Analysis and Modeling of Non-Native Speech for Automatic Speech Recognition

by

Karen Livescu

A.B., Princeton University (1996)

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

Master of Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

August 1999

[September 1999]

Author ....
Department of Electrical Engineering and Computer Science
August 13, 1999

Certified by ....
nes R. Glass
Principal Research Scientist
Thesis Supervisor

Accepted by ....
Arthur C. Smith
Chairman, Departmental Committee on Graduate Students

Analysis and Modeling of Non-Native Speech for Automatic Speech Recognition

by

Karen Livescu

Submitted to the Department of Electrical Engineering and Computer Science on August 13, 1999, in partial fulfillment of the requirements for the degree of

Master of Science

Abstract

The performance of automatic speech recognizers has been observed to be dramatically worse for speakers with non-native accents than for native speakers. This poses a problem for many speech recognition systems, which need to handle both native and non-native speech. The problem is further complicated by the large number of non-native accents, which makes modeling separate accents difficult, as well as the small amount of non-native speech that is often available for training. Previous work has attempted to address this issue by building accent-specific acoustic and pronunciation models or by adapting acoustic models to a particular non-native speaker.

In this thesis, we examine the problem of non-native speech in a speaker-independent, large-vocabulary, spontaneous speech recognition system for American English, in which a large amount of native training data and a relatively small amount of non-native data are available. We investigate some of the major differences between native and non-native speech and attempt to modify the recognizer to better model the characteristics of non-native data. This work is performed using the SUMMIT speech recognition system in the JUPITER weather information domain.

We first examine the modification of acoustic models for recognition of non-native speech. We show that interpolating native and non-native models reduces the word error rate on a non-native test set by 8.1% relative to a baseline recognizer using models trained on pooled native and non-native data (a reduction from 20.9% to 19.2%). In the area of lexical modeling, we describe a small study of native and non-native pronunciation using manual transcriptions and outline some of the main differences between them. We then attempt to model non-native word pronunciation patterns by applying phonetic substitutions, deletions, and insertions to the pronunciations in the lexicon. The probabilities of these phonetic confusions are estimated from non-native training data by aligning automatically-generated phonetic transcriptions with the baseline lexicon. Using this approach, we obtain a relative reduction of 10.0% in word error rate over the baseline recognizer on the non-native test set. Using both phonetic confusions and interpolated acoustic models, we further reduce the word error rate to 12.4% below baseline. Finally, we describe a study of language model differences between native and non-native speakers in the JUPITER domain. We find that, within the resolution of our analysis, language model differences do not account for a significant part of the degradation in recognition performance between native and non-native test speakers.

Thesis Supervisor: James R. Glass
Title: Principal Research Scientist

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The study of automatic recognition of non-native speech is motivated by several factors. First, any deployed speech recognizer must be able to handle all of the input speech with which it is presented, and in many cases, this includes non-native as well as native speech. Recognition accuracy has been observed to be drastically lower for non-native speakers of the target language than for native ones [12, 29, 31]. Research on both non-native accent modeling and dialect-specific modeling, a closely related issue, shows that large gains in performance can be achieved when the acoustics [3, 8, 22, 31] and pronunciation [15, 21, 29] of a new accent or dialect are taken into account. A motivation for studying non-native speech in particular, as opposed to dialect, is that non-native speech has been less widely studied and has some specific properties [16] that appear to be fundamentally different from those of dialectal speech. From a speech recognition perspective, non-native accents are more problematic than dialects because there is a larger number of non-native accents for any given language and because the variation among speakers of the same non-native accent is potentially much greater than among speakers of the same dialect.

The problem of recognition of non-native speech is one of mismatch between the recognizer's expectations and the test data. This includes two kinds of mismatch. The first is mismatch between models automatically determined from training data and the characteristics of the test data. The second is mismatch between knowledge-based, hand-made or partially hand-made models (for example, for word pronunciation) and the corresponding aspects of the test data. It has been previously observed that the performance of a speech recognizer varies with the degree of similarity between the data on which its models have been trained and the data on which it is tested. For example, in the case of the JUPITER conversational system (described in Chapter 2), the training data for the deployed system consists largely of native or lightly accented adult male speakers of American English; as a result, performance is significantly worse for female speakers, children, and non-native speakers with heavy accents [12]. A similar effect has been noted for recognizers trained on one dialect and tested on another dialect of the same language [8, 15].

1.1 Previous Work

Perhaps the simplest way to address the problem of training/testing mismatch is to pool the native training data with any available non-native training data. This approach treats native and non-native data as equivalent. As we show in Chapter 3, this can improve recognition of non-native speech relative to acoustic models trained on native data and even upon that of models trained on only non-native data, presumably because the non-native training data alone are insufficient to train robust models. This does not eliminate the mismatch, but it ensures that the recognizer has seen at least some training data similar to the test data.

Another approach to the problem of mismatch is to use multiple models, with each model optimized for a particular accent or class of accents. An accent-independent recognizer can then be built by using an accent classifier to detect the accent of a test speaker and then recognizing with the appropriate models. This approach is most commonly applied to the acoustic model. If there are sufficient training data for each accent class, separate acoustic models can be trained on accent-specific data. This method has been applied with some success to the recognition of both non-native accents [29] and dialectal accents [3]. In [29], Teixeira *et al.* show an improvement in isolated-word recognition over baseline British-trained models, using both whole-word and subword models and using either several accent-specific models or a single model for all non-native accents. Beattie *et al.* [3] compare gender- and dialect-dependent models to a baseline model trained on pooled multi-dialect data in a small-vocabulary (about 200-word) continuous speech recognition task for American English. In this work, the gender-dependent and gender- and dialect-dependent models achieved higher recognition accuracy than the pooled.

Very often, however, it is difficult to obtain sufficient data to robustly train the accent-specific models. In these cases, model adaptation is often used. In this approach, a single set of initial models is trained on a large set of data, but some transformation of the models is performed based on a smaller set of adaptation data from a single speaker or speaker category, determined by characteristics such as gender or accent. The adaptation can be either supervised, in which a set of previously transcribed data is used, or unsupervised, in which the recognizer's output is used to transform the models. This approach has been used, for example, by Diakoloukas *et al.* [8] to build a set of dialect-specific models for a new dialect of Swedish by adapting a set of existing models for another dialect with various amounts of development data. In this work, the adapted models are shown to outperform both the original unadapted models and a set of models trained only on data from the new dialect. Adaptation has also been used successfully to build speaker-dependent models for non-native speakers, using both supervised [3, 22, 31] and unsupervised [22, 31] approaches.

In the area of lexical modeling, several attempts have been made to better account for non-native word pronunciations. Liu and Fung [21], for example, obtain an improvement in recognition accuracy on English speakers with a Hong Kong accent by expanding a native lexicon using phonological rules based on knowledge of the speakers' native Cantonese. Teixeira *et al.* [29] use a data-driven approach to customize word pronunciation to specific non-native accents, using the Baum-Welch algorithm to retrain expanded hidden-Markov models representing the allowable pronunciations. Similar approaches have also been used for dialect-specific recognition. Humphries *et al.* [15], for example, use a tree clustering method to train probabilistic phonological rules for one British dialect and apply these rules to an existing lexicon designed for another dialect.

1.2 Goals and Overview

In the previous work we have mentioned, most of the approaches have involved modeling either a particular accent or a particular speaker. In this thesis, we investigate speaker-independent recognition, with no speaker adaptation. For a system dealing with non-native speakers from many diverse backgrounds, however, an approach using accent-specific models has two major disadvantages. First, in order to effectively account for all of the different accents of the test speakers, the number of models must be very large. This means both that the models are more computationally cumbersome and that a much larger set of training data is needed to adequately cover all of the accent categories. This problem is more marked for non-native accents than for dialectal accents because, typically, there are many fewer dialects of a given language than there are non-native accents. Furthermore, whereas dialects appear to be fairly stable and predictable, there can be marked differences between non-native speakers with the same accent. These differences are due to different levels of familiarity with the target language, as well as individual tendencies in mapping unfamiliar sounds and linguistic rules.

For these reasons, we are interested in the insights and performance gains we can obtain by analyzing and modeling all non-native speakers as a single group. Specifically, the main goals of this thesis are:

- To understand, in a qualitative and quantitative way, some of the major differences between native and non-native speakers of American English and the ways in which they affect the performance of a speech recognizer
- To investigate modifications to a speech recognizer geared toward improving its performance on non-native speakers

From the point of view of improving recognition performance, the main question we would like to address is: When faced with a domain in which we have a large amount of native data and a small amount of varied non-native data, how can we best use the limited non-native data to improve the recognizer's performance as much as possible? Although we occasionally examine the effects of our methods on the recognition of native speech, we do not attempt to build an accent-independent speech recognition system, that is, one that can handle both native and non-native speakers. Our main concern is to improve the recognition of non-native speech.

Chapter 2 describes the domain in which the work was performed and the baseline speech recognizer. The remainder of the thesis is divided into three parts, corresponding roughly to the three main components of a speech recognition system: Acoustic modeling, lexical modeling, and language modeling. Chapter 3 discusses experiments performed on the acoustic models. Chapters 4 and 5 discuss issues in lexical modeling: Chapter 4 describes a study of pronunciation patterns using manual transcriptions of native and non-native data, and Chapter 5 describes a method for automatic acquisition of such patterns and incorporation of this knowledge into the recognizer. Chapter 6 describes measurements showing the effect of language model differences between native and non-native speakers on the performance of the recognizer. Chapter 7 summarizes the thesis and suggests directions for future work.

Chapter 2

The JUPITER Domain and SUMMIT Recognizer

All of the work described in this thesis was performed in the JUPITER weather domain using the SUMMIT speech recognition system. The following sections describe the JUPITER conversational system and the associated database; the SUMMIT recognizer; and the baseline recognizer used for most of the experiments in the following chapters.

2.1 JUPITER

JUPITER is a telephone-based conversational system providing weather-related information for approximately 500 cities worldwide [33, 12, 11]. Using a toll-free number, users call JUPITER and inquire about various aspects of current weather forecasts, such as temperature, humidity, wind speed, precipitation, sunrise and sunset times, and weather advisories. In addition, users can ask JUPITER for help or clarification in various ways, such as "What cities do you know in New Jersey?", "What can I ask?", or "Repeat that please." During a call, JUPITER keeps a history of dialogue-specific information, so that questions such as "What about tomorrow?" can be answered appropriately, depending on the city and type of information that have been established in previous queries. Table 2-1 shows an example of a successful JUPITER conversation.

2.1.1 The JUPITER Corpus

Calls made to JUPITER are stored in a database, or corpus, for training and system development. The corpus used in this thesis contains 62,324 utterances collected between February 1997 and July 1998. Most of the utterances were collected live via the toll-free number. The corpus also contains approximately 3,500 read utterances, as well as about 1,000 utterances obtained through wizard data collection, in which the input is spontaneous but a

User: I'd like the weather in New York.
JUPITER: I know of the following places in New York: Albany, Binghamton, Buffalo, Elmira, Ithaca, New York City, Niagara Falls, Rochester, Syracuse, Utica, and Watertown. Please select one.
User: Albany.
JUPITER: In Albany today: Patchy fog early. Very warm and hazy and humid. Scattered showers early this morning. A little sunshine midday with chance of more showers and thunderstorms in the later afternoon. High upper 80s. 30 percent chance of rain. Tonight: Showers likely and possibly thunderstorms. Lows upper 60s. 60 percent chance of rain. Is there something else I can do for you?
User: Tomorrow's weather.
JUPITER: In Albany Friday: 30 percent chance of shower morning. Then clouds mixing with increasing sunshine afternoon. Becoming less humid. High lower 80s. What else would you like to know?
User: Nothing thank you.
JUPITER: Thanks for using JUPITER! Have a nice day!

Table 2-1: Example of a conversation with JUPITER.

human typist replaces the computer [12]. In our experiments, we used only the live data for training and testing. For each utterance, the corpus contains an orthographic transcription produced by a human transcriber.

In order to study native and non-native speech separately, we tagged each utterance in the corpus with an accent class label. We used the following accent classes: `amer` for native-sounding speakers of American English; `uk_aus` for native-sounding speakers of another dialect of English such as British or Australian English; `other` for all non-native speakers of English; and `undef` to indicate that the accent is "undefined", which we define below. We classified the accents by listening to those utterances which were noted by the transcriber as being possibly non-native, British, or Australian. We tagged an utterance as `other` whenever we could discern a non-native accent, whether weak or strong.

We note the following details about the classification of accents. First, we treat the `uk_aus` accent class as neither native nor non-native. This is because, while these speakers are not native speakers of *American* English, they are native speakers of another dialect of English and are not attempting to approximate American English. Although the acoustics, pronunciation, syntax, and semantics of non-American English dialects are all to some extent different from those of American English, they are presumably both more predictable and more similar to those of American English than are those of non-native English. For these reasons, the `uk_aus` utterances were excluded from further analysis. We use the term "non-native" to describe only those speakers whose accent is labeled `other` in the corpus, and the term "native" to describe only the `amer` utterances.

Second, we assume that all non-native speakers in the corpus are attempting to speak *American* English. Although some of the speakers may have learned a different dialect

Set	Number of Utterances	In-Vocabulary
all	62,324	46,036
amer	50,704	36,325
uk_aus	463	392
other	5146	4339
undef	6011	4980

Table 2-2: Breakdown of the JUPITER corpus into accent classes and in-vocabulary versus out-of-vocabulary utterances.

such as British or Australian English, we assume that the differences between non-native approximations of different English dialects are no larger than, or qualitatively different from, the differences between various non-native approximations of American English. We therefore treat all non-native speakers as a single group in our analysis and experiments.

The `undef` accent was applied to all:

1. Read (non-spontaneous) utterances. We perform all analysis and experiments only on spontaneous speech; because of the time-consuming nature of manual accent classification, we classified only spontaneous utterances.

2. Utterances spoken in a language other than English.

3. Utterances for which the accent class could not be determined for any reason; for example, because the utterance was too short or noisy to discern the accent, the speaker seemed to be using an artificial accent, or the recording contained only non-speech sounds.

Table 2-2 shows the number of utterances in the corpus in each of the accent classes. The table also contains the number of *in-vocabulary* utterances in each accent class, which are those utterances that contain only words within the 1956-word JUPITER vocabulary.

2.1.2 Division of the Corpus into Training, Development, and Test Sets

We divided the native and non-native utterances in the corpus into training, development, and test sets. The training sets were used to train acoustic and language models, as well as lexicon arc weights and the phonetic confusion weights discussed in Chapter 5. The development sets are used to perform intermediate tests, during which various parameters may be optimized. The test sets are used to obtain the final results and are not tested on or analyzed prior to the final experiments. The utterances in a single call to JUPITER are always placed in the same set, so that we do not test on utterances from calls on which we have trained. However, since the same speaker may call more than once, and since we do not know the identity of the speaker(s) in each call, the development and test sets may contain some of the same speakers as the training sets.

Set	Number of Utterances
native-train	33,692
nn-train	2,717
pooled-train-iv	36,409
pooled-train	50,269
native-dev	730
nn-dev	609
pooled-dev	500
native-test	1649
nn-test	1013

Table 2-3: Number of utterances in the training, development, and test sets. The abbreviations used in this table are: nn = non-native; dev = development; iv = in-vocabulary. The `pooled-train-iv` set is the subset of `pooled-train` consisting of only the in-vocabulary utterances.

Table 2-3 shows the number of utterances in each of these sets. All of the sets except `pooled-train` contain in-vocabulary utterances only. The `pooled-train-iv` set is simply the union of `native-train` and `nn-train`. The `pooled-dev` set is a random 500-utterance subset of the union of `native-dev` and `nn-dev`.

2.2 SUMMIT

SUMMIT is a segment-based speech recognition system [9]. That is, it attempts to segment the waveform into predefined subword units, which may be phones, sequences of phones, or parts of phones. This is in contrast to frame-based recognizers, which divide the waveform into equal-length windows, or frames [2, 24]. Both approaches first extract acoustic features at small, regular intervals. However, segment-based recognition approaches next attempt to segment the waveform using these features, while frame-based recognizers use the frame-based features directly.

The problem that a continuous speech recognizer attempts to solve is: For a given input waveform with corresponding acoustic feature vectors $A = \{\vec{a}_1, \vec{a}_2, \ldots, \vec{a}_N\}$, what is the most likely string of words $\vec{w}^* = \{w_1, w_2, \ldots, w_M\}$ that produced the waveform? In other words,

$$\vec{w}^* = \arg\max_{\vec{w}} P(\vec{w}|A), \tag{2.1}$$

where $\vec{w}$ ranges over all possible word strings. Since a single word string may be realizable as different strings of units $\vec{u}$ with different segmentations s, this equation becomes:

$$\vec{w}^* = \arg\max_{\vec{w}} \sum_{\vec{u},s} P(\vec{w}, \vec{u}, s, |A), \tag{2.2}$$

where $\vec{u}$ ranges over all possible pronunciations of $\vec{w}$ and s ranges over all possible segmentations for all of the pronunciations. This summation can be computationally expensive. As a simplification, SUMMIT assumes that, given a word sequence $\vec{w}$, there is an optimal segmentation and unit sequence, which is much more likely than any other s and $\vec{u}$. The summation is therefore replaced by a maximization, in which we attempt to find the best triple of word string, unit sequence, and segmentation, or the best path, given the acoustic features:

$$\{\vec{w}^*, \vec{u}^*, s^*\} = \arg\max_{\vec{w},\vec{u},s} P(\vec{w}, \vec{u}, s, |A) \tag{2.3}$$

Applying Bayes' Rule several times, this can be rewritten as

$$\begin{aligned} \{\vec{w}^*, \vec{u}^*, s^*\} &= \arg\max_{\vec{w},\vec{u},s} \frac{P(A|\vec{w}, \vec{u}, s)P(s|\vec{u}, \vec{w})P(\vec{u}|\vec{w})P(\vec{w})}{P(A)} && (2.4) \\ &= \arg\max_{\vec{w},\vec{u},s} P(A|\vec{w}, \vec{u}, s)P(s|\vec{u}, \vec{w})P(\vec{u}|\vec{w})P(\vec{w}) && (2.5) \end{aligned}$$

The second equality arises from the fact that $P(A)$ is constant for all $\vec{w}$, $\vec{u}$, and s, and therefore does not affect the maximization.

This derivation describes the approach used in Viterbi decoding [2], and is identical for frame-based and segment-based recognizers. The difference lies in the fact that segment-based recognizers explicitly consider segment start and end times during the search, whereas frame-based methods do not. Segment-based methods can therefore require significantly more computation during the search.

The estimation of the four components of the last equation is performed by, respectively, the acoustic, duration, lexical (or pronunciation), and language model components. In this thesis, we do not use a duration model; that is, $P(s|\vec{u}, \vec{w})$ is constant. The following four sections describe the other components of the SUMMIT recognizer: segmentation, acoustic modeling, lexical modeling, and language modeling. We finally describe the methods SUMMIT uses to search for the best path and the implementation of SUMMIT using finite-state transducers.

2.2.1 Segmentation

One way to limit the size of the search space is to transform the input waveform into a constrained network of hypothesized segments. In SUMMIT, this is done by first extracting frame-based acoustic features (*e.g.*, MFCC's) at small, regular intervals, as in a frame-based recognizer, and then hypothesizing groupings of frames which define possible segments. There are various methods for performing this task; two that have been used in SUMMIT are acoustic segmentation [10, 19], in which segment boundaries are hypothesized at points of large acoustic change, and probabilistic segmentation [5, 19], which uses a frame-based phonetic recognizer to hypothesize boundaries. In this thesis, we use acoustic segmentation.

2.2.2 Acoustic Modeling

After segmentation, a vector of acoustic features is extracted for each segment or boundary in the segmentation graph, or for both the segments and boundaries in the graph. In this thesis, we use only the boundaries in the segment graph. Each hypothesized boundary in the graph may be an actual transition boundary between subword units, or an internal "boundary" within a phonetic unit. We refer to both of these as boundaries or diphones, although the latter kind does not correspond to an actual boundary between a pair of phones.

The acoustic features are now represented as a set of boundary feature vectors $Z = \{\vec{z}_1, \vec{z}_2, \ldots, \vec{z}_L\}$. SUMMIT assumes that boundaries are independent of each other and of the word string, so that the acoustic component in (2.5) becomes

$$P(Z|\vec{w}, \vec{u}, s) = \prod_{l=1}^{L} P(\vec{z}_l|\vec{u}, s) \tag{2.6}$$

We further assume that the conditional probability of each $\vec{z}_l$ is independent of any part of $\vec{u}$ and s other than those pertaining to the lth boundary, so that

$$P(Z|\vec{w}, \vec{u}, s) = \prod_{l=1}^{L} P(\vec{z}_l|b_l), \tag{2.7}$$

where b_l is the lth boundary label as defined by $\vec{u}$ and s. The vector function of probabilities of feature vectors $\vec{z}$ conditioned on boundary labels b_i from a set of R boundary labels,

$$\vec{P}(\vec{z}) = \begin{bmatrix} P(\vec{z}|b_1) \\ P(\vec{z}|b_2) \\ \vdots \\ P(\vec{z}|b_R) \end{bmatrix}, \tag{2.8}$$

is estimated by the acoustic model, or set of acoustic models if we wish to emphasize the fact that each b_i is modeled separately. During recognition, the acoustic model assigns to each boundary in the segmentation graph a vector of scores corresponding to the probability of observing the boundary's features given each of the b_i. In practice, however, not all of the scores may be generated for each boundary. Instead, the scores are computed on demand, generating scores only for the hypotheses that are being considered in the recognition search at a given time.

2.2.3 Lexical Modeling

Lexical modeling is represented by the lexicon, a dictionary of allowable pronunciations for all of the words in the recognizer's vocabulary [32]. The lexicon consists of one or more basic pronunciations, or baseforms, for each word, as well as any number of alternate pronunciations created by applying phonological rules to the baseform. The phonological rules account for processes such as place assimilation, gemination, and alveolar stop flapping.

The alternate pronunciations are represented as a graph. In addition, the arcs in the lexicon can be weighted to account for the different probabilities of the alternate pronunciations. The lexical probability component in (2.5) is therefore the probability of a given path in the lexicon.

2.2.4 Language Modeling

The language model provides an estimate of $P(\vec{w})$, the probability of observing the word sequence $\vec{w}$. One widely-used class of language models is n-grams [2]. The probability of an N-word string can be rewritten as

$$P(\vec{w}) = \prod_{i=1}^{N} P(w_i | w_{i-1}, w_{i-2}, \ldots, w_1) \tag{2.9}$$

n-grams assume that the probability of a given word depends on a finite number $n-1$ of preceding words:

$$P(\vec{w}) = \prod_{i=1}^{N} P(w_i | w_{i-1}, w_{i-2}, \ldots, w_{i-(n-1)}) \tag{2.10}$$

The probability estimates for n-gram language models are typically trained by counting the number of times each n-word sequence occurs in a set of training data. To deal with the problem of n-grams that do not occur in the training data, smoothing methods are often used to redistribute some of the probability from observed n-grams to unobserved n-grams [6]. In all of the experiments in this thesis, we use word n-grams with $n = 2$ or $n = 3$, referred to as bigrams and trigrams, respectively.

