Improving the Performance of LVCSR Using Ensemble of Acoustic Models

Thesis Proposal

By

Rong Zhang

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
May, 2006

Committee:
Alexander Rudnicky
Tanja Schultz
Richard Stern
Karthik Visweswariah, IBM
Improving the Performance of LVCSR Using Ensemble of Acoustic Models

Abstract

Recent advances in Machine Learning have brought to attention new theories of learning as well as new approaches. Among these, the Ensemble method has received wide attention and has been shown to be a promising method for classification problems. Simply speaking, the ensemble method is a learning algorithm that constructs a set of “weak” classifiers and then combines their predictions to produce a more accurate classification. The underlying idea of the ensemble method is that the combination of diversified classifiers that have uncorrelated, and ideally complementary, error patterns can offer improved performance and a robust generalization capability.

Given its successes for many classification problems, we began investigating the problem of adapting ensemble techniques to continuous speech recognition. Continuous Speech Recognition has been acknowledged as one of the most challenging tasks in classification. The performance of an ASR system is negatively impacted by a number of issues, such as corruption of noise, variability of speaker and speaking mode, change of environment conditions, transmission of channel, inaccuracy of model assumption, complexity of language, etc.. The primary goal of our research is to discover methods suitable for ensemble construction and combination that meet these special requirements of continuous speech recognition. We propose several novel ensemble-based acoustic model training and combination schemes, and test their effectiveness using real-world speech corpora. Preliminary results are described in this proposal, in particular

- Utterance-level Boosting training algorithm for large scale acoustic modeling
- Frame-level Boosting training algorithm using a Word Error Rate reduction criterion
- N-Best list re-ranking and Rover combination to generate a better hypothesis

Encouraging experimental results convince us that the ensemble technique is a promising method and that it has the potential to substantially improve the performance of a LVCSR system. However research on ensemble methods for speech recognition is still in its early stage and unsolved questions on ensemble generation and hypothesis combination remain to be addressed. This proposal sets out several key research topics that, if successfully addressed will have the potential to significantly increase the accuracy of ensemble-based speech recognition systems. These include the following:

- Training criteria targeted at reducing Word Error Rate rather than Sentence Error Rate.
- Integrating data manipulation and feature manipulation methods for continuous speech recognition.
- Combination methods working on different objects, different levels and different decoding stages.
- Ensemble-based semi-supervised acoustic model training algorithm using labeled and unlabeled data.
Contents

1 Introduction

2 Ensemble Methods for Classification
   2.1 Why ensembles
   2.2 Methods for Constructing Ensemble
      2.2.1 Manipulating the Training Data
         2.2.1.1 Bagging
         2.2.1.2 Boosting
      2.2.2 Manipulating the Input Features
      2.2.3 Manipulating the Output Targets
      2.2.4 Others

3 Existing Ensemble Methods in Speech Recognition
   3.1 Construction
      3.1.1 Multi-Band Models and Multi-Stream Models
      3.1.2 Boosting for Acoustic Modeling
      3.1.3 Miscellaneous Methods
   3.2 Combination
      3.2.1 Feature Combination
      3.2.2 Likelihood / Posterior Combination
      3.2.3 Hypothesis Combination
         3.2.3.1 ROVER
         3.2.3.2 Word Lattice Based Combination

4 Preliminary Experiments of Ensemble Based Acoustic Model Training
   4.1 Utterance Level Boosting Training
   4.2 Frame Level Boosting Training
   4.3 N-Best List Re-Ranking and Rover Combination

5 Open Questions
   5.1 Investigation on Training Criteria for Reducing Word Error Rate
   5.2 Investigation on Combination Methods
   5.3 Investigation on Ensemble Based Semi-Supervised Acoustic Model Training

6 Expected Contributions

7 Timeline

8 References
1 Introduction

The goal of automatic speech recognition is to translate an acoustic spoken utterance into other representations, such as a sequence of words, which are easier for further computer or human processing. Figure 1.1 illustrates the general architecture of an ASR system [after Ney, 1990].

As Figure 1.1 shows, a speech recognition system consists of four major building blocks: Feature Extraction module, Search module, Acoustic Model and Language Model.

- **Feature Extraction.** The task of this module is to detect the analog speech signal, and parameterize it into a sequence of acoustic feature vectors \( \mathbf{x} = (x_1, x_2, \ldots, x_T) \).

- **Search.** The task of this module is to find the word string \( \hat{h}^* = (w_1, w_2, \ldots, w_K) \) with the highest joint probability \( \hat{h}^* = \arg \max_h P(h)P(\mathbf{x} \mid h) \) over all possible hypotheses \( h \).

- **Lexical Model.** A dictionary that defines the pronunciations and their variants for the words to be recognized in terms of sub-word units, such as triphones or senones.

- **Acoustic Model.** A statistical model that estimates \( P(\mathbf{x} \mid h) \), the conditional probability of observing a sequence of acoustic feature vectors \( \mathbf{x} \) for a given word string \( h \). Gaussian Mixtures based Hidden Markov Model (HMM) is the most widely used acoustic model in speech recognition.

- **Language Model.** A statistical model that provides measurement of \( P(h) \), the prior probability of a particular word string \( h \) being spoken. N-gram model has been established as the de facto standard for large vocabulary continuous speech recognition.
Continuous Speech Recognition has been acknowledged as one of the most challenging tasks in classification. The performance of an ASR system is negatively impacted by a number of issues, such as corruption of noise, variability of speaker and speaking mode, change of environment conditions, transmission of channel, inaccuracy of model assumption, complexity of language, etc. Similar problems also exist in other classification tasks, but are particularly serious in continuous speech recognition. In addition, some distinct properties of continuous speech recognition add further challenges to the task.

- Both the input instance $x = (x_1, x_2, ..., x_T)$ and output hypothesis $h = (w_1, w_2, ..., w_K)$ of a continuous ASR system are sequences that can be arbitrarily long. In contrast, for many common classification problems, the length of the input feature vector is fixed, and the output is only a simple class label.

- The space of sentence hypothesis for a consecutive utterance can be very large. For example, suppose an ASR system has a vocabulary with $M$ words and the maximum length of a hypothesis is artificially set to $K$ words. The number of possible hypothesis for searching is about (including the null string):

$$1 + M + M^2 + M^3 + ...... + M^K = \frac{M^{K+1} - 1}{M - 1}$$

Please note the number listed above does not consider the time segment information associated with each word. If we take account of segment information into calculation, the number of possible hypotheses would be much larger. As a practical matter, it is impossible for a recognizer to enumerate all hypotheses to estimate their probabilities.

- In LVCSR systems, sub-words, e.g., triphones or senones, are chosen as the speech unit for acoustic modeling. On the other hand, the speech corpora used for LVCSR system training are organized as a set of utterances that are transcribed into string of words. Usually, the segment information necessary for sub-word model training is not available in the speech dataset due to the expense of transcription. One has to use Viterbi based forced-alignment or other methods to guess a likely boundary for each sub-word appearing in an utterance.

- There is also a considerable mismatch between the training criteria of acoustic modeling and the measurement of recognition performance. The speech community often uses Word Error Rate (WER), a word level metric to evaluate system performance by computing recognition errors in terms of substitution, deletion and insertion.

$$Word\ Error\ Rate = \frac{Substitutions + Deletions + Insertions}{Number\ of\ Spoken\ Words}$$

However, as we will show later, current acoustic modeling techniques, e.g. MLE, MCE and Boosting, focus on how to reduce sentence level training errors rather than word level errors. This mismatch leads to an often observed phenomenon where the hypothesis with the highest likelihood is not the one with the lowest word error rate.

In the past several years, the machine learning community has extensively investigated and advocated the use of ensemble methods for solving complicated classification problems. Some general methods for constructing ensembles have been developed, such as Boosting, Bagging, Random Space, Random Forests, etc.. Meantime, speech-specific methods that explore the advantage of multiple recognizers were also investigated by the speech community, e.g. Multi-Band model and ROVER combination. The effectiveness of ensemble methods has been demonstrated through their applications to a variety of classification problems. However, research into ensemble methods for time series or sequence recognition problems, especially for continuous speech recognition, is still at an early stage. For example, the general ensemble methods proposed by the machine learning community were initially designed for binary or multiple-class classification. Such methods do not take into account the characteristic of continuous speech. The primary goal of this proposal is to investigate suitable ensemble-based acoustic model training and combination strategies that can fulfill the special requirements of continuous speech recognition.

