCA model with bias parameters.

In training Models 2, 4, and 5, the set $S = \{1 \dots N\}$, and the truncation level $m = 20,000$. The learning rate η was chosen by validation on the development set and was equal to 0.1. The $\mathbf{A}_j$ matrices had dimension 20×61 for Models 4 and 5, and dimension 50×61 for Model 2. In these models truncation was also applied during *testing*, with the truncation value $m = 50$. Hence for each test point $\mathbf{x}$, only the top 50 highest scoring training examples under the criterion $\alpha_j(\mathbf{x})$ were used to contribute to the estimate of $p(y|\mathbf{x})$ (we found this gave a slight improvement over using the full training set). The value $m = 50$ was again chosen using validation on the development set. Models 2, 4, and 5 were each trained using the SGD algorithm where development data was used to find the optimal value for L. Typically the value chosen for L was between five and ten.

Models 4 and 5 are competitive, with only hierarchical large-margin GMMs (Chang and Glass [2007]) giving lower error rates on test data. These models perform similarly to one another indicating that perhaps the bias parameters are not necessary when using the entire set as support points for this task. Models 4 and 5 give an almost 3% absolute improvement in error over the NCA baseline, demonstrating the value of allowing the parameters $\mathbf{A}_j$ to vary with the training example j. The restriction in Model 2 to having a single projection matrix for each class significantly degrades performance, with roughly 2% higher absolute error than Models 4 and 5, and a small improvement over NCA.

The LA-NCA model makes some improvements over the multiclass SVM model, however, the SVM model has many fewer parameters. This suggests that intelligent sub-sampling of support points may allow the LA-NCA model to still perform well while reducing the number of parameters. Also the SVM model more directly optimizes classification performance whereas the LA-NCA model is used primarily to optimize conditional log-likelihood.

Number of support points	Error Rate (development set)	Error Rate (test set)
1000	20.6	21.9
10000	19.3	20.3
50000	18.2	19.6
140225 (full training set)	17.7	19.0

Table 6.3: Performance of LA-NCA on TIMIT phonetic classification task for various values of the number of training points subselected for learning distance metrics.

Experiments with Subselection

In Table 6.3, the performance of an LA-NCA classifier for multiple values of m, the number of points sub-selected as support points is reported. The sub-selected points are chosen randomly. As the number of points increases, the performance of the classifier also improves on both the training and test sets. This suggests that further increases in the size of the training set would allow the LA-NCA method to still improve performance. It is also interesting that the system performs reasonably well, even with only 1000 randomly selected support points.

6.4.2 Lecture Recognition

Results are now reported on a large-vocabulary speech recognition task. The experiments on TIMIT were useful when developing the LA-NCA model, and are useful for comparison to previous approaches, but the TIMIT data has limitations: it involves phone classification rather than full recognition, and the size of the TIMIT data sets is small in comparison to the data sets now typically used in full recognition experiments. The lecture data used consists of 120 hours of training data (11.5 million samples, 80 times more samples than TIMIT) from multiple speakers, and a vocabulary size of over 37,000. The SUMMIT system (Glass [2003]) is used for recognition. We compare to both maximum-likelihood and discriminatively-trained baselines. The academic lecture task is described in detail in Section 3.5.2. While not quite at the scale of some speech recognition tasks (e.g., (Kim et al. [2005]) describe experiments using 1,350 hours of training data), the lecture recognition task is nevertheless of a significant size and complexity.

The SGD algorithm in Figure 6-2 is used to train the model. A subset S of size $|S| \approx 140,000$ (1.2% of all training points) is selected. These points were chosen at random, under the constraint that each of the 1871 classes should receive at least 10 points. The truncation size m was set to be $20,000$ for training and testing, and $L = 3$ iterations of training were performed. The matrices $\mathbf{A}_j$ were of dimension 20×112. The relatively small number of support points and training iterations were selected because of overfitting as well as computational efficiency concerns.

The following models are then designed for use in full recognition experiments with the SUMMIT system. Results for the models are shown in Table 6.4.

ML-GMMs. The baseline maximum-likelihood (ML) Gaussian mixture model (GMM) is a commonly-employed baseline in acoustic modeling. The acoustic vectors were first projected using a 50-dimensional NCA projection (results from Chapter 4 show that this method is competitive with HLDA (Kumar [1997]), a commonly used method for dimensionality reduction in speech recognition), followed by a class-based PCA projection (this has been found to improve performance, especially when using diagonal covariance GMMs).

MCE-GMMs. MCE is a discriminative training method for GMMs (McDermott et al. [2007]). It has been shown in previous work to give significant improvements in recognition performance over ML training on the lecture recognition task, and in other settings. The MCE model we use was originally developed (McDermott and Hazen [2004], McDermott et al. [2007]). The input representation used in the experiments with MCE consisted of 50 dimensional vectors produced by class-based PCA and the GMM model makes use of diagonal covariance matrices as well.

Model A: Interpolation of LA-NCA and ML-GMMs. The LA-NCA model produces an estimate $p_{lanca}(y|\mathbf{x})$ for any phone label y for any acoustic input $\mathbf{x}$. We can also use a maximum-likelihood trained GMM to derive the following conditional

Acoustic Model	WER (Development Set)	WER (Test Set)
Baseline ML-GMM	36.3	35.4
Model A: LA-NCA + ML-GMM	33.8	32.9
Baseline MCE-GMM	33.3	33.1
Model B: LA-NCA + MCE-GMM	31.2	30.4
Model C: LA-NCA	36.2	35.0

Table 6.4: WER of recognizer for different acoustic models on the lecture data.

estimate.

$$p_{gmm}(y|\mathbf{x}) = \frac{p_{gmm}(\mathbf{x}|y)p(y)}{\sum_{y' \in \mathcal{Y}} p_{gmm}(\mathbf{x}|y')p(y')}$$

As described in Chapter 5, the recognizer requires a generative model score $p(\mathbf{x}|y)$, as opposed to a discriminative model score of $p(y|\mathbf{x})$. The following value is therefore employed within the recognizer:

$$\beta_1 \log \left(\frac{\gamma_1 p_{gmm}(y|\mathbf{x}) + (1-\gamma_1) p_{lanca}(y|\mathbf{x})}{p(y)} \right) \tag{6.11}$$

The parameter $0 \leq \gamma_1 \leq 1$ gives relative weights for the GMM and LA-NCA models and is optimized for WER on the development set. β_1 is the acoustic model scaling parameter and is also tuned to optimize WER on the development set. The numerator gives an interpolated estimate of $p(y|\mathbf{x})$. Each $p(y)$ is proportional to the number of times y is seen in training. The term $p(y)$ is motivated by Bayes rule ($p(\mathbf{x})$ is constant w.r.t. y, and can be ignored during decoding). This type of acoustic model was previously applied in Chapter 4.