2.2.5 Searching for the Best Hypothesis

During recognition, the combined constraints of the scored segmentation graph, the lexicon, and the language model produce a weighted graph representing the search space. The recognizer must then find the best-scoring path through this graph. In the version of SUMMIT used in this thesis, the language model is applied in two passes. In the first pass, a forward Viterbi beam search [25] is used to obtain the partial-path scores to each node using a bigram language model, followed by a backward A^* beam search [30] using the scores from the Viterbi search as the look-ahead function. The A^* search produces a smaller word graph, which can then be searched with a more computationally demanding language model such as a trigram. In the second pass, the new language model is used to rescore the word graph with forward partial path scores, and a backward A^* search is again used to obtain the final hypotheses.

2.2.6 Recognition Using Finite-State Transducers

For most of our recognition experiments, we used an implementation of SUMMIT using weighted finite-state transducers, or FST's [11]. An FST recognizer R can be represented as the composition of four FST's,

$$R = A \circ D \circ L \circ G, \tag{2.11}$$

where: A is a segment graph with associated acoustic model scores; D performs the conversion from the diphone labels in the acoustic graph to the phone labels in the lexicon; L represents the lexicon; and G represents the language model. The process of recognition then becomes a search for the best path through R. SUMMIT allows the composition of D, L, and G to be performed either during or prior to recognition.

2.3 The Baseline Recognizer

This section describes the baseline recognizer used in all of the experiments other than the ones pertaining to the language model; see Chapter 6 for a description of the baseline recognizer used in the language model experiments.

2.3.1 Acoustic Models

The features used in the acoustic models are the first 14 Mel-frequency cepstral coefficients (MFCC's) averaged over 8 different regions near each boundary in the phonetic segmentation of each utterance [11]. These 112 features ($14\ MFCC's \times 8\ regions$) are reduced to a 50-dimensional vector using principal components analysis (PCA). The PCA transforms the resulting feature space, producing an identity covariance matrix for the training data [4]. The acoustic models are diagonal Gaussian mixture models with up to 50 mixture components per model, depending on the amount of training data available.

The procedure for training the models is as follows. In order to train new models, we need a time-aligned phonetic transcription of each of the training utterances. Since we only have manual word transcriptions for the utterances in the corpus, we create forced phonetic transcriptions or *forced paths*: We begin with a set of existing acoustic models, and we perform recognition on each training utterance using the known word transcription as the language model. In our case, the existing JUPITER acoustic models were trained on both native and non-native data, including some of the non-native data in our test set; however, we believe that the forced paths are sufficiently restricted by the pronunciations in the lexicon that the existing acoustic models do not significantly affect the final models that we train.

We next train the PCA coefficients. Finally, we train the mixture Gaussian models using K-means clustering [25] followed by estimation of each Gaussian's parameters via the expectation-maximization (EM) algorithm [7].

The acoustic models for the baseline recognizer was trained on the `pooled-train-iv` set. We will henceforth use the terms "pooled" and "baseline" interchangeably to refer to this set of acoustic models. The boundary labels for the acoustic models are based on a set of 62 phone labels (61 acoustic labels and a sentence-boundary marker); see Appendix A for a listing and description of the phone labels. From these phone labels, we compiled a set of 2,300 boundary labels, including both internal and transition boundaries. Although there are 3,906 possible boundary labels (62^2 transition boundaries and 62 internal boundaries), many of them do not occur in the lexicon and therefore need not be modeled. Since there were not a sufficient number of training tokens for all of the boundary labels, the labels were manually grouped by subjective similarity into classes. For each class, a single model was trained on the training tokens for all of the labels in the class. In order to be able to maintain the same label classes for subsequent training runs using only native or non-native training utterances, the classes were chosen in such a way that each class had at least 25 training tokens in both the `native-train` and `nn-train` sets. This resulted in 448 classes, a smaller set than the 715-class set currently used for JUPITER recognition with this label set [28]. Experiments conducted with both sets of classes show a small increase in recognition error rate when using the smaller class set. Specifically, for the `poooled-dev` set, the word error rate increased from 13.5% to 14.0%.

2.3.2 The Lexicon

For the baseline recognizer, we used a lexicon containing 1956 words. The arc probabilities in the lexicon were trained on the `pooled-train-iv` set. In order to account for pronunciations that do not occur at all in the training set, we smoothed the weights by adding a floor, or a minimum number of occurrences, to the count of occurrences of each possible pronunciation. The floor was optimized by trial and error, choosing the value yielding the lowest word error rate in a set of experiments on the `pooled-dev` set. Using this criterion, a floor of 0.1 was chosen.

2.3.3 The Language Model

The baseline recognizer uses a word bigram for the forward Viterbi search and a word trigram for the backward A^* search. For training of the n-grams, we used the `pooled-train` set, which contains both in-vocabulary and out-of-vocabulary utterances. All out-of-vocabulary words were treated as a single word during training. Smoothing is done using n-gram interpolation [6] using smoothing and floor counts of 2 for the bigram and 5 for the trigram. These values were chosen based on experiments on the `pooled-dev` set.

2.3.4 Baseline Performance

Table 2-4 shows the error rates obtained using the baseline recognizer on the native and non-native test sets. The error rates on the development sets are also shown, as these are used for comparison in subsequent chapters. In all cases, the beam width in the Viterbi

Test Set	S	I	D	WER
native-test	5.8	2.0	2.7	10.6
nn-test	12.4	3.9	4.6	20.9
native-dev	5.1	1.7	2.8	9.6
nn-dev	12.1	4.3	3.8	20.2

Table 2-4: Rates of substitution (S), insertion (I), and deletion (D) and word error rates (WER) obtained with the baseline recognizer on native and non-native sets. Each error rate is a percentage of the number of reference (actual) words in the test set.

search, measured in terms of the score difference between the best- and worst-scoring path considered, is 20. The word transition weight, a parameter controlling the tradeoff between deletions and insertions, is 4. We note the values of these parameters as they will be varied during the course of subsequent experiments.

Chapter 3

Acoustic Modeling

One respect in which native and non-native speakers may differ is in the acoustic features of their speech (*e.g.*, [1]). In other words, when intending to produce the same phone, native and non-native speakers may produce it with different formant frequencies (in the case of sonorants), voice onset times (for stops), durations, and so on. From a speech recognizer's point of view, this difference can affect the behavior of the acoustic models. In this chapter, we investigate the possibility of improving the recognizer's performance on non-native utterances by changing the way in which the acoustic models are trained.

To this end, we created several sets of acoustic models. Section 3.1 compares models trained on the native training set, models trained on the non-native training set, and the baseline models described in Chapter 2. Section 3.2 describes the creation of a combined model by interpolation of the native and non-native models and shows the results of using this model.

3.1 Native and Non-Native Acoustic Models

The native and non-native acoustic models were trained on the `native-train` and `nn-train` sets described in Chapter 2. The measurements, label classes, and training procedure were the same as those used to train the baseline models. Table 3-1 shows the performance of recognizers using the native, non-native, and pooled models. All of the recognizers are identical to the baseline recognizer described in Chapter 2, except for the different acoustic models.

As expected, the native models perform better on the native sets than they do on the non-native sets, and the non-native models perform better on the non-native sets than on the native set. The non-native models also perform better on the non-native sets than do the native models. This result is not a foregone conclusion, for two reasons. First, the native models would presumably perform better than the non-native models on non-native test data if the non-native models had sufficiently little training data. This result, therefore,

Test Set	Native Models				Non-Native Models				Pooled Models			
	S	I	D	WER	S	I	D	WER	S	I	D	WER
native-dev	5.0	1.6	2.7	9.3	12.9	3.7	5.3	21.8	5.1	1.7	2.8	9.6
nn-dev	13.8	4.9	4.3	23.0	12.6	4.8	3.0	20.4	12.1	4.3	3.8	20.2
native-test	5.4	1.8	2.5	9.7	14.9	3.9	5.1	23.9	5.8	2.0	2.7	10.6
nn-test	14.0	4.4	5.0	23.4	12.8	4.5	4.3	21.7	12.4	3.9	4.6	20.9

Table 3-1: Rates of substitution (S), insertion (I), and deletion (D) and word error rates (WER) for recognizers using the native, non-native, and pooled acoustic models. The sets are the native and non-native development sets (native-dev, and nn-dev, respectively).

provides some assurance that the non-native models are adequately trained. Second, since the non-native models are trained on speakers with many very different accents, it is possible that the resulting models would simply be too broad to accurately model the acoustics of the speech of any one non-native speaker. Therefore, this result also confirms that it is reasonable to model the acoustics of non-native speech with various accents using a single set of models.

The pooled models perform slightly worse on the native sets than do the native models. This indicates that including non-native data in the training set, at least in the proportion we have used, is detrimental to recognition of native speech. However, when testing on non-native speech, the pooled models perform slightly better than the non-native models. This suggests that, although non-native training data is better than native training data for recognition of non-native speech, including native data in the training can be helpful.

3.2 Model Interpolation

While the non-native models perform better on non-native data than do the native models, and the pooled models perform even better by a small amount, there is reason to believe that we could do better than either of these. The improvement we have seen from pooling the native and non-native training data indicates that the recognition of non-native speech can benefit from native training data. However, the pooled training set gives very little weight to the non-native training utterances, since the native utterances outnumber them by a factor of about 12. Intuitively, we would like to give more weight to the non-native training utterances than they receive in the pooled training. It would be undesirable to remove some of the native utterances from the pooled training set to achieve a better balance, since these add to the robustness of the models. One way to achieve the desired weighting is by interpolating the native and non-native models.

3.2.1 Definition

Interpolation of models refers to the weighted averaging of the PDF's of several models to produce a single output. Interpolation is closely related to model aggregation, the averaging of model PDF's using equal weights, which has been shown to improve the performance of a recognizer when applied to models from different training runs using the same data [13]. Model interpolation has also been used successfully to combine well-trained context-independent models with less well-trained but more detailed context-dependent models [14]. The case of native and non-native models is similar, in the sense that the native models are better trained but the non-native models are more appropriate for the test population.

Given an observation vector of acoustic features $\vec{x}$, a set of N models $\vec{P_1}(\vec{x}), \vec{P_2}(\vec{x}), \ldots, \vec{P_N}(\vec{x})$, and the corresponding weights $w_1, w_2, \ldots, w_N$, the interpolated model is defined as:

$$\vec{P_I}(\vec{x}) = \sum_{i=1}^{N} w_i \vec{P_i}(\vec{x}), \tag{3.1}$$

where

$$\sum_{i=1}^{N} w_i = 1 \tag{3.2}$$

In our case, there are only two models, $\vec{P}_{nat}(\vec{x})$ and $\vec{P}_{nn}(\vec{x})$, where the subscript *nat* refers to the native model and *nn* to the non-native one. The expression for the interpolated model is therefore:

$$\vec{P_I}(\vec{x}) = w_{nat}\vec{P}_{nat}(\vec{x}) + w_{nn}\vec{P}_{nn}(\vec{x}), \tag{3.3}$$

where

$$w_{nat} + w_{nn} = 1 \tag{3.4}$$

3.2.2 Experiments

We created a series of interpolated models by varying the relative weights assigned to the native and non-native models. Figure 3-1 shows the results of using these models in recognition experiments on the native and non-native development sets.

As expected, the WER's obtained with $w_{nn} = 0$ and $w_{nn} = 1$ are identical to those obtained with native and non-native models alone, respectively. Also as expected, the WER on `native-dev` displays almost monotonically increasing behavior as the non-native models are given increasingly more weight. The minimum WER, however, does not occur for $w_{nn} = 0$ but for $w_{nn} = 0.03$. With this weighting, the WER is 9.1%, which is a 2.1% relative error rate reduction (RERR) from the WER obtained with the native models. This may indicate that there is a benefit to interpolating the native and non-native models even for recognition of native speech, although it is not clear that the difference in this experiment is significant.

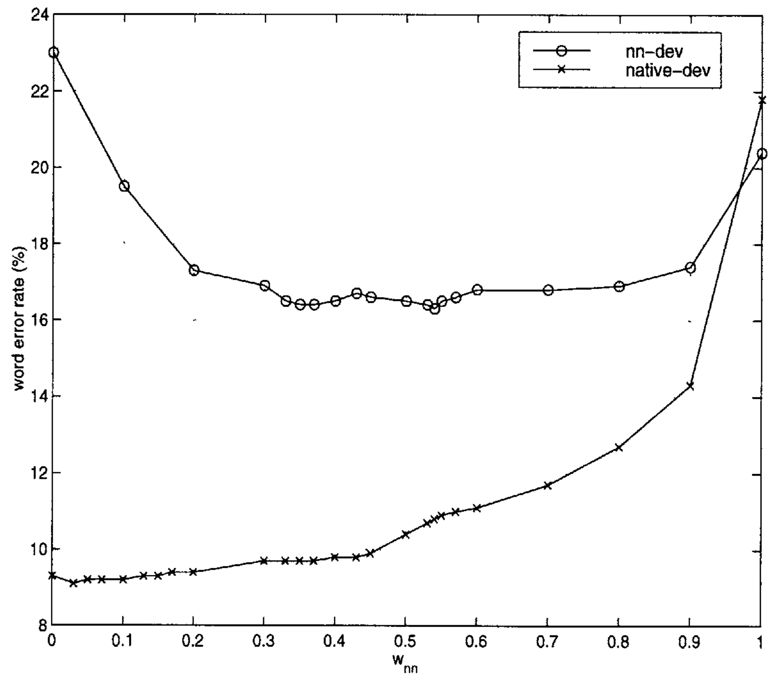

Figure 3-1: Word error rates obtained on native and non-native development sets using interpolated models with varying relative weights.

For nn-dev, the lowest WER, obtained with $w_{nn} = 0.54$, is 16.3%. This is a 20.0% RERR with respect to the non-native models and a 19.3% RERR with respect to the baseline models, indicating that the recognition of non-native speech benefits greatly from the interpolation of native and non-native models.

It is interesting to note that, when testing on nn-dev, almost *any* amount of interpolation is much better than using either of the individual models. The WER on nn-dev decreases drastically as we go from no interpolation ($w_{nn} = 0$ or 1) to interpolation with $w_{nn} = 0.2$ or 0.9, while varying much less within a large range of w_{nn} between those values. We also note that there appears to be a local minimum in the WER on nn-dev at $w_{nn} = 0.37$–0.35, in addition to the absolute minimum at $w_{nn} = 0.54$. It is not clear whether or not this can be attributed to statistical variation alone.

Tables 3-2 and 3-3 show the error rates obtained on the non-native and native test sets using interpolated models with $w_{nn} = 0.54$ and $w_{nn} = 0.03$, respectively. For nn-test, the WER using the interpolated models is 19.2%, an 8.1% improvement over the pooled models and an 11.5% improvement over the non-native models. For nat-test, using the interpolated models yields a WER of 9.6%, a 9.4% improvement over the pooled models and a 1.0% improvement over the native models.

In order to determine whether these results are the best that we could have obtained on the test sets, we tested the interpolated models on nn-test and nat-test with varying weights, as we did for the development sets. Figure 3-2 shows the results of these experi-

Pooled Models				Non-Native Models				Interpolated Models			
S	**I**	**D**	**WER**	**S**	**I**	**D**	**WER**	**S**	**I**	**D**	**WER**
12.4	3.9	4.6	20.9	12.8	4.5	4.3	21.7	11.5	3.6	4.1	19.2

Table 3-2: Comparison of error rates on the non-native test set obtained with the baseline models, non-native models, and interpolated models with $w_{nn} = 0.54$

Pooled Models				Native Models				Interpolated Models			
S	**I**	**D**	**WER**	**S**	**I**	**D**	**WER**	**S**	**I**	**D**	**WER**
5.8	2.0	2.7	10.6	5.4	1.8	2.5	9.7	5.3	1.8	2.5	9.6

Table 3-3: Comparison of error rates on the native test set obtained with the baseline models, native models, and interpolated models with $w_{nn} = 0.03$

ments. The lowest WER for `nn-test` is 18.9%, an improvement of 9.6% over baseline, and occurs at $w_{nn} = 0.57$. For `nat-test`, the lowest WER is 9.5%, an improvement of 2.1% over the native models alone, and occurs at $w_{nn} = 0.05$.

It is not clear why even the best-performing interpolated model for `nn-test` produces only about half the improvement that was obtained on `nn-dev`. This may be due to the large variability in non-native speech; that is, the utterances in the development set may happen to be better modeled by the acoustic models than those in the test set.

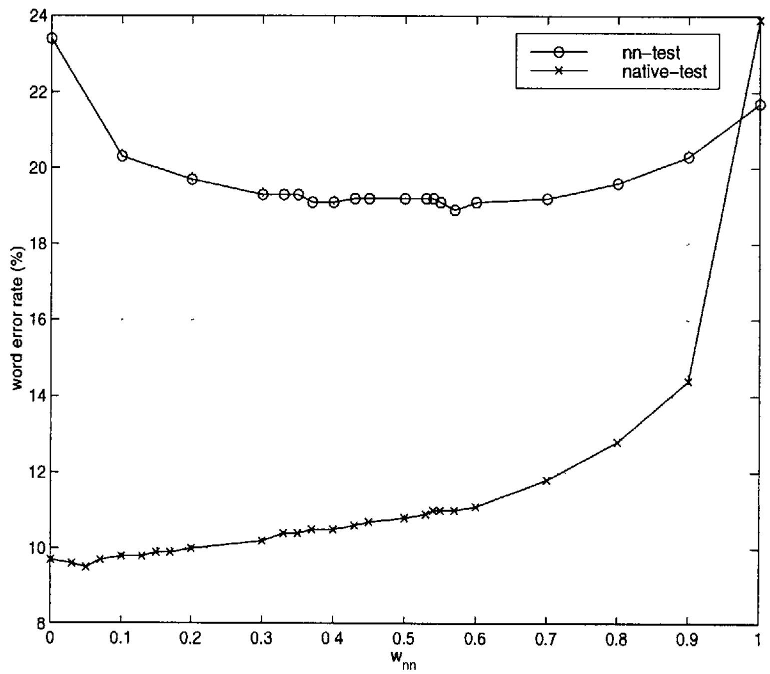

Figure 3-2: Word error rates obtained on native and non-native test sets using interpolated models with varying relative weights.

Chapter 4

Comparison of Native and Non-Native Lexical Patterns Using Manual Transcriptions

In this chapter, we investigate the differences between native and non-native patterns of word pronunciation through manual phonetic transcriptions of a small subset of the utterances in the JUPITER corpus. We examine the transcriptions both qualitatively, to gain a general impression of the phenomena present, and quantitatively, to measure the frequencies of these phenomena. This is an informal investigation, in that only a small number of utterances were transcribed by a single person. However, this type of investigation can provide a general picture of the types of pronunciation differences that exist between native and non-native speakers.

4.1 Manual Transcriptions

The data for this study consist of 100 utterances from 19 native speakers and 96 utterances from 24 non-native speakers, drawn from the JUPITER corpus. Each utterance was transcribed by both listening to portions of the utterance and visually examining a spectrogram of the waveform. The label set for the phonetic transcriptions is based on the one used in the baseline recognizer. However, labels were added as needed to better account for the speech sounds present in the data. This resulted in somewhat different label sets for the native and non-native transcriptions. For example, we found that several of the native speakers tended to use diphthong-like sounds instead of pure vowels. We therefore added several labels corresponding to these sounds. For non-native speakers, we added several short, tense vowels (such as IPA [i], a non-diphthongized, tense, high front vowel), as well as labels for non-native varieties of /r/ and /l/. Several non-standard phones were added for both native and non-native transcriptions, such as IPA [ʏ] (a lax, rounded, mid-high, mid-front vowel) and IPA [x] (a voiceless velar fricative). The full label sets used for the transcriptions are described in Appendix A. In the discussion and examples of the following sections, we use the manual transcription labels.

4.1.1 Guidelines for Preparation of Transcriptions

One of the first issues that arose in the preparation of the manual transcriptions is the level of specificity of the transcriptions. For example, the transcriptions cannot be on the phonemic level, since it is often not clear what the underlying phonemes are. For example, if an American speaker devoices a final [z], we may infer that the intended phoneme was a [z] although the sound produced more closely resembles an [s]. If, however, a non-American speaker devoices the same [z], we cannot know if he/she intended to produce a [z] or an [s]. A broad phonetic transcription is also possible; in this case, for example, a non-native speaker's pure [i] and a native speaker's diphthongized [iy] might receive the same label. However, this would have two main drawbacks. First, we would lose information about the presence of sounds that may be difficult for a speech recognizer to handle. Second, the non-American sounds may be difficult to classify under a broad classification scheme. For example, of two slightly different [i]'s, one may more closely resemble an [iy] while the other may more closely resemble an [ih]; using [i] allows us to both better describe the two segments and indicate that they are members of the same class.

For this reason, we have chosen to transcribe all of the utterances at the most specific level possible; that is, using the largest set of phones that we were able to distinguish. This has the advantage of retaining as much information as possible, but the disadvantage that it is biased by the transcriber's familiarity with various non-American phones. For example, while the transcriptions distinguish between [a] and [aa], [i] and [iy], and [ue] and [uhe], all non-American /r/'s are labeled [r*]. The utterances no doubt also contain many phones that were transcribed as American sounds although, to the speakers, they may be phonemically different (for example, aspirated and non-aspirated stops).

In some cases, the identity of a segment was very ambiguous. In general, we used the convention that the speaker is given the benefit of the doubt. Therefore, in cases where a segment could have been either a "correct" phone or an "incorrect" one (according to the lexicon), the correct phone was chosen for the transcription. Stop closures proved very difficult to transcribe, with some voiceless stop closures appearing to contain voice bars. As a rule, the correct closure was used in the transcription, except for a few voiceless stop closures that were labeled as their voiced counterparts because the waveforms appeared to contain several periods of voicing within the closures. Some segments were difficult to transcribe due to the limited bandwidth and various background noises. For example, alveolar and labiodental fricatives were easily confusable, and breath noises or background noise may have made some segments appear fricated even if they were not; in these cases, again, the correct label was used in the transcription. Finally, in cases where a phone's identity was particularly ambiguous, the label [unk] (for "unknown") or another label indicating the possible phones (such as [nas] for any nasal, [sch] for any schwa, and [dh_v] for [dh] or [v]) was used. Finally, pauses and non-speech sounds such as coughs and lip smacks were not transcribed and are not addressed in our analysis.

