VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV RADIOELEKTRONIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF RADIO ELECTRONICS

PHONOTACTIC AND ACOUSTIC LANGUAGE RECOGNITION

FONOTAKTICKÉ A AKUSTICKÉ ROZPOZNÁVÁNÍ JAZYKŮ

DOKTORSKÁ PRÁCE
DOCTORAL THESIS

AUTOR PRÁCE AUTHOR	PAVEL MATĚJKA
VEDOUCÍ PRÁCE SUPERVISOR	PROF. MILAN SIGMUND

BRNO 2008

Abstract

This thesis deals with phonotactic and acoustic techniques for automatic language recognition (LRE).

The first part of the thesis deals with the phonotactic language recognition based on co-occurrences of phone sequences in speech. A thorough study of phone recognition as tokenization technique for LRE is done, with focus on the amounts of training data for phone recognizer and on the combination of phone recognizers trained on several language (Parallel Phone Recognition followed by Language Model - PPRLM). The thesis also deals with novel technique of anti-models in PPRLM and investigates into using phone lattices instead of strings. The work on phonotactic approach is concluded by a comparison of classical n-gram modeling techniques and binary decision trees.

The acoustic LRE was addressed too, with the main focus on discriminative techniques for training target language acoustic models and on initial (but successful) experiments with removing channel dependencies. We have also investigated into the fusion of phonotactic and acoustic approaches.

All experiments were performed on standard data from NIST 2003, 2005 and 2007 evaluations so that the results are directly comparable to other laboratories in the LRE community. With the above mentioned techniques, the fused systems defined the state-of-the-art in the LRE field and reached excellent results in NIST evaluations.

Keywords

language recognition, language identification, phonotactics, feature extraction, phone recognition, TIMIT, neural networks, temporal patterns

Bibliographic citation

Pavel Matějka: Phonotactic and Acoustic Language Recognition, **Doctoral thesis**, **Brno**, **Brno University of Technology, Faculty of Electrical Engineering and Communication**, 2008

Abstrakt

Práce pojednává o fonotaktickém a akustickém přístupu pro automatické rozpoznávání jazyka.

První část práce pojednává o fonotaktickém přístupu založeném na výskytu fonémových sekvenci v řeči. Nejdříve je prezentován popis vývoje fonémového rozpoznávače jako techniky pro přepis řeči do sekvence smysluplných symbolů. Hlavní důraz je kladen na dobré natrénování fonémového rozpoznávače a kombinaci výsledků z několika fonémových rozpoznávačů trénovaných na různých jazycích (Paralelní fonémové rozpoznávání následované jazykovými modely (PPRLM)). Práce také pojednává o nové technice anti-modely v PPRLM a studuje použití fonémových grafů místo nejlepšího přepisu. Na závěr práce jsou porovnány dva přístupy modelování výstupu fonémového rozpoznávače – standardní n-gramové jazykové modely a binární rozhodovací stromy.

Hlavní přínos v akustickém přístupu je diskriminativní modelování cílových modelů jazyků a první experimenty s kombinací diskriminativního trénování a na příznacích, kde byl odstraněn vliv kanálu. Práce dále zkoumá různé druhy technik fúzi akustického a fonotaktického přístupu.

Všechny experimenty jsou provedeny na standardních datech z NIST evaluaci konané v letech 2003, 2005 a 2007, takže jsou přímo porovnatelné s výsledky ostatních skupin zabývajících se automatickým rozpoznáváním jazyka. S fúzí uvedených technik jsme posunuli state-of-the-art výsledky a dosáhli vynikajících výsledků ve dvou NIST evaluacích.

Klíčová slova

rozpoznávání jazyků, identifikace jazyků, fonotaktika, extrakce příznaků, rozpoznávání fonémů, TIMIT, neuronové sítě, časové trajektorie

Bibliografická citace

Pavel Matějka: Fonotaktické a akustické rozpoznávání jazyků, **Disertační práce**, **Brno**, **Vysoké Učení Technické v Brně, Fakulta Elektrotechniky a Komunikačních Technologií**, 2008

Acknowledgments

I would like to thank Honza Černocký, who was my unofficial supervisor and adopted me to the speech group that originated at the Faculty of Electrical Engineering and Computer Science and later moved to the Faculty of Information Technology. Although there were mostly others (Hynek, Lukas and Petr) coming with original research ideas, he was the one that was behind all the time and managed to keep the "Speech@FIT machine" working.

The following big thanks must go to Hynek Heřmanský who actually started my research career by inviting me to the internship at OGI and who was the first who initiated my interest in language recognition. At OGI, we have also participated at our first NIST evaluation.

I want to thank official supervisor Milan Sigmund for the guidance and for many valuable advices he gave me.

Special thanks must go to Petr Schwarz who was my team-mate in phone recognition and to Lukaš Burget who is behind many (if not all) innovative ideas in our group. I would also like to thank my colleagues Martin Karafiát and Ondřej Glembek for many discussions and ideas we shared. I would like thank to all the other members of the Speech@FIT group at Brno University of Technology, among all to František Grézl, Igor Szöke, Valiastina Hubeika, Oldřich Plchot, Jiří Kopecký, Michal Fapšo, Kamil Chalupníček and Petr Motliček and also members of the Anthropic Speech Processing Group at OGI, namely to Andre Adami, Pratibha Jain and Sunil Sivadas for the unforgettable time and experience. I would like to thank Tomáš Kašpárek, Petr Lampa, Pavel Chytil and others for always working computer infrastructure, and to our secretaries Sylva Otáhalová and Jana Slámová for the service they gave me. I would like to thank also my family and my sweetheart Martina for the patience they showed when I was writing this thesis.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Automatic spoken Language Recognition (LRE) is the process of classifying an utterance as belonging to one of a number of previously encountered languages. Automatic, because the decision is performed by machine. It is implied that the process is independent of content, context, task or vocabulary and robust with regard to speaker identity, sex, age as well as to noise and distortion introduced by the communication channel.

1.1 Problem Specification

As with speech recognition, humans are the most accurate language recognition systems on the world today [2, 3], assuming that they know the language (speak it). Within a second of hearing, people are able to determine if it is a language they know. If it is a language they do not know or are not familiar with, they often can make subjective judgment about a similarity to a language they know, e.g. ”sounds like Czech” or might describe that it is nasal (French), harsh (German), sing-song (Mandarin), rhythmic, guttural, etc. But the accuracy of their judgments is much less precise than the machine [3].

An obvious difference between languages are different words, but when confronted with an unknown language, it is nearly impossible to tell where words begin and end [4]. Perceptual experiments provide some clues [2, 5]. Speakers are sensitive to sounds not found in languages familiar to them, like click-sounds in a number of southern African languages. There are more subtle patterns in the frequency of occurrence of certain sounds and combinations. For instance Hawaiian has very small number of consonants. The cluster /sr/ is very common in Tamil, but not found in English at all. Hindi has four different consonants that are all likely to sound like /t/ to person speaking with stress language.

The differences are not necessarily only in phonemes or words, but also in the sounds of them. There are tonal languages like Mandarin or Japanese and stress ones like English or German. The function of tone is different across languages. In some languages, tone has a predominately lexical function. It is used almost exclusively to distinguish and contrast word meanings, for example word *ba* in Mandarin (Table 1.1) [6]. Tone may also be used grammatically: it is used to distinguish words, but is also used to mark sentences as in Japanese (Table 1.2).

In general, there are a variety of cues that humans or machines can use to distinguish languages. We know that the following characteristics differ from language to language:

- **Phonetics** - Though the human speech system is potentially capable of an unlimited range of sounds, in any language there is a limited number of recurrent, fairly distinctive speech units (phones/phonemes). Many languages share a common subset of phonemes.

Word	Tone intonation	Meaning
ba	high	to uproot
ba	mid	eight
ba	fall-rise	to hold
ba	low	a harrow

Table 1.1: Mandarin tone use

Word	Grammatical function	Meaning
iimono - intonation 1	sentence	It is good thing.
iimono - intonation 2	Adj+Noun	good thing
iimono - intonation 3	Adj Phrase	good quality

Table 1.2: Japanese grammatical use of tone

Phoneme frequencies may differ, i.e., a phoneme may occur in two languages, but it may be more frequent in one language than in the other. The number of phonemes in a language ranges from about 15 to 50, with peak at 30 [7].

- **Phonotactics** - Not only do phoneme inventories differ from language to language, but also does the phoneme combination or sequence of allowable phonemes. Some combinations that occur frequently in one language are illegal in another. Phonotactics helps to recapture some of the dynamical nature of speech lost during feature extraction.
- **Prosody** - Prosody is concerned with the "music" as opposed to the "lyrics" of speech. Languages have characteristic sound patterns which can be analyzed in terms of duration of phonemes, speech rate, intonation (pitch contour), and stress (short term energy).
- **Morphology - Vocabulary** - Conceptually, the most important difference between languages is that they use different sets of words - that is, their vocabularies differ. Thus, a non-native speaker of English is likely to use the phonemic inventory and prosodic patterns of her/his native language, but will be judged to speak English if the vocabulary used is that of English.
- **Syntax** - The ways in which words can be legally strung together also potentially distinctive information. Even when two languages share a word, e.g. the word "bin" in English and German, the set of words which can precede and follow the word will be different.

All information extracted from sources mentioned above are complementary and can be valuable to a LID system.

1.2 Motivation

There are 6000 million people on the world and 64% of them speak 14 languages (see Figure 1.1) from around 3000 world known languages [1]. To communicate with the others, it is at first necessary to know which language we are dealing with.

There are three main field of applications of language recognition nowadays. Probably the most dominant field are national security services for monitoring communications. The main focus is to route the call to appropriate officers knowledgeable in particular language or to large vocabulary speech recognition (LVCSR), keyword spotting (KWS), ...

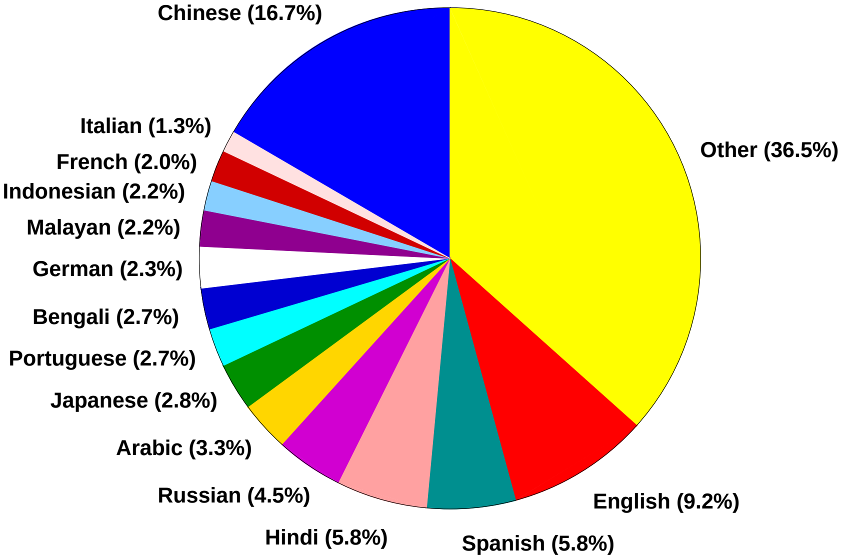

Figure 1.1: Word language distribution [1]

Alternatively, LID might be used in "call-centers" to route an incoming telephone call to a human switchboard operator fluent in the corresponding language. When a caller of a language line does not speak any English, a human operator must attempt to route the call to an appropriate interpreter. Much of the process is trial and error and requires connections to several human interpreters before the appropriate interpreter is found. The delay in finding appropriate human interpreter can be in the order of minutes as reported by Muthusamy [8]. Such delay can be devastating in emergency situation. An automatic language recognition system that can quickly determine the most likely languages of the incoming call might cut the delay by one or two orders of magnitude and ultimately save human lives.

There is lot of audiovisual data accessible through Internet and browsers similar to Google or Yahoo will need information about language of audio in which they are going to look for information. These system might use LID as a pre-processing of its own search.

In recent years there is increasing number of published papers on international conferences which means that the demand and requirements for automatic language recognition systems are steadily increasing.

1.3 Original contributions of this thesis

In my opinion, the original contributions – "clamis of this thesis" can be sumarized as follows:

- Detailed study of phone recognizers on different multilingual telephone databases with varying amount of training data.
- Analyzing the suitability of different phone recognizers for language recognition.
- Application of anti-models in phonotactic language recognition.
- Comparison of different representations of phone recognizer output (string vs. lattices) and of different modeling techniques (n-gram language models vs. binary decision trees).

- Detailed analysis of feature extraction for acoustic language recognition.
- Study of discriminative training for acoustic language recognition.
- Experimental validation on standard NIST 2003, 2005, 2007 LRE data.

1.4 Structure of the thesis

Approaches to Language Recognition are described together with the detailed literature overview in Section 2. Since the phone recognizer is the most important part of phonotactic approach Section 3 describe the development of our phone recognizer. The description of the experimental setup is given in Section 4. Next two Section 5 and 6 describe in detail approaches based on phonotactic and acoustic respectively. The results of the system used for Language Recognition Evaluation 2005 and 2007 are presented in Section 7 and Section 8. Conclusions and discussions can be found in Section 9.

Chapter 2

Approaches and State of the Art

2.1 Structure of the LID system

There are lot of approaches to language recognition. Generally we can say that the process of language recognition (Figure 2.1) can be described as follows:

- **feature extraction** – speech signal is converted into stream of vectors which should contain only that information about given utterance that is important for its correct recognition. An important property of feature extraction is the suppression of information irrelevant for correct classification. Currently the most popular features are Mel Frequency Cepstral Coefficients (MFCC) and their modification Shifted Delta Cepstra (SDC) [9], but features based on energy, fundamental frequency, articulatory features and time evolution of all mentioned features are not rare too (see Section 2.7).
- **classification** – directly to final classes (target languages) or to meaningful units, which are used for further statistical processing. For this classification, any kind of classifier such as Gaussian Mixture Models (GMM), Neural Networks (NN), Support Vector Machines (SVM), Radial Basis Functions (RBF), etc. can be used.
- **statistical models** – in some structures of LID, this sub-system is used further as statistical modeling on units produced by classification. Conventional ways of modeling are Language model (n-grams, binary decision trees) or SVM.
- **fusion** – possible merging with other systems implemented often by Liner Logistic Regression (LLR), NN, GMM.
- **thresholding and decision** is used for making hard decisions (True or False).

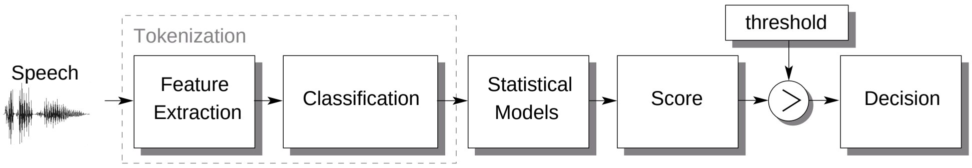

Figure 2.1: General language recognition scheme

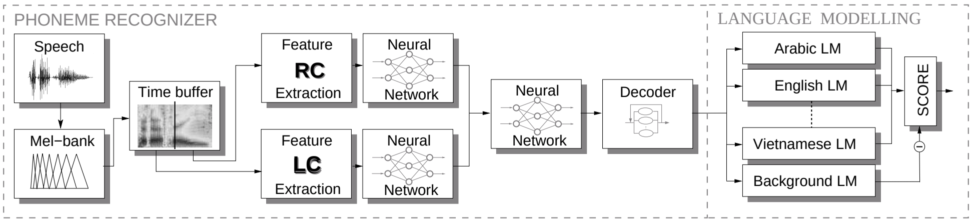

Figure 2.2: Phone Recognition followed by Language Model (PRLM)

2.2 Modeling the acoustics

This modeling attempts to find the discriminative information in acoustic data. Features derived from short-term spectra, prosodic information like fundamental frequency and its time trajectory, loudness on particular parts of the speech and its time evolution, intonation, can be taken into account. Gaussian Mixture Models is the state of the art system [10, 11, 12, 13, 14] to capture acoustic events in the language and different acoustic events across the recognized languages. For example, thousands of Gaussian components can be trained for each language on features which prove discriminability between languages, usually the same features as in speech recognition. The test utterance is assigned to the language represented by the GMM with the highest likelihood. Recently, other technique were used to model acoustic information for LID such as Support Vector Machines and Neural Networks [12].

2.3 Phonotactics

2.3.1 Phone Recognition followed by Language Model (PRLM)

We can consider phonemes as meaningful units, because words in all languages consist of different sets of phonemes [15]. We can use phone recognizer to tokenize speech into phonemes even if we do not know the target language. Such transcription is then similar to a phonetic transcription of the unknown words or sentences by known phonemes.

The goal is to have the most precise phone recognizer or at least a recognizer making consistently the same errors. The statistical modeling of phonemes is on the top of recognizer's output. N-gram language models of phoneme sequences are conventional way to compute similarity of unknown incoming speech to known languages. The whole process is shown in Figure 2.2. When test (unknown) speech comes, a phoneme string is produced and compared to all trained statistical models. Unknown sentence belongs to the language with the highest similarity score. The threshold on this score can be applied to decide if the unknown speech belongs to language in or out of our set.

Parallel Phone Recognition followed by language models (PPRLM) (see Figure 2.3) is a way how to increase accuracy and robustness of this approach [12, 16, 17, 18]. If we have phone recognizers trained on different languages, we can run them in parallel and fuse output score of all PRLMs. Recognizers trained with different languages can behave differently on incoming unknown speech and a mistake of one recognizer can be corrected by another one. On contrary one well trained phone recognizer can perform better then the fusion of several poor ones [18].

There is also a possibility to use tokenizers trained on unlabeled data. Since labeling is the most expensive part of collecting the database, we can have access to much more training material for the tokenizer. Such tokenizer can be an Ergodic Hidden Markov Model (EHMM) [19,

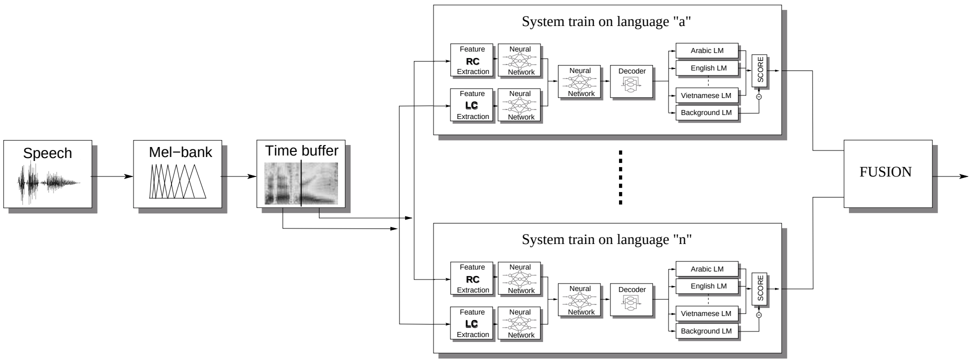

Figure 2.3: Parallel Phone Recognition followed by Language Models (PPRLM)

20]. Another approach to train tokenizer on huge amount of data is to make transcription with known well trained recognizer and train new one with this transcription [21, 22].

2.4 LVCSR : Large Vocabulary Continuous Speech Recognition

Almost all big laboratories on the world have experience with LVCSR systems. With more accurate LVCSR systems, the approach based on tokenizing speech on the word level and then modeling sequence of words became popular method for LID [23, 24]. Generalization of such systems is however more difficult and fewer languages can be reliably detected with them as developing full-fledged LVCSR is far more difficult than phone recognition.

The best way could be to use LVCSR systems of all languages represented in the test set and then fuse score out of all to decide about the result, but this is a bit utopia with the present state of multi-lingual LVCSR.

2.5 Evaluation Metric

The LID performance is evaluated separately for test segments of each duration and is done according to NIST [25, 26] per-language, considering each system is a language *detector* rather than recognizer. A standard detection error trade-off (DET) curve [27] is evaluated as a plot of probability of false alarms against the probability of misses with the detection threshold as parameter and equal priors for target and non-target languages. Equal error rate (EER) is the point where these probabilities are equal. The total EER of the whole LID system is the average of language-dependent EERs.

In addition, the false alarm and miss probability are combined into a single number that represents the cost performance of a system, according to an application-motivated cost model:

$$C(L_T, L_N) = C_{Miss} \cdot P_{Target} \cdot P_{Miss}(L_T) + C_{FA} \cdot (1 - P_{Target}) \cdot P_{FA}(L_T, L_N) \tag{2.1}$$

where L_T and L_N are the target and non-target languages, and C_{Miss}, C_{FA} and P_{Target} are application model parameters. The application parameters are usually set to:

$C_{Miss} = C_{FA} = 1$ and $P_{Target} = 0.5$

For LRE 2007 in addition to the performance numbers computed for each target/non-target language pair, an average cost performance will be computed:

$$C_{avg} = \frac{1}{N_L} \sum_{L_T} \left\{ \begin{array}{l} C_{Miss} \cdot P_{Target} \cdot P_{Miss}(L_T) \\ + \sum_{L_N} C_{FA} \cdot P_{Non-Target} \cdot P_{FA}(L_T, L_N) \\ + C_{FA} \cdot P_{Out-of-Set} \cdot P_{FA}(L_T, L_O) \end{array} \right\} \tag{2.2}$$

where N_L is the number of languages in the (closed-set) test, L_O is the Out-of-Set "language" (including both "unknown" languages and "known" but out-of-set languages),

2.6 NIST Campaigns

National Institute of Standards and Technology (NIST) [1] started Language Recognition Evaluations for comparative results all over the word and give support to new efforts in this field. The first NIST Language Recognition Evaluation took place in **1994** and the evaluation data contained speech from OGI multi-language telephone speech corpus [28] - monologue speech. The results are listed in Table 2.1.

Test Site	**Error %**
Lincoln Labs	3
OGI Y.Yan	8
ITT	8
OGI Berkling	8
MIT	16
Lockheed	24

Table 2.1: Error on standard NIST 1994 evaluation test set

The evaluation in **1996** demonstrated that systems using parallel language dependent tokenizers had the best language recognition performance. The evaluation data contained conversational speech merged from several different databases. These were mainly Switchboard (wide-band and narrow-band) and OGI multi-language telephone speech data. Selected results are reported in Table 2.2.