The proposal is organized as follows. We begin with a short survey of some general approaches for constructing ensemble of classifiers in Chapter 2. Chapter 3 provides a review of the state-of-the-art
ensemble methods in speech recognition. Chapter 4 presents our preliminary research results including several novel acoustic model training and combination schemes. Chapter 5 and 6 discuss the open questions that we will work on in the next stage, as well as the expected contribution. Finally, we conclude the proposal with a tentative research time table.


2 Ensembles Methods for Classification

Ensemble methods are learning algorithms that construct a set of classifiers then combine their individual decisions in some fashion, in order to classify new examples. It has been shown that such a combination of “weak” classifiers can result in a “strong” composite classifier whose classification performance is much better than that of any single classifier. As one of the most active research areas in Machine Learning, the ensemble approach is also known as the Fusion of Models, Mixture of Experts, Committee of Learners, Multiple Classifier System, Consensus Theory, as well as by other names.

2.1 Why Ensembles

The observation that ensembles perform better than any of its individual members is not an accidental phenomenon. To see why, imagine we have an ensemble of $2n+1$ classifiers, which have uncorrelated error patterns. Supposing $e$ is the error rate of the worst classifier, the performance of ensemble after combination with majority voting would be no worse than

$$\sum_{k=n+1}^{2n+1} C^k_{2n+1} e^k (1 - e)^{2n+1-k}.$$ 

For example, in the case of combination of three independent classifiers which error rates are equal to 20%, the overall error rate of the ensemble will be dramatically reduced to 10.4%.

Beyond this the intuitive explanation, [Dietterich, 1998] gives a deeper analysis for why ensembles can improve performance, or why it is not possible to find a single classifier that works as well as an ensemble. [Dietterich, 1998] shows that the strength of ensemble lies on its competence and flexibility in dealing with the following three situations: the training data may not provide sufficient information to choose a single best classifier; the learning algorithm we adopted may not be able to solve the difficult search problem we pose; and the hypothesis space may not contain the true function.

The example above also indicates the key issues to be considered in constructing a successful ensemble: the individual classifiers need be accurate and diverse [Kuncheva et al., 2002; Shipp & Kuncheva, 2002; Kuncheva & Whitaker, 2003]. An accurate classifier is one that has an error rate at least better than random guessing. For binary classification, this means the error rate of any component of ensemble should be lower than 50%. However, this condition alone can not guarantee that the performance of ensemble is superior to its components. The overall performance of ensemble depends not only on the individual performance of each component, but also on how well different classifiers complement to each other. For example, a combination of identical classifiers will not provide any help to the classification even though they have high accuracy. Research has shown that the error pattern of an individual classifier plays a critical role in ensemble based classification. For an ideal ensemble, we expect that individual classifiers are diverse so that the errors made by them can be uncorrelated and complementary. Obviously, this is not an easily-fulfilled requirement in the case of certain real-world applications such as continuous speech recognition. Investigations on measures of diversity and on the realization of diverse classifiers have been an important branch of ensemble research.

2.2 Methods for Constructing Ensembles

Many methods for constructing ensemble have been developed. These methods can be roughly categorized into three classes in accordance with the different objects being exploited in construction procedure. In addition, there are some ad hoc methods specific to particular algorithms or applications. This section presents a brief review of some representative ensemble learning methods.

2.2.1 Manipulating the Training Data

This class of methods adopts a strategy that generates multiple hypotheses by running the learning algorithm several times, each time with a different subset or distribution of training data. Well-known examples include Bagging and Boosting.
2.2.1.1 Bagging

Bagging is a straightforward way of manipulating training set for ensemble construction [Breiman 1996]. In each round of Bagging learning, a new training set is created using the technique called Bootstrap Sampling, which randomly draws a sample of \( N \) examples with replacement from the original training set consisting of \( N \) examples. Note that some examples in the original set may not appear in a Bootstrap sample while others may appear more than once. A new single classifier is then learned from the sampled training set. Ensemble is constructed by running this procedure repeatedly, and the final hypothesis of the ensemble is determined by selecting the one best agreed on by individual classifiers. The algorithm is illustrated in Table 2.1, in which function \( f(x, y) \) could be interpreted as a classifier or recognition model that maps a feature/label pair to some confidence or probabilistic metrics such that \( 0 \leq f(x, y) \leq 1 \).

Bagging is showed to be effective when used with “unstable” learning algorithms such as Decision Tree and Neural Network, for which a small change of training data may lead to a large change of learned concept. However, it may cause degradation when applied to “stable” learning algorithm such as K-Nearest Neighbor.

Input:
- Training set of \( N \) labeled examples \( \Psi = \{(x_i, y_i) \mid 1 \leq i \leq N\} \), where feature vector \( x_i = (x_{i1}, x_{i2}, \ldots, x_{iD}) \in \mathbb{R}^D \), a \( D \)-dimension space, and class label \( y_i \in Y = \{c_1, c_2, \ldots, c_M\} \).
- A learning algorithm.
- An integer \( K \) specifying the number of individual classifiers in ensemble.

For \( k=1 \) to \( K \):
- Create a Bootstrap replicate \( \Psi_k \) with replacement from the original training set \( \Psi \).
- Learn a new classifier \( f_k(x, y) \) on \( \Psi_k \).

Generalization:
- The class label for a new example \( x \) is determined by majority voting:
\[
y^* = \arg \max_{y \in Y} \sum_{k=1}^{K} f_k(x, y)
\]

Table 2.1 Bagging Algorithm

2.2.1.2 Boosting

Boosting has been, so far, the most successful approach developed so far for constructing ensemble s [Freund and Schapire, 1996; Freund and Schapire, 1997; Schapire et al., 1997; Breiman, 1998; Mason et al., 1999; Schapire, 1999; Schapire and Singer, 1999; Collins et al., 2000]. In Boosting, single classifiers are iteratively trained in a fashion such that hard-to-classify examples are given increasing emphasis. More specifically, the algorithm maintains a probability distribution for the training data, and initially every example is assigned equal weight. In each round, a new single classifier is learned from the current distribution. Meantime, a parameter that measures the classifier’s importance is determined in respect of its classification accuracy. The single classifier is then used to classify every training example. The probability distribution is updated in such a way that the weight of an example will be enhanced if it is misclassified, or reduced otherwise. As a result those examples that are difficult to classify will receive more attention from the training of subsequent classifiers. The AdaBoost algorithm for multi-class classification is illustrated in Table 2.2 [Freund and Schapire, 1996; Schapire and Singer, 1999].

Boosting has some important theoretic properties that allow it to outperform Bagging in classification. It has been shown that the training error of the Boosting algorithm drops exponentially fast to zero as the number of combined classifiers increases. More important, bounds of generalization error of Boosting
algorithm have been formulated in terms of VC-dimension and margin, which suggests that Boosting is not sensitive to the problem of overfitting.

**Input:**
- Training set of \( N \) labeled examples \( \Psi = \{(x_i, y_i) | 1 \leq i \leq N\} \), where feature vector \( x_i = (x_{i1}, x_{i2}, \ldots, x_{iD}) \in \mathbb{R}^D \), a \( D \)-dimension space, and class label \( y_i \in Y = \{c_1, c_2, \ldots, c_M\} \).
- A learning algorithm.
- An integer \( K \) specifying the number of individual classifiers in ensemble.

**Initialization:**
- Let \( B = \{(i, y) | 1 \leq i \leq N, y \in Y \text{ and } y \neq y_i\} \).
- Initialize distribution of training data: \( D_1(i, y) = 1/|B| \) for all \((i, y) \in B\).

For \( k = 1 \) to \( K \):
- Train a new classifier \( f_k(x, y) \) with respect to distribution \( D_k(i, y) \).
- Compute pseudo loss for classifier \( f_k(x, y) \):
  \[ \varepsilon_k = \frac{1}{2} \sum_{(i, y) \in B} D_k(i, y)(1 - f_k(x_i, y_i) + f_k(x_i, y)) \].
- Set \( \beta_k = \varepsilon_k / (1 - \varepsilon_k) \).
- Compute importance factor \( \alpha_k \) for \( f_k(x, y) \): \( \alpha_k = -\log \beta_k \).
- Update distribution \( D_k(i, y) \) by:
  \[ D_{k+1}(i, y) = \frac{D_k(i, y)}{Z_k} \beta_k^{1 + f_k(x_i, y) - f_k(x_i, y)} \]
  where \( Z_k \) is a normalization factor chosen to make \( D_{k+1}(i, y) \) a distribution function.