4.2 Observations on Common Pronunciation Phenomena in Transcriptions

This section describes some of the phenomena we noted in the transcriptions of native and non-native utterances.

4.2.1 Non-Native Speakers

This section lists some common phenomena in the non-native utterances, along with examples of each one. Many of these are intuitive if we consider the properties of languages other than English [27]. For example, certain languages have a simpler syllable structure than English and tend to use simpler consonant clusters, or none at all. It might be expected, therefore, that speakers of these languages may delete consonants in clusters or insert schwas to break up strings of consonants. Many languages also do not tend to reduce vowels in unstressed positions to the extent that American English does, so it is to be expected that speakers of these languages may substitute full vowels for schwas.

The data also demonstrate the unpredictability of non-native pronunciations. In some cases, we find different productions of the same word or phone within the same speaker. For example, one speaker in the corpus, who seemed to have a Spanish or South American accent, had a great deal of difficulty with the liquids in the word *Florida*. In several productions of this word, he: replaced the [l] with an American [r]-like sound; deleted the [r]; used a flap or trill for the [r]; or produced an unidentifiable phone in place of the [l]. Another speaker in the test set, who seemed to have a French accent, produced two different types of /r/ in the utterance *Paris France*: A trilled one in *Paris* and an American-sounding [r] in *France*.

The following is a list of some of the common pronunciation phenomena in the non-native transcriptions. The examples are of the form

words: ["typical" pronunciation] → [non-native pronunciation] (phenomenon)

where the "typical" pronunciation is drawn from the baseline lexicon. The symbol ε represents the null phone, used when there is an insertion or deletion.

- Use of the non-American tense vowels [i], [o], [uu], [a], and [e], often used instead of lax vowels:

  city: [s ih tcl t iy] → [s i tcl t i] ([ih] →[i], [iy] →[i])
  no: [n ow] → [n o] ([ow] →[o])
  tomorrow: [tcl t ax m aa r ow] → [t uu m a r ow] ([ax] →[uu], [aa] →[a])
  something: [s ah m th ih ng] → [s a m th_t ih ng] ([ah] →[a])

name: [n ey m] → [n e m] ([ey] →[e])

- Use of full vowels instead of reduced ones:

  Phoenix: [f iy n ix kcl k s] → [f iy n iy kcl k s] ([ix] →[iy])

  about: [ax bcl b aw tcl t] → [eh bcl b ahw uhx_ins dcl d ix_ins] ([ax] →[eh])

- Deletion of final consonants or of consonants in clusters:

  how_about: [hh aw ax bcl b aw tcl t] → [hh aw bcl b aw] ([tcl t] →[ε])

  New_York_City: [n y uw y o r kcl k s ih tcl t iy] → [n y uhe y o r* s i tcl t i] ([kcl k] →[ε])

  Portland: [pcl p o r tcl l ax n dcl d] → [pcl p o r l eh n dcl d] ([tcl] →[ε])

- For some speakers, difficulty producing liquids and non-American productions of [l] and [r]:

  Florida: [f l o r ix dx ax] → [f r unk unk i dh a] ([l o r] →[r unk unk])

  Peru: [pcl p rx uw] → [pcl p eh r* uw] ([rx] →[eh r*])

- Frequent schwa insertions:

  how_about the weather in: [hh aw ax bcl b aw tcl t dh ax w eh dh rx ih n] → [ahw eh b ahw uhx_ins d ix_ins dh_v eh w eh dh rx* ax_ins q i n* ix_ins] ([ε] →[uhx_ins], [ix_ins], [ax_ins])

4.2.2 Native Speakers

The following are some of the common phenomena that appear in the native utterances, along with examples of each. Note than some of these are also common in non-native speakers.

- Deletion of segments or compression of strings of segments in unstressed syllables:

  tomorrow in: [tcl t ax m aa r ow ih n] → [tcl t mm aa r nn] ([ow ih n] →[nn])

  could you: [kcl k uh dcl d y uw] → [k dcl d ue] ([uh] →[ε], [y uw] →[ue])

- Devoicing of final consonants, especially [z]:

  directions: [dcl d ix r eh kcl k sh ix n z] → [dcl d eh r eh k sh nn s] ([z] →[s])

- Use of diphthongs in place of pure vowels:

  and: [ae n dcl d] → [ihae n] ([ae] →[ihae])

- Schwa insertions:

  what is: [w ah dx ih z] → [w ah dx ix_ins ih z ax_ins] ([ε] →[ix_ins], [ax_ins])

4.3 Quantitative Analysis

In order to obtain quantitative estimates of the frequencies of the various phenomena in the transcriptions, we aligned each transcription with a subset of the lexicon corresponding to the known word transcription of each utterance. The procedure was a simple string alignment, in which any lexical phone can be aligned with any surface (transcription) phone. Alignments with ε represent deletions and insertions. A match between a surface phone and an identical lexical phone is given a cost of 0 and a mismatch is given a cost of 1, and the best alignment is found by minimizing the score. The only exceptions are that inserted schwas must be aligned with ε, and that [er] →[rx], [t-] →[t], [k-] →[k], and [p-] →[p] are considered to be matches since the manual transcriptions do not differentiate between these pairs of phones. After aligning all transcriptions, the probabilities of all phone-to-phone alignments, or confusions, are computed.

Figures 4-1 and 4-2 show the estimated confusion probabilities in bubble plot form. The calculation of the estimated probabilities is described in greater detail in Chapter 5. In this chapter, we are not concerned with the exact values of the probabilities, but only with their relative magnitudes. Note that the plots contain some spurious confusions (such as [uw] → [gf]) due to alignment errors.

Table 4-1 shows the frequencies of some of the common phenomena we have discussed, as well as some additional phenomena. The frequencies in the table were obtained from the automatic alignments. For each type of phenomenon, the number of occurrences of the phenomenon is divided by the number of times the conditioning context occurred. For example, for the use of [i, iy] instead of [ih], the conditioning count is the number of times a lexical [ih] occurred. For the frequencies of certain sounds that do not occur in the lexicon ([i], [r*], etc.), the conditioning count is simply the total number of surface labels. This table suggests that non-natives are more likely to use the vowels [a, o, e, i, uu]; replace a lax vowel with a tense vowel; replace a schwa with a full vowel or diphthong; use non-American liquids [r*, l*]; replace a stop with a fricative (such as [gcl g] → [gf]); or replace a dental fricative with an alveolar stop or stop-like fricative. Native speakers, on the other hand, are more likely to delete a schwa or use a diphthong instead of a pure vowel. For the other phenomena measured, it is difficult to determine from such a small number of occurrences whether there is a clear difference.

These phenomena suggest that we may be able to model non-native pronunciation patterns by incorporating additional rules into the recognizer's lexicon. However, discovering such rules and applying them to the lexicon is labor-intensive and prone to human and alignment error. A more attractive solution may be to learn rules from training data automatically. In the next chapter, we describe an attempt to automatically learn a particular class of phonetic replacement rules from the non-native training data.

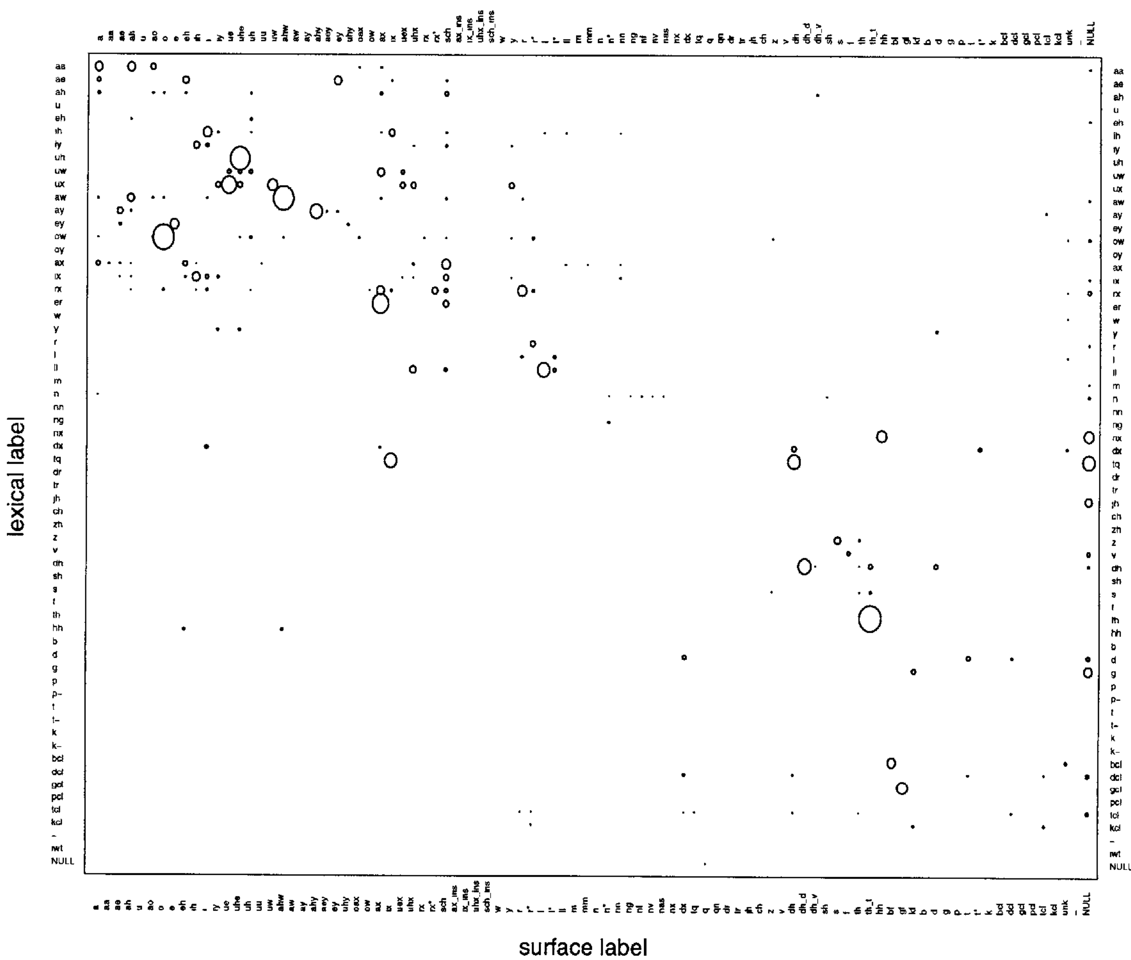

Figure 4-1: Bubble plot showing the estimated confusion probabilities for non-native speakers. The radius of each bubble is proportional to the probability of the corresponding confusion. The largest bubble corresponds to $P(th\_t|th) = 0.4545$.

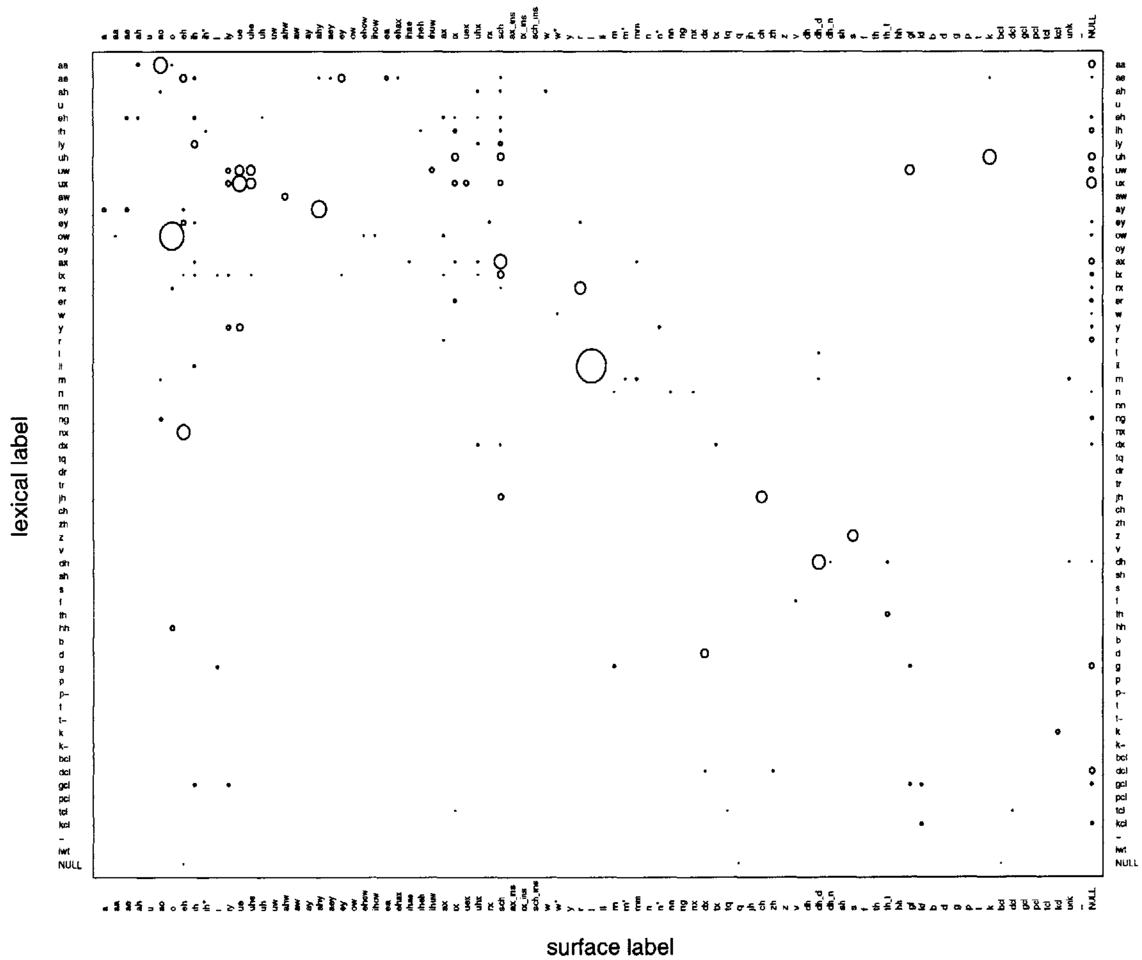

Figure 4-2: Bubble plot showing the estimated confusion probabilities for native speakers. The largest bubble corresponds to $P(l|ll) = 0.5833$. The scale is the same in this plot as in Figure 4-1.

Phenomenon	**Frequency (Native)**	**Frequency (Non-Native)**
(1) use of [a, o, e, i, uu]	38/2089 = 1.8%	93/2140 = 4.3%
(2) [eh] → [e, ey]	0/60	0/50
(3) [ih] → [i, iy]	0/52	15/76 = 19.7%
(4) [uh] → [uu, uw, ue]	0/8	0/5
(5) [ah] → [a, aa, ao]	1/32 = 3.1%	3/39 = 7.7%
(6) lax → tense (sum of (2)–(5))	1/152 = 0.7%	18/170 = 10.6%
(7) schwa → full vowel/diphthong	10/133 = 7.5%	27/157 = 17.2%
(8) [r*]	0/2089	14/2140 = 0.7%
(9) [l*]	0/2089	4/2140 = 0.2%
(10) schwa insertion	13/2153 = 0.6%	17/2212 = 0.8%
(11) schwa deletion	9/133 = 6.8%	2/157 = 1.3%
(12) voiced fric. → voiceless fric.	7/91 = 7.7%	13/134 = 9.7%
(13) use of [ehow, ihow, ea, ehax, ihae, iheh, ihuw]	9/2089 = 0.4%	0/2140
(14) stop frication	5/279 = 1.8%	10/240 = 4.2%
(15) [dh, th] →[d, t, dh_d, th_t]	19/78 = 24.4%	42/98 = 42.9%

Table 4-1: Frequencies of various phenomena noted in manual transcriptions of native and non-native data.

Chapter 5

Automatic Acquisition of Non-Native Pronunciation Patterns

In this chapter we investigate the possibility of modeling the pronunciation patterns of non-native speakers within a speech recognizer. One natural way to attempt to do this is by training the lexicon arc weights on the non-native training data. However, as Table 5-1 shows, this actually increases the word error rate on the non-native development set. This is most likely because of the limited size of the non-native training set; that is, it may not contain enough occurrences of each pronunciation in the lexicon to robustly train the arc weights.

Furthermore, even given sufficient non-native training data, a redistribution of the probabilities of alternate pronunciations is not likely to account for most of the pronunciation patterns of non-native speakers. The baseline lexicon contains several pronunciations per word, derived by applying common American English phonological rules to baseform pronunciations. However, as discussed in Chapter 4, and as shown in previous research (*e.g.*, [29]), non-native speech may contain many more pronunciations per word and may follow different phonological rules than native speech. Ideally, we would like to collect entire word pronunciations and train their probabilities from real non-native data. However, since we do not have enough instances of each word from the JUPITER lexicon in the non-native

Baseline Lexicon				Lex. Trained on `nn-train`			
S	**I**	**D**	**WER**	**S**	**I**	**D**	**WER**
12.1	4.3	3.8	20.2	14.1	4.6	3.9	22.6

Table 5-1: Rates of substitution (S), insertion (I), and deletion (D) and word error rates (WER) on `nn-dev` for recognizers using the baseline lexicon and a lexicon with arc weights trained on `nn-train`. In both cases, the baseline acoustic models were used.

training data, we attempt to derive rules from the training data, which we can then apply to the lexicon much as phonological rules are added to the baseforms.

5.1 Approach

The specific approach we chose is to augment the baseline FST recognizer with an additional weighted FST, C, which is introduced between the lexicon and the phonetic graph. That is, whereas the baseline recognizer can be represented as

$$R = A \circ D \circ L \circ G, \tag{5.1}$$

as described in Chapter 2, the modified recognizer is

$$R_C = A \circ D \circ C \circ L \circ G \tag{5.2}$$

The FST C may represent any number of phonological phenomena, and its arc weights represent the probabilities of occurrence of these phenomena.

In this initial investigation, we consider a very simple type of C. Namely, we build a C consisting of a single state, with self-loops representing allowed mappings between lexical phones and surface phones, the phones that actually occur in the utterance. A portion of such an FST may look like the following:

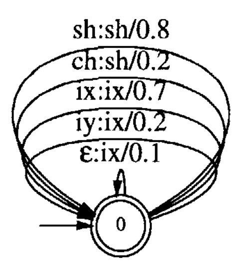

Figure 5-1: Example portion of a simple C.

In this and all FST's we show, an arc labeled $(x : y)/p$ indicates that a y can be realized as an x with probability p. The notation $(x : y)$ represents the *conditional* event that, given a lexical label y, its surface realization is x. In cases where a lexical label has a single surface realization, the weight on the corresponding arc may be omitted. This FST indicates that a [sh] in the lexicon can be realized as a [sh] with probability 0.8 or as a [ch] with probability 0.2, and that a lexical [ix] can be realized as an [ix] with probability 0.7 or as an [iy] with probability 0.2. An [ix] can also be deleted with probability 0.1; this is represented as substitution with a null phone, ε. The weight on an arc need not be its probability, although it is often a function of the probability. In this approach, insertions are represented as transitions from a null phone to a non-null phone.

This type of C models only *context-independent* probabilities. For example, it does not have different probabilities for the mapping $(ch : sh)$ in *Chicago* and in *Shanghai*, although the former is likely to be more probable than the latter. As we will show, however, even this simplistic approach can be useful in the recognition of non-native speech.

A full C would contain at least one arc for every lexical label. The maximum size of C is $N_l N_s - 1$, where N_l is the number of lexical labels and N_s is the number of surface labels, including a null phone in both label sets. The maximum is one less than $N_l N_s$ because the mapping between two null phones is not included. N_l and N_s may differ, for example, if the lexicon contains phonemic pronunciations and the phone graph contains phonetic pronunciations. However, we consider only the case where the lexical and surface label sets are identical, so that $N_l = N_s = N$. We refer to this FST as C because it attempts to model the phonetic *confusions* that non-native speakers may exhibit, or, as will be discussed, that the recognizer may exhibit in recognizing non-native speech. We will also refer to C as a confusion FST or CFST.

The effect of composing C with L is to simply add to each arc in L a set of parallel arcs corresponding to the possible surface realizations of the lexical phone on that arc. So, for example, L may contain the following pronunciation graph for *Chicago*:

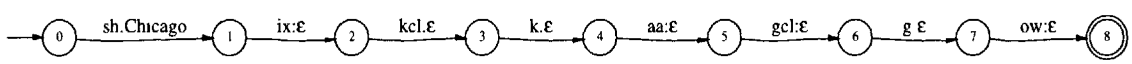

Figure 5-2: Lexical pronunciation graph for the word *Chicago*.

This graph indicates that the word *Chicago* can only be realized as the phone string [sh ix kcl k aa gcl g ow]. If C contains only the confusions shown in Figure 5-1, plus self-mappings only for all of the other lexical labels, then the composition $C \circ L$ yields the following pronunciation graph for *Chicago*:

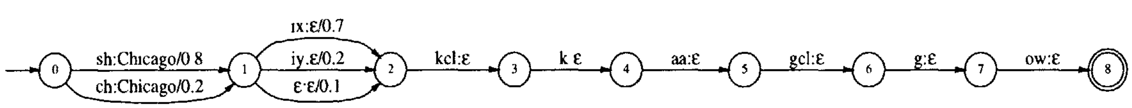

Figure 5-3: Expanded lexical pronunciation graph for the word *Chicago*.

This FST indicates that *Chicago* can have any of the pronunciations [sh ix kcl k aa gcl g ow], [sh kcl k aa gcl g ow], [sh iy kcl k aa gcl g ow], [ch ix kcl k aa gcl g ow], [ch kcl k aa gcl g ow], or [ch iy kcl k aa gcl g ow]. In practice, the $(\varepsilon : \varepsilon)$ arcs are removed, and alternate paths that skip over node 1 or 2 are inserted.

An alternate way of looking at C, rather than as an expansion of the lexicon, is that it allows an imperfect match between the lexicon and the surface phones. For example, while the lexicon may only have the pronunciation [sh ix kcl k aa gcl g ow] for the word *Chicago*, the incorporation of C allows the common non-native pronunciation [sh iy kcl k aa gcl g ow] by permitting a mismatch in the second phone.