Test Site EER [%]	**Segment length**		
	30s	**10s**	**3s**
MIT 1996	9.9	19.4	29.4
OGI 1996	11.8	20.9	30.7
MIT 2003	6.6	14.3	25.5
OGI 2003	7.7	11.9	22.6

Table 2.2: Equal Error Rate on NIST 1996 evaluation test set for PPRLM systems from 1996 and 2003

The **2003** NIST Language Recognition Evaluation [25] was very similar to the one in 1996. The primary evaluation data consisted of excerpts from conversations in twelve languages from the CallFriend Corpus [29]. These test segments had durations of approximately three, ten and thirty seconds. Six sites from three continents participated in the evaluation. The results were

[1] http://www.nist.gov

significantly better than from the previous evaluation. The comparative results from selected sites, which were published immediately after the evaluation, are listed below in Table 2.3:

- MIT[2] PPRLM [12] – based on six HMM phoneme recognizers trained on OGI Stories
- MIT GMM – acoustic approach with Gaussian Mixture Models
- MIT SVM – acoustic approach with Support Vector Machines
- MIT Fuse – merged system incorporating all MIT systems mentioned above
- OGI[3] PPRLM [17] – based on six HMM phone recognizers trained on OGI Stories
- OGI 3BT TRAP PRLM – one English phone recognizer trained on NTIMIT based on long temporal trajectories [30]

SYSTEM EER [%]	**30s**	**10s**	**3s**
MIT Fuse	2.8	7.8	20.3
MIT GMM	4.8	9.8	19.8
MIT SVM	6.1	16.4	28.2
MIT PPRLM	6.6	14.3	25.5
OGI PPRLM	7.71	11.88	22.60
3BT TRAP PRLM	12.71	22.71	32.19

Table 2.3: Equal Error Rates on NIST 2003 evaluation test set

The most successful was system MIT Fuse incorporating acoustic and phonotactic approach, but the complexity and real-time factor was disadvantage of this system. Acoustic based systems from MIT outperformed all mentioned phonotactic systems.

The **2005** and **2007** NIST Language Recognition Evaluation are described in detail in Sections 7 and 8.

NIST evaluations clearly show that they are the main technology push in this scientific field. If one group shows that some technology is very powerful then all at the next possible conference other groups report better performance with this technique, often extended and more thoroughly tuned and tested.

2.7 The State of the Art

According to Muthusamy, there were only fourteen published papers in English in the field of automatic language recognition before year 1992 [5, 31]. The main reason for that is probably that there were no big or standard multilingual databases designated for language recognition.

Early attempts to LID were based on matching spectral frames of test message to templates created during training [32, 33]. Cimarusti [34] ran a polynomial classifier on 100-element LPC derived feature vectors. Foil [35] examined both formant and prosodic feature vectors and Goodman [36] extended it by better classification distance metric and refining feature vector.

[2] MIT - Massachusetts Institute of Technology

[3] OGI - Oregon Graduate Institute

With the improvement of speech processing techniques, more sophisticated methods were used. Sugiyama [37] performed vector quantization classification based on LPC features. Nakagawa [10] and Zissman [11] applied Gaussian mixture classifiers. Almost all systems mentioned above used frame-level classification. Hidden Markov Models which are able to model sequence of features in time, have also been applied to LID. House and Neuberg [38] applied HMM to original phone transcription without using real speech and proved phonotactic information can be excellently used for LID. Savic [39], Nakagawa [10] and Zissman [11] applied HMMs to feature vectors automatically derived from unlabeled speech. Li [40] tested a new syllabic spectral features with k-nearest neighbor matching.

With the creation of transcribed multi-language speech databases [28], new LID systems have been proposed. Muthusamy [15] presented the broad-category segmentation and fine phonetic classification which are very capable of discriminating between English and Japanese. Hazen [41], Zissman [16] and Tucker [21] used a phone recognizer with following language modeling by n-grams (PRLM). Later Zissman [16, 42] and Yan [17, 43] extended this by using multiple language-dependent phone recognizers (PPRLM). Jayram [44] used a sub-word recognizer instead of phone recognizer. Andersen [45] have explored the possibility of using just phonemes, which discriminate the best between languages. Tucker [21] and Lamel [22] used single-language phone recognizer to label multilingual training data, which were then used to train language-dependent phone recognizers.

Large-vocabulary continuous speech recognition (LVCSR) has been also employed in language recognition [23, 24]. Test utterance in such system is recognized in a number of languages and the language recognition decision is based on the likelihood of the output word sequences reported by each recognizer.

In recent years, there have been renewed approaches to extract prosody information from speech by Cummins [46] and Adami [47]: both use fundamental frequency and band amplitude envelope. Berkling [48] and Metze [49] use confidence measures to improve LID and Hombert [50] uses only 'rare' segments which are discriminative (i.e. rare) and robust (i.e. easy to identify). Navratil [51] uses binary decision tries instead of conventional n-gram language models and Harberck [52] explored variable length n-gram (multigrams). Wong [53] presented methods to improve Gaussian mixture models such as unknown language rejection and background modeling. Dan and Bingxi [54] used discriminative criterion for training GMM system and Campbell et al. [55] used with success support vector machines instead of GMM for classifying acoustic features. Kirchhoff [56] tried to use parallel streams of articulatory features followed by n-gram modeling.

PPRLM systems performed very well during several last NIST evaluations and there is space to improve it, because usually used OGI database to train tokenizer does not contain enough data to train phoneme recognizers well as I showed in [18]. I used more data to train phone recognizers on different databases with relative improvement 57% on EER. Another way to avoid problem with the amounts of transcribed data is to use methods which do not need transcriptions. Torres–Carrasquillo [19] used an Ergodic HMM to tokenize speech to units based on the data. Another approach to improve PPRLM is to use phoneme lattices instead of hard strings as Gauvain proposed in [57].

Modeling the inter-language, -speaker and -session variability became a necessity of the automatic systems as it happened in speaker recognition. One of the latest comparison of several approaches were performed by MIT and BUT groups [13, 58].

In contrast to the machine based systems, Leeuwen [3] did a human benchmark tests and compared the human and machine performance on NIST 2005 data. The result is that if human know the language they perform far better than machine and if they do not know it then machine outperforms them.

2.7.1 Detailed Literature Overview

- **before 1992:** detailed description of main papers before year 1992 is part of Muthusamy work [5, 31].
- **1993: Muthusamy's** [5] dissertation on segmental approach to LID discusses which acoustic, broad phonetic and prosodic information is needed to achieve automatic LID. First experiments were conducted on 4 language task with high quality speech. On the basis of these results he further investigated these approaches with 10 language corpus of telephone speech [28]. Experiments with features based on pairs and triples of broad phonetic categories, spectral features (PLP) and pitch-based features were carried out on two languages (English vs. Japanese) and on ten language task. The extension of frequency occurrence, segment ratios and duration were also explored. With a system containing all above features merged together, he obtained accuracy 48.5% on short utterances (avg. 13.4 sec) and 65.6% on long utterance (avg. 50 sec) on ten-language task. He came up with conclusion that information on phonetic level instead of broad phonetic might be required to distinguish between languages with greater accuracy. A perfect overview of the literature and development of multi-language database OGI are great parts of his thesis. Perceptual experiments were also conducted, in which trained listeners identified excerpts of speech of one-, two-, four- and six-second durations as one out of ten languages. The average performance over all languages rose from 37.0% to 43.0% to 51.2% to 54.6% with increasing duration of speech segments.
- **1995: Yan's** [43] dissertation provided a partial unification by studying the roles of acoustic, phonotactic and prosodic information. Two novel information sources (backward LM and context-dependent duration model) were introduced. The best accuracies of 91% (45 second segments) and 77% (10 second segments) on nine language task were published. For the best system, he used a set of six phone language-dependent recognizers based on HMM followed by language modeling of phone sequence for each language.
- **1996: Schultz et al.** [24] used large vocabulary continuous speech recognition system (LVCSR). They compared language recognition system based on phone level and word level both with and without language model (LM). In the first attempt, bigram LM was implemented, but trigram in the second stage gained better results. On four language task, word-based system with trigram modeling of words (accuracy 84%) outperformed the phone-based system with trigram modeling of phones (82.6%) significantly. They claim: The more knowledge is incorporated in the word-based language recognition system, the better performance.
- **1999: Berkling** [48] examined various ways to derive confidence measures for LID system. Three types of confidences were proposed. (1) Scores are polled according to the winner – the target set contains scores of the correctly identified utterances. (2) Scores are pooled according to the input – the target set contains all scores where the input and the language model correspond to the same language. (3) The third method does not separate target and background but pools all winning scores into a single set regardless of whether or not the input utterance was correctly or incorrectly classified. She used phone recognizer followed by language models (PRLM) to evaluate which confidence measure is better. Experiments were conducted on NIST 1996 evaluation data. She also studied adding new

features (phone duration, phoneme frequency of occurrence . . .) for improving confidence measure.

- **1999: Hombert** and **Maddieson** [50] described using 'rare' segments for LID system. Detailed description of broad phonetic classes their representation and behavior in different language families is provided, because segments which are rare and easy to identify are extremely valuable in LID system.
- **1999: Harbeck** and **Ohler** [52] proposed to use multigrams (n-grams with variable length) for phonotactic modeling of token string. English and German from OGI Stories were used for testing with accuracy 73% on 10 second utterances and 84% on 30 seconds. They got 84% and 91% respectively using interpolated 3-grams. As they supposed, the results significantly dropped while using short utterances. It does not outperform baseline system, but it can be used as additional information for merging.
- **2001: Navrátil** [59] deals with a particularly successful approach based on phonotactic-acoustic features and presents systems for language recognition as well as for unknown-language rejection. An architecture with multi-path decoding, improved phonotactic models using binary-tree structures and acoustic pronunciation models serves for discussion on these two tasks.
- **2002: Jayram et al** [44] proposed a parallel sub-word recognition (PSWR) language recognition system which is an alternative to conventional Parallel Phone Recognition (PPR) system. Sub-Word Recognizer (SWR) is based on automatic segmentation followed by segment clustering and HMM modeling. PSWR outperformed PPR on six language task with 10 sec of testing utterance only on training set (90.2%) about 4% and it was 1% worse on testing set (62.3%).
- **2002: Torres-Carrasquillo** [9] used with success the Shifted Delta Cepstra (SDC) for Gaussian mixture modeling and in [19] he used Ergodic-HMM to train some meaningful classes based on the untranscribed data. He compared this method with conventional PRLM and GMM approaches. **Santosh Kumar** wrote a paper about the theory behind EHMM for LID in 2005 [60].
- **2003: Dan & Bingxi** [54] used a discriminatively trained GMM for LID. They used Minimum Classification Error (MCE) criterion to estimate new GMM parameters. The results with GMM-UBM acoustic system trained under Maximum Likelihood was 73.1% on 3 languages (English, Mandarin, French) and the one trained under MCE criterion was 75.5%. The conclusion was that discriminative training helps in distinguishing between languages.
- **2003: Adami** [47] proposed to use the temporal trajectories of fundamental frequency and short-term energy to segment and label the speech signal into a small set of discrete units that he used also for speaker recognition. He had 5 tokens which were combination of rising and falling of these two trajectories. He obtained 35% equal error rate on NIST 2003 LID evaluation with 30s utterances on 12 languages. He derived new features with his own segmentation.

  Adding duration of these 5 symbols decreased EER to 30%. He verified this information was complementary with phone-based system (24%) and by fusing these two systems he obtained 21.7%.
- **2003: MIT group** [12, 55] evaluated three approaches – phone recognition, Gaussian mixture modeling and support vector machine classification and fusion of all above. They

outlined the differences and progress from the NIST 1996 evaluation to NIST 2003 evaluation. Results for NIST evaluation 2003 are in Table 2.3. Main improvement in GMM approach is due to using gender-dependent GMM and feature mapping techniques to channel independent feature space. In phone-based LID system, new phoneme sets were used and trigram distributions were added to the language models, with language dependent weights for the trigrams, bigrams and unigrams.

- **2004: Gauvain** [57] used a phonotactic approach, trained on 3 languages (Arabic, Spanish and English). But for training and testing phonotactic model, he used phone lattices. He improved baseline system from 6.8% to 4% on NIST 2003 LID 30 sec. test set. Further improvement was gained by replacing linear averaging of scores by neural net classifier, this resulted in equal error rate of 2.7%.
- **2004: Wong, Siu** [61] used a conventional phonotactic approach based on HMM (3states, 6mixtures, 53phonemes) trained on English with LID accuracy 84% on 45sec long utterances from OGI Stories (6 languages = English, German, Hindi, Japanese, Mandarin, Spanish). They used a discrete HMM (DHMM) to correct phone recognizer. The idea is to train DHMM to convert recognized phoneme to the true one according to transcription to particular language. This can be done in language dependent and independent way. Then the language model is trained on original transcription or further training data can be taken from text corpora of target languages. Language independent DHMM performed with accuracy 67.5% and language dependent 66.7%.
- **2004: Campbell et al** [12, 55] used a support vector machine (SVM) as a classifier instead of Gaussian mixture models (GMM). The features for SVM were Shifted delta cepstra used successfully in GMM. The results are in Table 2.3.
- **2006: MIT group** [13] describe their system for NIST 2005 LRE with several subsystems - Gaussian mixture components and support vector machines classifying shifted delta cepstra, parallel phone recognition followed by n-gram language model and binary tree language model and parallel phone recognition followed by language model or support vector machines operating on phone lattice.
- **2006: BUT group** [58] describe their system for NIST 2005 LRE with several subsystems, see Chapter 7.
- **2006: Leeuwen** and **Brümmer** [62] come up with two new approaches: Gaussian Mixture Model technique with channel dependency adopted from speaker recognition, and Multi-class Logistic Regression system, which operates similarly to a Support Vector Machine (SVM), but can be trained for all languages simultaneously. Both approaches brought significant improvement on NIST LRE data. They also address the important issue of calibration.
- **2006: Burget** [14] used discriminatively trained acoustic models with Maximum Mutual Information criterion and improved the state of the art results with acoustic system by 50% relative, see Chapter 6.
- **2006: Matějka** [63] used the boosting models in phonotactic language recognition, see Section 5.5.
- **2006: White** [64] presented idea of modeling cross-stream dependencies in parallel phone recognition followed by n-gram language model.
- **2007: Campbell** [65] explored the ability of a single speech-to-text system to distinguish multiple languages.

- **2007: Castaldo** [66] explored a set of properly selected time-frequency features as an alternative to the commonly used Shifted Delta Cepstral features. He shows that significant performance improvement can be obtained estimating a subspace that represents the distortions due to inter-speaker variability within the same language, and compensating these distortions in the domain of the features.
- **2008: Leeuwen** and **Brummer** [67] used Linear Discriminant Analysis back-end to train detector for language with very small amount of training data.
- **2008: Farris** [68] investigated methods of reducing the amount of labeled speech needed for training LID systems. Starting with a small training set, an automated method is used to select samples from a corpus of unlabeled speech, which are then labeled and added to the training pool. The process iteratively continues till convergence.

2.8 Conclusion

In the phonotactic approach, the tokenizer (phone recognizer) seems to be the week part of the systems. Lots of LID systems use monophone phone recognizer trained on OGI stories database [28] employing Hidden Markov Models with 3 states and up to 20 Gaussian mixtures. OGI stories have only approximately 1 hour of training data per language. This is a controversy to the wise sentence *"more data better data"* known in LVCSR. The first solution is to train phone recognizers on different database where more data is available [18, 57, 69]. The other thing is to use more complex structure of phoneme recognizer [18, 70, 71] and since we train on more data we can use for example more Gaussian components in HMM or more sophisticated training techniques [72]. Another weak parts are in the phone recognizer output representation and modeling. One of the first works modeling different from language models come from Navratil [59]. Output representation where strings are replaced by lattices was investigated by Gauvain [57]. The issue of amount of training data, different structures of phone recognizers and also modeling of phonotactic dependencies of languages are all addressed in this thesis.

The main improvement of acoustic approach was done in feature extraction [9], by increasing the number of Gaussian mixture components in GMM, different methods of training GMM [54, 14] and by employing a support vector machines [55] as classifiers of acoustic features. A part of this thesis focuses on discriminative training techniques of Gaussian mixture models. The problem feature extraction and discriminative training of GMM are addressed too.

To achieve the best performance, it is necessary to merge information from various sources. Table 2.3 and [12, 13, 58] show clearly how much improvement can be obtained by proper fusion.

Chapter 3

Phone Recognition

3.1 Introduction

Our goal is to design a front-end module that would deliver language and task independent posterior probabilities of sub-word units such as phonemes together with an information about their temporal extent. There should be no language model or any other constraint because this system will be used for language recognition[1]. In phonotactic language recognition, a good tokenizer (phone recognizer) is the most important part [18, 71]. I show later that the accuracy of the whole LID system depends crucially on the accuracy of the tokenizer. Therefore, developing a high performance phone recognizer is the first task in building successful phonotactic language recognition system.

In this chapter, we are looking closer at the input parameterization and the structure of classification. Our experimental system is an HMM - Neural Network (HMM/NN) hybrid [73]. It has less parameters comparing to traditional HMM systems, and is capable of handling correlated multiple frames of features. Context-independent phoneme models are used. In our preliminary experiments, this system achieved about the same results as a conventional HMM system [30].

The baseline setup uses 13 Mel Frequency Cepstral Coefficients (MFCCs), including C_0, deltas and double deltas (referred as MFCC39). Multi-frame input [73] is also studied and applied.

Next I investigated the TempoRAl Patterns (TRAP) parameterization technique [74]. In this technique, frequency-localized posterior probabilities of sub-word units (phonemes) are estimated from temporal evolution of critical band spectral densities within a single critical band. Such estimates are then used in another class-posterior estimator which estimates the overall phoneme probability from the probabilities in the individual critical bands. This technique was demonstrated efficient in noisy environment. The TRAP technique is compared with MFCC and with multiple frames of MFCC. The TRAP-based system was simplified with the goal of increasing processing speed and reducing complexity. Next, the amount of data for training, length of context for phoneme recognition, mean and variance normalization of features and effective structure and number of parameters in system is studied.

[1] Note that such recognizer may by also used for other applications like keyword spotting, speaker recognition or recognition of out-of-vocabulary words.

3.2 Systems description

3.2.1 GMM/HMM system

Conventional phone recognizer based on Hidden Markov Model [30] trained using HTK toolkit[2] were taken as a baseline. Conventional feature extraction consist of standard 12 cepstral coefficient plus energy and delta and double delta coefficient was used. These coefficient form 39 feature vector well known as *MFCC39*. I use two setups with one and three states per phoneme. Phoneme loop forms the recognition network, this means that phonemes can follow each other, even itself. Phoneme insertion penalty is a constant which plays role in the skipping between phonemes. This constant is tuned to minimal phoneme error rate. The number of Gaussian components was tuned to minimal phoneme error rate too.

3.2.2 TRAP system (1BT = One Band TRAP)

In this technique, frequency-localized posterior probabilities of sub-word units (phonemes) are estimated from temporal evolution of critical band spectral densities within a single critical band. Such estimates are then used in another class-posterior estimator which estimates the overall phoneme probability from the probabilities in the individual critical bands. This technique was demonstrated efficient in noisy environment [74].

Speech signals divided into 25 *ms* long frames with 10 *ms* shift. The Mel filter-bank is emulated by triangular weighting of FFT-derived short-term spectrum to obtain short-term critical-band logarithmic spectral densities. TRAP feature vector describes a segment of temporal evolution of such critical band spectral densities within a single critical band. The usual size of TRAP feature vector is 101 points. The central point is the actual frame and there are 50 frames in past and 50 in future. This results in 1 second long time context. The mean and variance normalization can be applied to such temporal vector. Finally, the vector is weighted by Hamming window[3]. This vector forms an input to a classifier. Outputs of the classifier are posterior probabilities of sub-word classes which we want to distinguish. In our case, such classes are context-independent phonemes. Such classifier is applied in each critical band. The merger is another classifier and its function is to combine band classifier outputs into one. The described techniques yields phoneme probabilities for the center frame. Both band classifiers and merger are neural nets. The complete system is shown in Figure 3.1.

3.2.3 Simplified system (FN = FeatureNet)

The disadvantage of the system described in Section 3.2.2 is its complexity. Two main requests for real applications are short delay (or short processing time) and low computational requirements. Therefore I evaluated [75] a simplified version of the phone recognition system.

Band classifiers were replaced by a linear transform with dimensionality reduction. The PCA (Principal Component Analysis) was the first choice. During visual check of the PCA basis, these were found to be very close to DCT (Discrete Cosine Transform). The effect of simplification from PCA to DCT was evaluated too and was found not to increase error rates reported in this thesis by more than 0.5 %, therefore the DCT is used further. A windowing is applied before DCT.

[2] HTK toolkit,`htk.eng.cam.ac.uk`

[3] From our experiments is evident that in this setup, Hamming window does not have any effect, because normalization before Neural net removes it. Only applying projection (DCT, PCA, LDA) preserves the effect of windowing.

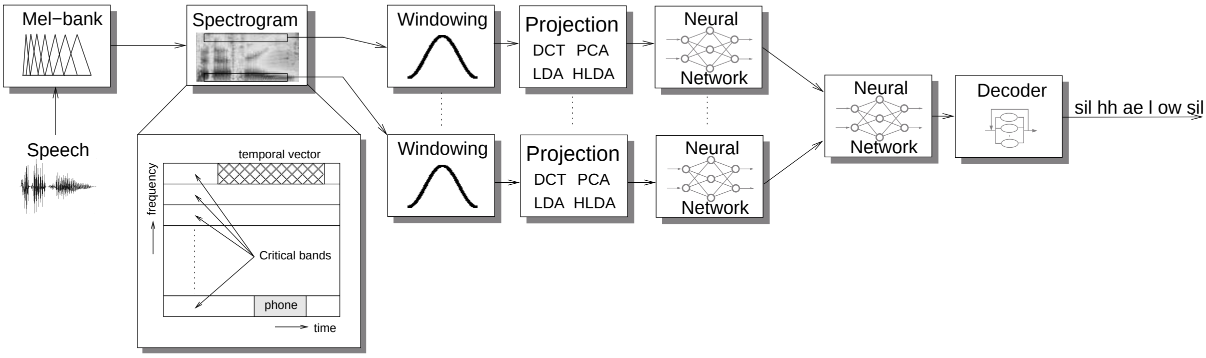

Figure 3.1: TRAP system

3.2.4 System with split temporal context (LCRC = Left and Right Context)

Many common techniques of speech parameterization like MFCC (Mel Frequency Cepstral Coefficients) and PLP (Perceptual Linear Prediction) use short time analysis. Our parameterization starts with this short term analysis but does not stop there – the information is extracted from adjacent frames. We have a block of subsequent mel-bank density vectors. Each vector represents one point in n-dimensional space, where n is the length of the vector. All these points can be concatenated in time order, which represents a trajectory. Now let us suppose each acoustic unit (phoneme) is a part of this trajectory. The boundaries tell us places where we can start finding information about the phoneme in the trajectory and where to find the last information. Trajectory parts for two different acoustic units can overlap – this comes from the co-articulation. The phoneme may even be affected by a phoneme occurring much sooner or later than the immediate neighbors. Therefore, a longer trajectory part associated to an acoustic unit should be better for its classification.

We attempt to study the amounts of data available for training classifiers of trajectory parts as a function of the length of these parts. As simplification, consider the trajectory parts to have lengths in multiples of phonemes. Then the amounts are given by the numbers of n-grams[4].

Table 3.1 shows the coverage of n-grams in the TIMIT test part. The most important column is the third, numbers in brackets – percentage of n-grams occurring in the test part but not in the training part.

If we extract information from trajectory parts approximately one phoneme long, we are sure that we have seen all trajectory parts for all phonemes during the training (first row). If the trajectory part is approximately two phonemes long (second row), we have not seen 2.26% of trajectory parts during training. This is still quite OK because even if each of those unseen trajectory parts generated an error, the PER would increase only by about 0.13% (the unseen trajectory parts occur less often in the test data). However, for trajectory parts of lengths 3 phonemes, unseen trajectory parts can account for 7.6% of recognition errors and so forth.