**Generalization:**
- The class label for a new example \( x \) is determined by weighted voting:
  \[ y^* = \arg \max_{y \in Y} \sum_{k=1}^{K} \alpha_k f_k(x, y) \]

<table>
<thead>
<tr>
<th>Table 2.2 AdaBoost Algorithm for Multi-Class Classification.</th>
</tr>
</thead>
</table>

### 2.2.2 Manipulating the Input Features

In this class of methods, the ensemble is constructed by manipulating the input features. A popular example is Random Space in which the individual classifiers are constructed from subspaces formed by randomly selecting a number of dimensions from the original feature space [Ho, 1995; Ho, 1998; Skurichina and Duin, 2002]. For a given feature space of \( D \) dimensions, there are possible \( 2^D \) such subspaces available for selection. In each round of training, a selection is made to learn a new classifier e.g. Decision Tree. In the generalization stage, the final decision on a questioned instance is reached through majority voting of individual classifiers. Table 2.3 shows the algorithm.

When the number of training instances is relatively small, Random Space may have the capability to solve the data scarcity problem by constructing classifiers through random selection from \( 2^D \) subspaces. On the other hand, in a high dimensional feature space, the possibility that all dimensions will be selected is very low. Moreover, the vast number of subspaces provides more choices than is needed in practice, which to some extent makes the use of all dimensionality unnecessary. Therefore, while other classification methods suffer from the curse of dimensionality, Random Space may be able to take advantage of high dimensionality.
Input:

- Training set of \( N \) labeled examples \( \Psi = \{(x_i, y_i) \mid 1 \leq i \leq N\} \), where feature vector \( x_i = (x_{i1}, x_{i2}, \ldots, x_{iD}) \in \mathbb{R}^D \), a \( D \)-dimension space, and class label \( y_i \in Y = \{c_1, c_2, \ldots, c_M\} \).
- A learning algorithm.
- An integer \( K \) specifying the number of individual classifiers in ensemble.

For \( k = 1 \) to \( K \):

- Randomly select a \( d_k \)-dimension subspace from original \( D \)-dimension feature space (\( d_k \leq D \)), and construct a new training set \( \Psi_k^{d_k} = \{(\tilde{x}_i, y_i) \mid 1 \leq i \leq N\} \) where \( \tilde{x}_i \) is the projected vector of \( x_i \) in the \( d_k \)-dimension subspace.
- Learn a new classifier \( f_k(x_i^{d_k}, y) \) on \( \Psi_k^{d_k} \).

Generalization:

- The class label for a new example \( x \) is determined by majority voting:
  \[
y^* = \arg \max_{y \in Y} \sum_{k=1}^{K} f_k(\tilde{x}_i^{d_k}, y)
  \]
  where \( \tilde{x}_i^{d_k} \) is the projected vector of \( x \) in subspaces.

Table 2.3 Algorithm of Random Space

**2.2.3 Manipulating the Output Targets**

The third class of methods for constructing ensemble is to manipulate the output target values of the classifier. Error-Correcting Output Coding is a successful example of this class of methods [Dietterich and Bakiri, 1995]. Suppose that the number of classes, \( M \), is large. The new learning problem can be constructed by randomly partitioning the \( M \) classes into two subsets, \( A \) and \( B \). The training data is then re-labeled so that the examples with the original classes in set \( A \) are given new label 0 while the other examples are given new label 1. Thus, a multi-class problem is converted into a binary class problem. The re-labeled data is feed to the learning algorithm for generating a new binary classifier, and consequently the ensemble is obtained by repeating the process many times. In generation stage, the outputs of each binary classifier give votes to the original classes. The class with the highest number of votes is selected as the final prediction for the entire ensemble.

**2.2.4 Other Techniques**

Methods for generating ensembles of classifiers can be viewed as means of injecting randomness into different level of learning. For example, in Bagging, the randomization is performed in Bootstrap sampling, and in Random Space, the randomization is performed in the selection of subspace. Moreover, Random Forests, investigated by [Dietterich, 2000; Breiman, 2001], showed that the randomness can also be injected into the learning procedure itself. Random Forests is a combination of tree-structured classifiers e.g. Decision Tree, each of which is grown by randomly selecting a feature-value test from the top-\( n \) best feature-value tests for node splitting. In contrast, classic tree-growing method always uses the best feature, measured by information gain, to split node. It has been shown that Random Forests can yield performance comparable to Boosting algorithm, and demonstrated more robustness to noise.

In addition to the ensemble methods listed above, there are also algorithm-specific and application-specific methods. In the next section, we will discuss methods developed for speech recognition.
3 Existing Ensemble Methods for Speech Recognition

Research in ensemble and combination methods for speech recognition has a long history that can be traced back to 1980s [Stolfo et al., 1989]. A large number of innovative and effective approaches that utilize the characteristic of continuous speech have been developed since then. Recently, the general ensemble methods advocated by machine learning community, such as Boosting and Bagging, have aroused extensive research interests in speech community. This section presents a brief review of some widely used methods for constructing and combining multiple recognition systems.

3.1 Construction

Following the classification scheme used in the previous chapter, ensemble construction methods for speech recognition can also be divided into categories such as feature manipulation (i.e. Multi-Band model) or data manipulation. In addition, there are also some speech-specific methods.

3.1.1 Multi-Band Models and Multi-Stream Models

Narrow band noise, noise that occurs in a certain frequency range, is a common reason for performance degradation of ASR systems. This is because, in conventional “full-band” recognition, feature extraction is carried out over the whole frequency domain. Thus corruption of the speech signal caused by noise in any sub-band is spread to all the components of acoustic feature vector. However, experiments on articulation index [Fletcher, 1953] have shown that human auditory perception is based on decisions within narrow frequency bands that are processed independently of each other. Moreover, humans are able to extract sufficient residual information from the clean frequency sub-bands, even with a considerable part of frequency domain being corrupted by noise.

Motivated by these observations, researchers have proposed and investigated Multi-Band models to enhance the robustness of ASR system in noise environment [Bourlard et al., 1996; Bourlard and Dupont, 1997; Tibrewala and Hermansky, 1997; Hagen et al., 1998; Dupont and Ris, 2001; Hagen et al., 2001; Hagen and Bourlard, 2001]. In Multi-Band recognition, the speech spectral domain is split into several frequency sub-bands, each of which is then processed separately to extract acoustic features. This is followed by estimation of frame level phoneme probabilities for each sub-band using corresponding sub-band features. These probabilities are then combined by some suitable rule, such as weighted sum or product, to yield a new vector representing the merged probability estimates which is further used in decoding. Figure 3.1 illustrates the architecture of Multi-Band system [Sharma, 1999]. The advantage of Multi-Band recognition is that the noise from one sub-band can be isolated from other bands in feature extraction and phoneme probability estimation. In addition, the impact of narrow band noise on overall recognition performance can be further offset by deemphasizing its weight in combination.

![Figure 3.1 Multi-Band Model](Sharma, 1999)
Multi-Stream models are another ensemble approaches which slightly differ from Multi-Band model in the features used in recognition. As compared to Multi-Band models in which each stream comprises features extracted from certain sub-band of speech spectrum, the streams of Multi-Stream model capture features from the entire frequency domain using different processing algorithms [Janin et al, 1999; Sharma, 1999; Christensen et al, 2000; Hagen et al, 2000; Neto and Meinedo, 2000; Shire, 2000]. More important, the Multi-Stream model provides a generalized framework that allows for the use of various kinds of combination strategies for continuous speech recognition, such as the combination of different type of features, different information sources, and different probability estimator or acoustic models. For example, multiple acoustic models trained by Boosting algorithm can be merged at the frame-state level to produce a better likelihood estimation using the Multi-stream framework [Dimitrakakis and Bengio, 2004].

3.1.2 Boosting for Acoustic Modeling

Boosting algorithm has been used to address a number of speech recognition problems and has been shown to be successful in improving system performance. Representative applications include speaker verification [Foo and Lim, 2002; Li et al, 2003; Asami et al, 2005], confidence annotation [Moreno et al, 2001], call routing [Rochery et al, 2002; Zitouni et al, 2000; Tur et al, 2004], speech detection [Xiong and Huang, 2002; Kwon and Lee, 2003], speech segmentation [Wang et al, 2003], emotion detection [Lisacombe et al, 2005], intent classification [Tur, 2005], spoke language generation [Walker et al, 2003; Mairesse and Walker, 2005], etc.. However, the complexity of these applications does not exceed the level of standard multi-class classification.