Model B: Interpolation of LA-NCA and MCE-HMMs. The MCE-trained GMM model also produces estimates $p_{mce}(\mathbf{x}|y)$ which can be interpolated with $p_{lanca}(y|\mathbf{x})$ estimates in the same way as in Model A, replacing p_{gmm} with p_{mce}.

$$p_{mce}(y|\mathbf{x}) = \frac{p_{mce}(\mathbf{x}|y)p(y)}{\sum_{y' \in \mathcal{Y}} p_{mce}(\mathbf{x}|y')p(y')}$$

$$\beta_2 \log \left(\frac{\gamma_2 p_{mce}(y|\mathbf{x}) + (1-\gamma_2) p_{lanca}(y|\mathbf{x})}{p(y)} \right) \tag{6.12}$$

γ_2 is restricted to be in $[0, 1]$ and together with β_2 is optimized as in Model A.

Model C: LA-NCA alone. In this model the LA-NCA model is used without interpolation with a GMM model. The model scores used within the HMM are defined as

$$\beta_3 \log \left(\frac{p_{lanca}(y|\mathbf{x})}{p(y)} \right) \tag{6.13}$$

6.4.3 Results

The results in Table 6.4 show that Model A gives a 2.5% absolute decrease in WER in comparison to the ML-GMM. This is a large improvement, slightly bigger than the gains seen from MCE training. Moreover, Model B gives a 2.7% absolute gain over the MCE trained GMM. MCE and LA-NCA combined give additive gains in performance, resulting in an absolute gain of 5% in WER over the ML-GMM. Finally, Model C performs only slightly better than the ML-GMM, showing that interpolation with either an ML-GMM or an MCE-GMM is important for performance. This may be because Model C is undersmoothed; future work may investigate this further.

Perplexity for the LA-NCA and ML-GMM models is also reported in Table 6.5. (Noter that perplexity is defined as $e^{-\kappa}$ where κ is the average of the log-likelihoods $\log p(y|\mathbf{x})$ on held out samples). ML-GMM has a perplexity of 9.96; LA-NCA has a perplexity of 8.89; an interpolation of the two models has a perplexity of 5.59. This is a large improvement in the perplexity of the acoustic model, more dramatic than the improvement in WER. More investigation is needed to determine exactly why such a large drop in perplexity does not lead to a bigger improvement in WER. However, the word-error rate depends on many other factors such as language and pronunciation models. It is possible the WER would still be significant even with the very low perplexity. NCA gives a perplexity value of 14.25 on this data, significantly worse than both ML-GMM and LA-NCA.

Model	Average CLL on held-out data	Perplexity
p_{gmm}	-2.299	9.96
p_{lanca}	-2.185	8.89
p_A	-1.721	5.59

Table 6.5: Average conditional log-likelihood (CLL) of p_{gmm} (ML GMM), p_{lanca}, and p_A (Model A) on DevSet1. The corresponding perplexity values are indicated as well where the perplexity is defined as $e^{-\kappa}$ given that κ is the average CLL.

6.5 Combining LA-NCA and NCA-ECOC Models

This section describes experiments on the academic lecture task that combine LA-NCA and NCA-ECOC approaches. The main idea is to replace the NCA estimate of $p(y|\mathbf{x})$ in Equation 5.7 with an LA-NCA estimate. The representative output code of a test sample then becomes

$$H_{ecoc'}(\mathbf{x};\mathbf{M},\Theta) = \sum_{y\in\mathcal{Y}} p_{lanca}(y|\mathbf{x};\Theta)\mathbf{M}_y \tag{6.14}$$

and the estimate of $p(y|\mathbf{x})$ of the combined model is the following.

$$p_{ecoc'}(y|\mathbf{x};\mathbf{M},\Theta) = \frac{e^{\langle \mathbf{M}_y, H_{ecoc'}(\mathbf{x};\mathbf{M},\Theta)\rangle}}{\sum_{y'\in\mathcal{Y}} e^{\langle \mathbf{M}_{y'}, H_{ecoc'}(\mathbf{x};\mathbf{M},\Theta)\rangle}} \tag{6.15}$$

The training estimates of $p(y|i)$ are similarly altered to use the LA-NCA estimates and the model is trained as in Chapter 5. Also the model can still be trained efficiently if all the estimates of $p_{lanca}(y|i)$ can be pre-computed and stored.

The estimates of $p_{lanca}(y|\mathbf{x})$ are interpolated with LA-NCA estimates $p_{ecoc'}(y|\mathbf{x})$ to create a model called p_{total}:

$$p_{total}(y|\mathbf{x}) = \lambda_{total} p_{lanca}(y|\mathbf{x}) + (1-\lambda_{total}) p_{ecoc'}(y|\mathbf{x}) \tag{6.16}$$

λ_{total} lies in $[0,1]$ and determines the mixing proportion between the the two estimates. λ_{total} is selected to optimize perplexity on held-out samples and is set to 0.65. The perplexity of the LA-NCA model is 8.89 and the perplexity achieved by the interpolated model p_{total} is 8.40.

Acoustic Model	WER (Dev)	WER (Test)
Model B: LA-NCA + MCE-GMM	31.2	30.4
Model C: LA-NCA	36.2	35.0
Model D: LA-NCA + NCA-ECOC + MCE-GMM	30.9	30.2
Model E: LA-NCA + NCA-ECOC	35.7	34.8

Table 6.6: WER of recognizer for different acoustic models on the lecture data.

Experiments are conducted on the academic lecture set using the interpolated model p_{total} and results are reported in Table 6.6. The results for Model B and Model C are repeated here for comparison. **Model D** is an interpolated model combining p_{total} with p_{mce}.

$$\beta_4 \log \left(\frac{\gamma_4 p_{mce}(y|\mathbf{x}) + (1-\gamma_4) p_{total}(y|\mathbf{x})}{p(y)} \right) \tag{6.17}$$

γ_4 is restricted to be in $[0, 1]$ and together with β_4 is optimized as in Model A.