This gave us a basic feeling how the parameterization with long temporal context works, showed that a longer temporal context is better for modeling of the co-articulation effect but also depicted the problem with insufficient amount of training data. Simply said, we can trust the classification less if the trajectory is longer because we probably did not see this trajectory during training. There are two approaches to deal with this problem. The first one is to weigh trajectory by some window - we are using Hamming one. The second solution is to split the

[4] Note that I never use those n-grams in phone recognition, it is just a tool to show amounts of sequences of different lengths!

N-gram order	# different N-grams	# not seen in the train part	Error (%)
1	39	0 (0.00%)	0.00
2	1104	25 (2.26%)	0.13
3	8952	1686 (18.83%)	7.60
4	20681	11282 (54.55%)	44.10

Table 3.1: Numbers of occurrence of different N-grams in the test part of the TIMIT database, number of different N-grams which were not seen in the training part, and error that would be caused by omitting not-seen N-grams in the decoder.

temporal context.

In this approach, an assumption of independence of some values in the temporal context was done. Intuitively two values at the edges of trajectory part, which represent the investigated acoustic unit, are less dependent than two values closed to each other. In our case, the trajectory part was split into two smaller parts – left context part and right context part. A separate classifier (again a neural net) was trained for each part, the target units being again phonemes or states. The output of these classifiers was merged together by another neural net (Figure 3.2). Now we can look at Table 3.1 to imagine what has happened. Let us suppose the original trajectory part (before split) was approximately three phonemes long (3rd row). We did not see 18.83% of patterns from the test part of database during training. After splitting, we moved one row up and just $2 times 2.26\%$ patterns for each classifier were not seen.

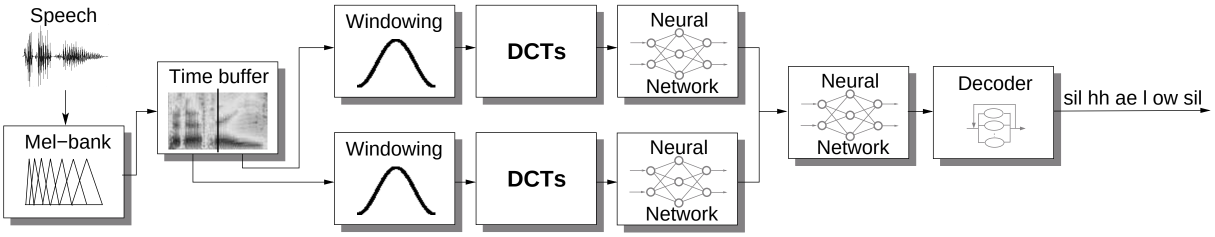

Figure 3.2: Phone recognizer based on split temporal context (LCRC)

3.3 Experiments

3.3.1 Experimental setup

Software

The Quicknet tool from the SPRACHcore[5] package, employing three layer perceptron with the softmax non-linearity at the output, was used in all experiments with neural networks. The STK toolkit[6] was used in experiments with conventional HMM. The decoder for experiments with neural networks was used from STK too.

[5] The SPRACHcore software packages, `www.icsi.berkeley.edu/~dpwe/projects/sprach`

[6] HMM Toolkit STK from Speech@FIT, `www.fit.vutbr.cz/speech/sw/stk.html`

Phoneme set

The phoneme set consists of 39 phonemes. It is very similar to the CMU/MIT phoneme set [76], but closures were merged with burst instead of with silence (bcl b → b). We believe it is more appropriate for features which use a longer temporal context such as TRAP or multi-frame MFCC.

Databases

TIMIT database [77] was used for experiments to compare different systems. All SA records were removed as we felt that the phonetically identical sentences over all speakers in the database could bias the results. The database was divided into three parts – training (412 speakers), cross-validation (50 speakers) and test (168 speakers). The original TIMIT training part was split into two subsets – the training subset and the cross-validation subset. Database was down-sampled to $8000Hz$, because in further experiments we will work with telephone data.

Evaluation criteria

Classifiers were trained on the training part of the database. In case of NN, the increase in classification error on the cross-validation part during training was used as stopping criterion to avoid over-training. There is one ad hoc parameter in the system, the phoneme insertion penalty, which has to be set. This constant was tuned to minimum phoneme error rate on the cross-validation part of the database. Results were evaluated on the test part of database. Numbers of substitution, deletion and insertion errors are reported, as well as a sum of these three numbers divided by the numbers of reference phonemes - the phoneme error rate (PER).

As it is difficult to compare results when the number of parameters in the classifier varies, an important issue, i.e. how to deal with sizes of a classifiers, had to be addressed. One possibility was to fix the number of parameters in the classifier and always reduce the input vector size by a linear transformation to fixed size. However, since the dimensionality reduction always implies a loss of information, a bottle-neck could be created. Therefore, in our experiments, we opted for a different solution in which the optimal size of classifier – number of neurons in the hidden layer and/or number of the Gaussian components – was found for each experiment. A simple criterion – minimum phoneme error rate – was used for this purpose.

3.3.2 Hidden Markov Models with more states

Using more than one state in HMM per acoustic unit (phoneme) is one of the classical approaches to improve PER in automatic speech recognition systems. A speech segment corresponding to the acoustic unit is divided into more coherent parts that ensure better modeling. In our case, phoneme recognition system based on Gaussian Mixture HMM and MFCC features was trained using the HTK toolkit [78]. Then, state transcriptions were generated using this system and neural nets were trained with classes corresponding to states. Coming up from one state to three states improved PER every time. Improvements are not equal and therefore these results are presented for each system separately. The improvement is up to 5% absolute (see Table 3.4).

3.3.3 HMM-GMM and HMM-NN

This experiment was done to compare HMM-NN hybrid with the ”conventional” HMM-GMM. The input consisted of MFCC39 features (12 MFCC coefficients, energy, derivative and acceleration coefficients). The number of parameters – Gaussian components in the case of GMM and neurons in hidden layer in case of NN – was increased until the decrease in PER was negligible. The final number of neurons in the hidden layer is 500 and the final number of Gaussian mixtures is 256 for one-state models and 500 neurons in hidden layer and 128 Gaussian mixtures for three-state models (see Table 3.2). This table contains also the numbers of parameters. The HMM system has about 2% better result in case of one state model at the expense of amount of parameters in the system.

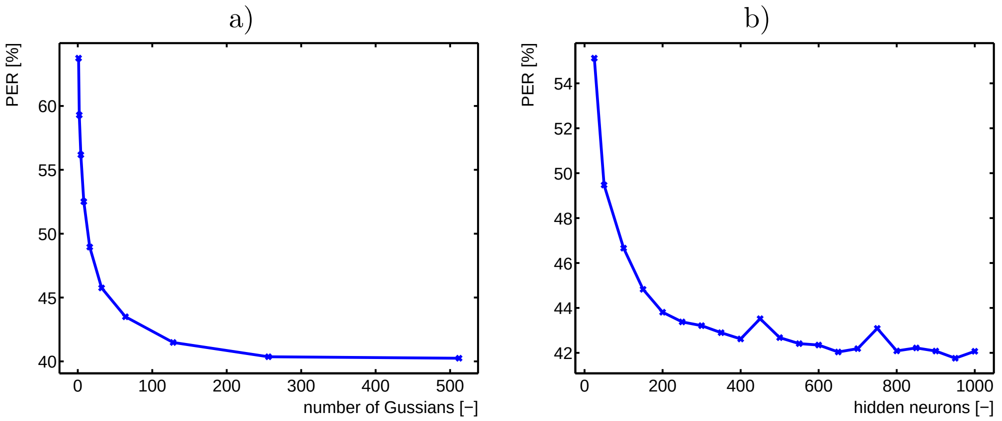

Figure 3.3: Phoneme Error Rates [%] for different number a) Gaussians in GMM and b) neurons in NN hidden layer

# states	system	ins	sub	del	PER	parameters
1	GMM 256G	5.8	20.3	14.3	**40.4**	819057
	NN 500N	10.7	23.6	8.4	**42.7**	40000
3	GMM 128G	6.6	21.4	7.3	**35.4**	1229057
	NN 500N	9.8	22.9	7.2	**39.9**	81000

Table 3.2: HMM-GMM and HMM-NN on MFCC39 with one- and three-state model

3.3.4 Single frame and multi-frame input with MFCC

Multiple frames of MFCC39 were joined together and formed the input to the neural net. We were looking for the minimal PER, therefore the number of subsequent frames joined together was being increased. The best PERs were systematically observed for 500 neurons (Table 3.3).

frames	1	3	5	7	9	15
PER [%] – 1 state	42.7	37.9	**37.6**	38.1	37.9	41.5
PER [%] – 3 states	39.9	36.0	**35.9**	36.2	36.6	39.4

Table 3.3: Effect of using multiframe with MFCC

3.3.5 TRAP and effect of length of the context

The TRAP system, as originally proposed, extracts information from long temporal context. The length of the context was set to be 1*s* in the original system [74]. But this length may depend on the task (recognizing phonemes, words, limited set of the words, . . .), on the size of classifiers, and on the amount of the training data. This experiment therefore evaluated the optimal length of the input trajectory for this task. The numbers of neurons in hidden layer of neural nets were constant – all had 500 neurons, and the TIMIT database was used, therefore the amount of training data was limited. The length of TRAP was being increased from 50 *ms* to 1 *s* and the PER was evaluated. As can be seen in Figure 3.4, the optimal length is about 200 *ms*–400 *ms*. Finding such an optimum length could indicate insufficient training data. However, the fact that shorter input is effective here, may have implications in applications where the minimal algorithmic delay is required.

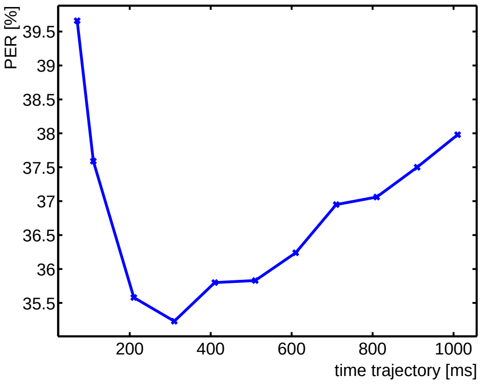

Figure 3.4: Phoneme Error Rates [%] for different lengths of TRAP

3.3.6 TRAP with more than one critical band

Recent results [79] suggest the advantage of use of up to three critical band trajectories in individual TRAP probability estimators. In our case, this is done by concatenating Hamming windowed 310 ms (31 point) long temporal trajectories from three adjacent critical bands to form a 93-dimensional input vector to each TRAP probability estimator. The individual three-band time-frequency patches overlap in frequency by two critical bands, thus combining the 1-3, 2-4, 3-5,…, (N-3)-(N-1), (N-2)-N critical bands. The number of individual TRAP probability estimators in the system is reduced by two since the inputs to the first and the last estimators overlap with their neighbors only at one critical band.

The resulting PER from the three-band TRAP system is 32.5%. This system has about 3% absolute improvement over one-band TRAP system. If we add 3 state modeling of phonemes we get improvement to 31.6%. It is another almost 1%. For comparison with other systems, see Table 3.4.

3.3.7 Simplified system (FN)

This system is a simplified version of 31 length TRAP system and contains weighting of values in temporal context by Hamming window where band classifiers were replaced by dimensionality reduction (DCT) to 15 coefficients. System with one state per phoneme has phoneme error rate 34.8% and the version with three states per phoneme has approximately the same result. The comparison with all other systems is in Table 3.4.

3.3.8 System with split temporal context (LCRC)

The feature extraction uses 15 Mel filter bank energies which are obtained in the conventional way. Temporal evolution of 31 frames of critical band spectral densities are taken as in TRAP processing, but the temporal trajectory is split into left and right contexts. This allows for more precise modeling of the whole trajectory while limiting the size of the model (number of weights in the NN) and reducing the amount of necessary training data. Both parts are processed by discrete cosine transform (DCT) to de-correlate and reduce dimensionality to 10 coefficients and concatenated over all bands. Two NNs are trained to produce the phoneme posterior probabilities for both context parts. Third NN functions as a merger and produces final set of posterior probabilities (see Figure 3.2). An evaluation of such system and comparison with others is shown in Table 3.4.

3.4 Conclusions on TIMIT

TIMIT database was chosen to compare our phone recognition scheme to the state-of-the-art. I can say that our LCRC system with three states favorably compares to the best known systems. Table 3.4 gives a track of improving our system and comparison with some results found in literature. Since all results from other sides are with sampling frequency 16kHz I also present our LCRC system trained on 16kHz. It perform with phoneme error rate 24.2%. Further development of our phone recognizer based on hierarchical structure of neural nets [70] is also presented in Table 3.4.

3.5 Multi-language Telephone Speech Phoneme Recognition

Based on experiments above I have used LCRC system with three states in further experiments (if not stated otherwise). The length of 31 frames of the time trajectory in feature extraction is used in each critical band. This length was chosen based on experiments aiming at minimizing phoneme error rate (see Section 3.3.5 and [30]). All neural networks have 500 neurons in hidden layer (if not stated otherwise).

For each language from all mentioned databases, we used the same structure as for TIMIT – we divided data into three parts. Recognizers were trained on the training part. The increase in classification error on cross-validation part was used as a stopping criterion in NN to avoid over-training. Testing of performance was done on the test part.

3.5.1 OGI Stories

Database description

The OGI Multi-language Telephone Speech Corpus [28, 86] consists of telephone speech from 11 languages: English, Farsi, French, German, Hindi, Japanese, Korean, Mandarin, Spanish, Tamil, Vietnamese. The corpus contains fixed vocabulary utterances (e.g. days of the week) as well as fluent continuous speech. The current release includes utterances from about 2052 speakers, for a total of about 38.5 hours of speech. I used fluent continuous speech part from the database where each caller was asked to speak for one minute about any topic. In six languages, some files, referred to as "stories", were selected for hand generated fine-phonetic transcriptions. The languages were: English(208), German(101), Hindi(68), Japanese(64), Mandarin(70), Spanish(108). The numbers in parentheses indicate the number of stories transcribed for each lan-

System	TIMIT 8kHz PER [%]
HMM/GMM MFCC39 256G	40.4
+ 3 states (128G)	35.4
HMM/NN MFCC39	42.7
+ 3 states	39.9
+ multiframe 5	37.6
+ 3 states	35.9
1 band TRAP system	35.6
+ 3 states	33.9
Simplified system	34.8
+ 3 states	34.7
3 band TRAP system	32.5
+ 3 states	31.6
Split left and right context	32.7
+ 3 states	**30.3**
+ 3 states + Sentence mean normalization	31.6
+ 3 states + Sentence mean & variance norm.	32.9
+ 3 states + 16kHz	**24.2**
Schwarz,Matejka: Hierarchical structure of neural nets [70] (16kHz)	21.5
Lamel: Triphone CD-HMM (16kHz) [80]	27.1
Ming: Bayesian Triphone HMM (16kHz) [81]	27.1
Deng: Hidden trajectory model (16kHz) [82]	27.0
Chang: Near–Miss modeling (16kHz) [83]	25.5
Robinson: Recurrent neural nets (16kHz) [84]	25.0
Halberstadt: Heterogenous Measurements (16kHz) [85]	24.4

Table 3.4: Comparison of phoneme error rates on TIMIT

guage. Amount of data with division to training and testing parts for all transcribed languages can be seen in Table 3.5.

Initial experiments with more languages

First experiments with our best system are reported in Table 3.6. Chanel normalization is performed even we did not get any improvement on TIMIT database (see Table 3.4). This is because TIMIT is clean and we suppose the normalization could help on real telephone data. This was confirmed in experiments: channel normalization helped on real data with gain up to 4.8% absolute in phoneme error rate.

It is hard to compare these phone recognizers because it is hard to find accuracies of phone recognizers implemented in LID systems of other sites. I found only Yan's results in his thesis (1995) [43] see Table 3.6. Phoneme error rates of our recognizers are better from 2% to 7% absolute. The smallest difference is for German and the highest for Mandarin.

Issue of amount of data

Unfortunately, the amount of transcribed data per language is only about one hour (see Table 3.5) which is not enough to train phone recognizer properly [75]. We are not looking on

Database	**train** [hours]		**test** [hours]		**cv** [hours]		**phonemes**
	speech	silence	speech	silence	speech	silence	[count]
SPDAT RUSSIAN	14.02	11.89	3.89	3.11	1.57	1.31	53
SPDAT CZECH	9.72	11.31	2.26	2.67	0.91	1.08	46
SPDAT POLISH	9.49	8.98	2.33	2.30	0.88	0.84	41
SPDAT HUNGARIAN	7.86	6.08	1.97	1.50	0.77	0.60	62
OGI ENGLISH	1.71	0.50	0.42	0.11	0.16	0.05	40
OGI SPANISH	1.10	–	0.26	–	0.11	–	39
OGI GERMAN	0.98	0.07	0.24	0.02	0.10	0.01	44
OGI HINDI	0.71	–	0.17	–	0.06	–	47
OGI JAPANESE	0.65	–	0.15	–	0.06	–	30
OGI MANDARIN	0.44	0.20	0.11	0.05	0.03	0.01	45

Table 3.5: Description of OGI and SpeechDat-E databases

SYSTEM **PER** [%]	**LCRC**	**LCRC** **Smn**	**LCRC** **Smvn**	**Yan** **HMM** [43]
OGI English	47.0	45.3	**44.8**	49.6
OGI German	55.4	51.1	**50.6**	52.8
OGI Hindi	49.5	**45.7**	45.9	48.9
OGI Japanese	42.0	41.2	**38.8**	42.6
OGI Mandarin	54.0	49.9	**49.5**	56.6
OGI Spanish	42.0	39.6	**37.3**	43.3

Table 3.6: PER [%] of phoneme recognition trained on OGI database

phoneme error rate in language recognition therefore we can use all data from a database for training phone recognizers and evaluate the LID error rate. Therefore we merged train and test sets together. Our test sets are about 15 minutes long in average but this represents about 20% of transcribed data! After this move we can no more evaluate phoneme error rate on the test data, as it was seen during the training. However, we can compare results on cross-validation part and see at least tendencies of phone recognizers. The results (Table 3.7) prove correctness of our assumption: ”more data is better data”.

I used the best recognizer from previous section (LCRC) with sentence mean and variance normalization for this experiment, and I saw these tendencies in all our tested phone recognizers.

Language		**ENG**	**GER**	**HIN**	**JAP**	**MAN**	**SPA**
baseline	test	44.8	50.6	45.9	38.8	49.5	37.3
	cv	52.1	56.1	**48.9**	**36.1**	45.5	39.3
retrained	cv	**52.0**	**55.4**	**48.9**	37.0	**42.9**	**38.0**

Table 3.7: Phoneme error rate [%] on OGI Stories

amount of data [hours]	1	3	5	7	10
PER [%]	34.89	30.82	29.57	28.27	**27.44**

Table 3.8: Influence of amount of data for training phoneme recognizer on PER[%] shown on LCRC phone recognizer (Smn) trained on SpeechDat-E Czech

SYSTEM PER [%]	**LCRC**	**LCRC Smn**	**LCRC Smvn**
SPDAT Czech	29.2	**27.4**	27.5
SPDAT Russian	43.1	42.0	**40.9**
SPDAT Polish	40.0	39.9	**39.4**
SPDAT Hungarian	36.4	**35.9**	36.3

Table 3.9: PER [%] of phoneme recognition trained on SpeechDat-E database

3.5.2 SpeechDat-E

Database description

The SpeechDat(E) Database (Eastern European Speech Databases for Creation of Voice Driven Teleservices) [87, 88] consists of telephone speech from 5 languages: Czech(526/526), Hungarian(511/489), Polish(488/512), Slovak(498/502) and Russian(1242/1258). The numbers in parentheses indicate the number of male (first number) and female (second number) speakers. The databases are balanced also over age groups and dialects. Each utterance is stored in separate file and has an accompanying ASCII SAM label file. I used phonetically balanced sentences (referred as `s`,`z` sentences). There are 12 sentences with average duration 4 seconds of speech per speaker. Amounts of data with division to training and testing parts for 4 languages can bee seen in Table 3.5.

Issue of amount of data

The difference between OGI Stories and SpeechDat-E is mainly in amounts of transcribed data. Therefore I simulate increasing amount of data for phoneme recognition and watched decreasing phoneme error rate. The results are shown in Table 3.8. If we compare the systems trained on 1 hour and 10 hours, there is an absolute difference of almost **7.5%** in PER. It is evident that the system trained on 1 hour of data is not well trained which may allow us to suspect all the phone recognizers trained on OGI multilingual database to be also badly trained.

Experiments

We can see that if we have more data we can train phone recognizer well. The results for languages other than SpeechDat-E are reported in Table 3.9. Channel normalization helped also for SpeechDat-E database. The improvement is about 2% absolute. Table 3.10 presents the final system where I increased the size of classifier to 1500 neurons in hidden layer and the scheduler for neural network learning rate was changed to halve the learning rate if the decrease in the frame error-rate (FER) on the *training* (rather than on the cross-validation part) set is less than 0.5 %. The minimum number of training epochs was set to 12. These changes lead to improvement of the system of about 2%.

SYSTEM LCRC Smn PER [%]	**500 neurons**	**1500 neurons**
SPDAT Czech	27.4	**24.1**
SPDAT Russian	42.0	**39.0**
SPDAT Polish	39.9	**36.3**
SPDAT Hungarian	35.9	**33.3**

Table 3.10: PER [%] of LCRC phoneme recognition with sentence mean normalization trained on SpeechDat-E database with different number of neurons in hidden layer in NN

3.6 Conclusion

The core of the phonotactic approach is phone recognizer. I used phone recognizer based on the LCRC scheme and trained on the telephone speech from SpeechDat database for further experiments. The results of these phone recognizers are hard to compare across the languages since the amount of training data varied as well as channel type, phone set, etc. I compare the quality of these recognizers in terms of usability in language recognition in Chapter 5.

Chapter 4

LID Experimental Setup

The first part of development was done using the NIST 1996 and NIST 2003 data together with the CallFriend database. These data are rather easy in comparison with later evaluation data from NIST 2005 and NIST 2007. Chapters 5 and 6 describe the development before the NIST evaluation in 2005. This section is an experimental setup for Chapters 5, and 6. Chapters 7 and 8 describe the system development and evaluation for the NIST LRE 2005 and 2007, these chapters have their own experimental setups given by evaluation data for the particular year.

4.1 NIST 1996 data set

This set was used as development data. There are two subsets – development and evaluation consisting of 12 languages (Table 4.1) and 3, 10 and 30 second audio files. Development data consists of approximately 1200 files for each evaluation duration, with roughly 160 files each for English, Mandarin, and Spanish and 80 messages for each of the other nine languages. The evaluation set consists of approximately 1500 files for each duration: 480 for English, 160 each for Mandarin and Spanish, and 80 for each of the other nine languages. English messages were obtained from both the CallFriend corpus (160) and other English corpora (320) [12].

4.2 NIST 2003 data set

This data set [25] consists of 80 segments with duration of 3, 10 and 30 second duration in each of 12 target languages (Table 4.1). This data comes from conversations collected for the CallFriend Corpus but not included in its publicly released version. In addition, there are four additional sets of 80 segments of each duration selected from other Linguistic Data Consortium (LDC)[1] supplied conversational speech sources, namely Russian, conversations of CallFriend type, Japanese, conversations from the CallHome Corpus, English, from the Switchboard-1 Corpus and cellular English and from the Switchboard Cellular Corpus. Development set for this evaluation are data from NIST 1996 LID described above. All results in this thesis are reported on 30 second segments from LRE 2003 set except Chapter 7 and if not stated otherwise.