The Boosting algorithm was also applied to LVCSR with respect to the characteristics of continuous speech. [Cook and Robinson, 1996; Cook et al, 1997; Schwenk, 1999] used Boosting algorithm to improve the performance of Hybrid HMM/Neural Network based speech recognizers. [Zweig and Padmanabhan, 2000; Meyer, 2002] developed practical utterance level Boosting training schemes that enable the technique to be used in large scale speech recognition task. [Zhang and Rudnicky, 2004a] extended Boosting training to the frame level, as an attempt to reduce word error rate instead of sentence error rate. [Dimitrakakis and Bengio, 2004; Dimitrakakis and Bengio, 2005] investigated Multi-Stream model as the platform to combine acoustic models trained using Boosting algorithm. Substantial reductions of recognition error were achieved in these experiments.

It is instructive to compare Boosting with MCE (Minimum Classification Error), one of the most successful discriminative training methods, in order to understand the effectiveness of Boosting in speech recognition. Given a training set of \( N \) labeled examples \( \Psi = \{(x_i, y_i) | 1 \leq i \leq N \} \), where feature vector \( x_i \in \mathbb{R}^D \) and class label \( y_i \in Y = \{c_1, c_2, \ldots, c_M\} \), the loss function that the Boosting algorithm aims to minimize is as follows [Schapire and Singer, 1999; Collins et al, 2000] (other variants exist).

\[
L_{\text{Boosting}} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{M-1} \sum_{y \neq y_i} \exp(f(x_i, y) - f(x_i, y_i))
\]

(3-1)

One version of the loss function for MCE is defined in (3-2) [Juang and Katagiri, 1992; Schluter, 2000]:

\[
L_{\text{MCE}} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{1 + \exp\{f(x_i, y_i) - \frac{1}{M-1} \left[ \sum_{y \neq y_i} f(y_i, y) \right]^{\eta}\}}
\]

(3-2)

where \( \eta \) is a positive parameter that controls how the competing hypotheses are weighted. In (3-1) and (3-2), function \( f(x, y) \) defines a mapping \( \mathbb{R}^D \times Y \rightarrow [0,1] \) which provides probabilistic or confidence measure for a feature/label pair.

Comparison of loss functions \( L_{\text{Boosting}} \) and \( L_{\text{MCE}} \) shows that Boosting also belongs to the family of discriminative training methods. Specifically, the minimization of \( L_{\text{Boosting}} \) via certain optimization procedures will increase the probability of the desired class \( y_i \) being predicted, and meantime decrease the
probability of alternative classes \( y \neq y_i \) being predicted. Therefore, Boosting has the advantages of
discriminative methods, such as having the capability to enhance the separability between competing
classes, and outperforming MLE (Maximum Likelihood Estimation) in the situation that model assumption
isn’t accurate.

On the other hand, Boosting differs from MCE in the way that \( f(x, y) \) is generated. In MCE, \( f(x, y) \) is
realized by constructing and optimizing a single classification model. In contrast, Boosting attempts to
generate a set of models and combine their hypotheses to make a better prediction of \( f(x, y) \). Thus
Boosting has the strength of ensemble methods, and is more powerful and flexible for complicated
classification problems such as continuous speech recognition.

3.1.3 Miscellaneous Methods

Other ensemble methods were also developed for improving recognition performance. [Cook and Robinson,
1995] built multiple speaker dependent acoustic models via utterance clustering, and combined them for
speaker independent recognition. [Vergyri et al, 2000] investigated the feasibility of combining
multilingual acoustic models to address the problem that sufficient training data is not available for a target
information, e.g. phone-scale and syllable-scale information, into recognition by combining decoders with
different time windows. More recently, [Siohan et al, 2005] used the technique of Random Forest to build
multiple systems by injecting randomness into Decision Tree based state-tying procedure.

In practice, researchers have also proposed some simple but effective solutions. One well-known approach
is to build acoustic models based on the separation of gender (male or female), age (children or adult),
channel (telephone or cell phone), or dialect. For example, the CMU Communicator system adopted a
parallel decoding architecture such that the speech recognition module incorporates two independent
decoders, one for male speakers and another for female speakers. Each decoder has its own acoustic model,
but shares the same lexicon and language model. The two acoustic models are trained separately using
speech data collected from male speakers or female speakers. In recognition, the “better” hypothesis, with
the higher decoding score, is selected as the final hypothesis.

3.2 Combination

The combination methods used in continuous speech recognition can be placed into three classes: feature
combination, likelihood (or posterior) combination and hypothesis combination, in accordance with the
recognition stage at which these methods are performed.

![Feature Combination Diagram]

Figure 3.2 Feature Combination

3.2.1 Feature Combination

Feature combination is a pre-processing method performed before the start of recognition. In feature
combination, various acoustic features extracted from different information sources or calculated using
different front-end algorithms are concatenated into a larger single vector as the feature for training and decoding. The mechanism of this method is illustrated in Figure 3.2. Strictly speaking, feature combination is not an ensemble based method since it essentially works on the platform of single recognition model.

3.2.2 Likelihood / Posterior Combination

As shown in Figure 3.3, likelihood / posterior combination is performed within the recognition stage. In this class of methods, the probability for the occurrence of a particular acoustic event, denoted by $f(c; \mathbf{x})$, is estimated by combining the output of individual classifiers. In implementation, $f(c; \mathbf{x})$ can be either a likelihood estimation $P(\mathbf{x} | c)$ or a posterior probability estimation $P(c | \mathbf{x})$. There are two issues to be considered in order to apply this kind of methods to continuous speech recognition. The first is the selection of a sub-word unit that determines at what level the combination is performed. Common choices include state and phoneme. For example, in Multi-Band acoustic modeling, spectral features extracted from frequency sub-bands are processed by multi-layer perceptron (MLP) based classifiers to generate phonetic probabilities. These phonetic probabilities are then combined for subsequent classification [Sharma, 1999; Hagen et al, 2000]. The choice of a combination rule is another important issue which shows how to calculate $f(c; \mathbf{x})$ from $f_i(c; \mathbf{x}_i)$, the output of individual classifiers. Some commonly used rules are listed as follows [Kirchhoff and Bilmes, 2000; Kirchhoff et al, 2000; Zolnay et al, 2005]. In addition, more complicated combination techniques such as Neural Network can also be used for yielding better probabilistic estimate.

- **Product**
  \[
  f(c; \mathbf{x}) = \frac{\prod_{i=1}^{N} f_i^{w_i}(c; \mathbf{x}_i)}{Z} \quad (3-3)
  \]
  Where $w_i$ denotes the weight associated with $f_i(c; \mathbf{x}_i)$, and $Z$ is a normalization factor making $f(c; \mathbf{x})$ a probability function if necessary.

- **Sum**
  \[
  f(c; \mathbf{x}) = \sum_{i=1}^{N} w_i f_i(c; \mathbf{x}_i) \quad (3-4)
  \]

- **Max**
  \[
  f(c; \mathbf{x}) = \max_{i=1}^{N} f_i(c; \mathbf{x}_i) \quad (3-5)
  \]
3.2.3 Hypothesis Combination

The mechanism of hypothesis combination is illustrated in Figure 3.4. There are conflicting opinions with regard to the hypothesis combination performed when recognition has completed. On one hand, hypotheses at later stage of recognition are believed to be more robust because they carry wider temporal context. Thus, hypothesis combination can be more effective than early stage combination i.e. likelihood/posterior combination. However, this advantage is weakened from the observation that the correct hypothesis may have been pruned out before they reach the final state of search. Despite the debates, hypothesis combination is extensively used in speech recognition and demonstrates solid performance in many applications. Two representative methods, ROVER and word lattice combination, are discussed as follows.

3.2.3.1 ROVER

ROVER (Recognizer Output Voting Error Reduction) is a word-level combination approach developed at NIST that aims to yield reduced word error rate by exploiting differences in the nature of the errors made by multiple speech recognition systems [Fiscus, 1997; Schwenk and Gauvain, 2000]. Rover proceeds in two stages. First, the hypotheses from different recognizers are progressively aligned together to build a single composite word transition network (WTN) by using dynamic programming. Figure 3.5 shows an example of the procedure of constructing WTN for three hypotheses: H1: a b c d; H2: b z d e; H3: b c d e f. Null transition is denoted by symbol @. Once the network is generated, a voting scheme respecting frequency, word confidence and time information is adopted to select the words with highest number of votes as the new hypothesis. In this example, the final system output is b c d e.

A theoretic analysis of the effectiveness of ROVER was presented in [Goel and Byrne, 2000; Goel et al, 2000; Goel et al, 2004] in which ROVER is viewed as an effort to achieve Minimum Bayes Risk (MBR). Moreover, this research work shows that some other important post-processing techniques, such as consensus network [Mangu et al, 2000], can also be unified into the framework of MBR classification.