This approach is similar in some respects to previous research in pronunciation modeling. In [20], for example, Levinson *et al.* obtained word hypotheses by aligning the output of a phonetic recognizer with the lexicon and grammar. This is similar to the approach used here, in that it allows a mismatch between the phonetic hypotheses and the lexicon. However, [20] used a measure of acoustic similarity between phones to determine the confusion weights. The work of Teixeira *et al.* [29] is more similar to the current approach, in that the pronunciation weights are estimated from non-native training data. In [26], Riley and Ljolje train probabilistic, context-dependent phoneme-to-phone mappings to obtain a phonetic lexicon for native speakers, using a similar estimation procedure to the one we have described but using a classification tree to group contexts into feature-based classes. Finally, there are also similarities between our approach and that of ANGIE [18], a subword lexical modeling framework, in which the probabilities of phonological rules are trainable.

5.2 Methods

This section discusses the methods we use to build CFST's: The estimation of arc probabilities, computational considerations in recognition with CFST's, and the problem of sparse training data.

5.2.1 Estimation of CFST Arc Probabilities

The first problem we confront is that of training the confusion probabilities in C. In order to do so, we need to have an accurate phonetic transcription for each utterance, aligned with the corresponding pronunciation of each word as it appears in the lexicon. We cannot do this by creating forced paths using the baseline recognizer, since these will only contain pronunciations already in the baseline lexicon. One option is to manually transcribe all of the training utterances. This, however, is a very labor-intensive task and is prone to mismatch between the human transcriber's conception of each phone and the recognizer's acoustic models.

Our approach to this problem is to generate automatic transcriptions of the non-native training sentences, using either phonetic recognizer hypotheses or forced paths with an expanded lexicon; both of these methods are described in Section 5.3. We then use an automatic string alignment procedure to align the transcriptions with the lexicon. Finally, we derive the maximum likelihood estimates of the confusion probabilities from their frequencies in the alignments.

The alignment of each utterance is done by composing several FST's:

$$P \circ C_a \circ L_0 \circ W, \tag{5.3}$$

where P is an unweighted FST representing the string of phones in the utterance's phonetic transcription; C_a is an *alignment CFST* which contains an *a priori* probability $P_a(s:l)$ of each confusion $(s:l)$; L_0 is the lexicon with no arc weights; and W is an unweighted FST representing the known word transcription of the utterance. The probabilities in C_a can be set either manually or using an existing CFST from a prior training run. We use L_0 rather than L during alignment because we do not want to bias the alignment toward any particular pronunciation of a given word. If we used the weighted lexicon L, there could be a situation in which the speaker produced one of the pronunciations in the lexicon for a given word, but the word was aligned with a different pronunciation because it had a higher probability in L.

The probability of each confusion is then computed by counting the number of times it occurs in the alignments. For substitutions and deletions, if the lexical label l occurs n_l times in the alignments, and the confusion $(s:l)$ with the surface label s occurs $n_{s:l}$ times, the maximum likelihood estimate of the probability of $(s:l)$ is

$$\dot{P}(s:l) = \frac{n_{s:l}}{n_l} \tag{5.4}$$

The probabilities of insertions are treated somewhat differently than those of substitutions and deletions. This is because, in the case of a substitution or deletion, the lexical label and its *a priori* probability are given. In the case of an insertion, on the other hand, we can view it as adding an arc with a null phone, *ins*, to the baseline lexicon where none previously existed. Note that *ins* is similar to, but not the same as, ε. While *ins* is a null lexical phone, it is a special kind of null phone—namely, one that will be replaced by a non-null surface phone. There are infinitely many ε's in any lexical string, but only as many *ins*'s as there will be insertions. In this formulation, the probability of an inserted s is the *a priori* probability of traversing an *ins* arc in the lexicon, multiplied by the probability of an *ins* being realized as an s. The first step, therefore, is to calculate the *a priori* probability of *ins*, or, in other words, the probability of an insertion.

Consider a sequence of lexical labels $\vec{l} = \{l_1, l_2, \ldots, l_m\}$, which is realized as the sequence of surface labels $\vec{s} = \{s_1, s_2, \ldots, s_{m+n_{ins}}\}$, where $s_i = \varepsilon$ for a deletion. The n_{ins} additional surface labels represent inserted phones. Then the maximum likelihood estimate of the probability of an insertion is

$$\dot{P}(ins) = \frac{n_{ins}}{m + n_{ins}} \tag{5.5}$$

In the present case, n_{ins} is the number of insertions in the alignments and $m + n_{ins}$ is the total number of surface phones in the alignments, including ε as a surface phone when there is a deletion. Given that an insertion occurs, the estimated probability of the inserted phone being s is then

$$\dot{P}(s|ins) = \frac{n_{s:ins}}{n_{ins}}, \tag{5.6}$$

where $n_{s:ins}$ is the number of times an s was inserted. Letting $n_{tot} = m + n_{ins}$, the total probability of an s insertion is therefore

$$\dot{P}(s : ins) = \dot{P}(s|ins)\dot{P}(ins) \tag{5.7}$$

$$= \frac{n_{s:ins}}{n_{tot}} \tag{5.8}$$

5.2.2 Computational Considerations of CFST's

The composition of a lexicon and a CFST can be larger than the lexicon by a factor of up to N_s, the number of surface phonetic labels. In preliminary experiments, we found that this can make both the running time and memory requirements of the recognizer prohibitively large. In order to make recognition with CFST's feasible, we use two main space-saving measures: dynamic composition and CFST pruning. Instead of performing the composition $R = D \circ C \circ L \circ G$ offline prior to recognition, we perform the two smaller compositions $R_1 = D \circ C$ and $R_2 = L \circ G$ offline and perform $R = R_1 \circ R_2$ dynamically during recognition. In dynamic composition, portions of R are created as necessary to expand the hypotheses being considered. While this is less efficient than offline composition, since R is not optimized during dynamic composition, it limits the working size of the FST being searched. In addition, since dynamic composition creates many superfluous paths, the total number of active nodes in the Viterbi search is limited to 1000.

In order to further limit the size of R, we prune C by including only those arcs whose probabilities are above a given threshold. When pruning C, we always include the self-confusions $(l : l)$ even if their probabilities are below threshold, so as to always allow the baseline pronunciations. After pruning, the arc probabilities are redistributed so that the probabilities of all of the allowed confusions for each lexical label sum to 1. In the case of insertions, the redistribution is done so that the probabilities of the allowed insertions sum to $\dot{P}(ins)$.

In addition, in some of the experiments, we further reduce time and space requirements by using a narrower beam in the Viterbi search than the baseline recognizer does. While this initially increases the error rates, it allows us to experiment with a larger range of CFST sizes.

5.2.3 The Sparse Data Problem

The robust estimation of confusion probabilities depends on the availability of sufficient training data. Because the non-native training data are quite limited, some of the lexical labels in the alignments occur very few times, so that the probability estimates for the confusions of those labels are unreliable. While the maximum likelihood probability estimates from the previous section are optimal for predicting the training set, they may not reliably predict the probabilities of unseen data.

In order to combat this problem, we smooth the confusion probabilities; that is, we redistribute the probability so that some or all of the unobserved confusions receive some probability, while the observed confusions receive somewhat less than the maximum likelihood probabilities. We describe two smoothing approaches, padding and linear interpolation.

Padding

A simple solution to the problem of unobserved confusions is to simply assume that some or all of the unobserved confusions were seen some minimum number of times in the training data. We use two types of padding approaches. In the first method, which we refer to as `pad-1`, we add one occurrence of each self-confusion $(l : l)$ that did not occur. This is a very restricted form of smoothing, but it ensures that the baseline pronunciations are allowed, while adding a very small number of arcs to the CFST. Using this approach, the probability estimates for substitutions and deletions are

$$\hat{P}(s:l) = \begin{cases} \frac{n_{s:l}}{n_l}, & (l:l) \textit{ observed} \\ \frac{n_{s:l}}{n_l+1}, & (l:l) \textit{ unobserved}, \; s \neq l \\ \frac{1}{n_l+1}, & (l:l) \textit{ unobserved}, \; s = l \end{cases} \tag{5.9}$$

The probability estimates for insertions do not change; that is, $\hat{P}(s : ins) = \dot{P}(s : ins)$.

In the second approach, `pad-2`, we add a small number n_p of occurrences of each confusion that did not occur in the training data. This guarantees that all confusions are represented. If the number of unobserved confusions from lexical label l is u_l, the probability estimates become

$$\hat{P}(s:l) = \begin{cases} \frac{n_{s:l}}{n_l+n_p u_l}, & (s:l) \textit{ observed} \\ \frac{n_p}{n_l+n_p u_l}, & (s:l) \textit{ unobserved} \end{cases} \tag{5.10}$$

This holds for all types of confusions, where, for an insertion, n_l is replaced by n_{tot}.

Linear Interpolation

A smoothing approach often used in training n-gram language models is linear interpolation [17], in which the probability estimates of the n-gram model are linear combinations of the observed frequencies of that n-gram in the training set and the probability estimates of the next lower-order n-gram model. By analogy to n-grams, confusion probability estimates can be smoothed by combining the observed frequency of $(s : l)$ with the *a priori* probability estimate of s, $\hat{P}(s)$:

$$\hat{P}(s:l) = \lambda_2 \, \dot{P}(s:l) + (1-\lambda_2)\, \hat{P}(s), \tag{5.11}$$

where $\hat{P}(s)$ is similarly smoothed with $\hat{P}_0 = 1/N$, the *a priori* unigram probability (recall that N is the number of labels in the label set):

$$\dot{P}(s) = \frac{n_s}{n_{tot}} \tag{5.12}$$

$$\hat{P}(s) = \lambda_1 \dot{P}(s) + (1 - \lambda_1) \hat{P}_0 \quad (5.13)$$

λ_1 and λ_2 are the smoothing parameters, which determine the weighting given to each of the probabilities being combined. We use the approach of [23], in which

$$\lambda_2 = \frac{n_l}{n_l + r_l} \quad (5.14)$$

$$\lambda_1 = \frac{n_{tot}}{n_{tot} + r_{tot}}, \quad (5.15)$$

where r_l is the number of unique surface labels s for which $(s : l)$ occurs in the training data, and r_{tot} is the number of unique surface labels in the training data.

5.3 Experiments

This section describes recognition experiments performed with CFST's using the methods described in the previous section to compute the estimated confusion probabilities. Unless otherwise indicated, all of the recognizers use the baseline acoustic, lexical, and language models; the only differences between the baseline and the test recognizers are the incorporation of a CFST, and possibly the use of different beam widths or word transition weights during the search.

5.3.1 Training from Phonetic Recognition Hypotheses

The first method we used to create automatic transcriptions of the training data is phonetic recognition. The phonetic recognizer we used has the same diphone acoustic models as the baseline word recognizer. In order to constrain the transcriptions as little as possible, while minimizing obvious transcription errors, a phone bigram language model, trained on the `mixed-train-iv` set, is used for this task. We felt that, because the phonetic language model is trained from forced paths created using the baseline lexicon, a higher-order n-gram would force the transcriptions to conform too closely to the baseline pronunciations. On the other hand, in preliminary experiments, we found that a unigram tends to produce some obvious transcription errors such as strings of consecutive stop closures (this problem may have been alleviated by the use of a duration model, which would discourage single segments from being broken up into several shorter segments).

For the alignment CFST C_a, we used the simple weighting function

$$w(s : l) = \begin{cases} 0, & s = l \\ 1, & s \neq l \end{cases} \quad (5.16)$$

The arc probabilities in C were then estimated from the alignments as described above. We refer to this method of creating C as the "phonetic recognition (or PR) method." Note that, since the arc weights are trained from phonetic recognition outputs and not from "correct" transcriptions, C is modeling not only effects present in the actual speech data but also the degree of difficulty that the recognizer has in distinguishing certain phones from each other.

Using this method on the non-native training set with no smoothing, we obtain a CFST, C_{PR}, containing 2188 confusions. Note that the maximum possible number of confusions is $62^2 - 1 = 3843$, since there are 62 phone labels including the null phone. The confusion probabilities in C_{PR} are shown in bubble plot form in Figure 5-4. In this and subsequent bubble plots, the null phone is represented as $NULL$ for both deletions and insertions, and the phone label [_] is represented as *iwt* (for *inter-word trash*). The diagonal entries $(l : l)$ are excluded from the bubble plot, since they are typically so large as to overwhelm the off-diagonal entries. Also excluded are all lexical labels with $n_l < 30$—namely, [k-], [zh], [dr], [nn], [oy], and [p-]—since these tended to have very large confusion probabilities as well. As expected, $(sonorant : sonorant)$ and $(non-sonorant : non-sonorant)$ confusions tend to receive higher probability estimates than $(sonorant : non-sonorant)$ or $(non-sonorant : sonorant)$–that is, there is more probability mass in the upper left and lower right regions of the plot than in the lower left and upper right. However, because of errors in phonetic recognition and alignment, there are many more $(sonorant : non-sonorant)$ and $(non-sonorant : sonorant)$ confusions than we might expect.

Table 5-2 shows the most likely confusions for each lexical label, including the diagonal entries, as well as the number of occurrences of each lexical label in the alignments. As both this table and Figure 5-4 show, the most common confusions include both some that we expect to see in non-native speech, such as $(iy : ix)$ and $(d : dh)$, and some that are most likely errors in phonetic recognition or alignment, such as $(ll : aw)$ and $(s : f)$. Appendix B lists all of the confusions and their frequencies.

We tested a recognizer using CFST's derived from C_{PR} by pruning with varying pruning thresholds (cprune's). In this implementation, the pruning threshold is expressed as a negative log probability, so that the higher the threshold, the more arcs are included. In all cases, we used the pad-1 smoothing method. Figure 5-5 shows the average number of arcs per phone in the CFST—that is, the ratio of the number of arcs in the CFST to the number of phones in the label set—for each cprune. The largest CFST, with cprune $= 6$, contains a total of 1250 confusions, a little over half of the confusions in C_{PR}. Figure 5-6 shows the word error rates obtained on the nn-dev set as a function of cprune. The word error rates are shown for two Viterbi pruning thresholds (vprune's). Due to time and memory constraints, the word error rate was not measured at vprune $= 20$, cprune $= 6$.

For cprune $= 0$, the CFST is an identity mapping, so that the recognizer is almost identical to the baseline recognizer; the only difference is that, in this case, the composition is done on the fly and the number of active nodes during the Viterbi search is limited to 1000. Therefore, the slight increase in WER at cprune $= 0$, vprune $= 20$ with respect to the baseline (from 20.2% to 20.3%) is due only to dynamic composition.

For both vprune $= 15$ and vprune $= 20$, there is a minimum in WER at cprune $= 4$. The increase in WER for cprune > 4 may indicate that the low-probability arcs are not well-trained, and that adding them increases the confusability between words. The lowest WER obtained in this series of experiments is 18.2% at vprune $= 20$, cprune $= 4$. This is a 9.9% relative reduction from the baseline WER.

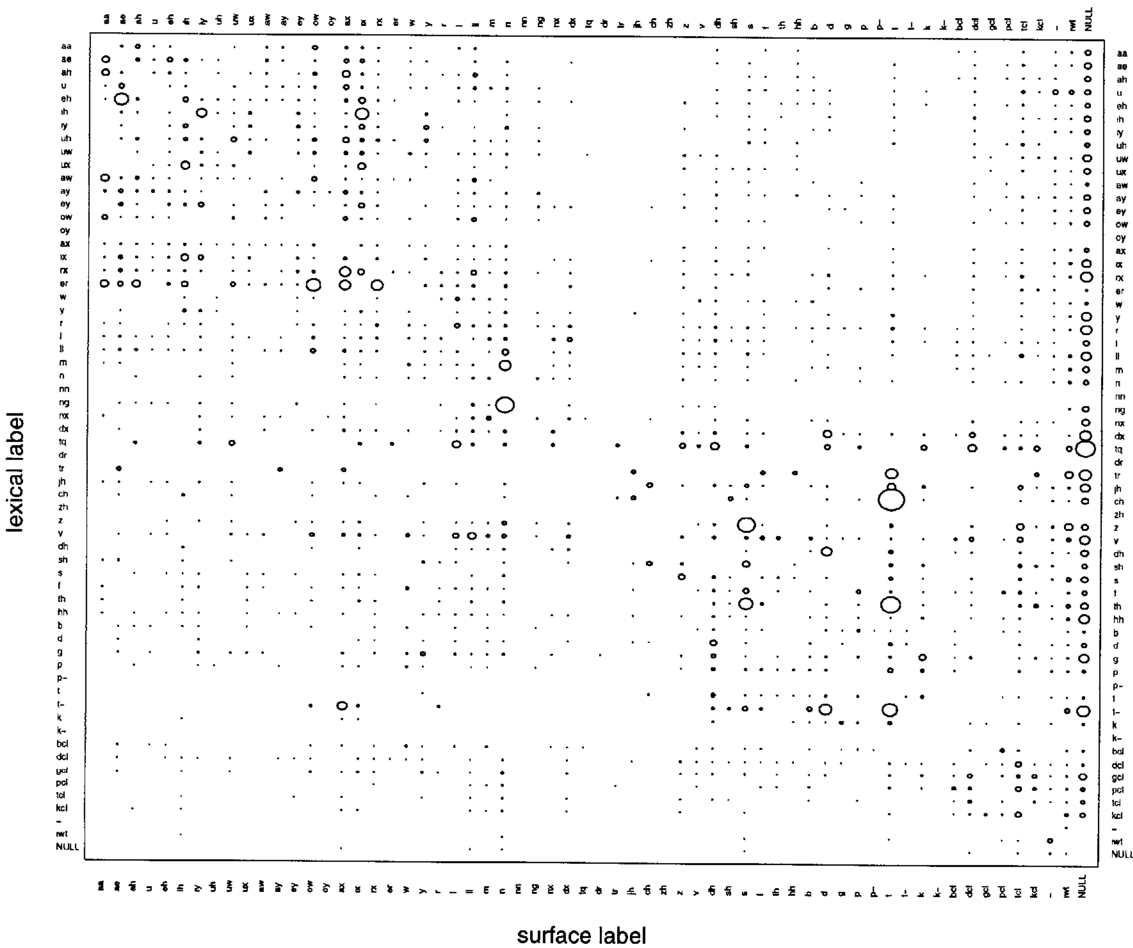

Figure 5-4: Bubble plot showing the estimated confusion probabilities using the phonetic recognition method. The radius of each bubble is proportional to the probability of the corresponding confusion. The largest bubble corresponds to $P(ch : t) = 0.155$.

Lex. Label	Freq.	Most Common Conf.	Lex. Label	Freq.	Most Common Conf.
aa	1146	aa, ε, ow, ah, ax, ix	dr	15	dh, tcl, dr, ε, _, d
ae	1205	ae, aa, ε, eh, ax, ix	tr	42	tr, ε, t, _, ay, f
ah	1338	ah, ax, aa, ε, ll, ow	jh	225	jh, ε, t, ch, tcl, s
u	265	u, ε, -, ax, ae, _	ch	271	ch, t, ε, sh, jh, ih
eh	1822	eh, ae, ix, ih, ε, ax	zh	8	tcl, kcl, n, p, s, ε
ih	2204	ih, ix, iy, ε, ux, ey	z	1110	z, s, _, tcl, ε, n
iy	1389	iy, ix, ε, ih, y, ey	v	338	v, ε, ll, l, tcl, dcl
uh	183	uh, ax, uw, ε, ix, ih	dh	2414	dh, d, ε, t, ih, n
uw	503	uw, ε, ix, ax, ow, w	sh	232	sh, s, ε, ch, t, tcl
ux	466	ux, ih, ix, ε, iy, tcl	s	2577	s, ε, z, _, t, tcl
aw	776	aw, aa, ow, ll, ε, ah	f	829	f, s, ε, p, tcl, w
ay	1051	ay, ε, ae, ax, ey, u	th	211	th, t, s, ε, _, kcl
ey	940	ey, ε, ix, iy, ae, ih	hh	634	hh, ε, _, k, dh, l
ow	2043	ow, aa, ε, ll, ax, uw	b	978	b, p, ε, dh, d, m
oy	22	oy, ay, ow, ll, n, ε	d	1017	d, dh, ε, t, b, iy
ax	2421	ax, ε, ih, eh, ae, ow	g	412	g, ε, k, dh, y, t
ix	2743	ix, ε, ih, iy, ae, ax	p	572	p, t, ε, k, dh, tcl
rx	2415	rx, ε, ax, ix, ll, ae	p-	27	p-, p, b, ae, f, _
er	185	er, ow, rx, ax, aa, ah	t	2409	t, dh, k, d, ε, p
w	2445	w, l, ε, v, m, b	t-	68	t-, t, ε, d, ax, s
y	830	y, ε, ih, t, n, iy	k	1375	k, t, g, ε, dh, p
r	1121	r, ε, l, t, n, rx	k-	1	_
l	1047	l, ε, dx, n, m, nx	bcl	906	bcl, pcl, w, ε, m, l
ll	835	ll, ε, n, ow, tcl, _	dcl	2378	dcl, tcl, ε, t, n, z
m	1233	m, n, ε, w, _, ll	gcl	325	gcl, ε, dcl, kcl, tcl, n
n	3680	n, ε, _, m, ll, ng	pcl	510	pcl, tcl, ε, bcl, kcl, dcl
nn	18	n, ax, d, ix, ε, -	tcl	3658	tcl, ε, dcl, kcl, n, dh
ng	498	ng, n, ε, ll, _, ey	kcl	1464	kcl, tcl, ε, _, gcl, pcl
nx	547	nx, ε, m, ll, dx, dcl	-	42881	-, _, s, n, dh, ax
dx	473	dx, ε, d, dcl, nx, n	_	36718	_, -, tcl, n, s, t
tq	61	tq, ε, dh, l, dcl, _	ε	146831	-, _, tcl, n, s, ix

Table 5-2: Frequency and most common confusions for each lexical label, using the phonetic recognition method.

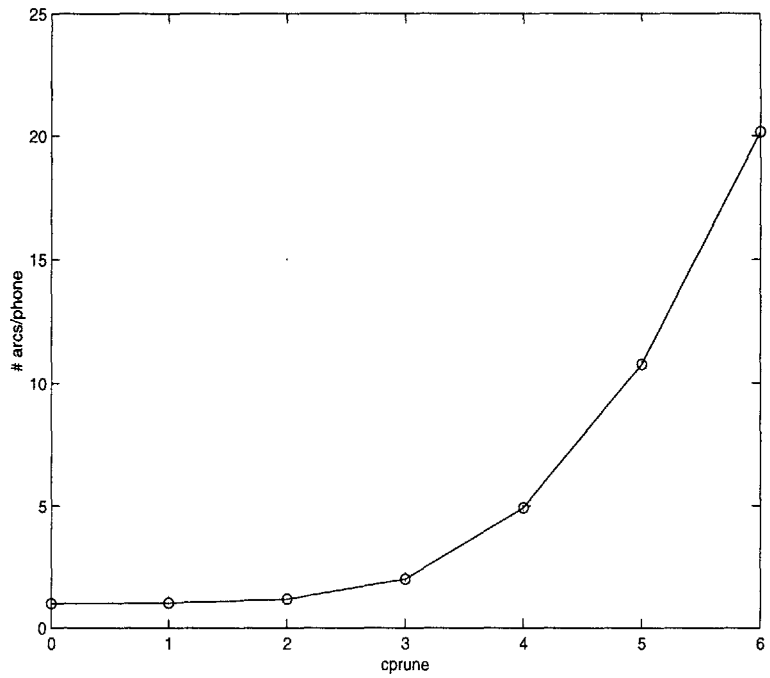

Figure 5-5: Average number of arcs per phone as a function of CFST pruning threshold, using the phonetic recognition method.