[1] http://www.ldc.upenn.edu

Arabic (Egyptian)	German	Farsi	French (Canadian French)
Hindi	Japanese	Korean	English (American)
Mandarin	Tamil	Vietnamese	Spanish (Latin American)

Table 4.1: The twelve target languages

4.3 Callfriend corpus

The CallFriend corpus of telephone speech was collected by LDC in 1996 primarily to support projects on Language Identification (LID) and was sponsored by the U.S. Department of Defense. There are 12 languages (see Table 4.1) with conversations lasting between 5 and 30 minutes. There are 60 unscripted conversations in each language. All speakers were aware that they were being recorded. They were given no guidelines concerning what they should talk about. Once a caller was recruited to participate, he/she was given a free choice of whom to call. Most participants called family members or close friends.

4.4 Evaluation

For evaluation metric see Section 2.5.

4.5 Score normalization

Final score takes into account likelihoods from all detectors:

$$\log P(L|\mathcal{O}) \approx \log p(\mathcal{O}|L)/T - \log \sum_{l} p(\mathcal{O}|l)/T \tag{4.1}$$

where $\log p(\mathcal{O}|L)$ is log-likelihood of speech segment $\mathcal{O}$ given by GMM (in case of acoustic approach) or LM (in case of phonotactic approach) for language L, T is number of frames (phonemes) in speech segment $\mathcal{O}$ and the term $\log \sum_l p(\mathcal{O}|l)/T$ can be interpreted as background model.

4.6 Fusion

Linear combination of scores from separate systems are used according Equation 4.2 where weights $\alpha, \beta, \gamma, \delta, \epsilon$ are tuned by simplex method to find minimal recognition error with the final score.

$$finalscore = \alpha GMM_{MMI} + \beta PRLM_{HU} + \gamma PRLM_{RU} + \delta PRLM_{CZ} + \epsilon PRLM_{PL} \tag{4.2}$$

The simplex method is a method for solving problems in linear programming. This method, invented by George Dantzig in 1947 [89], tests adjacent vertexes of the feasible set (which is a polytope) in sequence so that at each new vertex the objective function improves or is unchanged. The simplex method is very efficient in practice, generally taking $2m$ to $3m$ iterations at most (where m is the number of equality constraints), and converging in expected polynomial time for certain distributions of random inputs.

Chapter 5

Phone Recognition followed by Language Model - PRLM

I have used our phone recognizers described in Chapter 3 and language model described in next subsection. At first, different structures of phone recognizer were tested for language recognition based on one-best phoneme recognition output. Further the lattices instead of one-best recognition output were used. This chapter is concluded by the description of boosting models to improve the state-of-the-art language modeling in LID.

5.1 Language models

The goal of a Language Models (LM) is to estimate the probability of a symbol sequence, $\hat{P}(w_1, w_2, \ldots, w_m)$ which can be decomposed as a product of conditional probabilities:

$$\hat{P}(w_1, w_2, \ldots, w_m) = \prod_{i=1}^{m} \hat{P}(w_i \mid w_1, \ldots, w_{i-1}) \tag{5.1}$$

Limiting the context in Equation 5.1 results in:

$$\hat{P}(w_1, w_2, \ldots, w_m) \simeq \prod_{i=1}^{m} \hat{P}(w_i \mid w_{i-n+1}, \ldots, w_{i-1}) \tag{5.2}$$

for $n \geqslant 1$ with values of n in the range of 1 to 4 inclusive are typically used, and there are also practical issues of storage space for these estimates to consider.

Estimates of probabilities in n-gram models are commonly based on maximum likelihood estimates – that is, by counting events in context on some given training text:

$$\hat{P}(w_i | w_{i-n+1}, \ldots, w_{i-1}) = \frac{C(w_{i-n+1}, \ldots, w_i)}{C(w_{i-n+1}, \ldots, w_{i-1})} \tag{5.3}$$

where $C(.)$ is the count of a given word sequence in the training text.

5.2 String based system

This section presents the results with phone recognizer producing the one-best output (string) which is used for phonotactic modeling.

5.2.1 Training

Language model of 3rd order was used to capture phonotactic statistics of each language. Phoneme insertion penalty (PIP) in the decoder is a constant which must be tuned for the specific task. This constant influences the output phoneme strings and can vary for different applications such as phoneme recognition or language recognition. Here, it was tuned with the best LID performance as criterion (for more details see my Technical Report [90]).

5.2.2 Testing

During recognition, the test sentence is passed through the phone recognizer. The resulting string of phonemes is processed by all phonotactic models for each, the likelihoods of all trigrams are multiplied. The problem of unseen trigrams is solved by assign them the empirically tuned value -6 in the logarithm with base 10. If the trigram probability is lower than this value it is used instead of the estimated one. Likelihoods are normalized over all languages (see Section 4.5). Finally we have scores for all target languages. Test sentence belongs to target language with the maximal score.

5.2.3 Different phoneme tokenizers

I introduced several structures of phone recognizers in the previous chapter. Table 5.2.3 shows the performance of LID systems along with phoneme error rates of phone recognizers used. I used Hungarian SpeechDat-E database for this experiment, because I obtained the best EER with this language (see Table 5.4). All systems had 500 neurons in hidden layer in neural net which was trained to produce three phoneme state posterior probabilities for Viterbi decoder. Sentence mean normalization was used in the phone recognizers. This experiment proved the statement that EER is dependent on how good is the phone recognizer.

System	EER [%]	PER [%]
MFCC39 NN/HMM hybrid system	9.0	45.1
Simplified system = FN	5.6	38.6
Split left and right context = LCRC	4.8	36.4
+ Sentence mean normalization	4.4	35.9
+ Sentence mean & variance norm.	5.1	36.3

Table 5.1: Comparison of EER of several setups of Hungarian phone recognizer tested on NIST 2003 LID (30sec)

5.2.4 OGI Stories vs. SpeechDat-E - influence of amount of training data for phone recognizer

OGI Stories

We know that if we use more data for training phone recognizers it helps on the PER (see Section 3.5.1 and 3.5.2), but does it generalize for LID too? As equivalent to Table 3.7 with phoneme error rates, I report LID performance in Table 5.2 for increasing amount of training data for phone recognizer for OGI Stories database.

SpeechDat-E database

To simulate the EER for tokenizer trained on more data than 1 hour I present Table 5.3 and Figure 5.1 with tokenizer trained on Czech SpeechDat-E (PRLM BUT-CZ) from 1 to 10 hours. If we compare "PRLM BUT-OGI wholeDB" results with "PRLM BUT-CZ" trained on 1 hour, we can say, that the EER are similar. But with second system, we can go further with increasing amount of training data for phone recognizer. There is a saturation of EER after 7 hours of training data (with this database) with the best achieved results 5.42% which is almost 4% absolutely better than the equivalent system trained on 1 hour of data. It is evident that the systems trained on 1 hour of data are not well trained.

Parallel PRLM

The results of parallel ordering of phone recognizers are in the last column of Table 5.2 for phone recognizers trained on OGI Stories and in Table 5.4 for SpeechDat-E. If we compare EER in these tables, we can see that 3 out of 4 PRLM system trained on SpeechDat-E outperform PPRLM system trained on 6 languages from OGI Stories.

Language	**ENG**	**GER**	**HIN**	**JAP**	**MAN**	**SPA**	**fusion**
PRLM BUT-OGI	11.83	11.67	9.75	11.42	15.08	14.08	**6.92**
PRLM BUT-OGI wholeDB	10.58	10.33	8.92	9.08	12.83	11.33	**5.58**

Table 5.2: EER [%] of single PRLMs trained on OGI Stories and tested on 30 second task from NIST 2003 LID evaluation

amount of data [hours]	1	3	5	7	10
PER [%]	34.89	30.82	29.57	28.27	**27.44**
EER [%]	9.17	6.50	6.67	**5.42**	**5.42**

Table 5.3: Influence of amount of training data for phoneme recognizer on PER[%] and EER[%] (LID NIST 2003 30sec) with LCRC SpeechDat-E Czech phone recognizer

PRLM	duration [s]/EER [%]		
system	**30s**	**10s**	**3s**
Hungarian	4.42	13.8	28.9
Russian	4.75	15.6	28.3
Czech	5.42	16.7	33.3
Polish	6.75	17.8	31.9
fusion	**2.42**	**8.08**	**19.08**

Table 5.4: EER [%] of single PRLMs and PPRLM on NIST 2003 LID evaluation

5.2.5 Comparison of systems

Table 5.5 gives a comparison to the results of the best systems known from literature before 2005. All PRLM systems used in this experiment were tuned on NIST 1996 LID evaluation

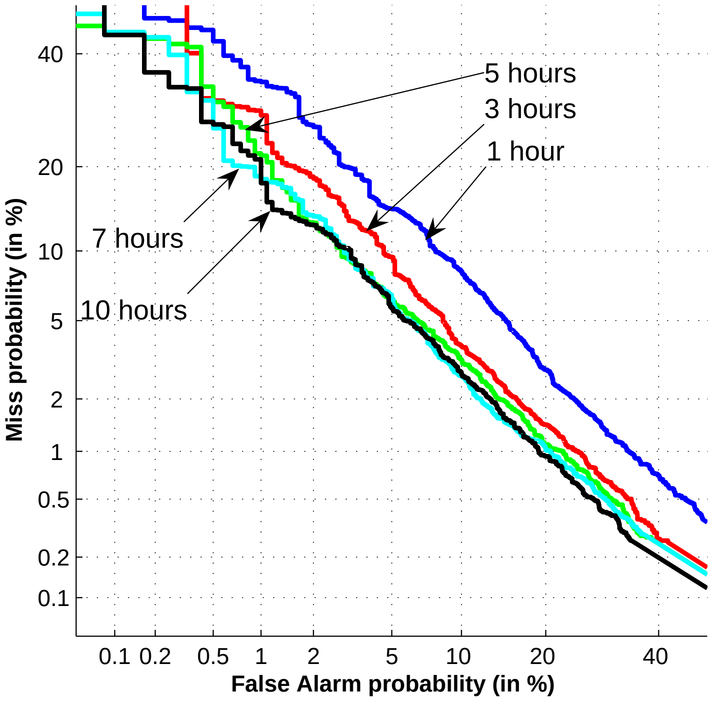

Figure 5.1: DET Plots of systems trained on different amounts of data on Czech Language on 30sec task from NIST LID 2003

data [90]. The testing was performed on NIST 2003 LID evaluation data. Results of OGI [91] and MIT PPRLM [12] employing HMM phone recognizers trained on 6 languages from OGI stories are in the first rows of the table. The system labeled ”FUSE MIT” was based on merging of output of PPRLM mentioned above, Gaussian Mixture Model and Support Vector Machine trained on acoustic features [12]. Our PPRLM system trained on OGI Stories (PPRLM BUT-OGI wholeDB) outperformed OGI and MIT ones by about 1% on the 30 second task, which is significantly better at the confidence level of 95% from Gaussian approximation. This is proving the superiority of our LCRC FeatureNet phone recognizer. Our best result was achieved with PPRLM trained on four languages from SpeechDat-E database – this system favorably compares to system ”FUSE MIT” even though no acoustic modeling was included.

SYSTEM EER(%)	**1996**			**2003**		
	30s	**10s**	**3s**	**30s**	**10s**	**3s**
PPRLM OGI	–	–	–	7.7	11.9	22.6
PPRLM MIT [12]	5.6	11.9	24.6	6.6	14.2	25.5
FUSE MIT [12]	2.7	6.9	17.4	2.8	**7.8**	20.3
PPRLM BUT-OGI	5.16	9.85	19.69	6.92	11.67	22.17
PPRLM BUT-OGI wholeDB	4.29	8.79	18.63	5.58	11.08	21.58
PPRLM BUT-SPDAT	**1.48**	**5.66**	**15.83**	**2.42**	8.08	**19.08**

Table 5.5: Comparison of EER [%] on NIST 1996 and 2003 evaluations

system / EER [%]	baseline	+ WB-LM	+ new phnrec + WB-LM		
duration [s]	30	30	30	10	3
Hungarian	4.4	3.7	3.1	10.6	23.7
Russian	4.8	4.4	3.0	9.6	22.2
Czech	5.4	4.8	3.8	11.9	25.0
Polish	6.8	6.7	6.0	14.4	28.4

Table 5.6: New baseline results for NIST LRE 2003.

5.3 String based system - New baseline

Table 5.6 summarizes new baseline results. The obvious shortcoming in previous phonotactic modeling — use of hard constant to replace unseen trigrams — was fixed by back-off language models with Witten-Bell discounting. This improved slightly the resulting EER. However, more improvement was obtained from improving the phone recognizers, mainly by increasing the number of hidden layer from 500 to 1500 (see Section 3.5.2 for description and Table 3.10 for improvement of phoneme error rates). Both changes together lead to about 30% relative improvement in EER. The right part of Table 5.6 denotes the new baseline of separate systems.

5.4 Phoneme Lattices

I have shown that it is not important when the tokens (phonemes) come from a language different from the target one. However, we have to take into account that the tokenizer, as all speech recognition techniques, is not 100% accurate. Common practice in LVCSR, acoustic information retrieval, etc. is to use richer structure at the end of decoder: lattices instead of strings. The idea behind using the phoneme lattices is to avoid some of the approximation made in the baseline (one-best) system. The language of spoken segment probably can be better approximated by taking the summation over the phone sequences presented in phone lattice instead of using just the most likely one. In LID, this approach was pioneered by Gauvain et al. [57] with good results.

Training

Let us consider the language recognition as a problem of finding maximum of Equation 5.4, where L^* is the hypothesized language, $f(\mathcal{O}|H, L, \lambda)$ is the likelihood of the speech segment $\mathcal{O}$ given the acoustic models λ, phone sequence H and the language L. The probability $P(H|L)$ is estimated using the language n-gram model (see Equation 5.2).

$$L^* = \underset{H}{argmax} \sum_{H} f(\mathcal{O}|H, L, \lambda) P(H|L) \tag{5.4}$$

Similarly to training n-gram language models from strings, we can use phone lattices to obtain better maximum likelihood (ML) estimates. In both cases, the ML training relies on finding maximum of $P(H|L)$ that maximizes Equation 5.2. If we consider that H is the string of phones (the best path through lattice) then the estimates of n-gram probabilities are just approximations. We can overcome this problem by summing likelihoods over all path in the lattice.

	training on string	training on lattice
scoring string	3.1	3.1
scoring lattice	3.6	2.3

Table 5.7: Experiments with phoneme strings and lattices on NIST LRE 2003 on 30sec duration.

Finding estimates of n-gram probabilities can be done iteratively by using EM algorithm. Given the current estimates M' of these probabilities for particular language, the next estimates are computed as the expectation of the n-gram frequencies $C(h_1, \ldots, h_n)$ which can be approximated by taking n-gram frequencies given the phone lattice $\mathcal{L}$. This gives us

$$E[C(h_1, \ldots, h_N)|\mathcal{O}, \lambda, M'] \simeq \sum_{h(e_i)=h_i} P(e_1, \ldots, e_N)|\mathcal{L}) \tag{5.5}$$

where the right part represent the sum of the lattice posterior probabilities of all sequences of n links corresponding to the phone n-gram $(h_1, \ldots, h_N)$ and is computed by means of the forward-backward algorithm

$$P(e_1, \ldots, e_N|\mathcal{L}) = \alpha(e_1)\beta(e_N) \prod_i \xi(e_i) \tag{5.6}$$

where $\alpha(e_1)$ is the forward probability of starting node of the link e, $\beta(e_N)$ is the backward probability of the end node of the link e and $\xi(e_i)$ is the posterior probability of the link e. Now the new estimates can be used to recompute the posterior probabilities in the lattice (acoustic probabilities stay unchanged, only the language probability change) for the new EM iteration. The EM procedure can be initialized with uniform distribution as suggested by Gauvain [57] or with the estimates computed from the string.

Testing

Scores for test segments are computed from phone lattices. Triphone expanded phoneme lattices are generated without any language model. Partial scores are given by trigram probabilities corresponding to triphone links weighted by their respective posteriors. The total score of segment is then computed as a sum of partial scores according to Equation 4.1.

Experiments

I have generated phoneme lattices only from acoustic scores without introducing any phonotactic constraints. Language models for each language are computed from n-gram frequencies given by all phone lattices belonging to language L.

All four combinations of LM-estimation and scoring (see Table 5.7) were tested. In the table, we see that training on lattice and scoring on raw strings does not bring any improvement and training on strings and scoring lattices even degrades performance of the system. The most intuitive lattice-lattice setup performs the best bringing approximately 25% relative improvement in EER (see Figure 5.2).

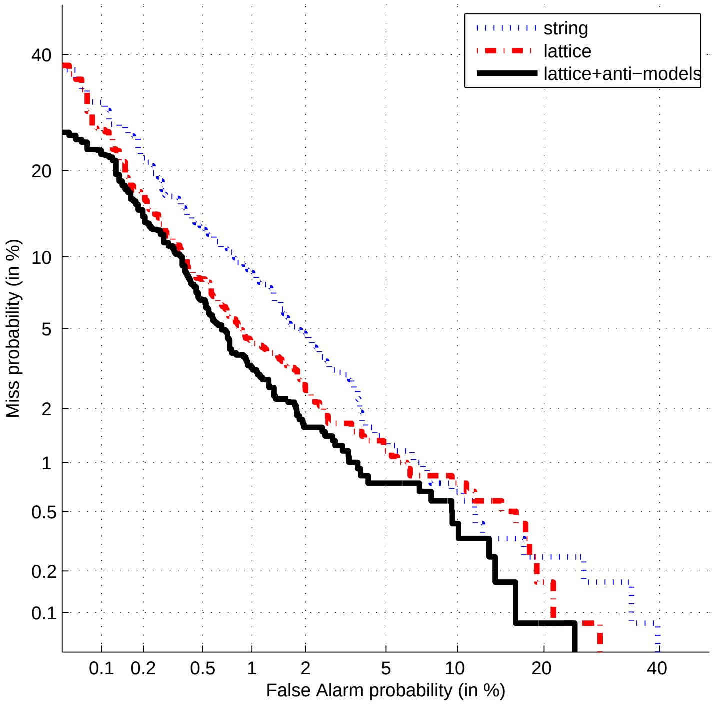

Figure 5.2: String vs. lattice based approach to LID on 30 sec segments from 2003 LID NIST

5.5 Anti-models

Anti-models are inspired by boosting training techniques and discriminative-like training. Anti-model is a language model modeling the space where target model makes mistakes [63]. Its training works in the following way: we will denote all utterances belonging to language L as set S_L^+ and all utterances not belonging to language L as set S_L^-. First, the training of phonotactic model LM_L^+ of each language L is done in standard way using only set S_L^+. Then, all *training* utterances are scored by all phonotactic models and posteriors of utterances are derived:

$$P(H_r|L) = \frac{p(H_r|LM_L^+)}{\sum_{\forall L} p(H_r|LM_L^+)}, \tag{5.7}$$

where H_r is the r-th training utterance representing phoneme sequence and $p(H_r|LM_L^+)$ is the likelihood provided by phonotactic model LM_L^+.

For language L, the parameters of anti-model LM_L^- should be trained on all segments from S_L^- mis-recognized as L. We can however use *all* utterances $H_r \in S_L^-$ and weight their trigram counts by posteriors $P(H_r|L)$. Obviously, an utterance from S_L^- with high probability to be mis-recognized as L will contribute more to the anti-model than an utterance correctly recognized as language G where $G \neq L$ (see Figure 5.3).

I have tested three flavors of anti-model training:

1. LM_L^- is estimated from segments of S_L^- but also from S_L^+. We could call this model “normalizing model” rather than anti-model.

2. LM_L^- is estimated only from segments of S_L^- with posterior weighting of trigram counts (Equation 5.7).

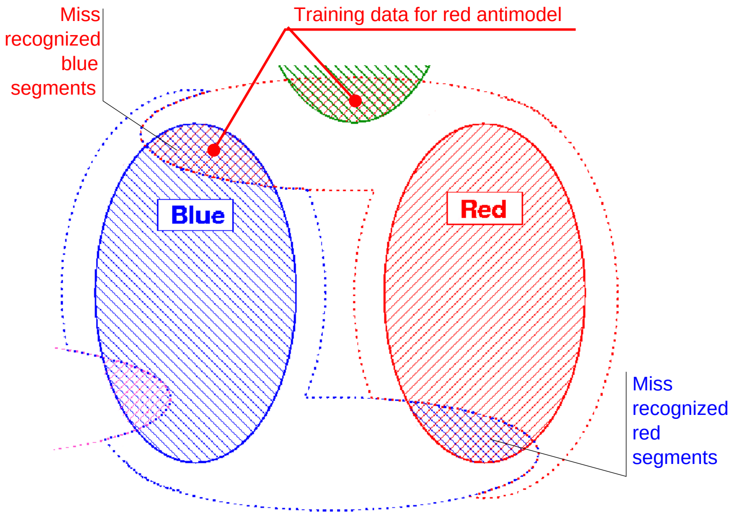
Miss recognized blue segments
Training data for red antimodel
Blue
Red
Miss recognized red segments

Figure 5.3: Illustration of the training of anti-models.

PRLM	lattice			lattice + anti-models		
duration **30s**	**10s**	**3s**	**30s**	**10s**	**3s**	
Hungarian	2.3	7.3	19.4	1.8	6.6	18.8
Russian	2.3	7.7	18.8	2.0	6.8	18.9
Czech	3.4	9.4	21.4	2.7	8.8	21.3

Table 5.8: EER [%] with using lattice and lattice + anti-models on LRE NIST 2003

3. LM_L^- is estimated only from segments of S_L^-, but in addition to posterior weighting, the trigram counts are also inversely weighted by the priors of different languages in S_L^-. For example, when we train anti-model for Arabic and we see 90% English and 10% of Tamil in S_L^-, the counts of English segments are divided by 0.9 and these of Tamil by 0.1.

In all three cases, the final score of test utterance H is obtained by subtracting the weighted likelihood of anti-model from the target model:

$$\log \mathcal{S}(H|L) = \log p(H|LM_L^+) - k \ \log p(H|LM_L^-), \tag{5.8}$$

where the constant k needs to be tuned experimentally.

In all anti-model experiments, language models were trained and evaluated on lattices and Witten-Bell discounting was used. Figure 5.4 presents the resulting EERs of the system for different settings of k. For $k = 0$ (no anti-model), all results are equal to EER=2.25% as already reported in Table 5.7. We see that all three anti-models improve the results. The normalizing LM_L^- is the worst, and the position of its minimum EER is very sensitive on optimal tuning of k. On the other hand, "pure" anti-model does well with a stable minimum at $k = 0.3$. The anti-model using *all data* from S_L^- is preferred. The results were verified also with other test segment durations (10s and 3s), another phone recognizers and different target data (NIST 1996), with the same stable peak at $k = 0.3$. The results for $k = 0.3$ for all recognizers are in Table 5.8. The improvement for Hungarian phone recognizer can be seen also on Figure 5.2.