Model E makes use of only the p_{total} estimates and is not interpolated with a GMM model. The scores of Model E are defined as follows:

$$\beta_5 \log \left(\frac{p_{total}(y|\mathbf{x})}{p(y)} \right) \tag{6.18}$$

The gains provided by combining the NCA-ECOC model with LA-NCA model are modest, with 0.2 percent reduction in WER from Model B to Model D on the test set. A 0.2 percent reduction in WER frm Model C to Model E on the test set is also seen. These results suggest that NCA-ECOC model as it is defined provide just a small benefit over the LA-NCA model. Future work may consider alternate methods of combining the output code information with the LA-NCA model.

6.6 Experiments with MNIST Digits

While the primary motivation for the methods developed in this thesis have been acoustic modeling, the method can also be applied to other tasks. Results are presented here on the MNIST handwritten digit classification task described in Sec-

NCA, $d = 2$, $k = 200$	27.9
NCA, $d = 3$, $k = 200$	17.1
NCA, $d = 5$, $k = 25$	9.1
NCA, $d = 10$, $k = 5$	4.3
NCA, $d = 20$, $k = 8$	2.6
LA-NCA, $d = 2$, $k = 9$	2.2
LA-NCA, $d = 3$, $k = 6$	2.2
LA-NCA, $d = 5$, $k = 4$	1.5
LA-NCA, $d = 10$, $k = 5$	1.8
LA-NCA, $d = 20$, $k = 6$	1.7

Table 6.7: Error rate of a NCA and LA-NCA classifiers on MNIST digits, trained with 60,000 samples and tested with 10,000 samples. The value d indicates the number of rows in the learned projection matrices. The value k indicates the number of nearest neighbors used to perform the classification and was selected on held-out data.

tion 4.4 using the LA-NCA model. In Table 6.7 the performance of various NCA and LA-NCA models are listed for d dimensional projections and k nearest neighbors used to perform the classifcation. The LA-NCA model using two dimensional projections outperforms the NCA model using twenty dimensional projections. Additional gains are seen as the value for d increases with the LA-NCA models but degrades after $d = 5$. Each of the LA-NCA models does much better than the NCA model with the same size projection. The two-dimensional LA-NCA model achieves 2.2% error whereas as the two-dimensional NCA model achieves 27.7% error, dramatically highlighting the importance of modeling localized structure. Performance of the LA-NCA models on held-out training samples for various values of k are depicted in Figure 6-4. As with the NCA model, the accuracy can vary with the value of k. The value of k used to classify test samples is chosen on these held-out samples.

In Figure 6-5, one randomly selected sample of each of the ten digits is depicted, with the original image pictured in the last column. One randomly selected row of the projection matrix $\mathbf{A}_j$ associated with each point is shown as an image, similar to Figure 4-5. The random initialization is shown in the first column; the second column shows the learned row for LA-NCA with $d = 2$; the third column shows the difference between columns one and two. The classification performance of the initialized repre-

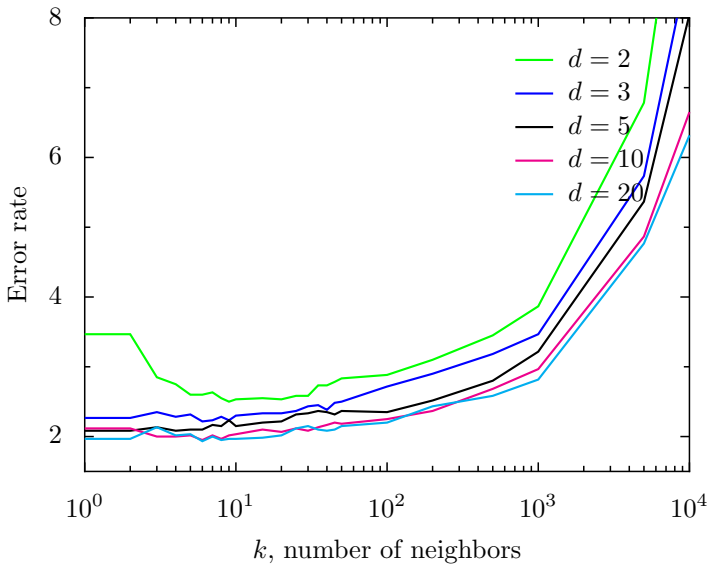

Figure 6-4: Classification accuracy of LA-NCA model on held-out MNIST training samples set for various values of d and k. The value d indicates the number of rows in the learned projection matrices. The value k indicates the number of nearest neighbors used to perform the classification.

sentation is 4.4%; the learned representation achieves 2.2% error; the original images achieve 2.8% error using a kNN classifier. It is interesting that both the initial and learned representations look fairly random. However the difference between the two clearly show a structure similar to the original image with a preference to differentiate with other possible confusable images.

6.7 Related Work

Now the relationship between LA-NCA and models other than NCA is discussed.

Discriminatively-trained GMMs are widely used for speech recognition (Cheng et al. [2009], Kim et al. [2005], McDermott and Hazen [2004], McDermott et al. [2007], Sha and Saul [2007a], Soltau et al. [2005], Valtchev et al. [1997], Woodland and Povey [2000, 2002]). A discriminative GMM can be used to provide an estimate