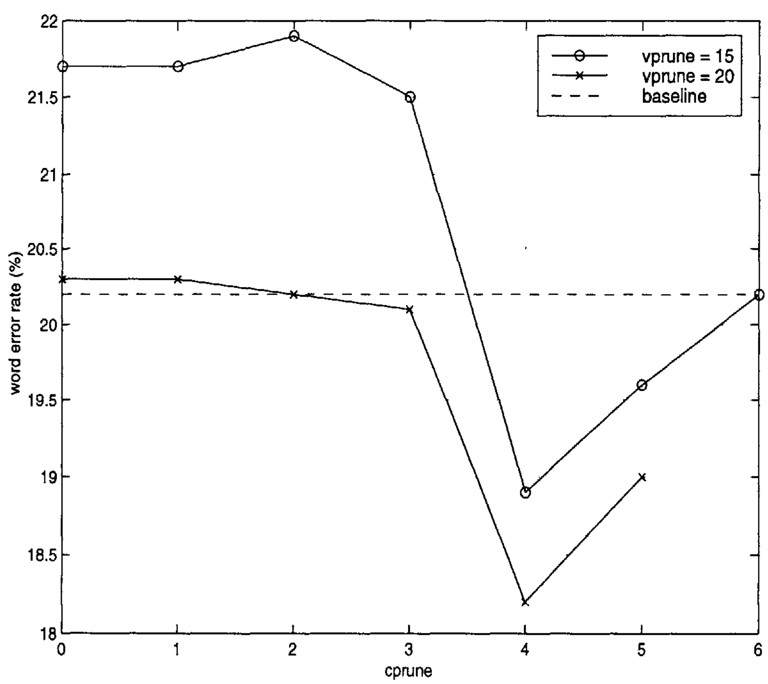

Figure 5-6: Word error rate on nn-dev as a function of CFST pruning threshold, using the phonetic recognition method.

vprune	S	I	D	WER
15	13.1	3.3	4.9	21.3
20	12.2	3.3	4.8	20.3

Table 5-3: Error rates obtained on nn-test using a CFST with cprune = 4, trained using the phonetic recognition method.

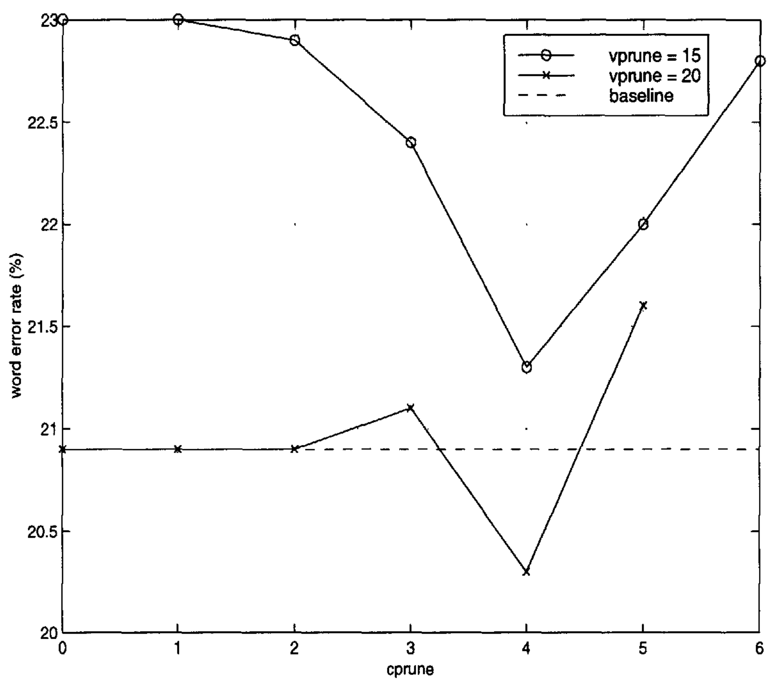

Figure 5-7: Word error rate on nn-test as a function of CFST pruning threshold, using the phonetic recognition method.

Based on these results, we chose a cprune of 4 to test the recognizer on the nn-test set. The results for both vprune = 15 and vprune = 20 are shown in Table 5-3. Compared to the baseline WER of 20.9%, the test recognizer reduces the WER by 2.9% with vprune = 20 but increases it by 1.9% with vprune = 15. To test whether this is the best performance we could obtain on nn-test, we repeated the experiment with the entire series of cprune's tested on the nn-dev set. The results of these experiments are shown in Figure 5-7. As this figure shows, for the pruning thresholds tested, this is the best performance we could obtain on nn-test. As in Chapter 3, we observe what appears to be a qualitative difference between the development set and the test set.

5.3.2 Training from Forced Paths

Although we are able to obtain some performance gains using a phonetic recognizer to generate automatic transcriptions, this results in large CFST's whose accuracy is limited by that of the alignments. Furthermore, there is reason to believe that the CFST's created this way are *too* large, since the phonetic recognizer does not use all of the information that the word recognizer has available to it. In the phonetic recognition approach, we build a CFST that the word recognizer would need in order to obtain the most likely word sequence, if the word sequence were constrained to match a *particular* string of phones. However, the word recognizer has the entire phonetic graph at its disposal, and can search for alternate phones that better match the lexicon when necessary. For this reason, we may not need to expand the lexicon to such a great extent.

An alternate approach is to generate forced transcriptions using a lexicon expanded with a pre-existing CFST C_0 trained with the phonetic recognition method. In other words, the forced paths are created by searching the FST

$$R_{FP} = A \circ D \circ C_0 \circ L \circ W, \tag{5.17}$$

where W represents the known word transcription of the utterance. Since the recognizer knows the correct word sequence when creating forced paths, it is more likely to choose only those confusions from C_0 that are "necessary", in the sense that it will prefer to use fewer confusions so as to minimize cost.

We used different CFST's for C_0 and C_a, the alignment CFST. For C_0, we used C_{PR}, pruned with a threshold of 6.5 and padded with pad-1. The pruning threshold was chosen so as to include most of the confusions in C_0 (1582 out of 2193, including padding), while still performing the forced paths in a reasonable amount of time. For alignment, however, C_a must allow all possible confusions, so that every transcription can be aligned with the lexicon. Therefore, for C_a we used C_{PR} padded using pad-2, with $n_p = 1$. We refer to this as the "forced path method" of confusion probability estimation.

Using this method with no smoothing, we obtain a CFST, C_{FP}, containing 840 confusions. This is a large decrease relative to the phonetic recognition method. Figure 5-8 shows a bubble plot of the confusion probability estimates in C_{FP}, excluding those of lexical labels with fewer than 30 occurrences in the alignments ([k-], [zh], [nn], [dr], [oy], and [p-]). A full listing of the confusion frequencies is given in Appendix B. Table 5-4 lists the most common confusions for each lexical label. From a visual inspection of both the plot and the table, the confusion statistics appear to conform better with our expectations about non-native speech. Overall, the plot is "cleaner" than the one obtained with the PR method, in the sense that the probability mass is more concentrated in the upper left and lower right quadrants. While many of the most common confusions still appear to be due to poor phonetic classification or alignment, such as $(l : w)$ and $(tcl : pcl)$, many of the expected confusions receive higher probability estimates than with the PR method, such as $(iy : ih)$ and $(uw : uh)$.

Figure 5-9 shows the number of arcs per phone for values of cprune between 0 and 12,

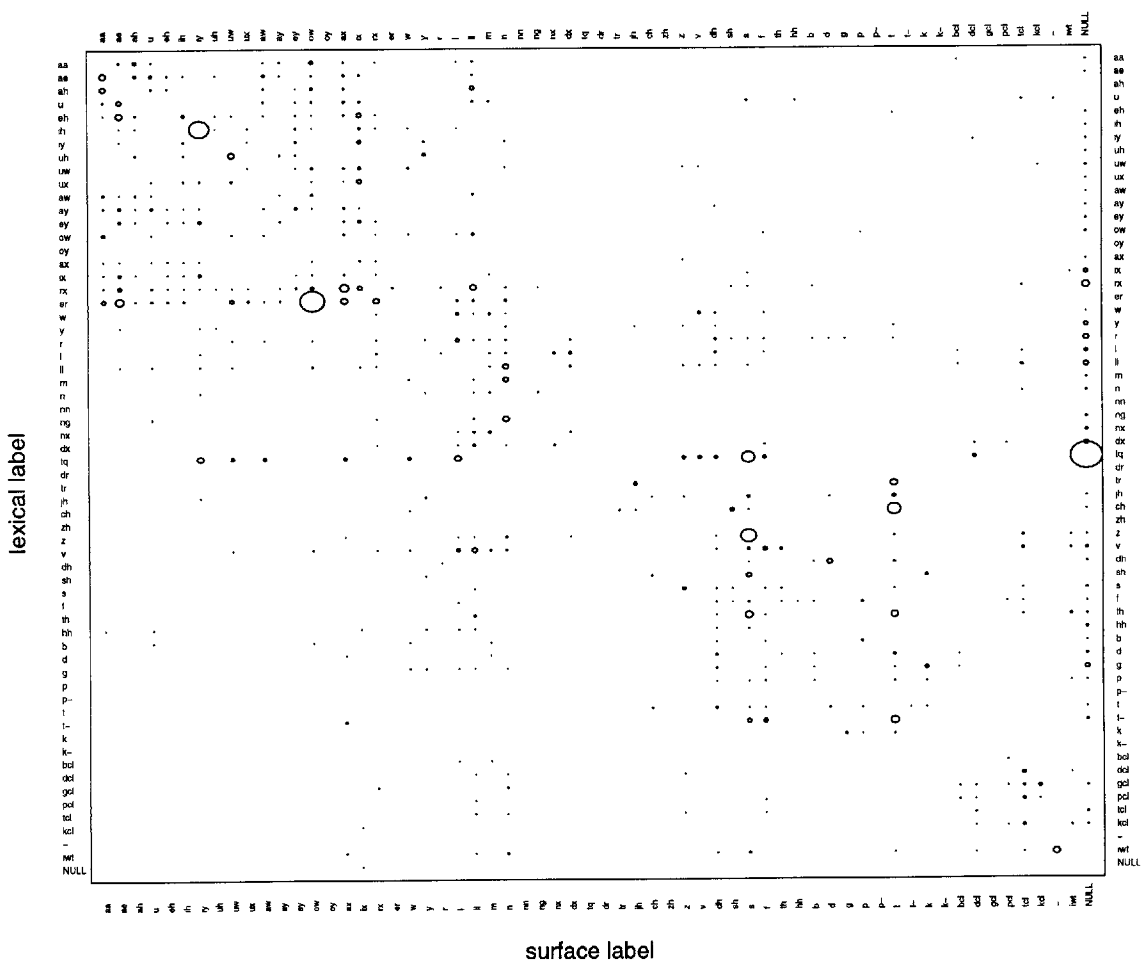

Figure 5-8: Bubble plot showing the confusion probabilities derived from alignments of forced paths obtained on the non-native training set. The largest bubble corresponds to $P(tq : NULL) = 0.2174$.

Lex. Label	Freq.	Most Common Conf.	Lex. Label	Freq.	Most Common Conf.
aa	1157	aa, ah, ow, ae, ll, ay	dr	15	dr, dh, tcl
ae	1230	ae, aa, u, aw, ah, ax	tr	42	tr, t, jh
ah	1304	ah, aa, ll, ow, ax, eh	jh	224	jh, t, s, y, ε, ch
u	265	u, ae, ax, ix, s, ll	ch	266	ch, t, sh, tr, jh, ε
eh	1814	eh, ae, ix, ih, ax, ah	zh	4	n, ix, p
ih	2113	ih, iy, ix, ey, ux, ε	z	1118	z, s, tcl, n, t, ε
iy	1379	iy, ix, ey, y, ih, ε	v	369	v, ll, f, l, ε, tcl
uh	212	uh, uw, y, ah, ih, ε	dh	2416	dh, d, t, ε, s, nx
uw	506	uw, ix, ow, ax, ε, w	sh	236	sh, s, k, ch, y
ux	474	ux, ix, uw, ih, iy, ε	s	2580	s, z, ε, dh, th, f
aw	773	aw, ow, aa, ll, ah, ay	f	829	f, p, s, ε, dh, pcl
ay	1053	ay, ae, ey, u, aa, ax	th	209	th, s, t, _, ll, ε
ey	938	ey, ix, iy, ae, ε, ay	hh	662	hh, ε, s, l, dh, aa
ow	2089	ow, aa, ll, ε, ax, uw	b	1014	b, p, ε, dh, m, f
oy	22	oy, ow, ay	d	1094	d, t, dh, ε, b, m
ax	2546	ax, ix, ε, eh, ae, ih	g	421	g, ε, k, dh, w, y
ix	2890	ix, ε, iy, ae, ax, ow	p	584	p, ε, f, t, s, dh
rx	2391	rx, ax, ε, ll, ix, ae	p-	27	p-, p, b, ε
er	175	er, ow, ae, ax, rx, aa	t	2540	t, dh, d, ch, p, k
w	2468	w, l, v, m, dh, f	t-	74	t-, t, f, s, ε, ax
y	829	y, ε, n, dh, jh, iy	k	1438	k, g, t, p, _, dh
r	1155	r, ε, l, dh, dx, n	k-	1	_
l	1111	l, ε, dx, nx, dh, n	bcl	913	bcl, pcl, m, _, l, ε
ll	803	ll, n, ε, tcl, dx, ow	dcl	2559	dcl, tcl, z, n, ll, _
m	1233	m, n, ε, w, ll, dcl	gcl	359	gcl, kcl, tcl, ε, n, rx
n	3713	n, ε, ng, m, ll, tcl	pcl	511	pcl, tcl, bcl, f, ll, dcl
nn	14	n, ax	tcl	3864	tcl, ε, dcl, ll, n, kcl
ng	499	ng, n, ε, ll, u, rx	kcl	1487	kcl, tcl, ε, _, pcl, gcl
nx	522	nx, ε, m, ll, dx, l	-	6365	-, _
dx	406	dx, ε, ll, nx, dcl, l	_	13378	_, -, s, n, ax, ll
tq	46	tq, ε, s, l, iy, aw	ε	82166	ix, s, -, n, _, tcl

Table 5-4: Frequency and most common confusions for each lexical label, using the forced paths method.

S	I	D	WER
11.4	3.2	4.2	18.8

Table 5-5: Error rates obtained on `nn-test` using a CFST, forced paths method. In this test, cprune = 12, vprune = 15, `wtw` = 2.

using the forced paths method. Figure 5-10 shows the word error rates obtained on the `nn-dev` set using CFST's derived from C_{FP} with **cprune** $\in [0, 12]$, **vprune** $= 15$ and with **cprune** $\in [0, 7]$, **vprune** $= 20$. As before, a smaller range of **cprune** was used with the larger **vprune** due to time and memory constraints. Note that, at **cprune** = 12, the CFST contains all of the confusions in the training data. All of the CFST's were padded with `pad-1`.

For **cprune** $\in [0, 5]$, the word transition weight (`wtw`) was 4, whereas for **cprune** > 5, the `wtw` was 2. This is because, as **cprune** increases, the ratio of deletions to insertions tends to increase. In order to balance the deletions and insertions, the `wtw` must be decreased. Ideally, we would optimize the `wtw` for each **cprune**; however, as the greatest change in the deletion/insertion ratio occurs between **cprune** = 5 and **cprune** = 6, we approximate the evolution as a step function.

In this case, the word error rate is almost monotonically decreasing with **cprune**. The lowest WER is 15.8% and is obtained with **vprune** = 15, **cprune** = 12. This represents a 21.8% relative reduction from the baseline WER. We therefore tested the recognizer on `nn-test` using these parameters. The results for this test are shown in Table 5-5.

In order to find out whether we could have obtained a lower WER on `nn-test`, we tested with the entire series of **cprune** values. The results of these tests are shown in Figure 5-11. In this case, only the **vprune** = 15 series was done. This experiment confirms that the lowest WER is obtained for **cprune** = 12.

5.3.3 Iterative Training

One factor that limits the accuracy of the confusion probability estimates is the quality of the alignments. This, in turn, is largely determined by the quality of the alignment CFST C_a. If we assume that a newly estimated C is more accurate than the C_a that was used in training it, in the sense that it predicts a higher probabillity for the training set, then we should be able to obtain an even more accurate CFST by using C as an alignment CFST and reestimating the probabilities. This can be repeated indefinitely and, if the assumption of monotonically increasing training set probability is correct, the process should converge at a local optimum.

Figures 5-12 and 5-13 compare the number of arcs per phone and word error rates on `nn-dev` for the first, second, and third iterations of CFST training using the forced paths method. In the second and third iterations, the forced paths were not remade, but only

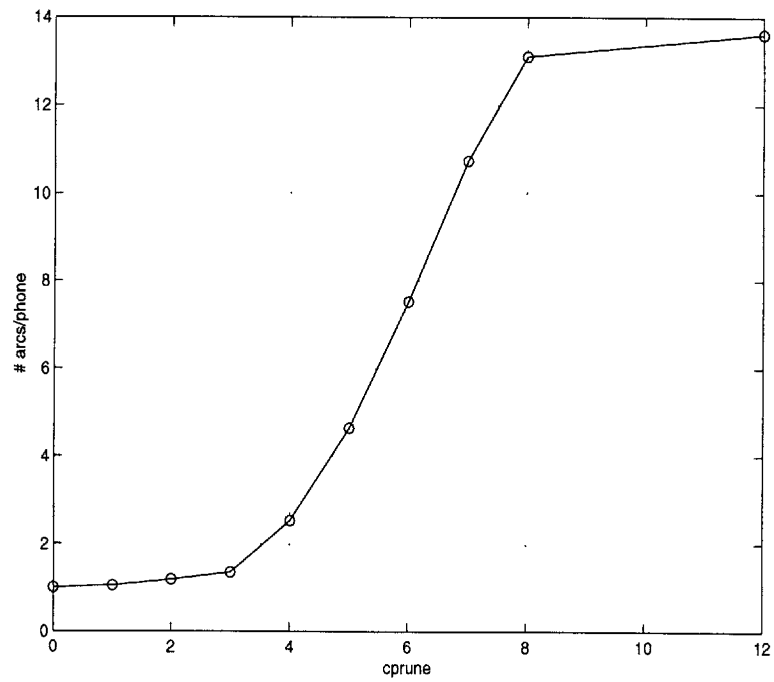

Figure 5-9: Average number of arcs per phone as a function of CFST pruning threshold, using the forced paths method.

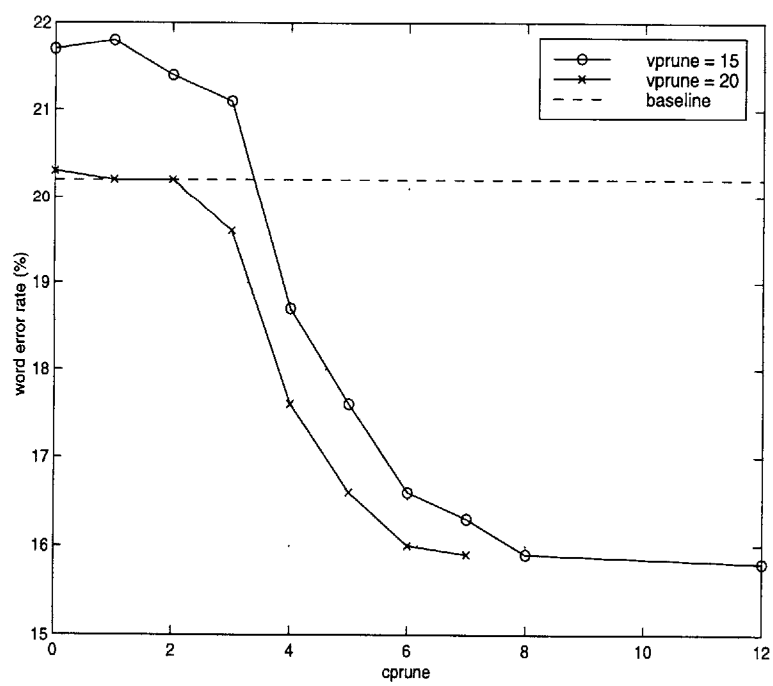

Figure 5-10: Word error rate on nn-dev as a function of CFST pruning threshold, using the forced paths method.

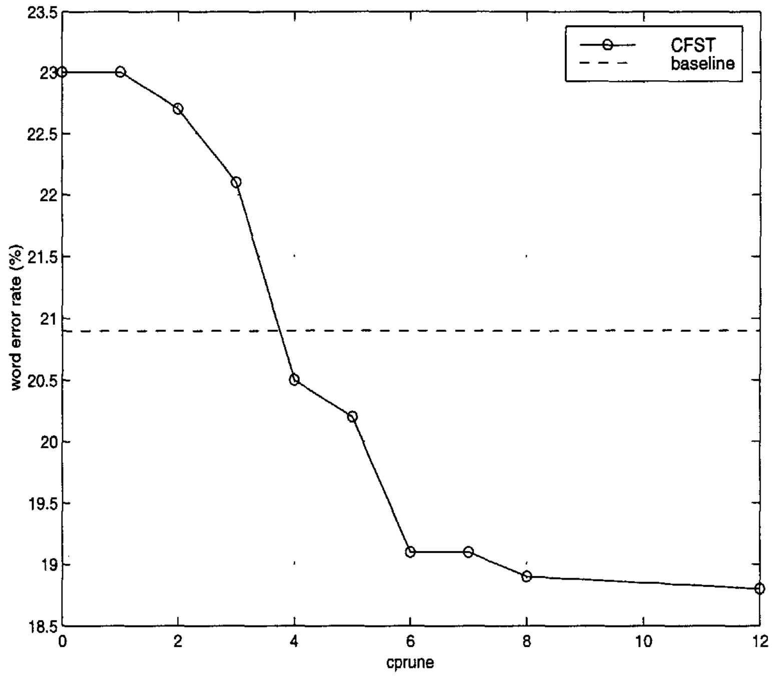

Figure 5-11: Word error rate on `nn-test` as a function of CFST pruning threshold, using the forced paths method.

realigned with an alignment CFST from the previous iteration. In building the alignment CFST's, the existing CFST was padded with `pad-2` with $n_p = 1$. As these figures show, we do not obtain the desired performance gains. This may be because the first-iteration confusion probabilities are already quite constrained, since the forced paths were made using a pruned CFST, so that retraining does not add a significant amount of information. Since the iterative training procedure did not appear to significantly affect the performance on `nn-dev`, this experiment was not repeated on `nn-test`.