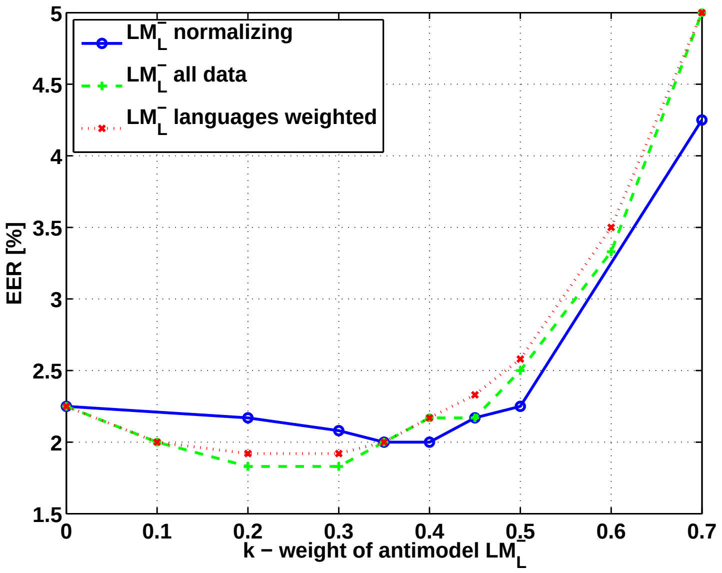

Figure 5.4: Results with different anti-models.

5.6 Conclusion

Experiments with increasing amount of data (see Table 5.4) clearly prove that there is a strong correlation between the performance of the phone tokenizer and performance of the whole language recognition system. We have proved that *Better phone recognizer => better LID system*. The conclusion from this is also that it is necessary to train phoneme recognizer on a lot of data. If we compare results with OGI Stories database and SpeechDat-E database we can conclude that it is better to use less well trained tokenizers than more poor one.

From comparison of our system with other systems in LID community we can conclude that we have good tokenizer which is very suitable for the LID task.

It is very beneficial to use lattices at the output of the phoneme recognizer for the LID. I observed about 25% relative improvement over the one-best output.

Anti-models provided initially promising results but as we will see in the following chapters, these could be too tightly bound to the NIST LRE 2003 data and were not fully reproduced on later (and more challenging) data. A brief discussion is provided in the final Conclusions.

In this chapter, the classical n-gram modeling of phoneme strings and lattices was used. We know however that this might be the weak part of the system and therefore, modeling using binary decision trees and adaptation of both n-gram models and trees from a Universal Background Model (UBM) is addressed in Chapter 8.

Chapter 6

Acoustic Modeling – GMM

6.1 Introduction

This chapter concentrates on acoustic modeling using GMM and complements the successful PPRLM described in previous chapter.

In acoustic modeling, we were inspired by the advantages brought by discriminative training into large vocabulary continuous speech recognition (LVCSR) systems where it leads to consistent improvement in accuracy [72, 92]. To our knowledge, training GMMs using MMI criterion has not been tested in LID so far[1]. Dan and Bingxi [54] reported results with Minimum classification error (MCE) criterion for the training, but the improvement they obtained was less than shown in this thesis. The performance of acoustic modeling was also improved increased by use of Heteroscedastic linear discriminant analysis (HLDA), also in common use in LVCSR.

This work on discriminative training for LID was facilitated by the experience with discriminative training applied in AMI-LVCSR system[2] [93]. I could also rely on our HMM toolkit STKthat implements discriminative training techniques.

6.2 Features

The most widely used features for LID (as well as for other speech processing techniques) are Mel-Frequency Cepstral Coefficients (MFCC). The works of Torres-Carasquillo [9] and others have, however, shown the importance of a broader temporal information for LID. Later Castaldo explored usage of time-frequency features which can probably extract the same information as SDC [66]. The shifted delta cepstra (SDC) features are created by stacking delta-cepstra computed across multiple speech frames. They are specified by a set of 4 parameters: N, d, P and k, where N is the number of cepstral coefficients, d is the window over which the deltas are computed for the delta-computation, k is the number of blocks whose delta-coefficients are concatenated to form the final feature vector and P is the time shift between consecutive blocks (see Figure6.1). In case we denote the original features $o_q(t)$[3], shifted deltas are defined:

$$\Delta o_q(t) = o_q(t + iP + d) - o_q(t + iP - d)$$

for $i = 0, P, 2P, \ldots, (k-1)P$. Obviously, the coefficients of SDC vectors are heavily correlated (most of elements are merely copied from one vector to another).

[1] We refer to state-of-the-art in 2005 [14]

[2] AMI is EC-sponsored project Augmented Multi-Party Interaction, http://www.amiproject.org

[3] $o_q(t)$ denotes the q-th element of feature vector $\mathbf{o}(q)$

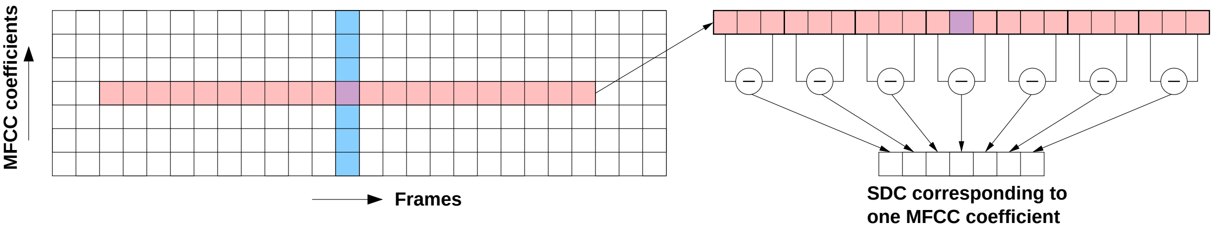

Figure 6.1: Shifted Delta Cepstra

Two widely used enhancements of features for LID are evaluated. RASTA filtering of cepstral trajectories providing simple channel normalization [94] and vocal-tract length normalization (VTLN) [95] which is a simple speaker adaptation.

6.2.1 VTLN

VTLN is a simple speaker normalization technique that can be implemented by modifying the frequency axis in the filterbank analysis. Vocal tract length normalization (VTLN) aims to compensate for the fact that speakers have vocal tracts of different sizes [95, 96].

6.2.2 RASTA

The word RASTA stands for RelAtive SpecTrAl Technique. RASTA filter removes slowly varying, linear channel effects from the feature vectors [97].

In this process, each feature vector's individual elements, considered to be separate streams of data, are passed through identical filters that remove near-DC components along with some higher frequency components.

I use the standard RASTA IIR filter

$$H(z) = 0.1 \times \frac{2 + z^{-1} - z^{-3} - 2z^{-4}}{z^{-4}(1 - 0.982z^{-1})} \tag{6.1}$$

6.3 Acoustic modeling

Language recognition can be seen as a classification problem with each language representing a class. In case, we base this classification on acoustic features, the straightforward way to model class L is to construct a Gaussian mixture model that represents feature vectors by a weighted sum of multivariate Gaussian distributions:

$$p_\lambda(\mathbf{o}(t)|L) = \sum_{m=1} c_{Lm}\mathcal{N}(\mathbf{o}(t); \boldsymbol{\mu}_{Lm}, \boldsymbol{\sigma}^2_{Lm}) \tag{6.2}$$

where $\mathbf{o}(t)$ is the input feature vector and the parameters λ of model of L-th class are c_{Lm}, $\boldsymbol{\mu}_{Lm}$ and $\boldsymbol{\sigma}^2_{Lm}$: mixture weight, mean vector and variance[4] vector respectively. The log likelihood of utterance $\mathcal{O}_r$ given a class L, is then defined as:

$$\log p_\lambda(\mathcal{O}_r|L) = \sum_{t=1}^{T_r} \log p_\lambda(\mathbf{o}(t)|L) \tag{6.3}$$

where T_r is the number of feature vectors in $\mathcal{O}_r$.

[4] we assume diagonal covariance matrices that can be represented by variances.

6.3.1 Maximum Likelihood training

In the standard Maximum Likelihood (ML) training framework, the objective function to maximize is the total (log) likelihood of training data given their correct transcriptions:

$$\mathcal{F}_{ML}(\lambda) = \sum_{r=1}^{R} \log \ p_{\lambda}(\mathcal{O}_r|L_r) \tag{6.4}$$

where λ denotes the set of model parameters, $\mathcal{O}_r$ is r-th training utterance, R is the number of training utterances and L_r is the correct transcription (in our case the correct language identity) of the r-th training utterance. To increase the objective function, the GMM parameters are iteratively estimated using well known EM algorithm reestimation formula [96].

In this algorithm, occupation probabilities, $\gamma_j(t)$, and feature vectors $\mathbf{o}(t)$ are used to estimate n-dimensional mean vector, $\boldsymbol{\mu}_j$, and full covariance $n \times n$ matrix, $\boldsymbol{\Sigma}_j$, of each Gaussian mixture component, j, according to the following equations:

$$\hat{\boldsymbol{\mu}}_j = \frac{\sum_{t=1}^{T} \gamma_j(t)\mathbf{o}(t)}{\gamma_j}, \tag{6.5}$$

$$\hat{\boldsymbol{\Sigma}}_j = \frac{\sum_{t=1}^{T} \gamma_j(t)(\mathbf{o}(t) - \hat{\boldsymbol{\mu}}_j)(\mathbf{o}(t) - \boldsymbol{\mu}_j)^T}{\gamma_j}, \tag{6.6}$$

$$\gamma_j = \sum_{t=1}^{T} \gamma_j(t) \tag{6.7}$$

where T is the number of feature vectors used for training.

For more detail see for example HTK book [96].

6.3.2 Discriminative training

In discriminative training, the objective function is designed in such a way that it is (or is believed to be) better connected to the recognition performance. One of the most popular discriminative training technique nowadays is Minimum Classification Error (MCE) and Maximum Mutual Information (MMI) training [72].

Minimum Classification Error - MCE

The Minimum Classification Error objective function is defined for task where the utterance can belong to one of a fixed number of classes: in our case languages $L = 1...K$. The class-conditional likelihoods are defined as a sum of all utterance likelihoods Equation 6.3 for the particular class (language).

$$g_L(\mathcal{O}; \lambda) = \sum_{r=1}^{R} \log p_{\lambda}(\mathcal{O}_r|L) \tag{6.8}$$

A *misclassification measure* for each class is defined as follows:

$$d_L(\mathcal{O}) = -g_L(\mathcal{O}; \lambda) + \log \left[\frac{1}{M-1} \sum_{k,k \neq L} \exp g_k(\mathcal{O}; \lambda)\eta \right]^{1/\eta} \tag{6.9}$$

which tends to be positive if the system does not classify the utterance as being class L, and negative if the utterance is classified as class L. The misclassification measure is then embedded in sigmoid function:

$$l_L(\mathcal{O}) = \frac{1}{1 + \exp(-\gamma d_L(\mathcal{O}))} \tag{6.10}$$

where $\eta > 0$ and $\gamma > 0$ are constants. The objective function, which is to be minimized, is the sum of $l_L(\mathcal{O})$ over all classes.

$$F_{MCE}(\lambda) = \sum_L l_L(\mathcal{O}) \tag{6.11}$$

The contribution to the objective function is close to zero for each sentence that is correctly recognized and one for an incorrectly recognized sentence. The transition between is controlled by the parameters γ and η.

Maximum Mutual Information - MMI

Unlike in the case of ML training, which aims to maximize the overall likelihood of training data given the transcriptions, MMI objective function to maximize is the posterior probability of correctly recognizing all training segments (utterances):

$$\mathcal{F}_{MMI}(\lambda) = \sum_{r=1}^{R} \log \frac{p_\lambda(\mathcal{O}_r|L_r)^{K_r} P(L_r)}{\sum_{\forall L} p_\lambda(\mathcal{O}_r|L)^{K_r} P(L)}. \tag{6.12}$$

where $p_\lambda(\mathcal{O}_r|L_r)$ is likelihood of r-th training segment, $\mathcal{O}_r$, given the correct transcription (in our case the correct language identity) of the segment, L_r, and model parameters, λ. R is the number of training segments and the denominator represents the overall probability density, $p_\lambda(\mathcal{O}_r)$ (likelihood given any language). We consider the prior probabilities of all classes (languages) equal and drop the prior terms $P(L_r)$ and $P(L)$. Usually, segment likelihood $p_\lambda(\mathcal{O}_r|L)$ is computed as simple multiplication of frame likelihoods incorrectly assuming statistical independence of feature vectors. Factor $0 < K_r < 1$, which is increasing the confusion between hypothesis represented by numerator and denominator, can be considered as a compensation for underestimating segment likelihoods caused by this incorrect assumption. In our experiments, this factor was empirically determined as $K_r = C/T_r$, where C is a constant dependent on type of features (6 in our case) and T_r is number of frames in r-th segment.

It can be shown [72] that MMI objective function (6.12) can be increased by re-estimating model parameters using extended Baum-Welch algorithm with the following formula for updating mean and variances:

$$\begin{aligned} \hat{\boldsymbol{\mu}}_{Lm} = & \quad \frac{\theta^{num}_{Lm}(\mathcal{O}) - \theta^{den}_{Lm}(\mathcal{O}) + D_j \boldsymbol{\mu}'_{Lm}}{\gamma^{num}_{Lm} - \gamma^{den}_{Lm} + D_{Lm}} \\ \hat{\boldsymbol{\sigma}}^2_{Lm} = & \quad \frac{\theta^{num}_{Lm}(\mathcal{O}^2) - \theta^{den}_{Lm}(\mathcal{O}^2) + D_{Lm}(\boldsymbol{\sigma}'^2_{Lm} + \boldsymbol{\mu}'^2_{Lm})}{\gamma^{num}_{Lm} - \gamma^{den}_{Lm} + D_{Lm}} - \hat{\boldsymbol{\mu}}^2_{Lm} \end{aligned} \tag{6.13}$$

where L and m are identities of model (language) and its mixture component, $\boldsymbol{\mu}'_{Lm}$ and $\boldsymbol{\sigma}'^2_{Lm}$ are old means and variances and D_{Lm} is smoothing constant controlling the speed of convergence, which is set to be greater than

- twice the value ensuring all variances to be positive
- $E\gamma^{den}_{Lm}$ where E is another constant (I use $E = 2$).

The terms:

$$\begin{aligned} \theta^{num}_{Lm}(\mathcal{O}) &= \sum_{r=1}^{R}\sum_{t=1}^{T_r}\gamma^{num}_{Lmr}(t)\mathbf{o}_r(t) \\ \theta^{num}_{Lm}(\mathcal{O}^2) &= \sum_{r=1}^{R}\sum_{t=1}^{T_r}\gamma^{num}_{Lmr}(t)\mathbf{o}_r(t)^2 \\ \gamma^{num}_{Lmr} &= \sum_{r=1}^{R}\sum_{t=1}^{T_r}\gamma^{num}_{Lmr}(t) \end{aligned} \tag{6.14}$$

are mixture component specific first and second order statistics and occupation counts corresponding to the numerator of the objective function (6.12). Denominator statistics can be expressed by similar equations, where all superscripts *num* are merely replaced by *den*. Note that the numerator statistic are ordinary ML statistics. Therefore the numerator posterior probability of occupying mixture component Lm by t-th frame of training segment r,

$$\gamma^{num}_{Lmr}(t) = \left\{ \begin{array}{ll} \gamma_{Lmr}(t) & \text{for } \; L = L_r \\ 0 & \text{otherwise} \end{array} \right. , \tag{6.15}$$

is nonzero only for mixture components corresponding to correct class (language). To estimate the posterior probabilities for the denominator:

$$\gamma^{den}_{Lmr}(t) = \gamma_{Lmr}(t)\frac{p_{\acute{\lambda}}(\mathcal{O}_r|L)^{K_r}}{\sum_{\forall q} p_{\acute{\lambda}}(\mathcal{O}_r|q)^{K_r}}, \tag{6.16}$$

likelihoods $p_{\acute{\lambda}}(\mathcal{O}_r|q)$ are evaluated using old parameters $\acute{\lambda} = \{\boldsymbol{\mu}', \boldsymbol{\sigma}'^2\}$. The factor K_r is discussed above. Finally,

$$\gamma_{Lmr}(t) = W_s \frac{c_{Lm}\mathcal{N}(\mathbf{o}_r(t); \boldsymbol{\mu}'_{Lm}, \boldsymbol{\sigma}'^2_{Lm})}{\sum_{j=1}^{J_L} c_{Lj}\mathcal{N}(\mathbf{o}_r(t); \boldsymbol{\mu}'_{Lj}, \boldsymbol{\sigma}'^2_{Lj})} \tag{6.17}$$

where c_{Lm} is mixture component weight, $\mathcal{N}(\cdot\,; \boldsymbol{\mu}'_{sm}, \boldsymbol{\sigma}'^2_{Lm})$ is evaluated for old mean and variance estimates and J_L is number of mixture components in model L.

The derivation of parameter update formula is described in detail for example in [72]. A formula for discriminative update of mixture component weights can be also derived [72], however, we train these weights only in ML iterations and keep them fixed in MMI iterations as their discriminative update is not expected to bring any significant improvement.

MCE and MMI Implementation

In principle, it is possible to implement MCE and MMI into a single criterion [72, 98]:

$$\mathcal{F}(\lambda) = \sum_{r=1}^{R} f\left(\log\frac{p_\lambda(\mathcal{O}_r|\mathcal{M}_{L_r})P(L_r)}{p_\lambda(\mathcal{O}_r|\mathcal{M}_{rec_r})}\right) \tag{6.18}$$

where for

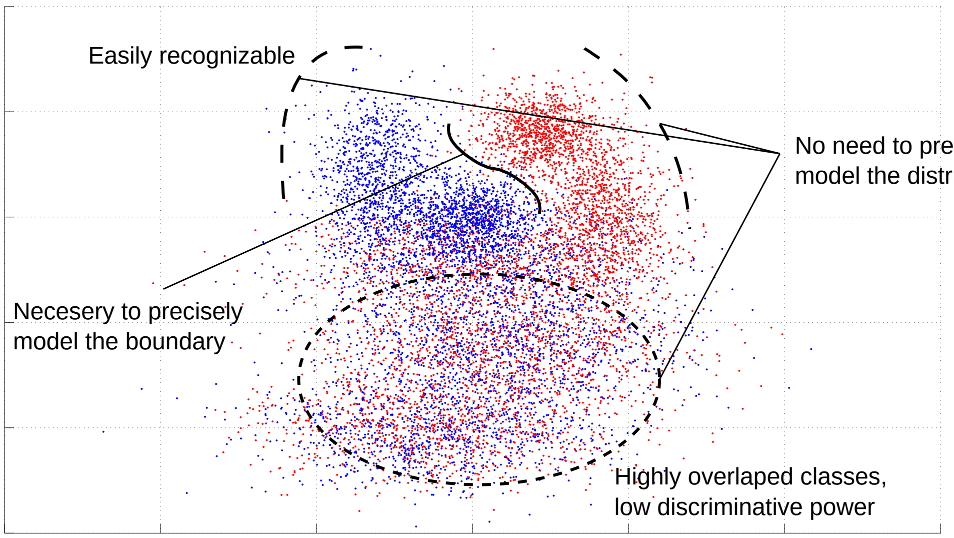

Figure 6.2: Highly overlapped feature distribution - differences between ML and discriminative training

- MMI we set the function f to $f(x) = x$ and include all sentences in the composite model $\mathcal{M}_{rec_r}$
- MCE we set $f(x) = -\frac{1}{1+e^{\gamma x}}$ and exclude the correct sentence from $\mathcal{M}_{rec_r}$. L_r is the correct transcription of r'th utterance.

The equivalence with MCE is valid if $\eta = \gamma$ and ignoring factor $\frac{1}{M-1}$ in Equation 6.9. Note that MCE training can be implemented as a modification of the Extended Baum-Welch (EB) procedure for MMI training, by scaling the state occupation probabilities and sums of data accumulated from each training file by the value of $\partial f(x)/\partial x$, i.e. the differential of the sigmoid function, where x is the value of the MMI criterion for that file [98].

Summary

Two discriminative training techniques were presented, but which one should theoretically perform the best for LID?

In **MCE** training, the system is trained to perform the best for the particular priors. MCE tries to recognize correctly all training examples. In case all training examples are correctly recognized, the model is not further updated. The sentences far from the classification boundary (distance from it depends on the slope of the sigmoid) do not have any effect on estimating new parameters of the model λ.

In **MMI** training, the criterion is the posterior probability of all training data being recognized correctly. MMI optimizes the system for detection and all operating points are equally important. MMI takes into account also priors and fixed number of languages.

Since I evaluate LID in terms of DET curves, where the recognition depends on different thresholds, the MMI criterion should be the better one, as it trains the model to be optimal for detection and all operating points.

The advantages and disadvantages of MMI are the following:

- It concentrates on precise modeling of decision boundary and does not waste the parameters on highly overlapped features with low discriminative power (see Figure 6.2).
- It optimizes parameters for good detection of whole segments (not individual frames) and therefore takes into account the enormous importance of correct speech segmentation.
- The drawback of MMI is that it also learns the (undesirable) language priors from training data. It is necessary to equalize data for this type of training.
- Another drawback are errors in reference labels in the data. Imagine for example the case where the Mandarin sentence has the English label. MMI forces to recognize Mandarin sentences as English and on contrary to ML training where this sentence can influence only one model, in MMI it influences all models.

The derivation of parameter update formula for all mentioned discriminative training methods is described in detail in [72].

6.4 HLDA

The Heteroscedastic Linear Discriminant Analysis [99] can be used to derive linear projection de-correlating feature vectors and performing dimensionality reduction. For HLDA, each feature vector that is used to derive the transformation must be assigned to a class. When performing the dimensionality reduction, HLDA allows to preserve useful dimensions, in which feature vectors representing individual classes are best separated (Figure 6.3). HLDA allows to derive such projection that the best de-correlates features associated with each particular class (maximum likelihood linear transformation for diagonal covariance modeling [99, 100]). To perform de-correlation and dimensionality reduction, n-dimensional feature vectors are projected into first $p < n$ rows, $\mathbf{a}_{k=1\ldots p}$, of $n \times n$ HLDA transformation matrix, $\mathbf{A}$. An efficient iterative algorithm [100] is used in our experiments to estimate matrix $\mathbf{A}$, where individual rows are periodically re-estimated using the following formula:

$$\hat{\mathbf{a}}_k = \mathbf{c}_k \mathbf{G}^{(k)-1} \sqrt{\frac{T}{\mathbf{c}_k \mathbf{G}^{(k)-1} \mathbf{c}_k^T}} \tag{6.19}$$

where $\mathbf{c}_i$ is the i^{th} row vector of co-factor matrix $C = |\mathbf{A}|\mathbf{A}^{-1}$ for the current estimate of $\mathbf{A}$ and

$$\mathbf{G}^{(k)} = \begin{cases} \sum_{j=1}^{J} \frac{\gamma_j}{\mathbf{a}_k \hat{\mathbf{\Sigma}}^{(j)} \mathbf{a}_k^T} \hat{\mathbf{\Sigma}}^{(j)} & k \leq p \\ \frac{T}{\mathbf{a}_k \hat{\mathbf{\Sigma}} \mathbf{a}_k^T} \hat{\mathbf{\Sigma}} & k > p \end{cases} \tag{6.20}$$

where $\hat{\mathbf{\Sigma}}$ and $\hat{\mathbf{\Sigma}}^{(j)}$ are estimates of global covariance matrix and covariance matrix of j^{th} class, γ_j is the soft count of training feature vectors belonging to j^{th} class and T is the total number of training feature vectors.