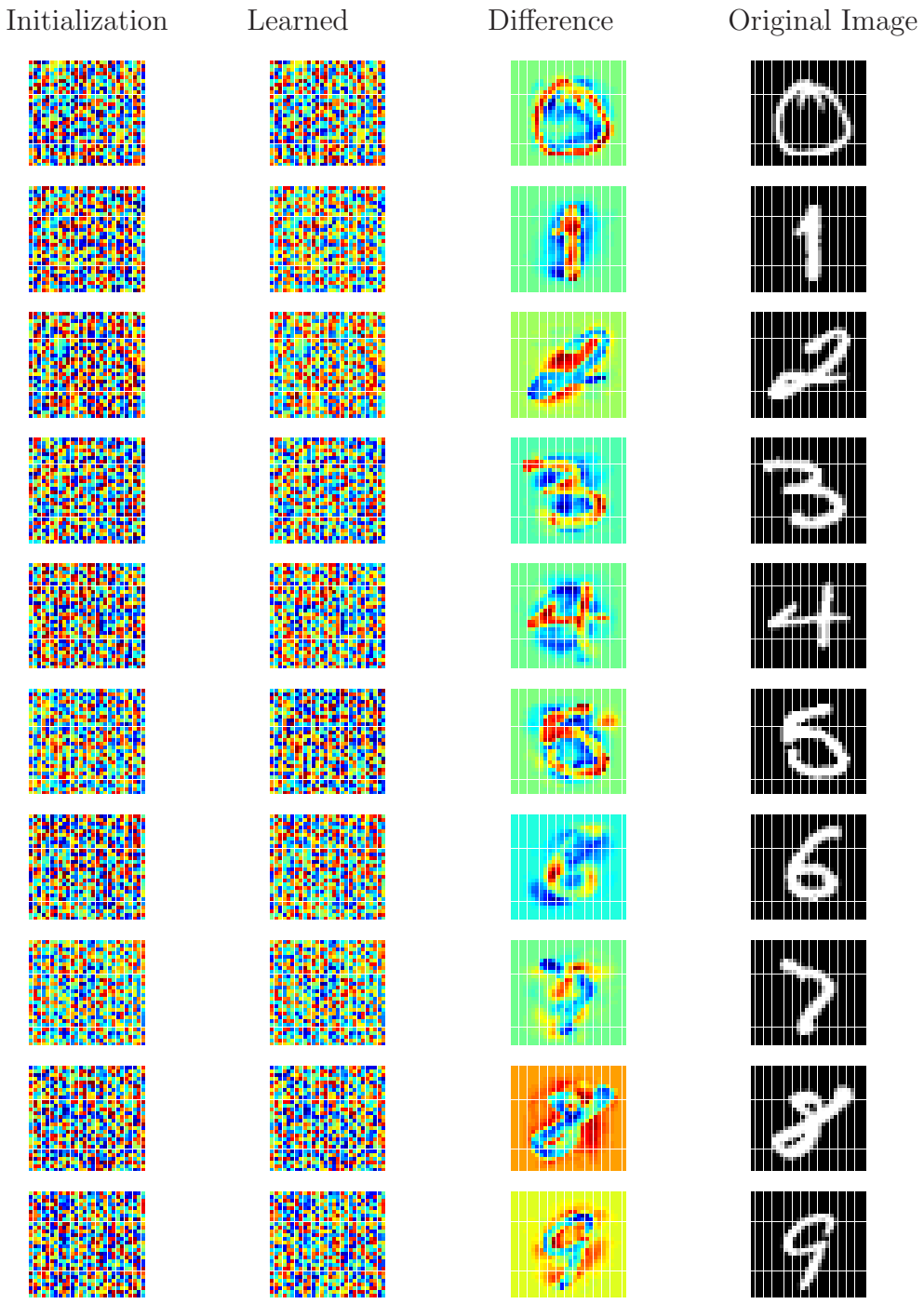

Figure 6-5: One randomly selected sample from each of the 10 MNIST classes, with the original image pictured in the last column. One randomly selected row of the projection matrix $\mathbf{A}_j$ associated with each point is shown. The random initialization is shown in the first column; the second column shows the learned row for LA-NCA with $d = 2$; the third column shows the difference between columns one and two. The classification performance of the initialized representation is 4.4%; the learned representation achieves 2.2% error; the original images achieve 2.8% error using a kNN classifier.

of $p(y|\mathbf{x})$ as follows:

$$p(y|\mathbf{x}) = \frac{p(y)\sum_{m=1}^{M_y}\lambda_{y,m}\mathcal{N}(\mathbf{x};\mu_{y,m},\Sigma_{y,m})}{\sum_{y'}p(y')\sum_{m=1}^{M_{y'}}\lambda_{y',m}\mathcal{N}(\mathbf{x};\mu_{y',m},\Sigma_{y',m})} \tag{6.19}$$

where $N(\mathbf{x};\mu,\Sigma) = \exp\{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\}/(2\pi)^{d/2}|\Sigma|^{1/2}$ is the multivariate Gaussian. The parameters of the model are mixing weights $\lambda_{y,m}$, means $\mu_{y,m}$, and covariances $\Sigma_{y,m}$ for the different mixture components. The $p(y)$ terms are priors for the different classes. Models which take a similar form to this, but where normalization is performed at the sentence level, are referred to as MMIE models in the speech recognition literature. A common method for optimizing log-likelihood under this model is to use the extended Baum-Welch algorithm (Gopalakrishnan et al. [1991]).

The LA-NCA approach in Equations 6.1 and 6.2 has close connections to discriminative GMMs. In particular, in the case where each projection matrix $\mathbf{A}_j$ is of dimension $D \times D$ (i.e., no dimensionality reduction is performed), it is possible to define a discriminative GMM which specifies the same distribution $p(y|\mathbf{x})$ as LA-NCA. The basic idea is to define the centers of the mixture components in the GMM to be the training points $\mathbf{x}_j$ for $j = 1 \ldots N$, and the associated covariance matrices to be $\Sigma_j^{-1} = \mathbf{A}_j^T\mathbf{A}_j$. A suitable choice for the $p(y)$ parameters and mixing weights can then be used to capture the effect of the β_j parameters. In this case LA-NCA can be viewed as using a reparameterization of a discriminative GMM, where the parameters $\mathbf{A}_j, \beta_j$ replace the parameters Σ_j and the mixing weights. The advantage of this reparameterization is that it is relatively easy to derive gradient updates for the LA-NCA model. ((Cheng et al. [2009]) discuss this reparameterization of discriminative GMMs at length. They do not, however, explore the case involving $A \in \mathbb{R}^{d\times D}$ where $d < D$.) Under the model in Eq. 6.19, naive gradient updates do not retain positive-definiteness constraints for Σ.

There are, however, several substantive differences between LA-NCA and MMI-trained GMMs, in particular: the use of centers $\mathbf{x}_j$ associated with training points; the use of a leave-one-out training criterion; the use of a different parameterization that allows gradient-based optimization; the use of projection matrices $\mathbf{A}_j$ of dimension

$(d \times D)$ where $d < D$, in which case the correspondence to GMMs is not so clear; and the use of a training objective that considers log-likelihood of training examples at the frame level (MMI training generally aims to optimize the conditional log-likelihood over entire training utterances).

Other criteria for training discriminative GMMs include large-margin methods (Sha and Saul [2007a]), and perceptron-style training (Cheng et al. [2009]). The TIMIT results in section 6.4 show that LA-NCA is competitive with large-margin GMMs.