It is possible that a more refined iteration process would achieve better results. For example, a better method of smoothing, instead of the simple padding method we used, may have helped by creating more realistic alignment CFST's. A training algorithm that considers multiple possible transcriptions or recomputes the forced paths between iterations may also be considered in the future.

5.3.4 Comparison of Smoothing Approaches

In order to compare the effects of different smoothing approaches, we performed a small experiment with a single medium-sized CFST, using each of the three methods described in Section 5.2.3, as well as no smoothing at all. Table 5-6 shows the performance obtained on `nn-dev` in these tests. In the case of `pad-2`, we used $n_p = 0.5$, so that, while all confusions are represented, those that occurred in the training data receive higher probabilities than

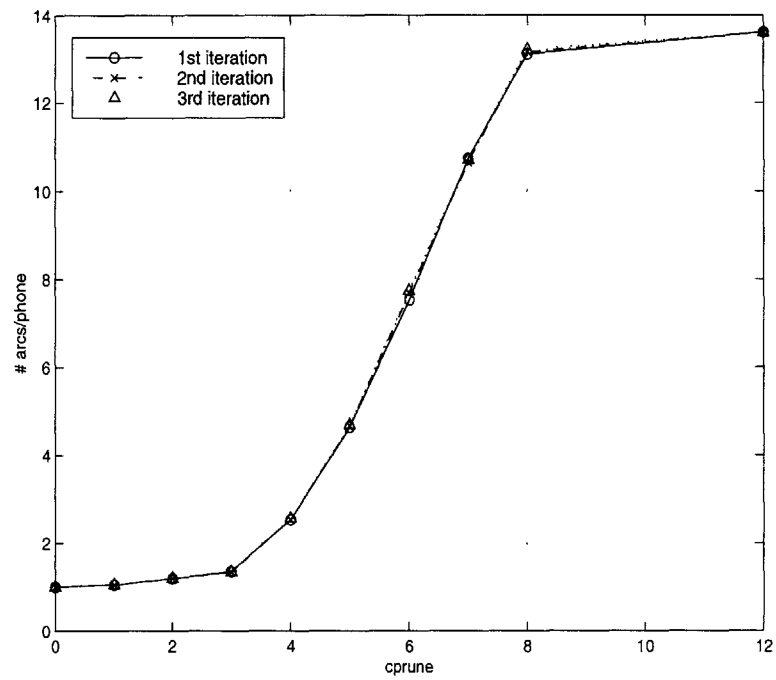

Figure 5-12: Average number of arcs per phone as a function of CFST pruning threshold, using the forced paths method, for 1st, 2nd, and 3rd iterations.

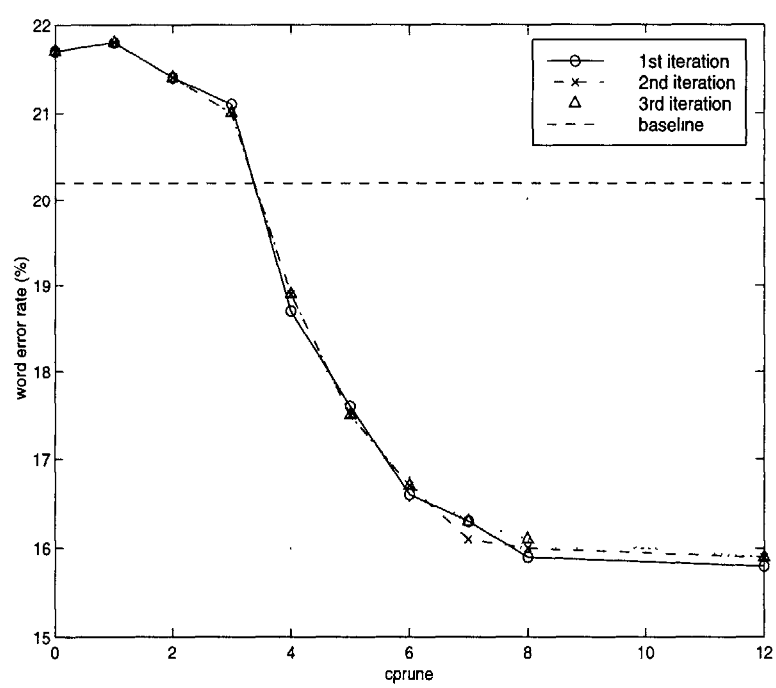

Figure 5-13: Word error rate on `nn-dev` as a function of CFST pruning threshold, using the forced paths method, for 1st, 2nd, and 3rd iteration.

Method	S	I	D	WER
No Smoothing	10.9	2.9	3.7	17.5
Pad-1	10.9	2.9	3.8	17.6
Pad-2	11.0	2.8	3.8	17.6
Linear Interpolation	11.0	3.0	3.7	17.7

Table 5-6: Comparison of error rates on `nn-dev` using various types of smoothing. In all cases, vprune = 15 and `wtw` = 4.

S	I	D	WER
10.6	3.6	4.0	18.3

Table 5-7: Error rates obtained on `nn-test` using a CFST, forced paths method, with interpolated acoustic models. In this test, cprune = 12, vprune = 15, `wtw` = 2.

those that did not. All of the CFST's are derived from C_{FP} with cprune = 5. The differences between the error rates are very small: The word error rates range from 17.5% with no smoothing to 17.7% with linear interpolation, a relative difference of just over 1%. It is not clear whether this difference is significant. This experiment was therefore also not repeated on `nn-test`. However, in future work, we may refine the smoothing approach, for example by optimizing the smoothing parameters on development data.

5.3.5 Recognition with Interpolated Models and CFST's

In Chapter 3, we showed that interpolated acoustic models can reduce the word error rate of a recognizer. In this chapter, we have shown that adding a CFST to the baseline recognizer can also reduce the word error rate. We now consider combining the two methods; that is, adding a CFST to a recognizer using interpolated acoustic models.

Figure 5-14 compares the error rates obtained on the `nn-dev` set using a CFST with cprune $\in [0, 12]$ and either the baseline acoustic models or the interpolated acoustic models with $w_{nn} = 0.54$. The interpolated models obtain much better performance than the baseline models; the lowest WER with the interpolated models is 15.2% at cprune = 12, a 24.8% improvement over the baseline recognizer and a 3.8% improvement over using just the CFST. Using the interpolated models with cprune = 12 on the `nn-test` set, we obtain a WER of 18.3%, a 12.4% reduction from baseline (see Table 5-7).

We again wanted to know whether this result is the best we could obtain on `nn-test`. Figure 5-15 shows a series of results obtained on the `nn-test` set using the interpolated acoustic models and the same set of CFST's as in Figure 5-11. For comparison, the figure also reproduces the results of Figure 5-11. For all values of cprune, the recognizer with the interpolated acoustic models achieves a lower word error rate. The lowest WER obtained

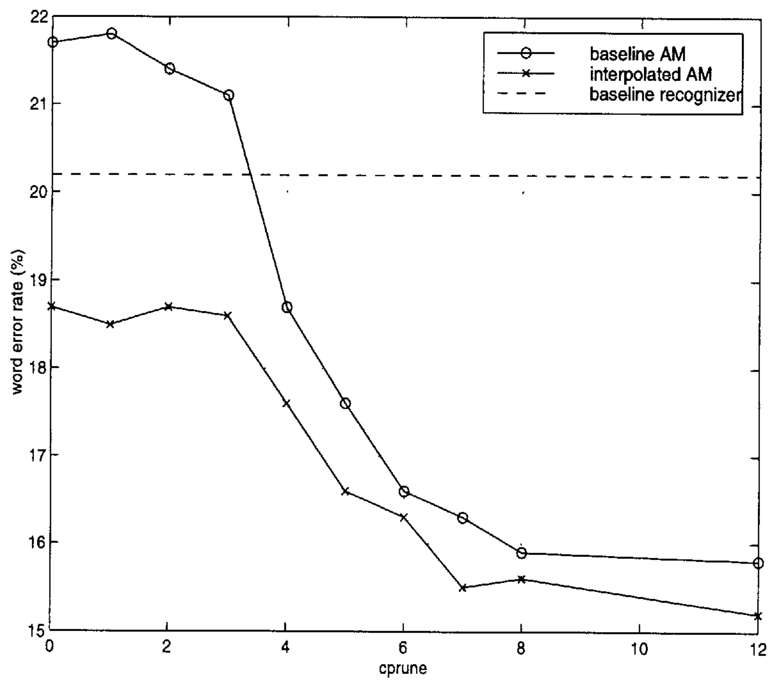

Figure 5-14: Word error rate on nn-dev as a function of CFST pruning threshold, using either the baseline acoustic models or interpolated acoustic models with $w_{nn} = 0.54$.

with the interpolated models is 18.1%, a 13.4% reduction from the baseline recognizer. This is somewhat better than the results obtained using the parameters that gave the best performance on nn-dev, but it is still not as large a reduction as we obtained on nn-dev.

5.4 Summary

The main results of this chapter for the nn-dev and nn-test sets are summarized in Table 5-8. In the first training run, in which transcriptions are generated by a phonetic recognizer, we obtain up to a 9.9% reduction in word error rate on nn-dev and up to a 2.9% reduction on nn-test, with respect to the baseline recognizer. By using the resulting CFST to create forced paths, we obtain relative reductions in word error rate of up to 21.8% on the development set and 9.6% on the test set. Finally, using this CFST and the interpolated acoustic models from Chapter 3, the WER is further reduced to 15.2% on nn-dev and 18.3% on nn-test, representing 24.8% and 12.4% relative reductions from the baseline, respectively. Table 5-8 also contains the results of Chapter 3, obtained using the interpolated acoustic models alone, for comparison. As the table shows, using a CFST, either alone or in combination with interpolated acoustic models, achieves a larger improvement than using the interpolated acoustic models alone.

The work described in this chapter demonstrates that some of the pronunciation patterns of non-native speakers as a group can be modeled using a simple single-state weighted

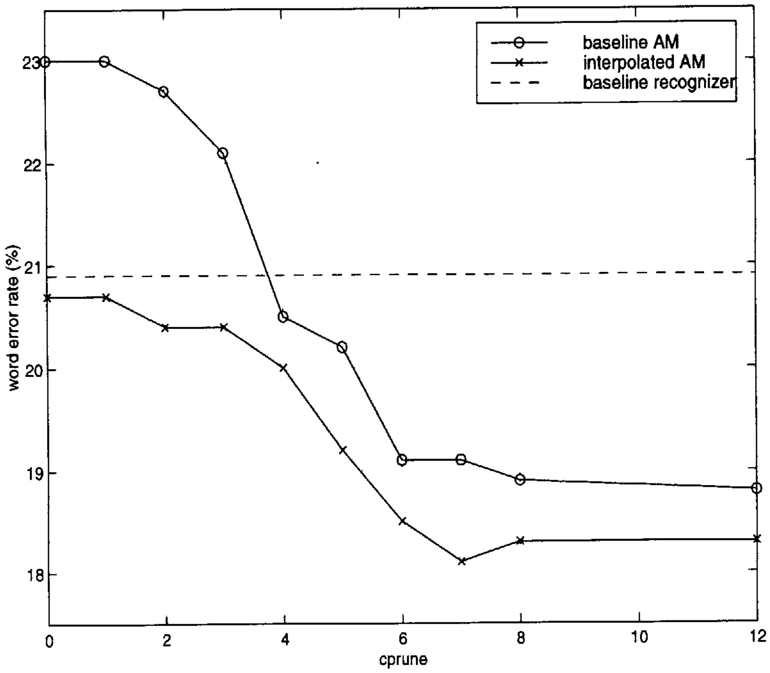

Figure 5-15: Word error rate on `nn-test` as a function of CFST pruning threshold, using either the baseline acoustic models or interpolated acoustic models with $w_{nn} = 0.54$.

	nn-dev				nn-test			
Method	**S**	**I**	**D**	**WER**	**S**	**I**	**D**	**WER**
Baseline	12.1	4.3	3.8	20.2	12.4	3.9	4.6	20.9
PR	11.1	3.4	3.7	18.2	12.2	3.3	4.8	20.3
FP	10.1	2.7	3.0	15.8	11.4	3.2	4.2	18.8
Interpolated AM	9.8	3.8	2.7	16.3	11.5	3.6	4.1	19.2
FP+Interp. AM	8.9	3.6	2.7	15.2	10.6	3.6	4.0	18.3

Table 5-8: Summary of the main results on `nn-dev` and `nn-test`. The methods compared are: Baseline recognizer; PR = baseline recognizer with CFST trained using phonetic recognition method; FP = baseline recognizer with CFST trained using forced paths method; Interpolated AM = baseline recognizer using interpolated acoustic models with $w_{nn} = 0.54$; and FP+Interp. AM = CFST trained using forced paths method + interpolated acoustic models.

FST, representing the likelihoods of phone substitutions, deletions, and insertions. Furthermore, the confusion probabilities can be automatically trained, using no prior knowledge to initialize their values.

There are many possible refinements to the methods we have described. For example, it would be interesting to attempt to iteratively train the confusion probabilities via an EM algorithm, and to refine the smoothing methods used for both the alignment and recognition CFST's. The CFST approach has the potential to be used for native speech as well, for example for discovering phonological rules rather than manually specifying them. This and other applications and extensions of this approach are further discussed in Chapter 7.

Chapter 6

Effect of the Language Model on Recognition Performance

One factor that may affect speech recognition performance is the degree of similarity between the language model used by the recognizer and the "language model" of the test speakers, i.e. the distribution of words and phrases in the test utterances. If non-native speakers use words, or combinations of words, with different probabilities than do native speakers, the language model (LM) scores for their utterances are likely to be lower than those for native utterances. As a result, non-native utterances may be less likely to be recognized correctly.

One example of this problem is given by the sentence *How about the Ann Arbor?*, spoken by a non-native speaker in the corpus. This sentence is likely to receive a poor LM score since it is uncommon to place a determiner (*the*) before a city name. A native speaker would probably have been more likely to say, *How about Ann Arbor?* Using a word bigram LM trained on the native training set, the log probability of *Ann Arbor* following *how about* (represented as one word in the bigram model) is -6.6; for *Ann Arbor* following *the*, the log probability is -12.8. This is an example of a non-native speaker receiving a low LM score because of incorrect syntax.

There are several other factors that may cause non-native utterances to receive lower LM scores. First, non-native speakers may use different distributions of words (i.e. different unigram probabilities) or combinations of words (i.e. different bigram or higher n-gram probabilities) than native speakers do, because of either linguistic difficulties or "practical" considerations. An example of a "practical" consideration that may cause such differences in distributions is that non-native speakers may be more likely to ask about the weather in cities outside the U.S.; a word n-gram trained on mostly native sentences would therefore assign lower probabilities to such queries. Because of the recognizer's poorer performance on non-native speakers, they may also be more likely than native speakers to make "repairs" in the conversation. Examples of repairs from the non-native utterances in the corpus are:

I said that's about it, thank you.
No, this is not a city, I said good bye.

I was talking about Rochester.
Do you understand my question?

Such sentences, if they are in fact less common among native speakers than among non-native speakers, should receive lower scores and therefore tend to worsen recognition performance.

In short, there are many factors that may cause us to expect non-native speakers to receive lower LM scores than native speakers. In the following sections we present some measurements intended to determine whether or not this effect is seen in the data, and to what extent it affects recognition performance.

6.1 Experimental Setup

The recognizer used for the language model experiments is similar to the baseline recognizer described in Chapter 2, but differs in the following respects[1]:

- The phonetic inventory consists of 106 labels, and includes stressed and unstressed variants of many of the labels.
- The lexicon consists of 1892 words (rather than 1956) and has unweighted arcs.
- The training set used for all of the language models contains 19,081 in-vocabulary, native utterances.

A recognizer using this label set and vocabulary has been reported on previously [11]. The test sets used in this chapter are a native set of 1041 utterances and a non-native set of 1552 utterances.

6.2 Set Log Probabilities

One measure of the degree of difficulty of predicting the words in a given set using a given LM is the entropy of the set with respect to the LM. The estimated entropy H_e of a set containing N words $\{w_1, w_2, \ldots w_N\}$ with respect to a given LM is defined as

$$H_e = -\frac{1}{N} log_2 P_e(w_1, w_2, \ldots, w_N), \tag{6.1}$$

where $P_e(w_1, w_2, ..., w_N)$ is an estimate of the probability of the given sequence of words given the language model. In our case, we will talk about per-word negative log probability

[1]The use of a different recognizer in these experiments is a chronological artifact and is unrelated to the purpose of the experiments.

Test Set	Unigram PNLP (Perplexity)	Bigram PNLP (Perplexity)	Trigram PNLP (Perplexity)	WER
native	4.87 (130.17)	2.78 (16.19)	2.67 (14.40)	11.9
non-native	4.99 (147.47)	3.04 (20.83)	2.96 (19.35)	30.0

Table 6-1: PNLP's, perplexities, and WER's for the native and non-native test sets.

(PNLP) rather than entropy since our software uses natural logs. The PNLP is simply the entropy multiplied by a factor of $ln2$. A quantity closely related to entropy is the set perplexity, defined as

$$B = 2^{H_e} = P_e(w_1, w_2, \ldots, w_N)^{-1/N}, \tag{6.2}$$

Table 6-1 shows the PNLP's and perplexities of the native and non-native test sets with respect to word unigram, bigram, and trigram models. The table also shows the word error rates obtained by the recognizer on those sets. For all of the language models, the non-native sets have larger PNLP's than the native sets. This matches our expectation that a LM should perform better on data similar to its training data. The fact that the WER for these sets increases with increasing PNLP is also reasonable, as we expect the recognizer to have more difficulty recognizing low-probability (high PNLP) utterances. However, the degree of the correlation between PNLP and error rate is not clear from the limited data in this table. In the following section, we examine the per-speaker PNLP's and error rates to gain a more detailed picture of this relationship.

6.3 Per-Speaker PNLP's

Figures 6-1 and 6-2 show scatterplots of the per-speaker WER versus PNLP for the native and non-native test sets. We chose to plot the PNLP's, rather than perplexities, because the PNLP's have a smaller dynamic range, which makes the plots more suitable for visual inspection. For these and all subsequent plots, the PNLP's have been measured using the word bigram language model. The plots seem to indicate that, in general, WER and PNLP are positively correlated. In particular, the linear correlation coefficients between WER and PNLP for the native and non-native speakers are 0.4378 and 0.4850, respectively, both of which are significant at the 1% significance level.

Besides the general trend of positive correlation, there appears to be a disproportionately large subset of speakers in both sets that receive zero WER, regardless of PNLP. This disproportionate number of of zero-WER speakers can be seen in Figure 6-3, which shows the distributions of error rate and PNLP within the test sets. The PNLP distributions appear to have a Gaussian-like shape.

Figure 6-4 shows the average WER versus PNLP for the native and non-native speakers, along with superimposed least-squares lines fitted to the data. In fitting the line to the native data, the rightmost datum (6.75, 0.0) was ignored since it was based on only one

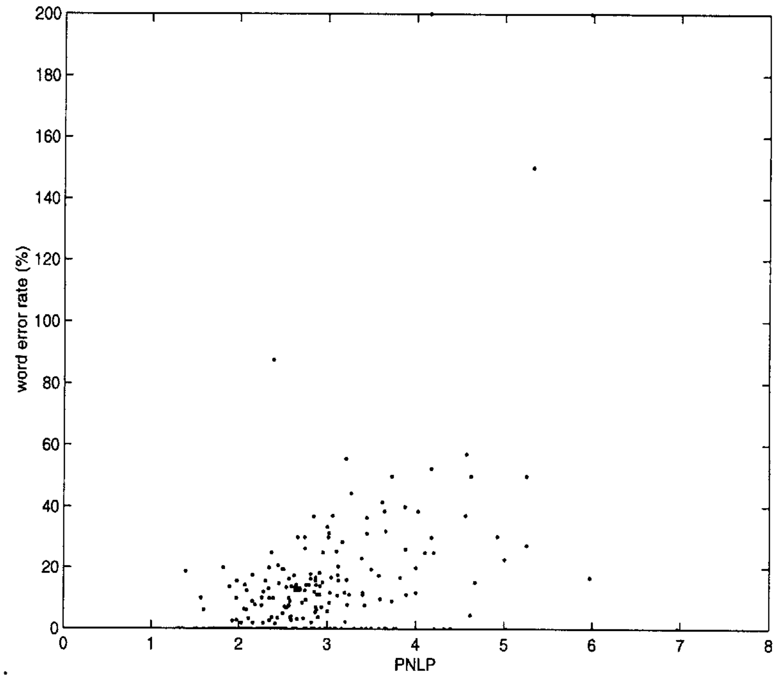

Figure 6-1: Scatterplot of WER versus PNLP for the native test set.

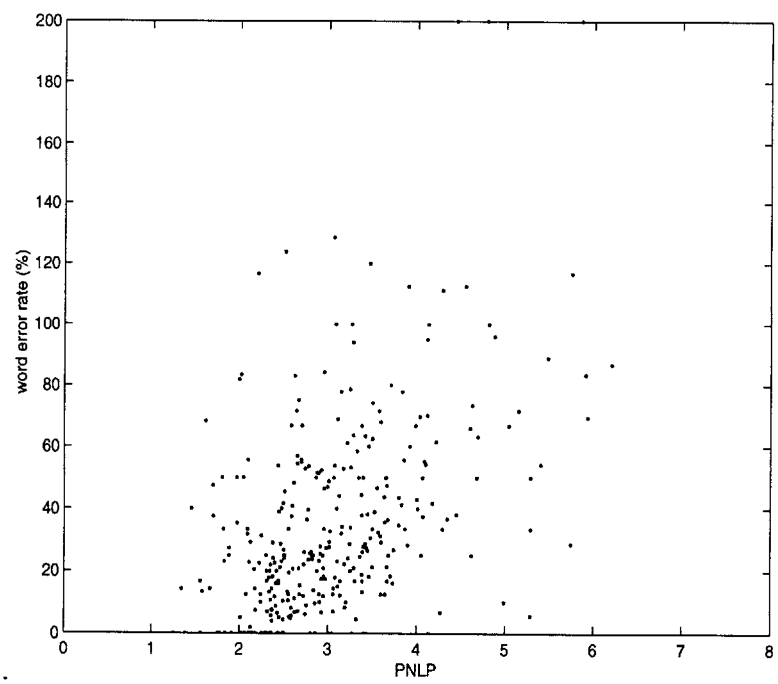

Figure 6-2: Scatterplot of WER versus PNLP for the non-native test set.