In our experiments, the classes are defined by each Gaussian mixture component m of each language. The selection, that feature vector $\mathbf{o}(t)$ belong to class j, is given by the value of occupation probability $\gamma_j(t)$ from the standard GMM training algorithm (see Section 6.3.1). New HLDA projection, $\mathbf{A}$, is then derived using the occupation probabilities and the estimated class covariance matrices, $\hat{\mathbf{\Sigma}}_j$.

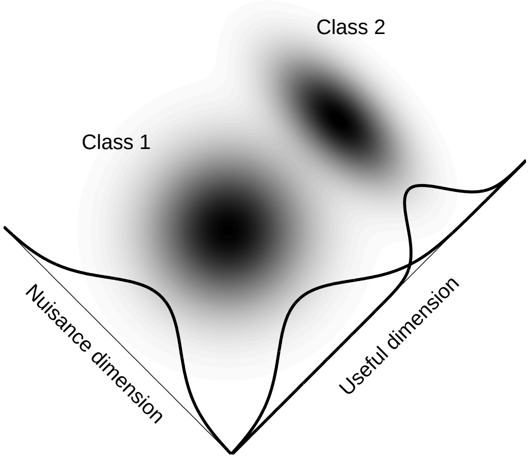

Figure 6.3: Heteroscedastic Linear Discriminant Analysis for 2-Dimensional Data

To obtain the correct estimates of HMM parameters in feature space corresponding to the newly derived transformation, $\mathbf{A}_p$, p-dimensional mean vector, $\hat{\boldsymbol{\mu}}_j^{HLDA}$, and variance vector, $\hat{\boldsymbol{\sigma}}_j^{HLDA}$, of each Gaussian mixture component is updated according to the following equations:

$$\hat{\boldsymbol{\mu}}_j^{HLDA} = \mathbf{A}_p \hat{\boldsymbol{\mu}}_j, \tag{6.21}$$

$$\hat{\boldsymbol{\sigma}}_j^{HLDA} = diag(\mathbf{A}_p \hat{\boldsymbol{\Sigma}}_j \mathbf{A}_p^T). \tag{6.22}$$

where $\mathbf{A}_p$ is matrix consisting of first p rows of matrix $\mathbf{A}$.

6.5 Experiments

6.5.1 Setup

Two training setups are used for experiments in this chapter. Both are derived from CallFriend corpus described in Section 4.3. All training data were end-pointed by our phoneme recognizer with all phoneme classes except `sil` linked to 'speech'. Only segments with label 'speech' are used for training. The **full set** contains training data from all languages, segments shorter than half second are left out. For **reduced set**, 1 hour of training data was selected from each target language by taking only segments longer than 2 seconds and balancing the amounts of data among speakers in each language. All results are reported on 30 second segments from LRE 2003 set (see Section 4.2).

6.5.2 Features

Only features derived from MFCC coefficients are used:

- For the `MFCC38` and `MFCC39` systems, the setup includes 12 direct coefficients, c_0, Δ and $\Delta\Delta$ computed with 3 frames window. The final coefficients exclude or include c_0, therefore 38 or 39 coefficients.

Features	c_0	RASTA	VTLN	# Gauss	
				512	2048
MFCC38	-	-	-	18.3	17.0
MFCC39	√	-	-	21.7	20.8
MFCC38	-	√	-	18.3	16.2
MFCC39	√	√	-	17.3	16.3
MFCC38	-	√	√	14.6	13.0
MFCC39	√	√	√	14.3	12.3
MFCC-SDC	-	-	-	12.5	10.5
MFCC-SDC	√	-	-	15.8	13.8
MFCC-SDC	-	√	-	11.3	8.7
MFCC-SDC	√	√	-	9.3	8.7
MFCC-SDC	-	√	√	8.8	**6.6**
MFCC-SDC	√	√	√	8.5	**6.6**
MFCC-SDC noDirect	-	√	√	11.4	9.3

Table 6.1: EER [%] comparison of different kind of features on reduced training set

- In the `MFCC-SDC` setup, several experiments were carried out with different parameters of shifted-delta computation. I ended up with the same setup as reported in [12]: parameters N, d, P and k set to 7, 1, 3, 7 producing 49-dimensional feature vectors. Results are usually reported only with SDC features, which do not include direct coefficients. I studied the influence of adding direct coefficients and c_0 too and found out that direct coefficients help, therefore `MFCC-SDC`.

Vocal-tract length normalization (VTLN) [95] warping factors are estimated using single GMM (512 Gaussians) with MFCC38, ML-trained on the whole CallFriend database (using all the languages). The model was trained in standard speaker adaptive training (SAT) fashion in four iterations of alternately re-estimating the model parameters and the warping factors for the training data.

6.5.3 Maximum likelihood training

This section describes the effects of different features and enhancements. The effect of varying number of Gaussians is also studied. Table 6.1 presents results with different feature extractions (MFCC, MFCC-SDC, adding c_0) and applying RASTA and VTLN. Results are obtained using reduced training data set with 512 and 2048 Gaussian mixtures.

The system with 2048 Gaussian mixture component behaves the following way: Adding c_0 to pure MFCC coefficients degrades the performance from 17% to 20.8% but by applying RASTA filtering the system performs about 1% better. There is almost no difference in performance by adding c_0 to this system. Adding VTLN to this system brings improvement about 3% absolute.

The baseline for SDC features is 10.5% which is much better than MFCC baseline and also than the best MFCC results with all enhancements. As in MFCC adding c_0 degrades performance and using RASTA improves it to 8.7% and suppresses the degradation by using c_0. Adding VTLN improves further the system by about 2% to final EER=6.6%. The comparison with SDC features without direct coefficients as proposed in [12] is also in the table.

Upper part of Table 6.4 reports the results for ML-training for the number of Gaussian components M varying from 16 till 2048. It is obvious that MFCC-SDC clearly outperform

Features	EER [%]
MFCC-SDC	10.5
MFCC-SDC RASTA	6.6
MFCC-SDC VTLN RASTA	**4.8**

Table 6.2: Influence of RASTA and VTLN using full training set

Features	EER [%]
MFCC-SDC no c_0	4.8
MFCC-SDC noDirect	6.8
MFCC39 D2	10.4
MFCC39 D3	10.8
MFCC39 D4	11.1
MFCC39 D5	11.8
MFCC39 no c_0 D3	11.3

Table 6.3: Comparison of SDC-MFCC with MFCC with different window sizes for computing delta coefficients, RASTA, VTLN and full training set are used.

MFCC which is coherent with the results of other groups. We see also decreasing EER for increasing number of Gaussian components. However, this increase is at price of much higher computational cost during training even for the reduced training data set. The advantage is that for small amount of data the system can benefit a lot from MMI without over-training (see the following section).

From the results obtained using reduced training data set, it is evident that RASTA and VTLN help for MFCC and also for MFCC-SDC features. Table 6.2 shows how much the technique improves the system trained on full training data set.

To compare SDC-MFCC with MFCC correctly in terms of captured time information (time context from which the coefficients are computed) I run an experiment where I increased the size of the window for computing delta coefficients see Table 6.3. Results for delta coefficients computed from window up to 5 frames are presented. Changing this size affects the results in the range of $\sim 1\%$ therefore SDC coefficients are able to extract more relevant information than delta coefficients.

The features with the best results are MFCC-SDC which are SDC coefficients with 7 direct MFCC and c_0. This makes the resulting feature vector 56 dimensional (7 SDC x 7 MFCC + 7 direct MFCC = 56). RASTA and VTLN are applied. In further experiments, this setup serves as a baseline.

6.5.4 Discriminative training

I chose MMI training method for detailed comparison with standard ML training, because of theoretically and also experimentally the best results (see Table 6.6).

Lower part of Table 6.4 shows the results for MMI-training. We see that systems with discriminatively trained models clearly outperform the standard ML-trained ones by $30-60\%$ relative. In LVCSR, discriminative training usually yields only between 6-15% *relative* improvement [72], I have therefore tried to explain such a dramatic improvement in LID: In our opinion,

system	16	32	64	128	256	512	1024	2048
ML-TRAINING								
MFCC39	24.7	21.4	18.7	17.2	15.8	14.3	13.1	12.3
MFCC-SDC	21.7	17.6	13.8	11.8	10.2	8.5	7.1	6.6
MMI-TRAINING								
MFCC39	12.3	10.0	8.0	7.6	7.3	7.5	-	-
MFCC-SDC	7.8	6.8	5.3	4.8	4.8	**4.6**	-	-

Table 6.4: Comparison of ML and MMI training on reduced training data set

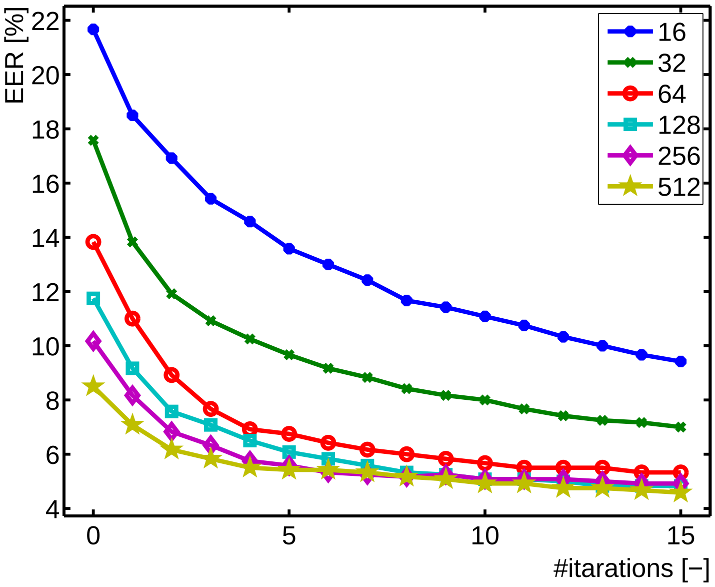

Figure 6.4: ERR [%] of different number of Gaussians trained under MMI framework on reduced training data set

the LVCSR-classes that are modeled by Gaussian mixtures are already relatively well separated in the acoustic space, so standard ML training does already well enough. In LID however, the classes are heavily overlapped and Gaussians occupy mostly the same acoustic space. The models need to concentrate on “tiny details” that help to separate the languages. With ML-training, the only possibility is the brute-force approach – increasing the number of Gaussians. On the other hand, discriminative training populates well these “tiny details” by definition.

Based on the results of this sections, I have continued with shifted delta-cepstra and discriminatively trained models. MMI is iterative, but usually requires only a few iterations to converge (Figure 6.4). Increasing the number of Gaussians in discriminative training improves the results only slightly (Tab. 6.4) with high computational cost during the training, therefore, I used $M = 128$ in the following experiments. Table 6.5 presents the results of the final system trained using full training data set for all tested durations from LRE 2003 with comparison to standard ML trained system with 2048 Gaussian components.

As mentioned in the summary of the theoretical part of this chapter, one potential disadvantage of MMI is that it learns language priors from training data. Therefore, I equalize the amounts of training data per language. The reduced data set was speaker and language bal-

system	**30s**	**10s**	**3s**
ML 2048	4.8	7.9	16.3
MMI 128	2.0	5.5	14.8

Table 6.5: Comparison of ERR [%] on LRE 2003 for ML and MMI trained systems on full training data set.

Training Criteria	# Gaussians [-]	EER [%]
ML	2048	6.6
MCE	128	5.3
MMI	128	**4.8**

Table 6.6: Different discriminative training criteria on reduced training data set

anced. In the full training data set, rather than equalizing the set by discarding parts of the training corpus, I appropriately weight segments in MMI re-estimation formulae [58].

Table 6.6 presents the results with different discriminative training criteria. MFCC-SDC features are used and the model is trained on reduced training data set.

6.5.5 HLDA

As the next step in the development of our LID system, I have employed Heteroscedastic linear discriminant analysis (HLDA) (see Section 6.4 for the theory). The reasons are obvious: our features are too highly-dimensional and (as it comes clearly from the nature of SDC) too correlated. HLDA provides a linear transformation that can de-correlate the features and reduce the dimensionality while preserving the discriminative power of features. In small- and large-vocabulary speech recognition [101, 102] HLDA consistently improved the recognition performance.

Results with HLDA for `MFCC-SDC` and $M = 128$ Gaussians are in Figure 6.5. Different lengths of the feature vector were tested with MMI training. Compared to non-HLDA result (reduced training data set, 4.8%), we see an improvement for wide range of feature dimensionalities; HLDA helps even in case we use it only to de-correlate, not to reduce the dimensionality (56 to 56 features). When the output dimensionality is tuned, we obtain EER of 4.1% on the reduced training data set which is 0.7% absolute improvement.

Table 6.7 shows results for HLDA on the full training data set, where HLDA did not improve the system — the number of dimensions (42) was tuned on the reduced data set, and for full training data, this reduction in dimensionality already seems to suppress useful information. This could be fixed by more careful tuning of the output dimensionality on the full training data set.

6.6 Conclusion

This chapter deals with acoustic modeling for language recognition. I have verified that the results and conclusions of other labs obtained with shifted delta-cepstra (SDC) features are valid and that these features are good for our task. I have concentrated on discriminative training methods and have shown, that MMI-based training of models for LID clearly outperforms widely used ML-training. MMI performed the best out of three tested discriminative methods. This

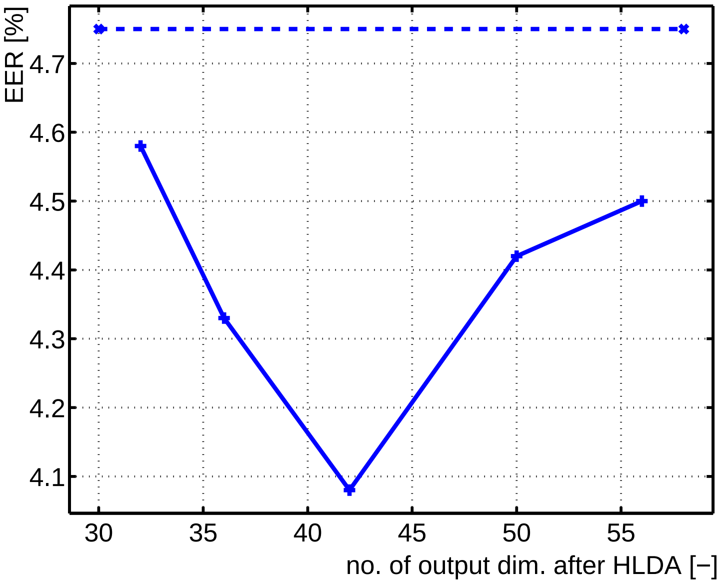

Figure 6.5: HLDA processing of features on reduced training data set. The dashed line shows the MMI result without HLDA.

system	data set	
	reduced	full
ML 2048	6.8	4.8
MMI 128	4.8	**2.0**
MMI 128 + HLDA	4.1	2.1

Table 6.7: ERR [%] for MMI-trained system, completed by HLDA. The results in the right column are for the full training data set. For comparison, results of ML-training are presented for M=2048 Gaussians.

verified also the assumption that for LID, discriminative training would bring more significant improvement than to LVCSR systems due to high overlap of classes in the feature space. I have also studied HLDA applied to de-correlate feature vectors and reduce their dimensionality. This technique helps for our reduced training data set, but the improvement did not fully generalize on the set with more training data.

Chapter 7

NIST Language Recognition Evaluation 2005

After evaluations in 1996 and 2003, NIST LRE 2005 was the 3^{rd} evaluation NIST conducted to establish a current baseline of performance capability for language and dialect recognition of conversational telephone speech[1].

Although we have worked on NIST 2003 data before and participated in the previous evaluation by contributing phone recognizer to OGI[2] submission to NIST 2003, NIST LRE 2005 was a land-mark as this was the first time my group – Speech@FIT – participated independently in the evaluation. We were encouraged by the obtained with the phonotactic system on NIST 2003 data (presented in previous chapters and in [18]). In the preparation phase for NIST LRE 2005, the acoustic system was defined with MMI training of models which proved to be the "best bet" in the evaluation.

My role in NIST LRE 2005 was to coordinate the whole Speech@FIT efforts and I was responsible for the phonotactic system (with great help of Petr Schwarz on the phone recognizer). The development of the acoustic system was lead by Lukáš Burget, although I contributed also to these efforts by setting-up the baseline maximum likelihood (ML) system, and running post-evaluation experiments.

This chapter is rather short as all the development of phonotactic and acoustic systems is described in respective chapters 5 and 6. The chapter concentrates on the comparison of different feature extraction techniques on NIST 2005 data and thoroughly analyzes the results obtained.

7.1 The data

NIST 2005 evaluation set contains test segments with three nominal durations of speech: 3 seconds, 10 seconds, and 30 seconds from a set of 7 languages and two dialects (English-American, English-Indian, Hindi, Japanese, Korean, Mandarin-Mainland, Mandarin-Taiwan, Spanish and Tamil). Actual speech durations varied but were constrained to be within the ranges of 2-4 seconds, 7-13 seconds, and 25-35 seconds of actual speech contained in segments, respectively. Opposed to the previous evaluations, the silence was not removed from speech so the actual length of the segment could be much longer.

Our **development data** comes from NIST 1996 and 2003 LRE (described in Section 4.1 and Section 4.2) plus 40 additional segments with duration around 20 sec from OGI Foreign

[1] NIST 2005 Language Recognition Evaluation: `http://www.nist.gov/speech/tests/lang/2005`

[2] Oregon Graduate Institute, Portland, Oregon, USA

Accented English database (Hindi part)[3] to compensate for the lack of English accented by Indian.

Our **training data** comes from the target languages from Call Friend, OGI Multilingual, OGI22 and OGI Foreign Accented English. From OGI Foreign Accented English database, only Hindi and Tamil English data were used as representatives of Indian English as we thought this dialect would be present in the evaluation data. However, in the evaluation Indian accented English were from all dialects but Hindi and Tamil.

For evaluation metric see Section 2.5.

7.2 System description

The system combines phonotactic and acoustic approaches. Two major contributions that made our system very successful in the evaluation are the use of discriminative techniques to train both acoustic GMM system (see Chapter 6 and [14]) and language model in phonotactic system (see Chapter 5 and [63]).

7.2.1 Acoustic approach

The features in our system are 7 MFCC coefficients (including coefficient C0) concatenated with SDC 7-1-3-7, which totals in 56 coefficients per frame. RASTA filtering of cepstral trajectories is used to alleviate channel mismatch [94] and Vocal-tract length normalization (VTLN) [95] performs simple speaker adaptation. VTLN warping factors are estimated using single GMM (512 Gaussians), ML-trained on the whole CallFriend database (using all languages). I used a segmentation generated by our Czech phone recognizer (see Section 3.2.4 and 7.2.2), where all phonemes are linked to 'speech' class and pause and speaker noises (breath, hesitation, etc.) to 'silence' class. The silence segments are not used for training and testing. Each speech segment generated by phone recognizer is taken as individual segment for MMI training and only segments longer than half second are used for training (about 4/5 of all available speech). Compared to the segments used in testing, the segments generated by phone recognizer are rather short (usually between 1 and 2 seconds).

One GMM with 256 mixture components was created for each of 7 target languages. Models were trained using data of target languages from Call Friend, OGI Multilingual, OGI22 and OGI Foreign Accented English[4]. We have however seen that the OGI databases did not provide much benefit, mainly because of their smaller size compared to CallFriend. Only seven target languages were used for building models, other languages were used for training out-of-set model.

Initial set of models was trained under conventional ML framework. These models served only as a starting point and were further discriminatively re-trained using Maximum Mutual Information estimation in about 20 iterations of MMI training (Section 6.3.2).

7.2.2 Phonotactic approach

The phonotactic system uses three phone recognizers trained on SpeechDat-E databases (Hungarian, Russian and Czech) and lattice based training and scoring of trigram language models. To further improve this system, I use language anti-models to correct mistakes of the target language model.

[3] Corpora from CSLU - Foreign Accented English : http://cslu.cse.ogi.edu/corpora/fae

[4] From OGI Foreign Accented English database, only Hindi and Tamil English data were used as representatives of Indian English (present in evaluation data).

Our phone recognizer uses a hierarchical structure of 3 neural nets where 2 of them process block of data to the past and future from the actual frame and the third functions as a merger and produces the final set of phoneme-state posterior probabilities (Section 3.2.4). All neural networks[5] have 1500 neurons in hidden layer.

A simple Viterbi decoder from our STK toolkit without any language model processes output of the merger and produces string of phonemes. Phoneme lattices are generated using HVite tool from HTK toolkit[6]. Full phone recognizer including the Hungarian, Czech and Russian nets used for this evaluation is available for download from BUT[7].

Each phonotactic system has its own segmentation derived from recognized phoneme strings. All silence classes, speaker and intermittent noises are merged together to form one silence class. If the silence is longer than 5 sec., the system flushes the previous segment and starts a new one.

Language models were computed from posterior weighted n-gram counts estimated on lattices (Section 5.4). I used smoothed trigram back-off language model with Witten-Bell discounting implemented in SRI LM toolkit[8] [103]. Anti-models, modeling the space where target model makes mistakes (Section 5.5) [63], were used. Models were trained using data of target languages from Call Friend, OGI Multilingual, OGI22 and Hindi part of OGI Foreign Accented English, other languages were used for training out-of-set model. As in the acoustic system, I have seen that the OGI databases did not provide much benefit.

Scores for test segments are computed from phoneme lattices, which are generated without any language model. Partial scores are given by the trigram probabilities weighted by the posterior weighted trigram count from the lattice. The total score of segment is then computed as a sum of partial scores.

7.2.3 Normalization and fusion

Normalization using likelihoods of all individual language detectors was done to obtain the final score of language L:

$$\log P(L|\mathcal{O}) \approx \log p(\mathcal{O}|L)/T - \log \sum_q p(\mathcal{O}|q)/T, \tag{7.1}$$

where $\log p(\mathcal{O}|s)$ is log-likelihood of speech segment $\mathcal{O}$ given by GMM or LM for language L. In case of PRLM, T is the expected number of phonemes in speech segment computed as sum of posterior probabilities of links in the phoneme lattice. T is the number of speech frames in the test segment in case of GMM.

To fuse scores from separate systems, a simple linear combination is done according to:

$$\begin{aligned} score \;&=\; \alpha\, GMM_{MMI} \;+\; \beta\, PRLM_{HU} \;+ \\ &\quad +\; \gamma\, PRLM_{RU} \;+\; \delta\, PRLM_{CZ} \end{aligned} \tag{7.2}$$

where weights $\alpha, \beta, \gamma, \delta$ are tuned by simplex method[9] on the development set. We are aware that linear combination is quite primitive and that better results should be obtained with more elaborate methods, such as GMM- or NN-based merging. We were however severely limited by the amounts of development data, especially by Indian-English (no development data at all).