While LA-NCA is applied to class-conditional probability estimation, it is related to other non-parametric statistical methods such as kernel regression and kernel density estimation. Kernel regression is a non-parametric method of estimation where the output value for a test point is estimated using a weighted average of the training samples (Benedetti [1977]). The weight on each training sample is determined using a kernel function that typically decays rapidly as the distance between the training and test sample increases. There has been recent work in kernel regression on learning both globally optimal kernel functions as well as locally adaptive functions such as the one proposed here (Navot et al. [2006], Takeda et al. [2006], Weinberger and Tesauro [2007]). Similar to kernel regression, kernel density estimation typically uses each training point to provide an estimate of the density of a distribution at a test point (Parzen [1961]). There have also been some attempts to train kernel density estimators discriminatively (Szummer and Jaakkola [2001]), and on adaptive kernels for these estimators (Brox et al. [2007], Sain [2002]). Our model differs from other proposed models mostly in the relative freedom of the kernel parameters and the way in which they are learned. There are other approaches to learning locally adaptive distance metrics such as (Domeniconi et al. [2005], Hastie and Tibshirani [1996]) which adapt the distance metric to each test point.

6.8 Lessons

This chapter introduced a generalization of NCA, LA-NCA, that learns multiple distance metrics that adapt to the local neighbourhood structure around each training point. We have demonstrated that LA-NCA is effective for acoustic modeling in speech recognition. There are several possible directions for future work. In terms of improvements in *computational efficiency*,[1] training and decoding times under the model depend linearly on the size of the set S; future work may consider methods for selecting S that are more effective than random sampling methods, and that may lead to sparser models. Parallelization methods may also be a natural way to increase efficiency of the algorithm in Figure 6-2. In terms of *model structure*, a better understanding of overfitting behavior, and of possible regularization of the $\mathbf{A}_j$ parameters, is needed. Finally, while the experiments in this chapter have focused on speech recognition, future work should consider other applications for LA-NCA.

[1] We note that our model is not particularly inefficient when compared to state-of-the-art large-scale discriminatively trained systems. In our experiments we chose the set S such that $|S| \approx 140,000$; it is not uncommon to have over $100,000$ Gaussian mixture components in modern speech recognition systems, e.g., see (Soltau et al. [2005]). Processing each training point in Figure 6-2 for the lecture-recognition task takes between 2x and 3x real time, which is comparable to other discriminative training methods for large-vocabulary speech recognition.

Chapter 7

Conclusion

7.1 Summary

This thesis was inspired by the neighborhood components analysis method (Goldberger et al. [2005]). This simple non-parametric method allowed us to do two things: (1) learn low-dimensional linear projections, and (2) provide nearest-neighbor based class-conditional probability estimates. Acoustic modeling has been traditionally approached as parametric estimation problem; but NCA provided an elegant non-parametric paradigm for developing nearest-neighbor based techniques for acoustic modeling.

Three problems related to acoustic modeling were tackled using NCA-based approaches. The first problem addressed was *performing dimensionality reduction* on acoustic vectors. NCA learns a low-dimensional linear projection of the feature space to improve the performance of a nearest neighbor classifier. The method avoids making parametric assumptions about the data and therefore can work well when the data is complex or multi-modal, which may be the case with acoustic data. NCA was shown to perform competitively with HLDA (Kumar [1997]) on the academic lecture task. The method also trained well with relatively few training samples suggesting that it could be applied successfully to problems such as speaker adaptation.

Second, we investigated the problem of *modeling the underlying structure of the acoustic-phonetic label space* in order to improve acoustic model estimates. The NCA-

ECOC model was developed to learn an output code to represent each acoustic-phonetic label. The similarity or dissimilarity between the output codes of each label could reveal structure in the label space. These output codes were used to improve the class-conditional probability estimates which are employed along with nearest neighbor information in the acoustic model. The end result was a relative reduction in word-error-rate on the academic lecture recognition task of 2.5%.

Third, the LA-NCA model was proposed to *locally adapt the distance metric to different parts of the feature space*. This model massively increases the number of parameters by learning a separate distance metric for each training point. Overfitting was controlled by subselecting support points and early stopping. The LA-NCA model delivered a 7-8% relative improvement in WER on the academic lecture recognition task.

7.2 Future Work

Future work could focus on improving the efficiency and performance of the models presented in this thesis.

Fast look-up of nearest neighbors could improve both training and test efficiency of the models and a number of algorithms have been published to this end (Andoni and Indyk [2006, 2008], Arya et al. [2002, 1998], Indyk and Motwani [1998], Kushilevitz et al. [1998], Mount [2006]). Many algorithms address the problem of quickly finding nearest neighbors under a single distance metric, though it is unclear how to perform fast search when using multiple distance metrics as in the LA-NCA model. Designing algorithms for finding nearest neighbors quickly under a locally adaptive distance metric could be an interesting direction for future research.

For the NCA-ECOC model, other definitions of similarity could be investigated. Additionally co-training of the distance metric and label space embedding could also be performed. Future work might also consider how the relationship between labels varies in different parts of the space.

For the LA-NCA model, methods for intelligent sub-selection of support points

could allow the model to work very well with many fewer parameters. Boosting approaches might be considered for the sub-selection of support points. It may also be possible to reduce the complexity of the model by constraining the distance metrics in some way. For instance, nearby points could somehow be constrained to have similar distance metrics. Regularization of the LA-NCA optimization criterion may also provide benefits. If efficiency of the method could be improved and overfitting concerns could be mitigated, the performance of the LA-NCA method could be even better.

Finally, evaluating these models on other applications such as some that arise in computer vision as well as on larger speech recognition tasks could be interesting avenues for future work. Using these models to perform speaker adaptation is also an important step towards incorporating these models into state-of-the-art recognizers.

Bibliography

Mit world, website: http://mitworld.mit.edu.

Mit open course ware, website: http://ocw.mit.edu.

I. S. Abramson. On bandwidth variation in kernel estimates–a square root law. *The Annals of Statistics*, 10(4):1217–1223, 1982.

E. L. Allwein, R. E. Schapire, and Y. Singer. Reducing multiclass to binary: a unifying approach for margin classifiers. *Journal of Machine Learning Research*, 1: 113–141, 2000.

A. Andoni and P. Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Communications of the ACM*, 51(1):117–122, 2008.

A. Andoni and P. Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In *Proceedings of the Symposium on Foundations of Compuer Science*, 2006.

S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y Wu. An optimal algorithm for approximate nearest neighbor searching in fixed dimensions. *Journal of the ACM*, 45(6):891–923, 1998.