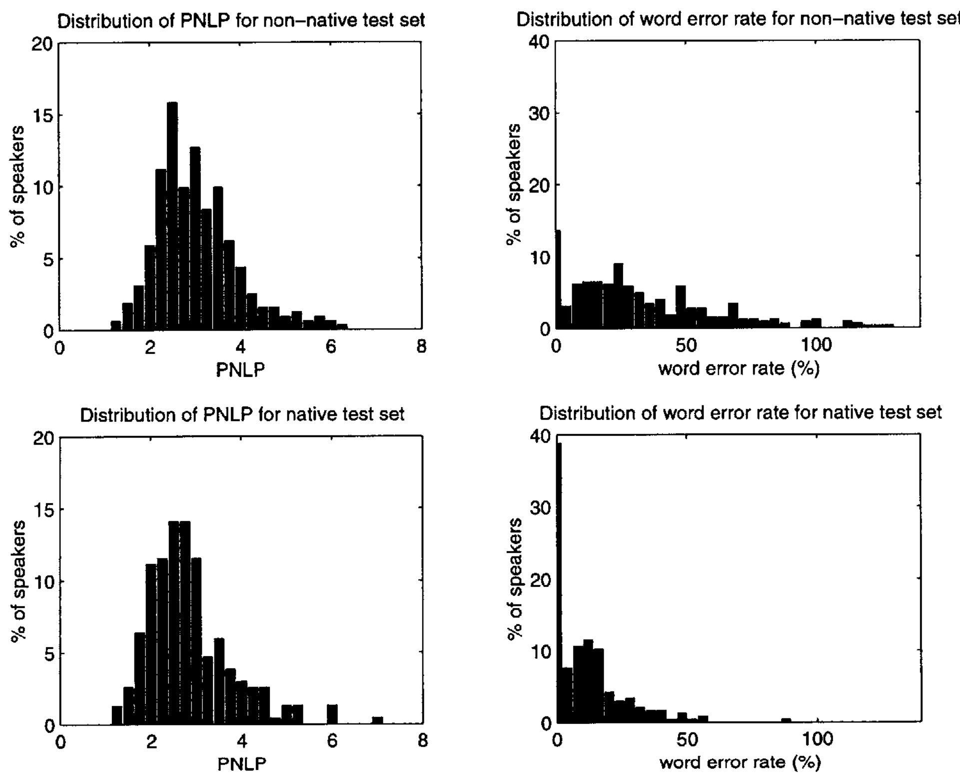

Figure 6-3: Histograms of WER and PNLP for the native and non-native test sets.

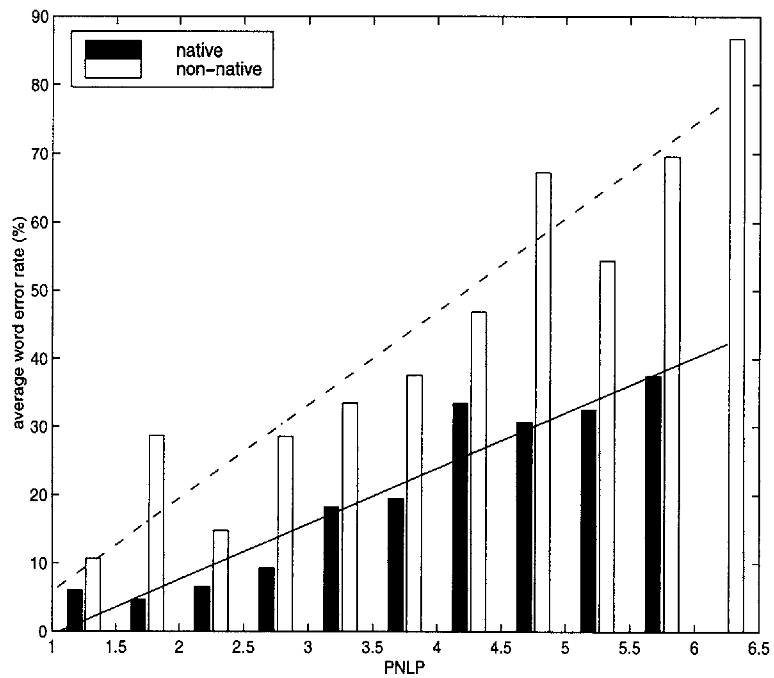

Figure 6-4: Average WER versus PNLP for native and non-native speakers, along with best-fit lines.

uterance and was therefore considered unreliable. The relationship between WER and PNLP is much clearer in this plot: The higher the PNLP, the larger the average WER for both native and non-native speakers.

As Figure 6-4 shows, the WER for non-native speakers is always much higher than that for native speakers, regardless of the PNLP. Specifically, we can calculate the mean percent difference D_p in average WER between native and non-native speakers at the same PNLP. We use the following method to calculate this mean: Let b_{max} be the highest PNLP bin number for which there are both native and non-native data; e_b^a and e_b^{nn} the average WER in bin b for native and non-native speakers, respectively; N_b^a and N_b^{nn} the number of native and non-native speakers, respectively, in bin b; and N^a and N^{nn} the total number of native and non-native speakers, respectively. Then:

$$D_p = 100 \sum_{b=1}^{b_{max}} w_b\left(\frac{e_b^{nn} - e_b^a}{e_b^a}\right), where \quad w_b = \frac{1}{2}\left(\frac{N_b^{nn}}{N^{nn}} + \frac{N_b^a}{N^a}\right) \tag{6.3}$$

In other words, each bin is weighted by the number of speakers in that bin, represented by the average of the proportions of native and non-native speakers in that bin. For the data shown in Figure 6-4, we find that D_p = 167%, with a standard deviation of σ_{D_p} = 40%. This is within error of the overall percent difference in WER between native and non-native speakers, which is 152%. Within the resolution of this analysis, therefore, differences in entropy do not account for a significant amount of the difference in WER between native and non-native speakers.

6.4 Summary

Our analysis of the effects of language model differences between native and non-native speakers demonstrates the following:

- As expected, native speakers have lower entropy than non-native speakers with respect to word n-gram models trained on native utterances.
- Word error rate generally increases with entropy for the test sets that we examined.
- Differences in entropy between native and non-native speakers do not account for a significant portion of the difference in WER between the two sets.

Chapter 7

Conclusion

This thesis has explored several aspects of non-native speech and its interaction with the acoustic, lexical, and language modeling components of a speech recognizer. This work differs from most previous research on non-native speech in that we have treated all non-native speakers as a group, rather than dividing them into separate accent classes. This approach is well-suited to a speech recognition system with a great deal of native speech data, but little non-native data, at its disposal.

The main goal of this thesis has been to understand the ways in which the speech of non-native speakers of American English, as a group, may differ from that of native speakers, and to attempt to modify a speech recognizer to better handle non-native speech input. This work was done using the SUMMIT speech recognizer in the JUPITER weather domain. The following sections summarize our main results and suggest directions for future work.

7.1 Summary

In the area of acoustic modeling, we have shown that a reduction in the word error rate on non-native speech can be achieved by interpolating two sets of acoustic models, one trained on only native speech and the other trained on only non-native speech. Using interpolated models with a weight of 0.46 on the native models and 0.54 on the non-native models, we have achieved an 19.3% relative reduction in word error rate on a non-native development set, and an 8.1% relative reduction on a non-native test set, when compared to a baseline model trained on pooled native and non-native data.

In the area of lexical modeling, we began with an informal study of manual phonetic transcriptions. By manually transcribing small subsets of the native and non-native utterances in the JUPITER corpus, we have been able to deduce some of the general characteristics of the non-native data. As expected, the non-native utterances appear to contain a larger number of non-American phones such as the tense vowels [i, o, uu, a, o] and non-American varieties of the liquids /r/ and /l/. Non-native speakers also appear to replace lax vowels

Method	S	I	D	WER
Baseline	12.4	3.9	4.6	20.9
Interpolated AM	11.5	3.6	4.1	19.2
CFST, FP method	11.4	3.2	4.2	18.8
Interp. AM + CFST, FP	10.6	3.6	4.0	18.3

Table 7-1: Error rates obtained on `nn-test` using the baseline recognizer, interpolated acoustic models (AM), a CFST trained using the forced paths (FP) method, and both interpolated AM and a CFST. The CFST in this table is C_{FP}, unpruned and padded with the `pad-1` method.

with tense vowels, reduced vowels with full vowels, and voiced consonants with voiceless consonants to a greater extent than native speakers; while native speakers have a greater tendency to replace pure vowels with diphthongs and to delete reduced vowels.

We have also investigated a method for the automatic acquisition of non-native pronunciation patterns. Using a one-state weighted finite-state transducer within an FST-based recognizer, we model the phonetic substitutions, deletions, and insertions present in non-native speech. We refer to this FST as a confusion FST or CFST. The context-independent probabilities of the confusions are estimated from alignments of automatically generated phonetic transcriptions with the baseline lexicon. We have shown that a two-step training process can yield much greater improvement than a single probability estimation step. In the two-step process, an initial CFST C_{PR} is first trained from alignments of phonetic recognizer hypotheses. This CFST is used in a word recognizer to create forced paths, which are then aligned with the baseline lexicon using the same initial CFST. The probability estimates derived from this second alignment step yield a more compact CFST, C_{FP}, and lower word error rates than either the baseline recognizer or a recognizer using C_{PR}. The word error rates obtained using C_{FP} are up to 10.0% lower than the baseline. Using both C_{FP} and interpolated acoustic models, the word error rate is further reduced to 12.4% below baseline. Table 7-1 summarizes the results on the non-native test set with each method.

Finally, in the area of language modeling, we have examined the difference between native and non-native speakers using measurements of their per-word negative log probabilities (PNLP's), a measure closely related to perplexity. We have found that, as expected, the set PNLP of a non-native test set is higher than that of a native test set, and the distributions of per-speaker PNLP's indicate that word error rate is positively correlated with PNLP. However, within the resolution of our measurements, the difference in PNLP between native and non-native speakers does not account for a significant portion of the difference in word error rates obtained on these sets.

This work has shown that it is possible to improve recognition accuracy on non-native speech while treating all non-native speakers as a single group. This is potentially useful for the development of speech recognition systems in domains in which non-native data are limited, or in which it is undesirable for other reasons, such as computational or time considerations, to separate non-native speakers into specific accents.

7.2 Future Work

We believe that, of the methods we have attempted, the automatic acquisition of pronunciation patterns has the greatest potential for future work. We have investigated only the simplest form of CFST, a one-state FST modeling only context-independent probabilities, incorporated directly into the baseline recognizer. One natural extension, given sufficient training data, would be to model context-dependent probabilities. In this work, we have also considered only confusions between two identical label sets. Another application of CFST's would be to derive phonological rules from training data, using a phonemic label set on one side and a phonetic label set on the other side. These methods can be applied more widely, not only to accent-specific modeling but to lexical and phonological modeling in general. We plan to continue to investigate these possibilities.

One of the drawbacks of our implementation of CFST's is a large increase in running time and memory requirements. However, our setup seems to contain redundant information: Since the acoustic models have been trained from forced paths created with the baseline recognizer, they implicitly encode some of the confusability between phones. There are at least two ways to attempt to reduce this redundancy. First, it may be beneficial to retrain the acoustic models, using forced paths created with a CFST. This would hopefully create smaller, more accurate models, leaving the modeling of confusability to the CFST. Second, since a CFST can greatly expand the number of allowable pronunciations per word, it may not be necessary to consider the entire acoustic-phonetic graph during recognition. Instead, the acoustic-phonetic graph can be collapsed in an initial phonetic recognition pass to a single path (as in [20]) or an N-best graph.

Another area in which further study is needed is language modeling. A more detailed study than the one we have done may reveal more specific differences between native and non-native speakers, and suggest possible modifications to the language model. As a starting point, it would be interesting to perform a similar experiment to the one we have performed with the acoustic models; that is, to interpolate a native language model with a non-native one.

In this thesis, we have achieved an improved speaker-independent baseline recognizer for non-native speakers. Although the results show that it is possible to obtain significant gains in performance by modeling all non-native speakers as a single group, there are many accent- and speaker-specific phenomena that could be better modeled. While separate accent-specific models for a large number of non-native accents may not be feasible, a system that uses a single set of baseline models for all non-native speakers and adapts to the characteristics of a particular test speaker may be a good way to address the complexities of non-native speech. For a conversational system like JUPITER, in which the speaker usually remains constant throughout a multi-utterance interaction, an incremental adaptation approach may be particularly suitable.

Finally, an area that this thesis does not address is that of accent-independent recognition. We have concentrated on improving the performance of a speech recognizer on non-native speakers. However, a deployed recognizer such as the one used in the JUPITER conversational system must be able to appropriately handle both native and non-native speech input. While much work is still needed to bring the recognition of non-native speech to the level of native speech recognition, any accent-specific methods must ultimately be incorporated into an accent-independent recognizer.

Bibliography

[1] L. M. Arslan and J. H. Hansen. Frequency characteristics of foreign accented speech. In *Proc. ICASSP '97*, pages 1123–1126, Munich, Germany, April 1997.

[2] L. R. Bahl, F. Jelinek, and R. L. Mercer. A maximum likelihood approach to continuous speech recognition. *IEEE Trans. Pattern Analysis and Machine Intelligence*, PAMI 5(2):179–190, 1983.

[3] V. Beattie, S. Edmondson, D. Miller, Y. Patel, and G. Talvola. An integrated multi-dialect speech recognition system with optional speaker adaptation. In *Proc. Eurospeech '95*, pages 1123–1126, Madrid, Spain, September 1995.

[4] C. M. Bishop. *Neural Networks for Pattern Recognition.* Clarendon Press, Oxford, 1995.

[5] J. W. Chang. *Near-Miss Modeling: A Segment-Based Approach to Speech Recognition.* Ph. D. thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, June 1998.

[6] S. F. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. Technical Report TR-10-98, Harvard University, 1998.

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

[8] V. Diakoloukas, V. Digalakis, L. Neumeyer, and J. Kaja. Development of dialect-specific speech recognizers using adaptation methods. In *Proc. ICASSP '97*, pages 1455–1458, Munich, Germany, April 1997.

[9] J. Glass, J. Chang, and M. McCandless. A probabilistic framework for feature-based speech recognition. In *Proc. ICSLP '96*, volume 4, pages 2277–2280, Philadelphia, PA, October 1996.

[10] J. R. Glass. *Finding Acoustic Regularities in Speech: Applications to Phonetic Recognition.* Ph. D. thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, 1988.

[11] J. R. Glass, Hazen, T. J., and I. L. Hetherington. Real-time telephone-based speech recognition in the JUPITER domain. In *Proc. ICASSP '99*, pages 61–64, Phoenix, AZ, March 1999.

[12] J. R. Glass and T. J. Hazen. Telephone-based conversational speech recognition in the JUPITER domain. In *Proc. ICSLP '98*, pages 1327–1330, Sydney, Australia, December 1998.

[13] T. J. Hazen and A. K. Halberstadt. Using aggregation to improve the performance of mixture Gaussian acoustic models. In *Proc. ICASSP '98*, pages 653–656, Seattle, WA, May 1998.

[14] X. Huang, M.-Y. Hwang, L. Jiang, and M. Mahajan. Deleted interpolation and density sharing for continuous hidden markov models. In *Proc. ICASSP '96*, pages 885–888, Atlanta, GA, May 1996.

[15] J. Humphries, P. Woodland, and D. Pearce. Using accent-specific pronunciation modelling for robust speech recognition. In *Proc. ICSLP '96*, pages 2324–2327, Philadelphia, PA, October 1996.

[16] A. James and J. Leather. *Second-Language Speech: Structure and Process.* Mouton de Gruyter, Berlin, 1997.

[17] F. Jelinek. Self-organized language modeling for speech recognition. In *Readings in Speech Recognition*, pages 450–506. Morgan Kaufmann Publishers, San Mateo, CA, 1990.

[18] R. Lau. *Subword Lexical Modelling for Speech Recognition.* PhD thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, May 1998. URL http://www.raylau.com/thesis/thesis.pdf.

[19] S. C. Lee. *Probabilistic Segmentation for Segment-Based Speech Recognition.* M. S. thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, May 1998.

[20] S. E. Levinson, A. Ljolje, and L. G. Miller. Continuous speech recognition from a phonetic transcription. In *Proc. ICASSP '90*, pages 93–96, Alburquerque, NM, April 1990.

[21] W. K. Liu and P. Fung. Fast accent identification and accented speech recognition. In *Proc. ICASSP '99*, pages 221–224, Phoenix, AZ, March 1999.

[22] P. Nguyen, Ph. Gelin, J.-C. Junqua, and J.-T. Chien. N-best based supervised and unsupervised adaptation for native and non-native speakers. In *Proc. ICASSP '99*, pages 173–176, Phoenix, AZ, March 1999.

[23] P. Placeway, R. Schwartz, P. Fung, and L. Nguyen. The estimation of powerful language models from small and large corpora. In *Proc. ICASSP '93*, pages 33–36, Minneapolis, MN, April 1993.

[24] L. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. of the IEEE*, 77:257–286, 1989.

[25] L. Rabiner and B.-H. Juang. *Fundamentals of Speech Recognition.* Prentice-Hall, Englewood Cliffs, NJ, 1993.

[26] M. D. Riley and A. Ljolje. Automatic generation of detailed pronunciation lexicons. In C.-H. Lee, F. K. Soong, and K. K. Paliwal, editors, *Automatic Speech and Speaker Recognition*. Kluwer Academic Publishers, Boston, 1996.

[27] M. Ruhlen. *A Guide to the Languages of the World*. Ruhlen, Stanford, California, 1976.

[28] N. Ström, L. Hetherington, T. J. Hazen, E. Sandness, and J. Glass. Acoustic modeling improvements in a segment-based speech recognizer. Submitted to ASRU '99, Keystone, Colorado, December 1999.

[29] C. Teixeira, I. Trancoso, and A. Serralheiro. Recognition of non-native accents. In *Proc. Eurospeech '97*, pages 2375–2378, Rhodes, Greece, September 1997.

[30] P.H. Winston. *Artificial Intelligence*. Addison-Wesley, Reading, MA, third edition, 1992.

[31] G. Zavaliagkos, R. Schwartz, and J. Makhoul. Adaptation algorithms for BBN's phonetically tied mixture system. In *Proc. ARPA Spoken Language Systems Technology Workshop '95*, pages 82–87, Austin, TX, Jan 1995.

[32] V. Zue, J. Glass, D. Goodine, M. Phillips, and S. Seneff. The SUMMIT speech recognition system: Phonological modelling and lexical access. In *Proc. ICASSP '90*, pages 49–52, Alburquerque, NM, April 1990.

[33] V. Zue, S. Seneff, J. R. Glass, I. L. Hetherington, E. Hurley, H. Meng, C. Pao, J. Polifroni, R. Schloming, and P. Schmid. From interface to content: Translingual access and delivery of on-line information. In *Proc. Eurospeech '97*, pages 2227–2230, Rhodes, Greece, September 1997.

Appendix A

Label Sets

The following tables list the label sets used in the acoustic and pronunciation modeling work in this thesis. Table A-1 lists the labels used in the baseline recognizer of Chapters 2–5, and Table A-2 lists the additional labels used in the manual transcriptions in Chapter 4.

Label	IPA	Example/Description	Label	IPA	Example/Description
aa	[ɑ]	b*o*b	n	[n]	*noon*
ae	[æ]	b*a*t	ng	[ŋ]	si*ng*
ah	[ʌ]	b*u*t	nn	[n̩]	butt*on*
aw	[ɑ^{w}]	b*ou*t	nx	[r̃]	wi*nn*er
ax	[ə]	*a*bout	ow	[o^{w}]	b*oa*t
ay	[ɑ^{y}]	b*i*te	oy	[o^{y}]	b*oy*
b	[b]	*b*ee	p	[p^{h}]	*p*ea
bcl	[b$^{\square}$]	b closure	p-	[p]	s*p*ot
ch	[č]	*ch*oke	pcl	[p$^{\square}$]	p closure
d	[d]	*d*ay	r	[r]	*r*ay
dcl	[d$^{\square}$]	d closure	rx	[ɚ]	butt*er*, ca*r*
dh	[ð]	*th*en	s	[s]	*s*ea
dr	—	*dr*y	sh	[š]	*sh*e
dx	[ɾ]	mu*dd*y	t	[p^{h}]	*t*ea
eh	[ɛ]	b*e*t	t-	[t]	s*t*op
er	[ɝ]	b*ir*d	tcl	[t$^{\square}$]	t closure
ey	[e^{y}]	b*ai*t	th	[θ]	*th*in
f	[f]	*f*in	tq	[ʔ]	glottalized t (bu*tt*on)
g	[g]	*g*ay	tr	—	*tr*y
gcl	[g$^{\square}$]	g closure	u	—	filled-pause [ʌ] (*u*m)
hh	[h]	*h*ay	uh	[ʊ]	b*oo*k
ih	[ɪ]	b*i*t	uw	[u^{w}]	b*oo*t
ix	[ɨ]	deb*i*t	ux	[ü]	t*oo*t
iy	[i^{y}]	b*ee*t	v	[v]	*v*an
jh	[ǰ]	*j*oke	w	[w]	*w*ay
k	[k^{h}]	*k*ey	y	[y]	*y*acht
k-	[k]	s*k*i	z	[z]	*z*one
kcl	[k$^{\square}$]	k closure	zh	[ž]	mea*s*ure
l	[l]	*l*ay	-	—	utterance initial/final silence
ll	[l̩]	bott*le*, coo*l*	_	—	interword silence
m	[m]	*m*om			

Table A-1: Label set used for baseline recognizer in Chapters 2–5.