[5] All nets are trained using QuickNet from ICSI http://www.icsi.berkeley.edu/Speech/qn.html

[6] http://htk.eng.cam.ac.uk

[7] http://www.fit.vutbr.cz/speech/index.php?id=phnrec

[8] http://www.speech.sri.com/projects/srilm

[9] The Simplex Method is the earliest solution algorithm for solving linear program problems. It is an efficient implementation of solving a series of systems of linear equations. By using a greedy strategy the algorithm terminates at an optimal solution.

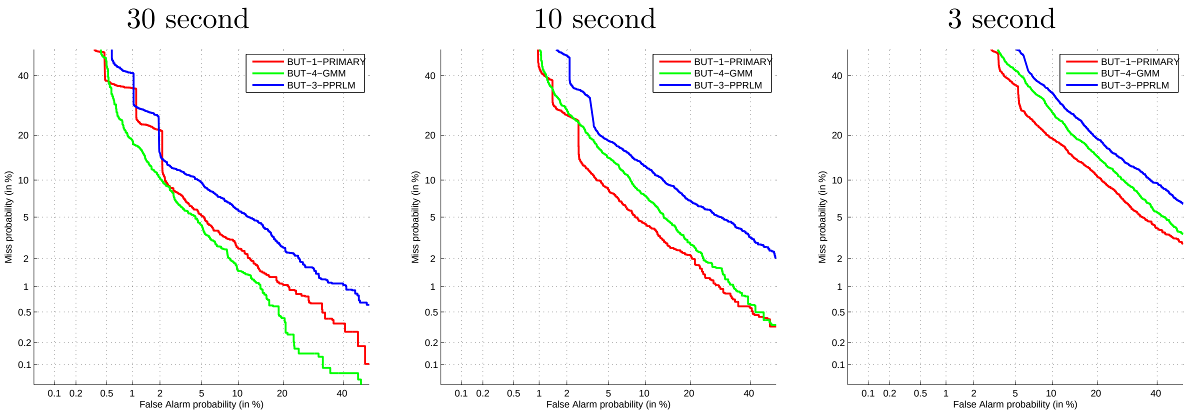

Figure 7.1: DET curve for submitted NIST LRE 2005 primary system for three test durations.

System EER [%]	Duration		
	30s	10 s	3 s
PRLM+lattice	9.83	14.93	23.44
PRLM+lattice+anti-model	8.49	13.82	22.98
PPRLM+lattice+anti-model	7.30	11.38	19.46
GMM-MMI 256	4.63	8.61	17.23
Fusion - submitted	5.01	6.54	14.14
Fusion - corrected	**3.09**	**6.54**	**14.14**

Table 7.1: EER for different system for NIST LRE 2005 - submission.

7.3 Primary condition - submitted

Table 7.1 shows the results of separate systems on LRE 2005 data in terms of EER. The presented PRLM system is based only on Hungarian phone recognizer (the best out of our 3 PRLM systems). PRLM with trigram language model derived from lattices and tested using lattices performs with EER=9.83%. Using anti-models results in relative improvement of 14%. Fusion of three phonotactic systems (PPRLM) gains EER=7.30% which is also almost 14% relative improvement. Although antimodels consistently helped, the results from lattice based PRLM have not fully met our expectation and are behind our previous work on NIST 2003 data (Section 5.4) [63]. The problems were addressed in post-evaluation (Section 7.4).

The EER of GMM-MMI 256 system is 4.63%, which confirmed our previous good results and superiority of discriminatively trained models over ML-trained ones [14].

Fusion of GMM-MMI and PPRLM systems gives 33% relative improvement over the best separate system and the final EER reaches 3.09%. Unfortunately, this is a post-evaluation results – due to bad fusion weights in Equation 7.2, the EER of our submitted 30 sec system was 5.01% (even worse than GMM-MMI 256 itself, see Figure 7.1). We have tuned weights for 30 sec condition on only 30 sec segments from the development data while we were in the range of EER below 1%, so we over-tuned the weights for the development data. The result 3.09% is with the weights for 10 sec condition applied in the 30s condition. The second and third columns in Table 7.1 show the results for 10 sec and 3 sec durations (here, the table contains the evaluation results – the weights were correct for these durations) and Figure 7.2 presents the

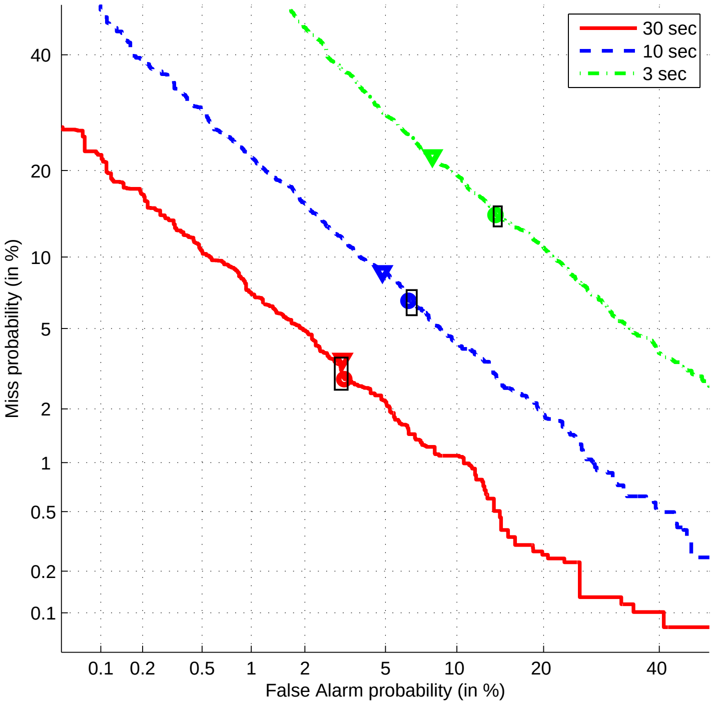

Figure 7.2: DET curve for NIST LRE 2005 primary system for three test durations. The triangle represents our actual decision, the circle is the minimum DET and the rectangle is the EER with its confidence interval.

DET curve for all systems. The triangle on the curve represents our actual decision, the circle represents the minimum DET and the rectangle is the EER with its confidence interval on the level of 95%.

7.4 Primary condition - post-evaluation

To simplify the comparison of the performance of our different systems, I will discuss only results obtained for NIST LRE2005 primary condition (30 sec. segments) throughout this section. Tables 7.2, 7.1 and 7.3, however, present results also for the other two conditions (10 and 3 sec. segments).

For comparison of GMM system trained using MMI (256 mixture components) and GMM trained under ML framework (2048 components) see the first two rows of table 7.2. MMI training provided more than 50% relative improvement compared to ML training, which confirmed our previous good results and superiority of discriminatively trained models over ML-trained ones [14].

The following lines of table 7.2 show the effect of different feature extraction techniques. Since MMI training is very time-consuming process, all these results are reported only for ML-trained system. A significant degradation (7% relative) can be seen for the system without VTLN and even more prominent degradation (27% relative) is caused by further omitting the RASTA processing. Our features are 7 MFCC coefficients concatenated with 49 SDC coefficients. The usual practice is to use only SDC coefficients alone. Doing so, however, leads again to

System EER [%]						Duration		
						30s	10s	3s
MMI	MFCC		SDC	VTLN	RASTA	4.6	8.6	17.2
ML	MFCC		SDC	VTLN	RASTA	10.8	13.9	21.0
ML	MFCC		SDC		RASTA	11.6	14.5	21.8
ML	MFCC		SDC			14.8	17.8	24.0
ML			SDC	VTLN	RASTA	14.0	17.8	25.0
ML	MFCC	no c_0	SDC	VTLN	RASTA	10.9	14.4	21.2
ML	MFCC		$\Delta, \Delta\Delta$	VTLN	RASTA	18.2	20.0	26.2

Table 7.2: EER for different GMM systems for NIST LRE 2005.

System EER [%]	Duration		
	30s	10s	3s
PRLM (string)	6.8	13.9	24.5
PRLM+lattice	5.7	10.7	21.2
PRLM+lattice+anti.m.	5.3	10.7	21.4
GMM-MMI 256	4.6	8.6	17.2
Fusion	**2.9**	**6.4**	**14.1**
MIT fuse posteval [13]	**4.0**	**9.5**	**20.7**

Table 7.3: EER for PRLM system for NIST LRE 2005 – post-evaluation.

significant degradation in performance (23% relative). For the purpose of the language or speaker recognition, the zero'th MFCC coefficient (c_0), which is carrying information about short term signal energy, is often discarded. I have obtained slightly worse results without c_0 coefficient, however, according to our experience, discarding c_0 coefficient would be important without RASTA processing. The last line in table 7.2 shows the results for features that are 13 MFCC augmented with their delta and delta-delta coefficients – the features commonly used for speech recognition. The performance with these features is far behind those with any of SDC based features, which confirms the superiority of SDC for language recognition task.

Table 7.3 shows the results of post-evaluation work. I analyzed the results of PRLM system and found two important problems. First, I found a bug in the implementation of language model back-offing. Fixing this bug resulted in decrease of EER from 9.8% to 7.7% on 30 sec. segments. The second problem was the segmentation of test data. For generation of lattices, I segmented test utterances to chunks smaller than the original segments. I tuned the length of chunks for optimal performance on Indian accented English as this was the only example of LRE 2005 data that we had available. This was clearly wrong decision. On chunk boundaries, we lost information about phoneme context for estimation of trigram statistics. Without segmentation of test utterances, the systems performs with EER=5.7%. For comparison, Table 7.3 contains also results for PRLM based on strings (language model are trained and test trigram statistics are estimated only using the best path through the lattice). The anti-models improved the final lattice based PRLM, however the gain I obtained was smaller compared to our previous work. It seems that anti-models mostly corrected our second problem in the submitted system. Since NIST 2003 test data match well to the training (CallFriend) data and do not contain long silences, I was not able to identify these problems before the evaluation.

The performance of the fused system making use of the corrected PPRLM system and original GMM-MMI system is EER=2.9%. The weights for fusion were tuned again on our development set.

7.5 Conclusion

Compared to other systems in the NIST 2005 evaluations, our system performed well. Especially the discriminative training brought substantial improvement over the standard ML scheme. Good performance of phonotactic approach on LRE 2003 [63] did not fully generalize during the evaluation, but helped us to find out some problems in PRLM approach.

Chapter 8

NIST Language Recognition Evaluation 2007

In NIST 2007, Speech@FIT was already considered as on the sites that has something to say in world's LRE business, so our task in NIST LRE 2007[1] was to confirm our results from 2005 and come up with new ideas that would increase the accuracy of the system but also be interesting from the scientific point of view.

The efforts put in these evaluations corresponded to this situation, there were 14 systems fused in the BUT submission. My task here was again to coordinate the whole efforts in the evaluations, which was a truly Speech@FIT-wide effort. Ondřej Glembek was responsible contributed to the binary decision trees and factor analysis therein, Tomáš Mikolov helped with adaptation of classical language models and Petr Schwarz and Mišo Fapšo worked on the SVM-based approaches. Valja Hubeika was working on channel compensation in acoustic systems and Oldřich Plchot prepared Thai telephone data derived from broadcasts, although they did not make it to our NIST submission. Finally, the "guru" Lukáš Burget was on the discriminative training in acoustic systems. Besides the coordination, my work was in the pre-processing steps (segmentation, feature extraction, generating super-vectors) and the complete phonotactic work (with contributions of Ondřej and Tomáš). After the usual description of training, development and evaluation data, the chapter contains a complete description of all submitted systems. For eigen-channels, binary decision trees, multi-models and factor analysis, this chapter includes also the necessary theoretical background[2]. On contrary to NIST 2005, we have put significant efforts into the calibration of scores which is essential for good performance of the sub-systems and for their good fusion.

8.1 The Data

8.1.1 Training data

The following training data (distributed by LDC and ELRA) were used to train our systems (see Table 8.1 for more details):

[1] NIST 2007 Language Recognition Evaluation, `http://www.nist.gov/speech/tests/lang/2007`

[2] Portions of text adopted with kind permission from other Speech@FIT members and Jiří Navrátil are marked wherever appropriate.

CF	CallFriend
CH	CallHome
F	Fisher English Part 1.and 2.
F	Fisher Levantine Arabic
F	HKUST Mandarin
SRE	Mixer (data from NIST SRE 2004,2005,2006)
LDC07	development data for NIST LRE 2007
OGI	OGI-multilingual
OGI22	OGI 22 languages
FAE	Foreign Accented English
SpDat	SpeechDat-East[3]
SB	SwitchBoard

Table 8.1: Training data in hours for each language and source

	sum	CF	CH	F	SRE	LDC07	OGI	OGI22	Other
Arabic	212	19.5	10.4	175	5.93	1.45		0.33	
Bengali	4.27				2.86	1.42			
Chinese	93.2	41.7	1.64	17.2	44.9	4.2	0.87	0.85	
English	264	39.8	4.68	162	34.9		6.77	0.52	15.6 (FAE)
Hindustani	23.5	19.6			0.64	1.32	1.53	0.42	
Spanish	54.3	43.8	6.71		2.63		1.18	0.38	
Farsi	22.7	21.2			0.03		1.00	0.42	
German	28.2	21.6	5.10				1.12	0.38	
Japanese	23.9	19.1	3.47				0.87	0.35	
Korean	19.7	18.4			0.09		0.72	0.5	
Russian	15.1				3.38	1.33		0.43	10.0 (SpDat)
Tamil	19.6	18.4					0.96	0.26	
Thai	1.45				0.15	1.23			
Vietnamese	21.6	20.6					0.79	0.27	
Other	62.5	20.7					1.10	3.29	37.4 (SpDat)

8.1.2 Development data

The development data for this evaluation were defined by MIT Lincoln Labs[4]. The development data have nominal duration 3, 10 and 30 seconds and were based on segments from previous evaluations plus additional segments extracted from longer files from training databases (which were not included in the training set). The set was splitted to two halves for the development. The 1^{st} part was based on NIST LRE 1996 and 2003 data and additional segments from Fisher, CallHome and Mixer databases. The set contains 5165 trials. The 2^{nd} part was based on NIST LRE 2005 and additional segments from OGI stories and Mixer database. The set contains 5884 trials

[3] see http://www.fee.vutbr.cz/SPEECHDAT-E or the ELRA/ELDA catalog

[4] I would like to thank to MIT Lincoln Labs for preparing the test and dev sets – we were really glad to be able to run our systems using them.

8.1.3 Evaluation data

The number of languages to be detected has been significantly increased since the last evaluation in 2005. There are 14 languages that were used as detection targets in LRE07[5]. These are listed in Table 8.2.

The evaluation set contains test segments with three nominal durations of speech: 3, 10 and 30 seconds. Actual speech durations varied but were constrained to be within the ranges of 2-4 seconds, 7-13 seconds, and 25-35 seconds of actual speech contained in segments, respectively. The silence was not removed from speech so a segment could be much longer. Unlike previous evaluations, the nominal duration for each test segment was not identified. There were more then 7500 segments to identify.

Table 8.2: A list of the target languages and dialects for LRE07

- Arabic
- Bengali
- Chinese
  - Cantonese
  - Mandarin
    - Mainland
    - Taiwan
  - Min
  - Wu

- English
  - American
  - Indian
- Hindustani
  - Hindi
  - Urdu
- Spanish
  - Caribbean
  - non-Caribbean

- Farsi
- German
- Japanese
- Korean
- Russian
- Tamil
- Thai
- Vietnamese

For evaluation metric see Section 2.5.

8.2 System description - Primary system

The system was a fusion of 13 sub-systems: 4 acoustic and 9 phonotactic ones. All sub-system descriptions are completed with a “code” of the system for easy identification. Fro all systems, the pre-processing was done by voice activity detector (VAD) based on our Hungarian phone recognizer with all the phoneme classes linked to ’speech’ class. The silence is not used in acoustic systems.

8.2.1 Acoustic systems

All acoustic systems used the shifted-delta-cepstra (SDC) [9] together with direct MFCC. The feature extraction was the same as in our LRE 2005 system (Chapter 7 or [58]): 7 MFCC coefficients (including coefficient C0) concatenated with SDC 7-1-3-7, which totals in 56 coefficients per frame.

Vocal-tract length normalization (VTLN) was done with the same models and in the same way as in NIST LRE 2005 (see Chapter 7) The warping factors are estimated using single GMM (512 Gaussians), ML-trained on the whole CallFriend database (using all the languages). The model was trained in standard speaker adaptive training (SAT) fashion in four iterations of alternately re-estimating the model parameters and the warping factors for the training data.

[5] NIST 2007 Language Recognition Evaluation Plan: www.nist.gov/speech/tests/lang/2007

GMM system with 2048 Gaussians per language and eigen-channel adaptation `GMM2048-eigchan`

The inspiration come from our GMM system for speaker recognition [104] which follow conventional Universal Background Model-Gaussian Mixture Modeling (UBM-GMM) paradigm [105] and employs number of techniques that has previously proven to improve GMM modeling capability (see Section 6 and [58]) .

Training speaker model and verification

Each model of language is obtained by traditional *relevance MAP* adaptation [106] of UBM using enrollment conversation. Only means are adapted with relevance factor $\tau = 19$.

In verification phase, standard Top-N Expected Log Likelihood Ratio (ELLR) scoring [106] is used to obtain the verification score, where $N = 10$ in our system. However, for each trial, both model of language and UBM are adapted to channel of test conversation using simple eigen-channel adaptation [107] prior to computing the log likelihood ratio score.

Eigen-channel subspace estimation

We adopted the term ‘eigen-channel’ as used in speaker recognition (SRE) from Kenny [108]. It was introduced to the SRE by Spescom Datavoice (SDV) in 2004 [107], revisited by Kenny and Vogt [109] in NIST SRE 2005, and again by several sites in various forms in NIST SRE 2006.

Let *supervector* be a MD dimensional vector constructed by concatenating all GMM mean vectors and *normalized by corresponding standard deviations.* M is number if Gaussian mixture components in GMM and D is dimensionality of features. Before eigen-channel adaptation can be applied, we must identify directions in which *supervector* is mostly affected by changing channel. These directions (eigen-channels), are defined by columns of $MD \times R$ matrix $\mathbf{V}$, where R is the chosen number of eigen-channels ($R = 50$ in our system). The matrix $\mathbf{V}$ is given by R eigen-vectors of average within class covariance matrix, where each class is represented by supervectors estimated on different segments spoken in the same language.

For each language, L, and all its conversations, $j = 1, \ldots, J_L$, UBM is adapted to obtain a supervector, $\mathbf{s}_{Lj}$. The corresponding average language supervector given by

$$\bar{\mathbf{s}}_L = \sum_{j=1}^{J_L} \mathbf{s}_{Lj} / J_L \tag{8.1}$$

is subtracted from each supervector, $\mathbf{s}_{Lj}$, and the resulting vectors form columns of $MD \times J$ matrix $\mathbf{S}$, where J is the number of all conversations from all languages. Eigen-channels (columns of matrix $\mathbf{V}$) are given by R eigen-vectors of $MD \times MD$ matrix $\mathbf{S}\mathbf{S}^T$ corresponding to R largest eigenvalues. Unfortunately, for our system, where $MD = 2048 \times 56 = 114688$, direct computation of these eigen-vectors is unfeasible. A possible solution is to compute eigen-vectors, $\mathbf{V}'$, of $J \times J$ matrix $\mathbf{S}^T\mathbf{S}$, eigen-channels are then given by $\mathbf{V} = \mathbf{S}\mathbf{V}'$. In case the MAP criterion is used for eigen-channel adaptation (see below), the length of each eigen-channel must be also normalized to the standard deviation of vectors forming columns of $\mathbf{S}$ along the direction of the eigen-channel:

1. scaling each eigen-channel vector to unity length,

2. estimating sample standard deviation from all columns of $\mathbf{S}$ projected to the eigen-channel,

3. scaling the eigen-channel by the standard deviation.

This normalization is irrelevant in the case of ML criterion.

Eigen-channel adaptation

Once the eigen-channels are identified, a model of language (or UBM) can be adapted to the channel a test conversation by shifting its supervector in the directions given by eigen-channels to better fit the test conversation data. Mathematically, this can be expressed as finding the *channel factors*, $\mathbf{x}$, that maximize the following MAP criterion:

$$p(\mathbf{O}|\mathbf{s}+\mathbf{V}\mathbf{x})\mathcal{N}(\mathbf{x};\mathbf{0},\mathbf{I}), \tag{8.2}$$

where $\mathbf{s}$ is supervector representing the model to be adapted[6], $p(\mathbf{O}|\mathbf{s}+\mathbf{V}\mathbf{x})$ is likelihood of the test conversation given the adapted supervector (model) and $\mathcal{N}(\cdot;\mathbf{0},\mathbf{I})$ denotes normally distributed vector. Assuming fixed occupation of Gaussian mixture components by test conversation frames, $\mathbf{o}_t, t=1,\ldots,T$, it can be shown [107] that $\mathbf{x}$ maximizing criterion (Equation 8.2) is given by:

$$\mathbf{x} = \mathbf{A}^{-1}\sum_{m=1}^{M}\mathbf{V}_m^T\sum_{t=1}^{T}\gamma_m(t)\frac{\mathbf{o}_t-\boldsymbol{\mu}_m}{\boldsymbol{\sigma}_m}, \tag{8.3}$$

where $\mathbf{V}_m$ is $M \times R$ part of matrix $\mathbf{V}$ corresponding to m^{th} mixture component, $\gamma_m(t)$ is the probability of occupation mixture component m at time t, $\boldsymbol{\mu}_m$ and $\boldsymbol{\sigma}_m$ are the mixture component's mean and standard deviation vectors and

$$\mathbf{A} = \mathbf{I} + \sum_{m=1}^{M}\mathbf{V}_m^T\mathbf{V}_m\sum_{t=1}^{T}\gamma_i(t). \tag{8.4}$$

In our implementation, occupation probabilities, $\gamma_m(t)$, are computed using UBM and assumed to be fixed for given test conversation. This allows to pre-compute matrix $\mathbf{A}^{-1}$ only once for each test conversation. For each frame, only Top-N occupation probabilities are assumed not to be zero. In the following ELLR scoring, also only the same top-N mixture components are considered. All these facts ensure that adapting and scoring T-norm models on a test conversation can be performed very efficiently.

Eigen-channel adaptation can be also performed by maximizing ML criterion instead of MAP criterion. This corresponds to dropping the prior term, $\mathcal{N}(\mathbf{x};\mathbf{0},\mathbf{I})$, in criterion (Equation 8.2) and term $\mathbf{I}$ in Equation 8.4. In our experiments, there is almost always enough adaptation data making the prior term in MAP criterion negligible. Therefore, we have not found any difference in performance when using the two criteria.

This system uses very simple scheme of modeling channel variability that affects only the verification phase. However, more sophisticated schemes can be considered [110, 108].

GMM-MMI: `GMM256-MMI`

This subsystem uses GMM models with 256 Gaussians per language, where mean and variance parameters are re-estimated using Maximum Mutual Information criterion - the same as for LRE2005 (see Section 6.5.4 and Chapter 7 or [58]).

[6] Note again that by our definition, "supervector" is mean supervector normalized by the corresponding standard deviations.

GMM-MMI with channel compensated features: `GMM256-MMI-chcf`

Similar set of GMM models with 256 Gaussians per language are trained with Maximum Mutual Information criterion. However, the features are first compensated using eigen-channel adaptation in feature domain [111, 66].