S. Arya, T. Malamatos, and D. Mount. Space-efficient approximate voronoi diagrams. In *ACM STOC*, 2002.

S. Axelrod, V. Goel, RA Gopinath, PA Olsen, and K. Visweswariah. Subspace constrained gaussian mixture models for speech recognition. *IEEE Transactions on speech and audio processing*, 13(6):1144–1160, 2005.

G. A. Babich and O. I Camps. Weighted parzen windows for pattern classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(5):567–570, May 1996.

L. R. Bahl, F. Jelinek, and R. L. Mercer. "a maximum likelihood approach to continuous speech recognition". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(2):179–190, 1983.

A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall. Learning distance functions using equivalence relations. In *MACHINE LEARNING-INTERNATIONAL WORKSHOP THEN CONFERENCE-*, volume 20, 2003.

M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6):1373–1396, 2003.

J.K. Benedetti. On the nonparametric estimation of regression functions. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 248–253, 1977.

Y. Bengio, H. Larochelle, and P. Vincent. Non-local manifold parzen windows. In *Advances in Neural Information Processing Systems*. MIT Press, 2006a.

Y. Bengio, M. Monperrus, and H. Larochelle. Nonlocal estimation of manifold structure. *Neural Computation*, 18:2509–2528, 2006b.

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

J. Bilmes. Graphical models and automatic speech recognition. *Mathematical foundations of speech and language processing*, 2003.

M. Brockmann, T. Gasser, and E. Herrmann. Locally adaptive bandwidth choice for kernel regression estimators. *Journal of the American Statistical Association*, 88 (424):1302–1309, 1993.

T. Brox, B. Rosenhahn, D. Cremers, and H.P. Seidel. Nonparametric density estimation with adaptive, anisotropic kernels for human motion tracking. *Human Motion–Understanding, Modeling, Capture and Animation*, pages 152–165, 2007.

Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001.

H. Chang. Large-margin gaussian mixture modeling for automatic speech recognition. Master's thesis, Massachusetts Institute of Technology, 2008.

H. A. Chang and J. R. Glass. Hierarchical large-margin gaussian mixture models for phonetic classification. In *ASRU' 07*, pages 272–277, 2007.

C. Cheng, F. Sha, and L. K. Saul. Matrix updates for perceptron training of continuous density hidden markov models. In *Proceedings of ICML-09*, 2009.

P. Clarkson and P. J. Moreno. On the use of support vector machines for phonetic classification. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 585–588, 1999.

T. M. Cover. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1), 1967.

K. Crammer and Y. Singer. On the learnability and design of output codes for multiclass problems. *Machine Learning*, 47(2-3):201–233, 2002.

K. Crammer and Y. Singer. Improved output coding for classification using continuous relaxation. In *Advances in Neural Information Processing Systems*. MIT Press, 2000.

T. Deselaers, G. Heigold, and H. Ney. Speech recognition with state-based nearest neighbour classifiers. In *Interspeech*, 2007.

T. G. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2:263–286, 1995.

C. Domeniconi, J. Peng, and D. Gunopulos. Locally adaptive metric nearest-neighbor classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(9):1281–1285, 2002.

C. Domeniconi, D. Gunopulos, and J. Peng. Large margin nearest neighbor classifiers. *IEEE Transactions on Neural Networks*, 16(4), 2005.

R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. John Wiley and Sons, 1 edition, 1973.

J. S. Garofolo et al. Timit acoustic-phonetic continuous speech corpus. 1993.

G. Evermann, HY Chan, MJF Gales, B. Jia, D. Mrva, PC Woodland, and K. Yu. Training LVCSR systems on thousands of hours of data. In *Proc. ICASSP*, pages 209–212, 2005.

J. Fan. Design-adaptive nonparametric regression. *Journal of the American statistical Association*, 87(420):998–1004, 1992.

R.A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7:179–188, 1936.

E. Fosler-Lussier, S. Greenberg, and N. Morgan. Incorporating contextual phonetics into automatic speech recognition. In *Proceedings of Int. Congress of Phonetic Sciences*, 1999.

J. Frankel, M. Wester, and S. King. Articulatory feature recognition using dynamic Bayesian networks. *Computer Speech & Language*, 21(4):620–640, 2007.

Y. Freund, S. Dasgupta, M. Kabra, and N. Verma. Learning the structure of manifolds using random projections. In *Advances in Neural Information Processing Systems 18*. MIT Press, 2007.

K. Fukunaga and R. R. Hayes. The reduced parzen classifier. *Pattern Analysis and Machine Intelligence, IEEE Transactions*, 11:423–425, 1989.

J. Glass. A probabilistic framework for segment-based speech recognition. *Computer, Speech, and Language*, 17(2-3):137–152, 2003.

J. Glass, T. J. Hazen, L. Hetherington, and C. Wang. Analysis and processing of lecture audio data: Preliminary investigations. In *HLT-NAACL 2004 Workshop on Interdisciplinary Approaches to Speech Indexing and Retrieval*, pages 9–12, 2004.

J. Glass, T. J. Hazen, S. Cyphers, I. Malioutov, D. Huynh, and R. Barzilay. Recent progress in the mit spoken lecture processing project. In *Proceedings of Interspeech*, 2007.

A. Globerson and S. Roweis. Metric learning by collapsing classes. In Y. Weiss, B. Scholkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 513–520. MIT Press, 2006.

J. Goldberger, S. Roweis, G. Hinton, and R. Salakhutdinov. Neighbourhood components analysis. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 513–520. MIT Press, 2005.

PS Gopalakrishnan, D. Kanevsky, A. Nadas, and D. Nahamoo. An inequality for rational functions with applications to somestatistical estimation problems. *IEEE Transactions on Information Theory*, 37(1):107–113, 1991.

A. Gunawardana, M. Mahajan, A. Acero, and J. C. Platt. Hidden conditional random fields for phone classification. In *Interspeech*, pages 1117–1120, 2005.

A. Halberstadt and J. Glass. Heterogeneous measurements and multiple classifiers for speech recognition. In *ICSLP*, 1998.

A. K. Halberstadt and J. R. Glass. Hetergeneous acoustic measurements for phonetic classification. In *Eurospeech*, pages 401–404, 1997.

S. Har-Peled. A replacement for voronoi diagrams of near linear size. In *Proceedings of the Symposium of Foundations of Computer Science*, 2001.

M. Hasegawa-Johnson, J. Baker, S. Borys, K. Chen, E. Coogan, S. Greenberg, A. Juneja, K. Kirchhoff, K. Livescu, K. Sonmez, et al. Landmark-based speech recognition. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2005.