Label	Description	Label	Description
a	low, front, unrounded vowel	m*	unusual [m]
aey	[ae]–[y]	n*	unusual [n]
ahw	[ah]–[w]	nas	undertermined nasal
ahy	[ah]–[y]	nf	[f]-like [n]
ao	mid-low, mid-back	nv	[v]-like [n]
ax_ins	inserted [ax]	o	mid-high, back rounded vowel
bf	voiced labial fricative	oax	[o]–[ax]
dh_d	stop-like [dh]	q	glottal stop
dh_n	[n]-like [dh], as in *in the*	qn	glottalization
dh_v	undetermined, either [dh] or [v]	r*	unusual [r]
e	mid-high, front vowel	rx*	unusual [rx]
ea	[e]–[a]	sch_ins	inserted schwa of undetermined type
ehax	[eh]–[ax]	t*	unusual [t]
ehow	[eh]–[ow]	th_t	stop-like [th]
gf	voiced velar fricative	tx	voiceless flap
i	high, front, short vowel	ue	rounded [i]
ih*	unusual [ih]	uex	[ue]-like schwa
ihae	[ih]–[ae]	uhe	rounded [ih]
iheh	[ih]–[eh]	uhx	[uh]-like schwa
ihow	[ih]–[ow]	uhx_ins	inserted [uhx]
ihuw	[ih]–[uw]	uhy	[uh]–[y]
ix_ins	inserted [ix]	unk	unknown sound
kf	voiceless velar fricative	uu	high, back, short vowel
l*	unusual [l]	w*	unusual [w]

Table A-2: Labels used for manual transcriptions which are not included in Table A-1. The label descriptions are often given in terms of other labels in this table or in Table A-1. For diphthongs, the descriptions are of the form [starting point]–[ending point]. The transcriptions made use of all of the labels shown here and all of the labels in Table A-1, with the following exceptions: [er] was not used—[rx] was used to represent both [er] and [rx]; and no distinction was made between aspirated and unaspirated stops, using the labels for the aspirated stops to represent both levels of aspiration. The labels were created as needed, so that some are only used in the native transcriptions, while others are only used in the non-native transcriptions.

Appendix B

Confusion Matrices

The following tables list the counts of each phonetic confusion, as well as the total count of each lexical label, in the automatic alignments described in Chapter 5. Table B-1 shows the counts obtained using the phonetic recognition method and corresponds to the CFST C_{PR}, described in Section 5.3.1. Table B-2 shows the counts obtained in the first iteration of the forced paths method and corresponds to C_{FP}, described in Section 5.3.2.

	aa	ae	ah	u	eh	ih	iy	uh	uw	ux	aw	ay	ey	ow	oy	ax	ix	rx	er	w	y	r	l	ll	m	n	nn	ng	nx	dx	tq		tot
aa	877	14	29	5	7	2	1	1	2		10	9		33		18	15	8	1	7	1	1	10	12	1			1		1		aa	1146
ae	53	807	25	7	37	23	3	3	1		17	7	2	8		33	30	5	1	4	4	1	2	9	2	9				2		ae	1205
ah	57	13	928	8	17	9	1	5		3	10	2	3	28		61	16	3	1	3		3	4	32	2	6		1				ah	1338
u	1	8		174	1	2			2		2	2	1	3		8	3	1		1	1		2	4	3	3		1				u	265
eh	6	149	27	2	1210	57	13	11	4	4	13	6	15	6		33	70	12	1		1		11	2	2	5		10	2	3		eh	1822
ih	2	21	14		12	1485	130	10	11	35	1	2	35	5		9	175	6		4	25			1		8		2	2	1		ih	2204
iy		6	1	1	5	40	1015	3	7	23		2	25	6		13	45	11	1		39	1		2	1	25		3	1	2		iy	1389
uh		1	3		1	4	2	119	6	1	1		1	4		7	4	3	1		4			1	1	1		1				uh	183
uw			2		2	1	6	1	364	7		1	4	9		10	11	3		7	3		2	3	2	5		2			1	uw	503
ux				1	1	24	6	2	5	356				1		3	21	2			2			1		2						ux	466
aw	38	6	15	5	3		2	1	1	1	605	3		23		5	2	4	1	4			5	16	1	1						aw	776
ay	15	26	12	16	11	9	3			1	12	718	16	8	7	25	14	5	1	2	1	1	2	7	1	8		13	1		1	ay	1051
ey	1	22	2		12	14	29		5	1	2	8	676	4		7	31	9	2		5	2	3	3	2	7		9	2	4		ey	940
ow	66	9	12	6	5	4		3	24	2	16	10	1	1541	1	48	16	8		11	2	2	18	53	4	11		4	1		1	ow	2043
oy												3		3	12									1		1						oy	22
ax	16	23	14	6	24	31	12	3	10	4	7	4	10	22		1957	20	18	2	2	4	1	3	14	6	22			1	2		ax	2421
ix	20	62	23	3	30	123	84	6	14	31	1	6	26	9	1	47	1851	27	1	1	25	1	6	7	7	10			4	4		ix	2743
rx	13	51	23	10	20	27	15	5	6	7	4	6	29	37	2	156	89	1345	15	1	2	23	15	77	7	26		4	3	13		rx	2415
er	9	6	9		3	7			5	1	2	1	1	16		12		13	73		2	2	2	3		3				2		er	185
w	2	6	2		2	10	3	2	4	3	3	2	1	10		3	5	4		2070	1	12	57	22	24	7		1	1	1	2	w	2445
y		5	1			16	11	2	2	1	1		3	1		3	9	2		3	642		3		3	12		3		2		y	830
r	3	4			1	4	6	1	6	1	3	2	3	10	1	4	1	18		8	3	753	33	14	14	18		1	4	17		r	1121
l	5	11	4	1	5	3	6		4	4	1	1		15		5	10	12		10	6	9	720	2	16	20		1	16	28		l	1047
ll	3	11	11	3	3	4	6	1	4		5	4	2	23		14	8	6	1	1	7	2	8	524	3	31		1	3	6		ll	835
m	3	5	1			2	1		2	1	2	1		3		5	5	1		17	3	2	7	9	958	84		1	3	3	1	m	1233
n	7	7	7	4	3	4	15		11	3	6	1	3	11		25	10	6		10	16	6	8	40	46	3025		40	14	15		n	3680
nn																3	2									5	1		1			nn	18
ng		2	1	2	1	1	3		2	1			4			2	1	1		1	1	1		8	1	54		376		2		ng	498
nx	3						3		1		2	1	1		1	2				1	1		2	7	13			5	447	6	1	nx	547
dx		1				1	3				3	1		3		6	2	4		2	1	1	5	6	2	8			8	303		dx	473
tq			1				1		2								1		1				3	1		1			1		20	tq	61
dr																																dr	15
tr		1										1				1																tr	42
jh	1	1		1	1		2		1					1		1	1				2		1			2						jh	225
ch		1				4			1					1				1		1						1						ch	271
zh										1																1						zh	8
z		2			2	1	6		5	2			3	2		7	12	1			3		3	4	6	28	1	3		7		z	1110
v		1		1	2		1		3	2	3			9		6	5	2		7	1		12	16	7	8		1		7		v	338
dh		7	3	3	2	24	4		4	3	4	1	9	8		2	13	2	1	4		7	10	9	9	23		1	18	22	2	dh	2414
sh	1	2				1			1												3	1	1			1				1		sh	232
s	3	5	4	4	6	6	16	1	2	6	6		6	1		17	23	5		2	3	1	6	10	3	26		3	2	3		s	2577
f	5	1		1	1	5	6			3	1	1		1		3	2			14	1	2	4	4	3	2						f	829
th	1					1				1				1			2	1				1		2		2						th	211
hh	3		1	3	2	2	1			1		1	1			3	1	1			3	2	4	4	1	1			3	3		hh	634
b		2	3	3		4	3			1				4		1		1		5		1	1		5	2		2				b	978
d	1	5			1	2	7			2			1			2	5	1			2		2		5	3			2	2		d	1017
g		2		1			4		3	3	2			1		4		2		4	11		4	2	3	5			2			g	412
p			2	1	1		1	1			1	1				2	3	1		6	1		1	1	1			1				p	572
p-		2																														p-	27
t	3	2				5	2	1	1	1	1					5	2		1		10	2	1	1		4				1		t	2409
t-														1		4	1					1										t-	68
k	2	1			1	3	1		1	2				2		4	4			2		1	2	2	1							k	1375
k-																																k-	1
bcl	1	5		2	1									3		4		1		11	1		7		8	2			1	2	1	bcl	906
dcl	1	5	2	2	2	6	2		3		1	2	1	2		4	11	1	1	2	1	1	1	9		16		1		6		dcl	2378
gcl		2				1			1					4		2	4	2			4	1		1		5				2		gcl	325
pcl			1		1				1					1		1	1	1		1			1	1	1	6						pcl	510
tcl	9	4	3	1	5	3	4	2	1	2	1	2	3	3		11	4	7		3	2		1	16	1	26		4		7		tcl	3658
kcl		2	3		2	4	1			1	2	2	3	1		4	7	1	1	2		1		6	3	11		1		3		kcl	1464
-	3	4	4	6	1		1		2	1	5			1		10	3	2	1	5	1	1	1	4	4	17		2				-	42881
-	22	24	5	16	9	46	26	5	16	9	13	5	17	24		61	36	13		27	20	14	20	57	31	108		5	5	12	3	-	36718
ϵ	83	102	37	45	49	157	137	23	50	58	32	31	43	115		196	225	91	10	69	123	53	138	212	78	295		44	24	61	14	-	146831

Table B-1: Confusion matrix showing the frequency of each confusion and the total frequency of each lexical label, using the phonetic recognition method. Lexical labels are on the left and right; surface labels are listed along the top and bottom. The table is continued on the following page.

	dr	tr	jh	ch	zh	z	v	dh	sh	s	f	th	hh	b	d	g	p	p-	t	t-	k	k-	bcl	dcl	gcl	pcl	tcl	kcl	-	_	ε		tot
aa				1			1	5			2		3			1			1		3		2	2		1	8	1	2	9	38	aa	1146
ae			1			1	1	5		4			3		2	1	2		8					3	1	3	13	1	5	6	50	ae	1205
ah						2		4		5	5	1	4	5	3		1		3		1		6	7	1	4	9	3	8	3	47	ah	1338
u								1		2	2		1	1	1				1		1			2			5	1	8	7	9	u	265
eh				1		14	1	6		5	1	2	1	3	4		2		5		6			7	1	2	19		6	8	53	eh	1822
ih						2		8		8		1			12		3		9		4		3	23	1	4	14	5	11	15	85	ih	2204
iy			1			5		2	1	4	5		1	3	6		1		4		2		1	5	1	1	8	1	9	8	42	iy	1389
uh										2	1		1						2					1		1	3	1			5	uh	183
uw						5	2	3					1		1									3	1		4	4	5	3	25	uw	503
ux						3		1	1	3	1		1								1				1		5		3	1	18	ux	466
aw							1	2		1	1			2			2							2		2		1	3	2	15	aw	776
ay			1			3		10		3	2	1	9	4	2		3		5		2			2		2	10	2	6	9	39	ay	1051
ey				1	1	4	1	4		1					2	2			2		1			1	2		8	1	6	5	36	ey	940
ow			1			2	3	10		9	5		2	2		1	4		8		2		1	8		4	10		9	19	64	ow	2043
oy																												1			1	oy	22
ax		1				10	3	5	1	10	3	1	2	1	5		4		6		1		2	13	1	3	14	2	8	20	67	ax	2421
ix			1	3	2	8	2	19	1	12	3	1	2	4	2		2		6		5		2	14		3	14	2	15	43	147	ix	2743
rx				1		13	2	12	10	24	1	5	3	7	27		4		12		2		3	10	4	6	37	2	7	26	166	rx	2415
er						1		1							1		1		1					1			2	2	1		2	er	185
w						2	26	17		4	15	2	5	23	5	5	4		3		1		3	5		1	7	1	8	11	37	w	2445
y			6			3		6		2	1			2	6	2	1		13					2			3	1	1	8	48	y	830
r						3	6	13	2	10	4	2	1	6	7	6	5		19		4		3	4	1	1	3	2	6	8	72	r	1121
l						5	3	15	2	6	3		2	4	4	2	1		12	1	4		5	7		1	10	1	4	3	37	l	1047
ll						8	4	7		9	2	1	1		2		1		5				5	5	2		19	1	5	14	49	ll	835
m						2	3	6		3		1	1	5	6	1	3						6	5		1	1	2	5	16	46	m	1233
n			1		1	28	5	9		12		1	7	3	4		5		13				1	14	2	10	30	4	23	57	107	n	3680
nn															2				1										1	1	1	nn	18
ng						1	1	1		1							1		1					1	1		1			5	19	ng	498
nx				1				2		3				3	2		1		1				1	5			2	1	2	1	25	nx	547
dx						6	1	7		2	3				23	2	2		4		3		2	15		3	3	2	1	2	32	dx	473
tq		1				2	1	3		1					2		1				2			3		1	1	2		2	7	tq	61
dr	2	1						4						1	1				1								3			1	1	dr	15
tr		27	1								1		1						3									1		2	3	tr	42
jh			147	7		1		3	2	5	1			1	2				11		4			1			6	1	2	3	13	jh	225
ch		2	7	182					7	2					2		1		42		1						1		2	1	11	ch	271
zh										1							1										2	1			1	zh	8
z			1		1	669	2	12	1	116	3	1	1	1	7		1		22		1			11		1	46	6	13	53	43	z	1110
v						5	140	6		6	8	7		6	1	2	2		3		1		5	9		1	11		3	8	20	v	338
dh		1		1		3	3	1802	2	11		3	1	7	147	3	3		59		1		10	16		3	21	1	10	9	102	dh	2414
sh			1	7		3			173	10			1						5		3						5	3	2		7	sh	232
s			3	3	1	96	2	40	8	1852	23	29	11	2	9	1	4		54	1	7		3	19		5	41	2	21	67	103	s	2577
f			1				1	10	2	27	617	3	2	2			20		11		4		2	2	1	14	15	3		9	23	f	829
th								2	1	17	3	125			1		1		24		1			1			3	5	1	5	9	th	211
hh			1			1		7	1	3			507	2	2		3		2		7			1		1	4		3	10	38	hh	634
b							1	9		1	4		2	873	7	3	15	1	2	2			1			2	1			4	12	b	978
d			3	1		1		39		3	1	2	1	10	833	6			19	6	2		3			1	7		3	5	28	d	1017
g	1					3		11		1	1	1		3	5	264	5		6		17		1	5			2	4	1	4	24	g	412
p						1		11		5	6	3	5	6	6	1	440	1	17		11		3	1			7		6	6	11	p	572
p-											1			6			6	10												1	1	p-	27
t		3	5	20		1		52	1	14	6	10	7	7	24	2	22		2084	13	39		2	5	1	2	3	2	10	9	22	t	2409
t-								1	1	2	1			2	5				6	36										2	5	t-	68
k			1	2		1	1	13		2		2	5	4	6	30	13		31		1197			4	5		8		3	4	14	k	1375
k-																														1		k-	1
bcl							3	3		2	4		1	1	2	1	5	1	1				780	5	1	20	7	1	1	7	10	bcl	906
dcl		1	1			15	5	2	1	8	1	2	1		2	1	1		19	3	3		10	2084	4	3	79	7	4	12	26	dcl	2378
gcl						1	1	2		2		1		1	2				3		4		2	9	231	3	7	9	1	3	14	gcl	325
pcl		1					2			1	2	2		3		1			5		3		10	9		410	17	9	1	5	12	pcl	510
tcl			1			17	3	20	4	6	6	4	1	1	8	2	9		1	1	5		6	74	2	19	3194	39	8	17	85	tcl	3658
kcl				2	1	6	1	2		3	1	2				2	4		9		1		4	12	21	12	53	1190	1	31	45	kcl	1464
-			2			3		16	1	18	3	1	6	4					3		2		3	4		3	7	4	42512	210		-	42881
-		3	6	3	1	32	5	60	2	76	32	12	31	13	37	7	19		69	1	23		27	63	13	32	116	29	980	34362	15	_	36718
ε		6	11	16	4	103	24	173	13	244	89	36	70	48	112	27	67	2	182	3	84	1	62	190	17	71	304	85	1356	326		_	146831

Table B-1: Confusion matrix obtained with the phonetic recognition method (continued).

	aa	ae	ah	u	eh	ih	iy	uh	uw	ux	aw	ay	ey	ow	oy	ax	ix	rx	er	w	y	r	l	ll	m	n	nn	ng	nx	dx	tq		tot
aa	1037	12	27	5							8	9		27		9	2	2		1			2	10								aa	1157
ae	53	1035	21	23	9	3		4			22	5		8		18	13			1				8		1						ae	1230
ah	49	1	1141	10	10	1		1			8		1	23		15	1	1		1				37								ah	1304
u	2	9		230							2		1	2		4	3							3	2							u	265
eh	4	82	15		1551	37	3	4	2		4		14	3		18	60	4					1									eh	1814
ih		1	7		3	1727	288	1	1	10			22	2			31	1		1	4		1			2						ih	2113
iy		2			2	10	1274		2	3			17			2	34	2			13					5						iy	1379
uh			2			1		193	9			1	1								4											uh	212
uw									464	1			2	7		5	10			5	1					1						uw	506
ux				2		3	3		6	439				1		1	15															ux	474
aw	10	2	7	2							728	3		11						1				8								aw	773
ay	10	20	4	16	3	2	2				4	947	17	2	1	10	3							1		1						ay	1053
ey		14	3		5	3	19		1		2	8	837			6	19	6	1			1				2		1				ey	938
ow	43	1	3	7					15		5	2		1919		16	2	2		6			4	35		1						ow	2089
oy												2		3	17																	oy	22
ax	4	11		3	11	8	2		1		1	1		5		2452	15	3						6	2	2						ax	2546
ix	14	57	9		15	17	61	1	3	1		1	4	18		24	2549	11			4			2	2							ix	2890
rx	11	60	1	19	9	2	4	2	3	5	2	7	21	49		143	74	1682	18			12		100	2	10		1				rx	2391
er	5	10	1	2	2	2			5	2	1	1		29		8		7	94				2	2		2						er	175
w						1				1				2				2		2282		3	47	5	28	3						w	2468
y		2				2	2	1													783					4						y	829
r							1		4		2			2				5		2	1	999	30	4	5	9			1	9		r	1155
l	1	1					5							2		1		8		1	2	5	979		6	9			17	21		l	1111
ll		2		2	1	1	5		3			1		8		2	5	2						664	4	32				9		ll	803
m														1						8			1	6	1156	45			1			m	1233
n		1					8							2		4	3				8			11	21	3569		24				n	3713
nn																4										10						nn	14
ng				3														1						3		21		465				ng	499
nx																							2	7	9				490	3		nx	522
dx																1		1					2	7		1			4	369		dx	406
tq							2		1		1					1				1			2								19	tq	46
dr																																dr	15
tr																																tr	42
jh							1														2											jh	224
ch																				1												ch	266
zh																	1									2						zh	4
z							1		2				1			2	1				1		2	1		10				2		z	1118
v									1					2				1		2			8	13	4	3						v	369
dh		5				1	2				1			1			1			1		2	1	3		3			8	4		dh	2416
sh																					1											sh	236
s			1	2			4			1	1					3	2						1	7		3						s	2580
f																				2			2									f	829
th																								3								th	209
hh	2			2													1				2		3	1								hh	662
b				2										2						2					3							b	1014
d		1					1									1									2	1						d	1094
g																1				2	2		1	1		1			1			g	421
p												1								1												p	584
p-																																p-	27
t																																t	2540
t-																1																t-	74
k																									1							k	1438
k-																																k-	1
bcl		1																		2			3		4							bcl	913
dcl						2										1	1							4		7				2		dcl	2559
gcl																	1	3								3						gcl	359
pcl																								1								pcl	511
tcl																3								14		7						tcl	3864
kcl																	1													1		kcl	1487
-																																-	6365
-	3			5		2			1		1					63				1		1		59	1	134						-	13378
ϵ	1	5	1	2			6							5		4	128	3			5		1	10		41				2		-	82166

Table B-2: Confusion matrix showing the frequency of each confusion and the total frequency of each lexical label, using the forced paths method. Lexical labels are on the left and right; surface labels are listed along the top and bottom. This table is continued on the following page.

	dr	tr	jh	ch	zh	z	v	dh	sh	s	f	th	hh	b	d	g	p	p-	t	t-	k	k-	bcl	dcl	gcl	pcl	tcl	kcl	-	_	ε		tot
aa																							1								5	aa	1157
ae													1																		5	ae	1230
ah																													1		3	ah	1304
u										3			1														2		1			u	265
eh																			1												11	eh	1814
ih								1																1			1				8	ih	2113
iy						1					3													1			1				7	iy	1379
uh																															1	uh	212
uw						1	1																	1				1	1		5	uw	506
ux										1																					3	ux	474
aw																															1	aw	773
ay								2					1				1											1			5	ay	1053
ey																											1		1		8	ey	938
ow								1			6						1													1	19	ow	2089
oy																																oy	22
ax						3				1																			1	2	12	ax	2546
ix								1		5											1									10	80	ix	2890
rx								4	5	7		3		1										2			13			2	117	rx	2391
er																																er	175
w							45	19			12		3	3			1							1			1				9	w	2468
y			3			1		3							2				2												24	y	829
r							2	15	2	4	4	2		1	1	1	2		1										1		45	r	1155
l								14		1	3			1					2		1		2	2			2				25	l	1111
ll						4	4	4		4	1												2				14			1	28	ll	803
m						2		1																2							10	m	1233
n						7				2																	10		2	5	36	n	3713
nn																																nn	14
ng																															6	ng	499
nx																								1							10	nx	522
dx						1		1			2													3		1				1	12	dx	406
tq						1	1	1		4	1													1							10	tq	46
dr	11							3																			1					dr	15
tr		39	1																2													tr	42
jh			208	1		1				4					1				5												1	jh	224
ch		2	1	229					7	1									24												1	ch	266
zh																	1															zh	4
z						936	1	3		116	2								8								16		1	5	7	z	1118
v							296	2		6	10	6															6			3	6	v	369
dh								2232		8				6	86				26				1				3				21	dh	2416
sh				2					221	8											4											sh	236
s						54		13	9	2399	12	12	3		1		1		8					2			7		1	2	31	s	2580
f								4	1	8	790	1	1	2			9									3	1				5	f	829
th								1		11	1	177							10								1			3	2	th	209
hh								2	1	4			633				1														10	hh	662
b								3			2			982			11													1	6	b	1014
d			1					16		2		2		3	1032				17				1								14	d	1094
g								4			1			1		380	1		1		10		1								13	g	421
p								2		2	3			1			565		3		2									1	3	p	584
p-														2			6	18													1	p-	27
t		1	2	17				36		8	5	2	1	2	22		11		2413	5	9									2	4	t	2540
t-										2	2								4	64											1	t-	74
k								2					1		1	17	3		9		1399			1	1					2	1	k	1438
k-																														1		k-	1
bcl											1												892			5				3	2	bcl	913
dcl						10	2			2			1											2463			56	2		3	3	dcl	2559
gcl																							2	2	331	1	5	8			3	gcl	359
pcl											2												3	1		494	9	1				pcl	511
tcl						4					2													34		4	3749	7		3	37	tcl	3864
kcl						1				1														1	3	4	26	1427		11	11	kcl	1487
-																													6364	1		-	6365
_						2		51		166	4		1		1				47		1			17		1	55		618	12143		_	13378
ε						3		1		86	1								3								29		62	38		_	82166

Table B-2: Confusion matrix obtained with the forced paths method (continued).

5230-14