3.2.3.2 Word Lattice Based Combination

[Singh et al, 2001] investigated a different methodology for hypothesis combination. The idea is to merge the word hypotheses obtained from various recognition systems into a new word graph, and then search the best path from it as the final decoding output. Initially, each word in each of the hypotheses is represented by a node in the graph. The acoustic score of the node is set to that associated with the original word. In the next step, all nodes representing identical words hypothesized between the same time instants are collapsed into a single node. Finally, links are formed between all node pairs where the word-end time of one node and the word-begin time of the next node are within 30 ms of each other. After the word graph is constructed in this manner, a standard language model is used to score the paths through the graph and the best path is obtained as the final hypothesis.
The weakness of ROVER and the Singh’s technique is that they are restricted to the use of a single best hypothesis of different recognizer. In many situations, the correct words do not appear in that single hypothesis. Consequently, they are unable to be selected for combination. Motivated by the fact that word lattice can preserve more information than other output formats such as single hypothesis and N-best lists, [Li et al, 2002] extended the idea of [Singh et al, 2001] to the direct combination of word lattices. This scheme is also carried out in two stages. The word lattices generated from different ASR systems are first merged into a larger mixed lattice through operations of merging edges, creating new edges and renormalizing scores. Once this is done, searching algorithm, e.g. Viterbi or A*, is performed to seek the path with maximum cumulative score in the new mixed lattice as the combination result.

Step 1: Initialize Hypotheses into WTNs

```
<table>
<thead>
<tr>
<th>WTN-1</th>
<th>WTN-2</th>
<th>WTN-3</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>b</td>
<td>c</td>
</tr>
<tr>
<td></td>
<td>z</td>
<td>d</td>
</tr>
<tr>
<td></td>
<td></td>
<td>e</td>
</tr>
<tr>
<td></td>
<td></td>
<td>f</td>
</tr>
</tbody>
</table>
```

Step 2: Align WTN-2 to WTN-1

```
<table>
<thead>
<tr>
<th>WTN-1</th>
<th>WTN-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>b</td>
</tr>
<tr>
<td></td>
<td>z</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

Step 3: Composite WTN Made from WTN-1 and WTN-2

```
<table>
<thead>
<tr>
<th>WTN-1+2</th>
</tr>
</thead>
<tbody>
<tr>
<td>@</td>
</tr>
<tr>
<td>b</td>
</tr>
<tr>
<td>z</td>
</tr>
<tr>
<td>d</td>
</tr>
<tr>
<td>e</td>
</tr>
</tbody>
</table>
```

Step 4: Align WTN-3 to WTN-1+2 and Form Final WTN

```
<table>
<thead>
<tr>
<th>WTN-1+2+3</th>
</tr>
</thead>
<tbody>
<tr>
<td>@</td>
</tr>
<tr>
<td>b</td>
</tr>
<tr>
<td>z</td>
</tr>
<tr>
<td>d</td>
</tr>
<tr>
<td>e</td>
</tr>
<tr>
<td>@</td>
</tr>
<tr>
<td>@</td>
</tr>
</tbody>
</table>

Figure 3.5 ROVER: Hypotheses Alignment and WTN Construction
4 Preliminary Experiments in Ensemble Based Acoustic Model Training

This section presents some preliminary results from experiments investigating ensemble based methods, in particular the Boosting algorithm, for acoustic modeling and hypothesis generation. Experiments with utterance level Boosting training will be described in Section 4.1. A frame level training scheme aiming to reduce word error rate rather than sentence error rate will be proposed in Section 4.2. Section 4.3 will discuss the experiment using N-best list re-ranking and ROVER combination for hypothesis generation.

All the experiments make use of the CMU Sphinx system, either Sphinx II or Sphinx III, for acoustic model training and testing. The speech data used in these experiments were selected from the CMU Communicator corpus, an 8K-16bits telephone based speech dataset collected between April 1998 and November 2000 for research in travel domain dialog processing [Rudnicky et al, 1999]. Most of the methods and corresponding experimental results discussed in this section have been published in [Zhang and Rudnicky, 2003a; Zhang and Rudnicky, 2003b; Zhang and Rudnicky, 2004a; Zhang and Rudnicky, 2004b].

4.1 Utterance Level Boosting Training

This section describes on investigation into the effectiveness of Boosting algorithm in acoustic model training. The standard Boosting algorithm, as illustrated in Table 2.2, which was initially designed for binary or multi-class classification problem, does not consider the special requirements of continuous speech recognition. An utterance level training approach has been investigated by [Zweig and Padmanabhan, 2000; Meyer, 2002; Zhang and Rudnicky, 2003b] to address this problem.

Suppose we have a training set \( \Psi = \{ (x_i, y_i) \mid 1 \leq i \leq N \} \) for continuous speech recognition, where, \( x_i \) is the sequence of feature vectors for the \( i \)-th training utterance, while \( y_i \) is the corresponding transcript. The essence of Boosting style acoustic model training is to minimize the value of the following loss function which is strongly related to sentence level recognition error.

\[
L = \sum_{i=1}^{N} \sum_{h \neq y_i} \exp[P_\Lambda(h \mid x_i) - P_\Lambda(y_i \mid x_i)]
\]  \hspace{1cm} (4-1)

where \( h \) denotes possible sentence hypothesis for input speech, \( \Lambda \) denotes the recognition model or ensemble to be learned, and \( P_\Lambda(h \mid x) \) denotes the posterior probability for a hypothesis. Table 4.1 illustrates the utterance level Boosting algorithm, which can be interpreted as a process to minimize the value of (4-1) by iteratively constructing an ensemble of \( K \) acoustic models.

A couple of obstacles need be addressed before we can apply this algorithm to acoustic modeling. First, for continuous speech recognition, the number of possible hypotheses for an utterance could be infinite. Assume that the recognizer has 5,000 words in its vocabulary, and that the maximum length for an utterance is restricted to 20 words. Theoretically, without taking into account the segmentation information associated with each word, the recognizer could output up to about 5000\(^{20}\) different hypotheses. Clearly, such a huge number makes it impossible for the recognizer to traverse all of the classes. To solve this problem, we have to compress the hypothesis space into a subset of limited size. In our experiments, the hypothesis set \( H_\Lambda \) contains only the hypotheses in the N-best list.

Second, the Boosting algorithm requires a probabilistic estimate of \( P_\Lambda(h \mid x) \) for each class, while most speech recognizers only output the log-likelihood scores provided by the acoustic and language models, whose value range is too large to be qualified as a good candidate for implementation. We therefore use the following scheme for converting the likelihood scores into posterior probability.

\[
P_\Lambda(h \mid x) = \frac{1}{\sum_{h'} \exp[P_\Lambda(h' \mid x)]}
\]
\[ P_A(h \mid x) = \frac{P_A(h, x)}{P_A(x)} = \frac{P_A(h, x)^\beta}{\sum_{h' \in \text{N-best list of } x} P_A(h', x)^\beta} \approx \frac{[P_A(h)P_A(x \mid h)]^\beta}{\sum_{h' \in \text{N-best list of } x} [P_A(h')P_A(x \mid h')]^\beta} \]

\[ \approx \frac{\exp[\alpha \log P_A(h) + \log P_A(x \mid h)]^\beta}{\sum_{h' \in \text{N-best list of } x} \exp[\alpha \log P_A(h') + \log P_A(x \mid h')]^\beta} \]

\[ (4-2) \]

Where \( \log P_A(x \mid h) \) and \( \log P_A(h') \) are acoustic model scores and language model scores, respectively. \( \alpha \) is the language model weight, and \( \beta \) is a smoothing parameter whose value is empirically set to control how the hypotheses in N-best list are weighted. In the case that the correct transcripts does not exist in the N-best list, one can run forced alignment to get the log-likelihood score, or simply choose a small default value for it.