T. Hastie and R. Tibshirani. "discriminant adaptive nearest neighbor classification". *"IEEE Transactions on Pattern Analysis and Machine Intelligence"*, 18(6):607–616, 1996.

T. J. Hazen and E. McDermott. Discriminative mce-based speaker adaptation of acoustic models for a spoken lecture processing task. In *Proceedings of Interspeech*, 2007.

X. He and P. Niyogi. Locality preserving projections. *Advances in neural information processing systems*, 16:153–160, 2003.

H. Hermansky, D. P. W. Ellis, and S. Sharma. Tandem connectionist feature extraction for conventional hmm systems. In *International Conference on Acoustics, Speech, and Signal Processing*, 2000.

G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, pages 504–507, July 2006.

B.J.P. Hsu and J. Glass. N-gram weighting: reducing training data mismatch in cross-domain language model estimation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 829–838. Association for Computational Linguistics, 2008.

X. Huang, A. Acero, and H.W. Hon. *Spoken language processing: A guide to theory, algorithm, and system development*. Prentice Hall PTR Upper Saddle River, NJ, USA, 2001.

P. Indyk and R. Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *30th Symposium on Theory of Computing*, 1998.

A. J. Izenman. Recent developments in nonparametric density estimation. *Journal of the American Statistical Association*, 86(413):205–224, 1991.

DY Kim, HY Chan, G. Evermann, MJF Gales, D. Mrva, KC Sim, and PC Woodland. Development of the CU-HTK 2004 broadcast news transcription systems. In *International Conference on Acoustics, Speech, and Signal Processing*, 2005.

A. Klautau, N. Jevtic, and A. Orlitsky. On nearest-neighbor error-correcting output codes with aplication to all-pairs multiclass support vector machines. *Journal of Machine Learning Research*, 4:1–15, 2003a.

A. Klautau, N. Jevtic, and A. Orlitsky. Discriminative gaussian mixture models: A comparison with kernel classifiers. In *MACHINE LEARNING-INTERNATIONAL WORKSHOP THEN CONFERENCE-*, volume 20, page 353, 2003b.

E. G. Kong and T. G. Dietterich. Probability estimation using error-correcting output coding. In *International Conference on Artificial Intelligence and Soft Computing*, 1997.

Y. Koren and L. Carmel. Robust linear dimensionality reduction. *IEEE Transactions on Visualization and Computer Graphics*, pages 459–470, 2004.

N. Kumar. *Investigation of silicon auditory models and generalization of linear discriminant analysis for improved speech recognition.* PhD thesis, Johns Hopkins University, Baltimore, Maryland, 1997.

E. Kushilevitz, R. Ostrovsky, and Y. Rabani. Efficient search for approximate nearest neighbor in high dimensional spaces. In *Proceedings of the Symposium on Theory of Computing*, pages 614–623, 1998.

L. F. Lamel, R. H Kassel, and S. Seneff. Speech database development: design and analysis of the acoustic-phonetic corpus. In *Proceedings of the DARPA Speech REcognition Workshop*, pages 100–109, 1986.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

K. F Lee and H. W. Hon. Speaker-independent phone recognition using hidden markov models. *IEEE Transactions on Audio, Speech, and Language Processing*, 37:1641–1648, 1988.

J. M. Leiva-Murillo and A. Artes-Rodriguez. A fixed-point algorithm for finding the optimal covariance matrix in kernel density modeling. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 705–708, 2006.

K. Livescu. Analysis and modeling of non-native speech for automatic speech recognition. Master's thesis, Massachusetts Institute of Technology, 1999.

K. Livescu, J. Glass, and J. Bilmes. Hidden feature models for speech recognition using dynamic Bayesian networks. In *Eighth European Conference on Speech Communication and Technology*, 2003.

D.G. Lowe. Similarity metric learning for a variable-kernel classifier. *Neural computation*, 7(1):72–85, 1995.

J. C. Marshall and M. L. Hazelton. Boundary kernels for adaptive density estimators on regions iwth irregular boundaries. 101:949–963, 2010.

S. Matsoukas and R. Schwartz. Improved speaker adaptation using speaker dependent feature projections. In *Workshop on Automatic Speech Recognition and Understanding*, pages 273–278, 2003.

E. McDermott. *Discriminative training for speech recognition*. PhD thesis, Waseda University, School of Engineering, 1997.

E. McDermott and T. J. Hazen. Minimum classification error training of landmark models for real-time continuous speeech recognition. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 937–940, 2004.

E. McDermott and S. Katagiri. Minimum classification error for large scale speech recognition tasks using weighted finite state transducers. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 113–116, 2005.

E. McDermott, T. J. Hazen, J. Le Roux, A. Nakamura, and S. Katagiri. Discriminative training for large-vocabulary speech recognition using minimum classification error. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(1), 2007.

A. Mohamed and G. Hinton. Phone recognition using restrited boltzmann machines. In *International Conference on Acoustics, Speech, and Signal Processing*, 2010.

D. M. Mount. Approximate nearest neighbor (ann) library. *http://www.cs.umd.edu/m̃ount/ANN*, 2006.

A. Navot, L. Shpigelman, N. Tishby, and E. Vaadia. Nearest neighbor based feature selection for regression and its application to neural activity. *Advances in Neural Information Processing Systems*, 18, 2006.

M. Ostendorf. Moving beyond the 'beads-on-a-string' model of speech. In *Proc. IEEE ASRU Workshop*, pages 79–84, 1999.

A. Park, T. J. Hazen, and J. R. Glass. Automatic processing of audio lectures for information retrieval: Vocabulary selection and language modeling. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 497–500, 2004.

E. Parzen. On estimation of a probability density function and mode. *Annals of Mathematical Statistics*, 33(3):1065–1076, 1961.

K. Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 2(6):559–572, 1901.

J. Peltonen, J. Goldberger, and S. Kaski. Fast semi-supervised discriminative component analysis. *MLSP*, pages 312–317, 2007.

D. Povey and B. Kingsbury. Evaluation of proposed modifications ot mpe for large scale discriminative training. In *International Conference on Acoustics, Speech, and Signal Processing*, 2007.

D. Povey and P. C. Woodland. Minimum phone error and i-smoothing for improved discriminative training. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 105–108, 2002.