**Initialize:**
- Let \( \Psi_0 = \Psi \) where \( \Psi \) is the original training set.

For \( k = 1 \) to \( K \):
- Train a new acoustic model \( \Lambda_k \) from data set \( \Psi_{k-1} \).
- Generate hypothesis set \( H_x = \{h\} \) for each utterance \( x \in \Psi_{k-1} \) using \( \Lambda_k \), and compute posterior probability \( P_{\Lambda_k}(h \mid x) \) for each hypothesis \( h \in H_x \).
- Compute pseudo loss \( \epsilon_k \) that

\[ \epsilon_k = \frac{1}{2} \sum_{x \in \Psi_{k-1}} \sum_{h \in H_x \text{ and } y \neq x} \frac{1}{|H_x|} \left[ 1 - P_{\Lambda_k}(y \mid x) + P_{\Lambda_k}(h \mid x) \right] \]

Where \( y \) denotes the correct transcripts for utterance \( x \).
- Set \( c_k = \epsilon_k / (1 - \epsilon_k) \).
- Calculate new weight for each utterance \( x \in \Psi_{k-1} \) that

\[ w(x) = \sum_{h \in H_x \text{ and } y \neq x} c_k \frac{1}{|H_x|} [1 + P_{\Lambda_k}(y \mid x) - P_{\Lambda_k}(h \mid x)] \]

- Resample training data according to normalized \( w(x) \), forming a new training set \( \Psi_k \).

**In generalization:**
- The hypothesis to a new utterance \( x \) is determined by

\[ h^* = \arg \max_h \sum_{k=1}^K \frac{1}{c_k} P_{\Lambda_k}(h \mid x) \]

**Table 4.1 Utterance Level Boosting Algorithm for Acoustic Modeling**

**Experiments**

The utterance level Boosting algorithm was investigated using the CMU Communicator corpus. We used 31248 utterances for acoustic model training and 1689 utterances for testing. The number of acoustic models was set to 5 to manage the amount of computation. The experiments were performed using CMU Sphinx II system. Please note that as a semi-continuous HMM based system, the performance of Sphinx II is worse than that using Sphinx III system (as we will see in the next two sections), since the latter employs fully continuous HMMs to model the speech signal. The language model was pre-trained and was not changed during the iteration of Boosting training.
Table 4.2 presents the algorithm’s performance as a function of $K$, the number of acoustic models in the ensemble. Baseline is the word error rate at $K = 1$, the one realized by single model. The result is also illustrated in Figure 4.1.

<table>
<thead>
<tr>
<th>Ensemble Size</th>
<th>$K=1$</th>
<th>$K=2$</th>
<th>$K=3$</th>
<th>$K=4$</th>
<th>$K=5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Word Error Rate</td>
<td>33.31%</td>
<td>30.89%</td>
<td>29.97%</td>
<td>29.46%</td>
<td>28.77%</td>
</tr>
</tbody>
</table>

Table 4.2 Performance of Utterance Level Boosting Algorithm on CMU Communicator Data

Table 4.2 shows that Boosting algorithm demonstrates substantial improvements over the baseline. Word error rate is down to 28.77% from 33.31% when 5 acoustic models are generated and combined, which represents a 13.63% relative reduction on the recognition errors.

Figure 4.1 Performance of Boosting Algorithm

4.2 Frame Level Boosting Training

The experimental results reported in the previous section demonstrate the effectiveness of the Boosting algorithm in acoustic modeling. However, analysis shows that utterance level Boosting algorithm has two weaknesses that hurt its capability for handling continuous speech recognition. First, as illustrated in (4-1), the loss function of the utterance level Boosting algorithm is designed to describe sentence error, that is a sentence is judged as correct only when all the words within it are correct. It does not model word error, the most widely accepted metric to evaluate the performance of a speech recognizer. Specifically, the posterior probability $P_A(h|\mathbf{x})$ used in the loss function is defined to measure how likely that the sentence hypothesis $h$ as a whole is correct for input speech $\mathbf{x}$. From $P_A(h|\mathbf{x})$ only, we are unable to tell how many words in the hypothesis are misrecognized. Moreover, in many cases the hypothesis with higher posterior probability can contain more word level recognition errors than the one with lower posterior probability. Even though there is a strong correlation between sentence-level and word-level recognition errors, training with a criterion aiming to boost posterior probability for sentence may not necessarily result in optimal acoustic models with lower word error rate. Second, in the utterance level Boosting algorithm, re-sampling is performed on the whole utterance. This means all the words within the same utterance always have equal weights. However, intuitively, the misclassified words should be given more attention than others in the training of subsequent model. To address these two problems, we propose a frame level Boosting training scheme for acoustic modeling in which the loss function is constructed to measure frame based recognition errors, and re-sampling is applied to focus on the more confusable parts within an utterance, where the errors actually occur [Zhang and Rudnicky, 2004a].

Frame level posterior probability
We define a new metric, *frame level posterior probability*, to quantitatively measure how likely the hypothesis given to a particular frame by the existing acoustic model is correct. This metric enables us to focus on the hard-to-learn parts from within the utterance and increase their weights in the model training.

Let \( \mathbf{x} \) be the sequence of feature vectors for an utterance with \( T \) frames that \( \mathbf{x} = (x_1, x_2, \ldots, x_T) \). For a particular frame \( x_t \), we use \( \theta_u(t) \) to denote the possible hypothesis on it, and \( r_u(t) \) to denote the desired one, where the sub-index \( u \) denotes the unit chosen for the hypothesis. In continuous speech recognition, there are at least four choices for \( u \): sentence, word, phoneme and state. For example, \( \theta_u(t) \) and \( r_u(t) \) represent the hypothesized and desired word at frame \( t \), respectively. It bears noting that most speech corpora don’t provide segmentation information at the word, phoneme and state levels. This means that, except for the \( r_s(t) \) whose value could be obtained directly from transcripts, for all the other three units we need to perform an forced-alignment to determine the \( r_u(t) \), and in these cases the \( r_u(t) \) is only an approximation to the correct result. The definition of frame level posterior probability of \( \theta_u(t) \) being the hypothesis at frame \( t \) is given as follows.

\[
P_A(\theta_u(t) \mid \mathbf{x}) = \frac{P_A(\theta_u(t), \mathbf{x})}{P_A(\mathbf{x})} \approx \frac{\sum_{h \in H_x, \text{and } h_u(t) = \text{the hypothesized result of } h \text{ at frame } t} P_A(h, \mathbf{x})^\beta}{\sum_{h \in H_x} P_A(h', \mathbf{x})^\beta}
\]

where \( \beta \) is the empirically determined smoothing factor, and \( H_x \) denotes the search space of hypotheses for utterance \( \mathbf{x} \). \( H_x \) is constrained to N-best lists in our experiments.

**Loss function**

The loss function for the frame level Boosting training algorithm is defined as follows.

\[
L = \sum_{i=1}^N \sum_{t=1}^{T_i} \sum_{\theta_u(i,t) \neq r_u(i,t)} \exp[P_A(\theta_u(i,t) \mid \mathbf{x}_i) - P_A(r_u(i,t) \mid \mathbf{x}_i)]
\]

where \( T_i \) is the number of frames in utterance \( \mathbf{x}_i \). \( \theta_u(i,t) \) and \( r_u(i,t) \) denote the hypothesized result and desired output for frame \( t \) of \( \mathbf{x}_i \), respectively. In Eq. (4-4),

\[
L_{i,t} = \sum_{\theta_u(i,t) \neq r_u(i,t)} \exp[P_A(\theta_u(i,t) \mid \mathbf{x}_i) - P_A(r_u(i,t) \mid \mathbf{x}_i)]
\]

is called pseudo-loss for frame \( t \), and describes the degree of confusion of this frame for recognition. A high value of \( L_{i,t} \) indicates that the questioned frame is possibly misrecognized and its weight needs be increased in the subsequent model training. Clearly, minimizing the value of this loss function will help to realize that \( P_A(r_u(i,t) \mid \mathbf{x}_i) \gg P_A(\theta_u(i,t) \mid \mathbf{x}_i) \) for \( \theta_u(i,t) \neq r_u(i,t) \), and increase the accuracy of recognition.

Please note that different unit \( u \) can lead the loss function to measure different types of recognition error. For example, the loss function used by utterance level Boosting algorithm for modeling sentence error, as shown in Eq. (4-1), can be viewed as a special case of (4-4) by setting the unit \( u \) to sentence. According to the definition, \( \theta_s(i,t) \) is one of the sentence hypotheses in \( H_x \) of utterance \( \mathbf{x}_i \), while the \( r_s(i,t) \) is actually the correct transcripts \( y_i \). Thus we have,
Similarly, we can also set the unit $u$ to word or phoneme, and construct loss functions as well as training schemes that aim to reduce word errors or phone errors for continuous speech recognition.

**Frame level Boosting training scheme**

The frame level Boosting training scheme is presented in Table 4.3. Please note that in this algorithm the weight $w_{i,d}$ is associated with the frame $t$ of utterance $x_i$, rather than with the whole utterance. This leaves us with the question of how to resample the training data for acoustic model training. We use a very simple strategy to solve this problem, in the way that a new feature sequence is created for utterance $x_i$ by duplicating frame $x_{t,i}$ for $\lfloor w_{i,t} \rfloor$ times. It should be pointed out that this method is rather ad hoc and that further investigation is necessary to improve this re-sampling method.

**Initialize:**
- Let $\Psi = \Psi$ where $\Psi$ is the original training set.

**For $k = 1$ to $K$:**
- Train a new acoustic model $\Lambda_k$ from data set $\Psi_{k-1}$.
- Determine the value of $r_u(i,t)$ for each frame of each utterance $x_i \in \Psi_{k-1}$. Run forced-alignment if necessary.
- Generate hypothesis set $H_x = \{h\}$ for each utterance $x_i \in \Psi_{k-1}$ using $\Lambda_k$, and compute posterior probability $P_{\Lambda}(\theta_u(i,t) | x_i)$ for every possible $\theta_u(i,t)$ at frame $t$.
- Compute pseudo loss

$$\epsilon_k = \frac{1}{2 |\Psi_{k-1}|} \sum_{x_i \in \Psi_{k-1}} \frac{1}{T_i} \sum_{t=1}^{T_i} l_{i,t},$$

where

$$l_{i,t} = \sum_{\theta_u(i,t) \in H_{x_i}} \left[1 - P_{\Lambda_k}(r_u(i,t) | x_i) + P_{\Lambda_k}(\theta_u(i,t) | x_i)\right].$$

- Set $c_k = \epsilon_k / (1 - \epsilon_k)$.
- Calculate new weight for each frame $t$ of each utterance $x_i \in \Psi_{k-1}$ that

$$w_{i,t} = \frac{1}{|\theta_u(i,t)|} \sum_{\theta_u(i,t) \not\in H_{x_i}} \frac{1}{c_k^2} \left[1 - P_{\Lambda_k}(r_u(i,t) | x_i) - P_{\Lambda_k}(\theta_u(i,t) | x_i)\right].$$
- Resample training data according to normalized $w_{i,t}$, forming new training set $\Psi_k$.

Table 4.3 Frame Level Boosting algorithm for Acoustic Modeling

**Experiment**
We performed a comparative experiment to investigate the effectiveness of the frame level Boosting training scheme. As in Section 4.1, we used 31248 utterances for acoustic model training and 1689 utterances for testing, both of which were selected from CMU Communicator speech corpus. The current experiment used CMU Sphinx III system, instead of the Sphinx II system. Since Sphinx III is a fully continuous HMM based system, its performance is much better than that of Sphinx II which is based on semi-continuous HMMs. The number of acoustic models in ensemble is set to 5 due to computation concern. This experiment used ROVER, a word level hypothesis combination method, to generate the final hypothesis for ensemble. More experiments of ROVER will be discussed in next section.

Three training schemes, all of which can be derived from the frame level Boosting algorithm by setting \( u \) to a different unit, were compared in this experiment, including the conventional utterance level Boosting (\( u = \text{sentence} \)), frame based word Boosting (\( u = \text{word} \)), and frame based phone Boosting (\( u = \text{phone} \)). Table 4.4 presents their word error rate varying with the ensemble size. The baseline word error rate is 15.0%, realized by single acoustic model.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>( K=1 )</th>
<th>( K=2 )</th>
<th>( K=3 )</th>
<th>( K=4 )</th>
<th>( K=5 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Utterance Boosting</td>
<td>15.0%</td>
<td>13.9%</td>
<td>13.2%</td>
<td>12.8%</td>
<td>12.7%</td>
</tr>
<tr>
<td>Word Boosting</td>
<td>15.0%</td>
<td>14.4%</td>
<td>13.7%</td>
<td>13.4%</td>
<td>13.0%</td>
</tr>
<tr>
<td>Phone Boosting</td>
<td>15.0%</td>
<td>14.0%</td>
<td>13.1%</td>
<td>12.9%</td>
<td>12.4%</td>
</tr>
</tbody>
</table>

Table 4.4 Performance of Three Boosting Training Algorithms on CMU Communicator Data

From Table 4.4 we observed that the frame based training methods also achieved attractive performance compared with conventional utterance level Boosting algorithm. Word Boosting and phone Boosting realized 13.3% and 17.3% relative reductions on word error rate respectively, when combining 5 acoustic models. More encouragingly, the frame based phone Boosting appears to outperform the conventional utterance level Boosting and establishes the best result for this experiment.

However, as Table 4.4 shows, none of the three methods performed significantly better than others. Even though they have some promising properties, the frame based methods did not demonstrate much advantage over the utterance level Boosting training. We think this is mainly due to the lack of accurate segmentation information for word or phone within an utterance. So we have to run forced alignment or other methods based on an existing model to guess the more likely \( r_u(t) \) (\( u = \text{word or phone} \)) for each frame. Obviously, the performance of the frame level Boosting training algorithm highly depends on the correctness of the estimated \( r_u(t) \).

### 4.3 N-Best List Re-Ranking and Rover Combination

As we discussed before, the conventional utterance level Boosting algorithm focuses on reducing sentence level error rate rather than reducing word level error rate, the most commonly used performance metric in speech recognition. This weakness is mainly due to the unavailability of accurate word or phone segmentation information within a continuous utterance. Namely, the characteristics of continuous speech force us to accept and use a practical but suboptimal criterion for acoustic model training. Section 4.2 proposed a solution to this problem that modifies the loss function, making it describe word or phone errors. In this section, we approach this problem from a different perspective, by investigating post-processing techniques to improve the performance of ensembles [Zhang and Rudnicky, 2004b].

Conventionally, the Boosting style ensemble uses the following combination method to generate the final hypothesis for a questioned utterance \( x \) (please see Table 2.2 and Table 4.1 for the entire algorithm).

\[
    h^* = \arg \max_h \frac{1}{c_k} P_{\lambda_k}(h \mid x)
\]

(4-7)

where \( h \) is the hypothesis predicted from an individual acoustic model, i.e. the top-1 hypothesis in N-best list. (4-7) shows that the conventional combination method is essentially a sentence level hypothesis
selection process: the final hypothesis $h^*$ is selected from the set consisting of top-1 hypothesis of individual models by using weighted majority voting. We improve the conventional method from two aspects: N-best list re-ranking and ROVER combination.

Examination of N-best list reveals that the best hypothesis, the one with the lowest word error rate, is not always in top-1 position. This phenomenon is caused by many reasons, such as inaccurate acoustic and language models, unavailability of sufficient training data, and lack of good features. N-best list re-ranking is a post-processing technique that attempts to locate the hypothesis with lowest word errors rather than to accept the top-1 result blindly, and its effectiveness has been proved by many independent experiments. In our experiment, we use Neural Network as the re-ranking method. The inputs are four features we have investigated previously, LM-Backoff-Mode, Utterance level posterior probability, Word level posterior probability and Frame level posterior probability [Zhang and Rudnicky, 2001], while the output is trained to approximate the word accuracy of each hypothesis. After re-ranking, the hypothesis with the highest estimated word accuracy will be chosen as the best hypothesis for combination.

The conventional method uses sentence level majority voting to combine hypotheses from each acoustic models. This method ignores some important information associated with individual word in the hypothesis, such as confidence and segmentation. ROVER is a successful method for realizing word level hypothesis combination. The hypotheses from different acoustic models or recognizers are first combined into a single word transition network (WTN) by using dynamic programming alignment. Once the network is generated, a voting scheme respecting frequency, confidence and time information is used to seek the best scoring word sequence. This is different from majority voting adopted by the conventional method that only selects the most probable result from the existing set of top-1 hypotheses: ROVER can create a new hypothesis by merging two or more hypotheses. ROVER was initially intended to reduce word error rate by exploiting the difference between outputs from multiple speech recognition systems representing different training and decoding approaches. However, we believe ROVER could also benefit Boosting training even though all the acoustic models are trained using the same technique, albeit with different views of the corpus.

Three experiments were designed to test the effectiveness of N-best list re-ranking and ROVER for Boosting style acoustic modeling. We used 31248 utterances for acoustic model training and 1689 utterances for testing, both of which were selected from the CMU Communicator speech corpus. These experiments were performed using the CMU Sphinx III system. The ensemble size was set to 4.

The first experiment was the training and test of multiple acoustic models using utterance level Boosting algorithm and the conventional combination method. The result obtained from this experiment was viewed as the baseline for following experiments. Table 4.5 shows the performance of Boosting training and combination as a function of the number of acoustic models in the ensemble. Please note that the result for $K=n$ means the word error rate obtained from hypotheses combination of $n$ models. As we reported before, cooperation of multiple models outperforms single model on recognition accuracy. When we use 4 acoustic models, the word error rate was down to 13.27% from 14.99%, the number for one model, which represents a 11.47% relative reduction.