D. Povey, B. Kingsbury, L. Mangu, G. Saon, H. Soltau, and G. Zweig. Fmpe: discriminatively trained features for speech recognition. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 961–964, 2005.

D. Povey, D. Kanevsky, B. Kingsbury, B. Ramabhadran, G. Saon, and K. Visweswariah. Boosted mmi for model and feature-space discriminative training. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 4057–4060, 2008.

W. H. Press, S. A. Teukolsky, W. T. Vetterline, and B. P. Flannery. *Numerical recipes: the art of scientific computing*. Cambridge University Press, 3 edition, 2007.

O. Pujol, P. Radeva, and J. Vitria. Discriminant ecoc: a heuristic method for application dependent design of error correcting output codes. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, 28(6), 2006.

V. Ramasubramanian, K. Kulkarni, and B. Kaemmerer. Acoustic modeling by phoneme templates and modified one-pass dp decoding for continuous speech recognition. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 4105–4108, 2008.

R. Rifkin. Speaker recognition using local models. In *Interspeech*, 2003.

R. Rifkin, K. Schutte, and M. Saad. Noise robust phonetic classification with linear regularized least squares and second-order features. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 881–884, 2007.

S.T. Roweis and L.K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323, 2000.

S. R. Sain. Multivariate locally adaptive density estimation. *"Computational Statistics and Data Analysis"*, 39:165–186, 2002.

R. Salakhutdinov and G. Hinton. Learning a nonlinear embedding by preserving class neighbourhood structure. *AI and Statistics*, 2007.

G. Saon and D. Povey. Penalty function maximization for large margin hmm training. In *Interspeech*, 2008.

G. Saon, G. Zweig, and M. Padmanabhan. Linear feature space projections for speaker adaptation. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 325–328, 2001.

F. Sha and L. K. Saul. Large margin gaussian mixture modeling for phonetic classification and recognition. In *International Conference on Acoustics, Speech, and Signal Processing*, 2007a.

F. Sha and L. K. Saul. Comparison of large margin training to other discriminative methods for phonetic recognition by hidden markov models. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 313–316, 2007b.

F. Sha and L. K. Saul. Large margin hidden markov models for automatic speech recognition. In *Advances in Neural Information Processing Systems*, 2007c.

J. Shlens. A tutorial on principal component analysis. *University of California, San Diego*, 2005.

N. Singh-Miller and M. Collins. Learning label embeddings for nearest-neighbor multiclass classification with an application to speech recognition. In *Neural Information Processing Systems*, 2009.

N. Singh-Miller, M. Collins, and T. J. Hazen. Dimensionality reduction for speech recognition using neighborhood components analysis. In *Interspeech*, 2007.

L.I. Smith. A tutorial on principal components analysis. *Cornell University, USA*, 2002.

H. Soltau, B. Kingsbury, L. Mangu, D. Povey, G. Saon, and G. Zweig. The IBM 2004 conversational telephony system for rich transcription. In *Proc. ICASSP*, volume 1, pages 205–208, 2005.

D. F. Specht. Generation of polynomial discriminant functions for pattern recognition. *IEEE Transactions of electronic computers*, pages 308–319, 1967.

K. N. Stevens. *Acoustic Phonetics*. MIT Press, 2000.

M. Sugiyama. Dimensionality reduction of multimodal labeled data by local Fisher discriminant analysis. *The Journal of Machine Learning Research*, 8:1061, 2007.

M. Szummer and T. Jaakkola. Kernel expansions with unlabeled examples. In *Advances in Neural Information Processing Systems 13*. MIT Press, 2001.

H. Takeda, S. Farsiu, and P. Milanfar. Robust kernel regression for restoration and reconstruction of images from sparse noisy data. In *2006 IEEE International Conference on Image Processing*, pages 1257–1260, 2006.

S. Tong and D. Koller. Restricted bayes optimal classifiers. In *Proceedings of the National Conference on Artificial Intelligence*, pages 658–664, 2000.

A. Torralba, R. Fergus, and Y. Weiss. Small codes and large image databases for recognition. *IEEE Computer Vision and Pattern Recognition*, June 2008.

L. Torresani and K. Lee. Large margin component analysis. *Advances in neural information processing systems*, 19:1385, 2007.

A.M. Turing. Computing machinery and intelligence. *Parsing the Turing Test*, pages 23–65, 2008.

A.M. Turing. Computing machinery and intelligence. *Mind*, 1950.

V. Valtchev, JJ Odell, PC Woodland, and SJ Young. MMIE training of large vocabulary recognition systems. *Speech Communication*, 22(4):303–314, 1997.

P. Vincent and Y. Bengio. Manifold parzen windows. In S. Becker, S. Thrun, and K. Overmayer, editors, *Advances in Neural Information Processing Systems 15*. MIT Press, 2003.

L. Wang and P. C. Woodland. Discriminative apative training using the mpe criterion. In *ASRU 2003*, pages 279–284, 2003.

L. Wang and P.C. Woodland. MPE-based discriminative linear transform for speaker adaptation. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 321–324, 2004.

K. Q. Weinberger, J. Blitzer, and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. In *Advances in Neural Information Processing Systems*. MIT Press, 2006.

K.Q. Weinberger and G. Tesauro. Metric learning for kernel regression. In *Eleventh international conference on artificial intelligence and statistics*, pages 608–615. Citeseer, 2007.

P. C. Woodland and D. Povey. Large scale discriminative training for speech recognition. In *Automatic Speech Recognition*, 2000.

P. C. Woodland and D. Povey. Large scale discriminative training of hidden markov models for speech recognition. *Computer Speech and Language*, 16:25–47, 2002.

E.P. Xing, A.Y. Ng, M.I. Jordan, and S. Russell. Distance metric learning with application to clustering with side-information. *Advances in neural information processing systems*, pages 521–528, 2003.

J. Yousafzai, Z. Cvetkovic, and P. Sollich. Improving the Robustness of Phoneme Classification Using Hybrid Features. 2010.

D. Yu, L. Deng, X. He, and A. Acero. Large-margin minimum classification error training for large-scale speech recognition tasks. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 1137–1140, 2007.

G. Zavaliagkos, Y. Zhao, R. Schwartz, and J. Makhoul. A hybrid segmental neural net/hidden markov model system for continuous speech recognition. *IEEE Transactions on Speech and Audio Processing*, 2(1):151–160, 1994.

B. Zhang and S. Matsoukas. Minimum phoneme error based heteroscedastic linear discriminant analysis for speech recognition. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 925–928, 2005.