<table>
<thead>
<tr>
<th>Ensemble Size</th>
<th>$K=1$</th>
<th>$K=2$</th>
<th>$K=3$</th>
<th>$K=4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Utterance Boosting</td>
<td>14.99%</td>
<td>13.54%</td>
<td>13.31%</td>
<td>13.27%</td>
</tr>
</tbody>
</table>

Table 4.5 Performance of Boosting Training and Combination

Neural Network based N-best list re-ranking was tested in the second experiment, in which hypothesis combination was applied to the new top-1 hypothesis of reordered N-best list. Table 4.6 presents the experimental result. Comparing Table 4.6 with Table 4.5, we can find that the re-ranking realized consistent improvement in system performance. This demonstrates the effectiveness of the Neural Network based re-ranking method, as well as that of the features we adopted in this experiment.

<table>
<thead>
<tr>
<th>Ensemble Size</th>
<th>$K=1$</th>
<th>$K=2$</th>
<th>$K=3$</th>
<th>$K=4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>N-best list Re-ranking</td>
<td>14.62%</td>
<td>13.14%</td>
<td>13.03%</td>
<td>12.98%</td>
</tr>
</tbody>
</table>

Table 4.6 Performance of Boosting Training + N-Best List Re-Ranking
On the basis of N-best list re-ranking, we further investigated the performance of ROVER combination. Table 4.7 shows the result. The results demonstrate that ROVER outperforms conventional sentence level method in combining multiple hypotheses and strongly support the viewpoint that word level combination is more suitable for continuous speech recognition. The final result was very encouraging. When we used 4 acoustic models, along with the N-best list re-ranking and ROVER, the word error rate is reduced to 12.52% which represents 16.48% relative reduction compared to the performance of a single model. One phenomenon deserving more attention is that when $K=2$ the word error rate achieved by ROVER is higher than that given in Table 4.6. We think this can be explained by the characteristic of ROVER which essentially intends to select word with high confidence score and frequency. In the case of two hypothesis combination, the frequency information isn’t very helpful. Therefore, if the confidence measure isn’t accurate, ROVER may be unable to generate a correct result.

<table>
<thead>
<tr>
<th>Ensemble Size</th>
<th>$K=1$</th>
<th>$K=2$</th>
<th>$K=3$</th>
<th>$K=4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Re-ranking + ROVER</td>
<td>14.62%</td>
<td>13.32%</td>
<td>12.70%</td>
<td>12.52%</td>
</tr>
</tbody>
</table>

Table 4.7 Performance of Boosting Training + N-Best List Re-Ranking + ROVER

The results of the three experiments are also illustrated in Figure 4.2.
5 Open Questions

The preliminary experimental results reported in previous section demonstrate that an ensemble method, e.g. Boosting algorithm, is a promising approach to acoustic model training. Nevertheless, there are some open questions to be addressed in order to make this kind of approach more effective and practical in fulfilling the special requirements of continuous speech recognition.

5.1 Investigation on Training Criteria for Reducing Word Error Rate

Our analysis in the previous section shows that there is a considerable mismatch between the training criteria of the conventional utterance level Boosting algorithm and the goal of acoustic modeling: the former is designed to minimize sentence error rate, while the latter aims to build a recognizer with low word error rate. We have proposed a frame level Boosting training algorithm that can partly address this problem. However, this new algorithm has its own weaknesses, such as it has to estimate the desired class $r_w(t)$ for each frame. Such information is unavailable for most speech corpora. The frame-level algorithm will remain as an important topic in our dissertation research.

Investigations on other ensemble construction methods, e.g. Bagging and Random Forests, are also scheduled in our research plan. These methods, which were proposed for solving general classification problem, don’t concern the characteristic of consecutive speech. There will be some interesting issues to be considered to apply them into continuous speech recognition. Moreover, we will also investigate the ensemble methods used by speech community, e.g. Multi-Band model, which can be interpreted as approaches to construct ensemble by splitting the feature space. One open question is to investigate if we can combine the two families of ensemble methods, data manipulation method represented by Boosting algorithm and feature manipulation method represented by Multi-Band model, into a unified framework targeting at reducing word error rate.

5.2 Investigation on Combination Methods

The preliminary experiments show that a good hypothesis combination method can significantly improve recognition accuracy. For example, 16.48% relative reduction on word error rate was obtained by using N-best list re-ranking and ROVER together (see Chapter 4.3 for detail). These encouraging results motivate us to continue work on this direction for further improvement. Research issues include:

- Investigations of different candidate of combination. In our preliminary experiments, the combination is performed to the best single hypothesis selected from N-best list. A straightforward extension is to exploit the entire N-best list for combination. Other candidates include word lattice and consensus network. Word lattice is the super-set of N-best list, which owns more decoding information than the latter. Consensus network is a compressed and time-aligned representation for word lattice with the advantage that the correct words are much easier to be identified [Mangu et al., 2000].

- Investigations of proper order of combination. It has been known that the pair-wise alignment procedure used by ROVER is to some extent affected by the order of combination. In our preliminary experiments, the hypotheses were combined in the order of their corresponding acoustic models being generated by Boosting training. Other choices exist. For example, hypotheses can be combined in the order of increasing word error rate of their recognizers. We can also borrow the method used by multiple DNA sequence alignment that iteratively selects the pair of sequences with least distance for combination [Thompson et al., 1994].

- Investigations of the appropriate decoding stage for combination being performed. Both N-best list and word lattice based combination are post-processing methods which are performed when the whole decoding process is completed. This means that the hypotheses that have been pruned from the Viterbi search are unable to be considered for combination. There are evidences showing that model combination, i.e. likelihood/posterior combination, performed within the search stage is superior to the post-processing methods. These types of method will be investigated in our research.
5.3 Investigation of Ensemble Based Semi-Supervised Acoustic Model Training

Traditional acoustic model training assumes the availability of large amount of manually transcribed training data. However, transcribing speech data is an expensive and time-consuming process. For example, in our meeting recognition research, it usually takes a skilled transcriber more than one week to generate and double-check the transcripts of a one-and-half hour meeting. On the other hand, massive amounts of unlabeled raw data are relatively easy to collect. An automatic learning procedure that can use both labeled and unlabeled speech data for acoustic model training is therefore of great interest. This kind of approach is referred as semi-supervised learning since it still needs a small faction of labeled data for the training of an initial seed model.

Semi-supervised acoustic model training has elicited wide interests in the speech community [Kemp and Waibel, 1999; Kamm and Meyer, 2001; Wessel and Ney, 2001; Lamel et al, 2002; Visweswariah et al, 2004; Zhang et al, 2005]. Most of the proposed approaches focus on the optimization of a single model. Ensemble methods, e.g. Boosting algorithm have been applied to solve semi-supervised learning problem by machine learning researchers, and appears to outperform other approaches in some question sets [Buc et al, 2001; Bennett et al, 2002]. Their results strongly suggest that ensemble based method can benefit semi-supervised acoustic model training. In the proposed dissertation research, we will investigate this problem and propose our solution for it.
6 Expected Contributions

Our expected contributions from the research of the open questions include:

- A set of improved ensemble based learning algorithms for acoustic model training, that differ from conventional discriminative training and utterance level Boosting training in the target of reducing word error rate, instead of sentence error rate.

- Novel methods that allow the combination of hypotheses to be performed on different levels, e.g. state, phone, word or utterance, at different decoding stages, e.g. within Viterbi search or after Viterbi search, to different objects, e.g. N-best list, word lattice or consensus network, and with different order, e.g. least distance or increasing recognition error.

- New semi-supervised learning algorithms for ensemble based acoustic model training that can improve recognition performance by exploiting un-transcribed speech data.

- Packaging of research results into self-contained toolkits that can be made publicly available.
7 **Timeline**

- April 2006 to December 2006, Research on open questions.

Please see Table 7.1 for detail.

<table>
<thead>
<tr>
<th>Research Activity</th>
<th>04/06</th>
<th>05/06</th>
<th>06/06</th>
<th>07/06</th>
<th>08/06</th>
<th>09/06</th>
<th>10/06</th>
<th>11/06</th>
<th>12/06</th>
<th>01/07</th>
<th>02/07</th>
<th>03/07</th>
<th>04/07</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supervised Training</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Criteria</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Combination Methods</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Semi-Supervised Training</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Thesis Write Up</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 7.1 Research Timeline
8 References