The adaptation of feature vector $\mathbf{o}'_t$ is obtained by subtracting either channel compensation offset value (in case of using only one best Gaussian) or a weighted sum of compensation offset values from the original feature vector $\mathbf{o}_t$ according to:

$$\mathbf{o}'_t = \mathbf{o}_t - \sum_{m=1}^{N} \gamma_m(t)\mathbf{V}_m\mathbf{x} \tag{8.5}$$

where $\mathbf{V}_m$ and $\mathbf{x}$ are estimated in the same way as for the adaptation in the model domain. N is the number of Gaussians used for the compensation. In our experiments, $N = 1$, which represent adaptation according the one best Gaussian, was the best.

With these features, we can proceed with standard way. Starting from target models of languages with means MAP adapted from UBM using the compensated features, only mean parameters are further re-estimated using MMI criterion.

SVM on GMM super-vectors: `GMM512-SVM`

In this type of system, GMM supervectors, similar as described in the previous section, are extracted not only from target-model training speech segments, but also for all other background and test speech segments. In other words, each speech segment is represented by a single GMM supervector. The target and background supervectors are then used to train support vector machine (SVM) models of language against which the test supervectors are scored [112, 113]. The SVM uses a linear kernel in supervector space. Each SVM is trained using the all available positive examples from the target language, and many negative examples from a other languages.

8.2.2 Phonotactic systems

The phonotactic systems were based on 3 phone recognizers: two left-context/right-context hybrids and one based on GMM/HMM context dependent models.

Hybrid phone recognizers

The phone recognizer is based on hybrid ANN/HMM approach, where artificial neural networks (ANN) are used to estimate posterior probabilities of phonemes from Mel filter bank log energies using the context of 310ms around the current frame. Hybrid recognizers were trained for Hungarian and Russian on the SpeechDat-E databases. For more details see Chapter 3 or [70, 75].

GMM/HMM phone recognizers

The third phone recognizer was based on GMM/HMM context-dependent state-clustered triphone models, which are trained in similar way as the models used in AMI/AMIDA LVCSR [114]. The models were trained on 2000 hours of English telephone conversational speech data from Fisher, Switchboard and CallHome databases. The features are 13 PLP coefficients augmented with their first, second and third derivatives projected into 39 dimensional space using HLDA transformation. The models are trained discriminatively using MPE criterion [72]. VTLN and

Table 8.3: Phonotactic systems for LRE 2007

type	recognizer	HU [LCRC]	RU [LCRC]	EN [GMM/HMM]
string	4-grams	HU_strLM		EN_strLM
lattice counts	2-grams LM	HU_LM-MMI		
	3-grams LM	HU_LM	RU_LM	
	decision trees	HU_TREE_A3E7M5S3G2_FA	RU_Tree	EN_Tree
	SVM	HU_SVM-3gram-counts		

MLLR adaptation is used for both training and recognition in SAT fashion. The triphones were used for phoneme recognition with a bi-gram phonotactic model trained on English-only data.

Phonotactic Modeling

All the recognizers were able to produce phoneme strings as well as phoneme lattices. In case of lattices, posterior-weighted counts (“soft-counts”) were used (see Section 5.4).

The modeling in the individual phonotactic subsystems is outlined in Table 8.3 and in the following paragraphs.

4-gram language model based on strings: `HU_strLM, EN_strLM`

These systems use 4-gram model estimated on phoneme strings from the Hungarian LCRC and English GMM/HMM phone recognizers. In the case of Hungarian phone recognizer, the LM for each language was derived by interpolating several LMs. At first, we estimated individual LMs using each data source and each language (separate LM for Arabic Fisher, Arabic CF,....). Then we interpolated these separate LMs with LMs estimated on all data from the target language. In the case of English phone recognizer, the final target language LMs were interpolated with single LM trained on all languages together. This was helpful because of the limited amount of data to train LMs (at most 3 hours per language). The interpolation weights were tuned to give minimal perplexity on the development set. Witten-Bell smoothing was used. Pruning using minimal count was applied. The thresholds 2,3 and 8 for bi-, tri- and four-grams respectively were found to perform well on the development data.

3-gram language model on lattice counts: `HU_LM, RU_LM`

The phonotactic models were based on soft-counts (see Section 5.4), but they were adapted from ”UBM” trained on all data in the same way as described for decision tree based phonotactic models [115] (see next paragraph).

Binary decision trees on lattice counts: `HU_TREE_A3E7M5S3G2_FA, RU_Tree and EN_Tree`

In all our systems, binary decision tree language modeling was based on creating a single language independent tree (referenced as “UBM”) and adapting its distributions to individual language training data, as described in Navrátil’s work [59, 115]. While the sub-systems built on Russian LCRC and English GMM/HMM phoneme recognizers use this basic approach only, the Hungarian output was processed in a more complex way using Multi-models and applying Factor Analysis (FA).

Binary decision trees (BT)[7]

Let $\mathcal{A} = \{\alpha_1, ..., \alpha_K\}$ be the phonetic vocabulary, and let $A = a_1, ..., a_t..., a_T$, $a \in \mathcal{A}$ be a sequence of phonemes corresponding to an utterance of length T.

The binary tree (BT) language models belong to a class of approaches aiming at reducing the model complexity via context *clustering.* Here, the probability of a current observation a_t (token) is conditioned on a *set* of token histories (a "cluster of histories"). Since the clusters may include histories of arbitrary lengths, information from longer contexts can be modeled while the model complexity is only determined by the number of such clusters and may be chosen appropriately. Obviously, a sensible choice of the clustering function is essential. The application of BTs for this purpose proved effective in [59, 115, 116].

Consider a sufficiently large training set $A = \{a_1, ..., a_T\}$ representing the decoded speech and define the distribution $Y_A = \{p(\alpha_j|A)\}_{1 \leq j \leq K}$ with the proportions of symbols $\alpha_j \in \mathcal{A}$ observed in A. The essential idea in the BT building process is to find two disjoint subsets $A_1 \cup A_2 = A$ using which two descendant leafs are created. To assess the goodness of such split, the entropy for a distribution of a symbol set $\mathcal{A}$ at leaf l is defined as

$$H_l = -\sum_{s_i \in A} P_l(s_i) \log_2 P_l(s_i) \tag{8.6}$$

the average BT prediction entropy is then

$$\bar{H} = \sum_t P_l \cdot H_l \tag{8.7}$$

with P_l denoting the prior probability of visiting the leaf l, and $P_l(s)$ the probability of observing a symbol s at that leaf. Equation 8.7 is to be minimized in the course of building the BT model. During this process the probabilities $P_l(s_i)$ and P_l are not known and have to be replaced by estimates, $\hat{P}_l(si)$ and $\hat{P}_l$, obtained from a training sample at $a_t, ..., a_T$. Assuming a BT model structure with certain parameters leads to partitioning of the training data into L leaves, each containing a data partition α_l , then the sample distribution estimates are calculated from counts as follows:

$$\hat{P}_l(si) = \frac{\#(s_i|\alpha_l)}{|\alpha_l|} \tag{8.8}$$

$$\hat{P}_l = \frac{|\alpha_l|}{\sum_{l=1}^{L} |\alpha_l|} \tag{8.9}$$

with $\#(s_i|\alpha_l)$ being the a_i count at leaf l, and $|\alpha_l|$ the total symbol count at l.

The remaining problem of finding an optimum tree structure and the corresponding node questions is solved by applying a greedy search algorithm at each node, combined with a recursive procedure for creating tree nodes. To limit the otherwise extensive search space, we restrict the binary questions to be elementary expressions involving a single predictor, rather than allowing for composite expressions. Stated in principal steps, the tree building algorithm is as follows [59, 117]:

1. Let c be the current node of the tree. Initially c is the root.

[7] This section is copied with kind permission of J. Navrátil from [115]

2. For each predictor variable X_i $(i = 1, ..., N)$ find the subset S_i^c, i.e. the binary question, which minimizes the average conditional entropy of the symbol distribution Y at node c. At the same time, it should be provided that each of the subsets satisfies some minimum data criterion. If such question cannot be found, training in this leaf is stopped.
3. Determine which of the N questions derived in Step 2 leads to the lowest entropy.
4. Compute the reduction in entropy. If this reduction is above some significance threshold , store question, create two descendant nodes, c_1 and c_2, pass the data corresponding to the conditions $X_k \in S_k^c$ and $X_k \notin S_k^c$, and repeat Steps 2-4 for each of the new nodes separately.

In [115], Navrátil shows that minimizing the overall average entropy is equivalent to maximizing training data likelihood. Furthermore, Step 3 also maximizes the mutual information between the split distribution and the node question.

The task of finding the best subset in Step 2 is the main source of computational complexity. Exhaustive search (going through all combinations) would involve 2^{K-1} entropy evaluations and thus is unsuitable for our task, where usually $K > 40$. Bahl et al. [117] described an iterative greedy search algorithm, which we have adopted in our work. Further speedup was achieved by adopting approaches used in LVCSR [118].

Increase of speed while building the tree was achieved by using sufficient statistics in form of N-gram counts instead of the sequences of phonemes. This way, exploitation of lattices was feasible.

Leaf adaptation

When little data is available for building the (sub-)tree, an adaptation scheme has been proposed [116]. A UBM is built on separate training set, where large amount of data is available. When adapting a new tree, the UBM structure is copied and the leaf distributions are estimated in the maximum a-posteriori (MAP) framework.

So far, the N-gram LM's have been estimated on training data only, using the maximum likelihood criterion. However, the BT adaptation scheme can be easily adopted to the N-gram LMs:

For each leaf (cluster, N-gram history) l and symbol s, compute the new conditional probability $\hat{P}'(s|l)$ as follows:

$$\hat{P}'(s|l) = \left[b_{s,l} \frac{\#(s|l)}{|l|} + (1 - b_{s,l}) \hat{P}(s|l) \right] / D \tag{8.10}$$

where

$$b_{s,l} = \frac{\#(s|l)}{\#(s|l) + r} \tag{8.11}$$

where $\hat{P}(s|l)$ is the original UBM probability of symbol s given leaf l, $\#(s|l)$ stands for counts of symbols s in leaf l, D normalizes the values to probabilities, and r is an empirical value controlling the strength of the update.

Smoothing of BT

When traversing the BT, each node splits the data into two subsets, causing data sparsity. Each node in the BT can hold the phoneme distribution before the split. Navrátil [116] proposes a smoothing technique, where the smoothed probability of a symbol s in node l $\hat{P}'_{sm}(s|l)$ is given as:

$$\hat{P}'_{sm}(s|l) = b_{s,l}\hat{P}(s|l) + (1 - b_{s,l})\hat{P}_{sm}(s|parent(l)) \tag{8.12}$$

where $b_{s,l}$ is as in Equation 8.11, with r being the smoothing factor. The equation is applied recursively until $parent(l) = root$, when $\hat{P}_{sm}(s|root) = \hat{P}(s|root)$

In the case of BT's, this approach is beneficial, having the smoothing constant r set to 2. However, when applied to N-gram LM's, the performance generally degrades.

Multi-models

Inspired by the principle of anchor models [119], we model language classes by a combination of "other" language models. Instead of merging all resources (databases) of one language together for UBM adaptation, those resources with large amount of data were "hand-clustered", and a single LM was created for each of these clusters (e.g. 7 LMs for English). Such hand-clustering reflected some specifics such as foreign-accented English, different dialects, etc. A linear backend is used to post-process these individual outputs to come up with one score per language.

Multi-models address the question of tweaking the weights for multiple training sources. The application of LDA backend is nothing but a linear combination of multiple model scores, which is equivalent to giving different weights to different data sources in the 1-model-per-language situation.

Factor analysis (FA)[8]

Inspired by the inter-session variation techniques in acoustic language recognition [120, 104], we have adopted the framework to the phonotactic LR. With the leaf log-likelihoods defining the model parameter space, we search for their subspace which best describes the inter-session variability. In the testing phase, we then let the model adapt to the test utterance in this subspace.

Let us denote the concatenation of leaf log-probabilities as a column super-vector $\mathbf{d}$, and let $\mathbf{n}_a$ be the concatenation of clustered N-gram statistics of the inspected utterance a. The standard way to evaluate the tree score for utterance a is to compute the inner product of $\mathbf{d}$ and $\mathbf{n}_a$:

$$S_a = \mathbf{n}_a^{\mathrm{T}}\mathbf{d} \tag{8.13}$$

In FA, we define a transform matrix $\mathbf{V}$ with the same number of rows as is the dimensionality of $\mathbf{d}$ and the number of columns corresponding to the desired number of vectors of factors. The vectors of factors are weighted by column vector $\mathbf{x}$, and then added to the original model parameter supervector $\mathbf{d}$. The FA objective function and output score for utterance a are computed as:

$$S_a = \sum_i n_a^i \log \frac{e^{l_a^i}}{\sum_{j \in cluster(i)} e^{l_a^j}}, \tag{8.14}$$

with

$$l_a^i = d^i + \mathbf{v}^i\mathbf{x}_a \tag{8.15}$$

where d^i is the ith element of $\mathbf{d}$, $\mathbf{v}^i$ is i-th row of matrix $\mathbf{V}$, $cluster(i)$ corresponds to the leaf to which i belongs, and $\mathbf{x}_a$ is a column vector of weights estimated for each utterance a.

[8] This section was adopted from our ICSLP 2008 paper with kind permission of co-authors.

Matrix $\mathbf{V}$ and vector $\mathbf{x}_a$ are estimated numerically using the R-prop algorithm [121] as an alternative to the traditional gradient descend method, where gradient sign is used instead of the gradient value.

We begin by estimating the vector $\mathbf{x}_a$. The gradient is computed as:

$$\frac{\partial}{\partial \mathbf{x}_a} S_a = \sum_i n_a^i \mathbf{v}^i - \sum_i n_a^i \frac{\sum_j \mathbf{v}^j e^{l_a^i}}{\sum_j e^{l_a^i}} \tag{8.16}$$

$\mathbf{V}$ is estimated by maximizing Equation 8.14 summed over a selected sub-set of training utterances A balanced among languages. The gradient for each row i of matrix $\mathbf{V}$ is computed as:

$$\frac{\partial}{\partial \mathbf{v}^i} S = \sum_{a \in \mathrm{A}} n_a^i \mathbf{x}_a^{\mathrm{T}} \left[1 - \frac{e^{l_a^i}}{\sum_j e^{l_a^i}} \right] \tag{8.17}$$

The scheme for one training iteration of matrix $\mathbf{V}$ comprised 10 sub-iterations of estimating the vector $\mathbf{x}_a$ for each utterance a from training set A, followed by 10 sub-iterations of re-estimating matrix $\mathbf{V}$. After 5 iterations, the recognition performance converged.

When scoring a test utterance $\hat{a}$, vector $\mathbf{x}$ was estimated in 10 iterations, and the score was obtained using Equation 8.14.

The critical issue was the initialization of $\mathbf{V}$. We observed that simple principal component analysis framework, as in [104], gave the best convergence speed. The weights $\mathbf{x}$ are initialized by a zeros vector.

Empirically, the number of factors for the evaluation was set 4.

3-gram lattice counts as super-vectors to SVM: `HU_SVM-3gram counts`

In this subsystem, the trigram-lattice-counts from Hungarian phoneme recognizer were used as a supervector for subsequent classification by SVMs, similarly as in MIT's work [65].

8.2.3 Normalization and Calibration

All systems were first processed by linear backend and then fused (or calibrated) using Multi-class Linear Logistic Regression (The detailed description is out of the scope of this theses and can be find in [122]). Both linear backend and fusion parameters were trained using our development data. The FoCal Multi-class toolkit by Niko Brummer[9] was used for the pre-processing and fusion. There are three schemes we compare:

1. Linear backend (LDA)
   - LDA followed Gaussian model with tied diagonal covariance matrix
   - or Gaussian model with tied full covariance matrices
   - No constraint on size of input vector
2. Linear Logistic Regression (LLR)
   - Parameters to estimate are single multiplicative constant and bias vector
   - Trained discriminatively (the same objective function as MMI)

[9] http://niko.brummer.googlepages.com/focalmulticlass

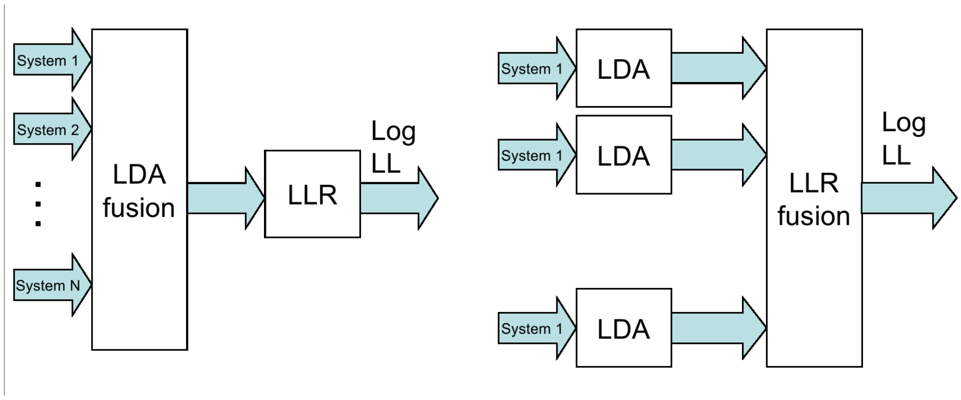

Figure 8.1: Types of fusion

3. LDA followed by LLR

For the fusion, there are two possible solutions (see Figure 8.1):

- LDA fusion with LLR calibration
- LDA calibration on separate systems and LLR fusion.

8.3 Post-evaluation Experiments

The first attention is given to the calibration and fusion, since it has the most impact to the post-evaluation analysis. All results in this section are reported in 100x C_{avg} (see Section 2.5).

8.3.1 Calibration and fusion

The effect of calibration was studied on one acoustic and one phonotactic system, the best ones from each category:

- GMM with 2048 Gaussian mixture model trained using MMI criterion on channel compensated features
- Binary decision trees on posterior weighted posterior counts from English HMM phone recognizer.

With proper calibration it is possible to reduce error by 60% for 30 second condition (see Figure 8.2 and Table 8.4).

The primary system was fusion of all subsystems we built for this evaluation. Detailed report of the results of all subsystems are in Table 8.6. We used LLR fusion which was trained on join set of 10 and 30 seconds from Dev set. The LDA fusion produced approximately the same results (see Table 8.5). The 30 seconds condition scored better but 10 and 3 seconds conditions were worse for the LDA fusion. We tried to do the duration dependent fusion, which we did for 2005 evaluation, but at that time we used different fusion strategy and over-trained it. Duration dependent fusion worked better for this evaluation. There are three data sets on which we can train fusion: Dev, Test and Dev+Test. Test data are closer to the evaluation data then the Dev

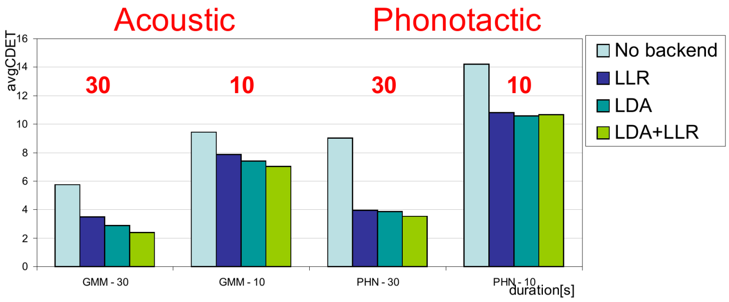

Figure 8.2: Effect of calibration for one acoustic and one phonotactic system on LRE 2007 data

Table 8.4: Effect of calibration for one acoustic and one phonotactic system on LRE 2007 data

	GMM2048-MMI-chcf			EN_Tree		
	30	10	3	30	10	3
No backend	5.75	9.45	18.44	9.02	14.21	24.37
LLR	3.49	7.90	17.65	3.96	10.83	22.97
LDA	2.88	7.42	16.94	3.85	10.55	22.58
LDA+LLR	2.41	7.02	16.90	3.54	10.69	22.66

data, therefore the results are better for this set for LLR fusion. But if we put Dev and Test sets together, which results in 2 times more training examples, both LLR and LDA fusion are better and produce almost the same results. The rest of the results in this section are calibrated on Dev+Test set.

8.3.2 Acoustic systems

Eigen-channel Adaptation

We need to estimate directions with large inter-session variability according to Section 8.2.1.

The eigen-channel adaptation is performed on the models with the eigen-channels derived in the following way:

1. UBM is trained using the original features.
2. for each utterance, a new GMM is obtained by MAP adaptation.

Table 8.5: Different fusions of primary systems on LRE 2007 data

	LLR fusion			LDA fusion		
calibrated on	30	10	3	30	10	3
Dev30+Dev10 - Primary system	2.01	4.74	14.20	1.71	5.21	16.39
Dev - duration dependent	1.94	4.87	13.84	2.02	5.58	16.06
Test - duration dependent	1.61	4.61	14.24	2.38	5.32	18.73
Dev+Test - duration dependent	1.41	4.43	12.98	1.31	4.51	14.69

Table 8.6: Performance of individual subsystems on LRE 2007 data

Acoustic	30	10	3
GMM256-MMI	4.15	8.61	18.43
GMM256-MMI-chcf	3.73	9.81	20.98
GMM2048-eigchan	2.76	7.38	17.14
GMM512-SVM	3.80	8.77	20.14

Phonotactic	30	10	3
HU_LM	5.54	11.75	23.54
HU_TREE_A3E7M5S3G2_FA	4.52	10.35	23.66
HU_strLM (4-gram)	6.35	13.86	27.12
HU_LM-MMI (2-gram)	6.85	14.27	26.37
HU_SVM-3gram-counts	5.41	13.26	26.92
RU_LM	6.06	13.04	24.47
RU_Tree	6.31	12.99	24.51
EN_Tree	4.56	12.32	24.54
EN_strLM (4-gram)	5.83	14.62	27.24

3. a super-vector of means normalized by corresponding standard deviations is obtained from each GMM.
4. maximum of 100 super-vectors per each database and language were selected. That means 100 super-vectors from Arabic CallFriend, 100 from Arabic Fisher, 100 from Arabic Mixer . . . . (Thai, Bengali, OGI, OGI22, LDC07 and databases with less than 5 utterances were omitted).
5. mean is subtracted from super-vectors over each database (not over language as one would expect)
6. eigen-channels (i.e. directions in which language models are adapted for each test utterance) are given by eigen vectors of covariance matrix estimated from super-vectors (see Section 8.2.1 or [104]).

The eigen-channels are estimated to reflect session variability within databases, but not across databases (even for databases of one language). We tested eigen-channels also to reflect session variability within languages, but the result was worse.

To speed up evaluation of individual models of languages, only 10-best scoring Gaussians from UBM were considered for eigen-channel adaptation and likelihood computation.

Further we applied the same matrix to compensate features instead of using it in a model domain [111, 66]. By replacing this compensation from model to feature domain we lost a little bit in performance (see Table 8.7 GMM2048-eigchan for compensation in model domain and GMM2048-chcf for feature domain) but we can benefit from retraining models with MMI criterion. The important things are that when retraining with channel compensated features, only means of GMM have to be re-estimated, not variances! Re-training of variances seems to cancel the effect of channel compensation. There is also baseline results with ML trained models for comparison in Table 8.7.

The best results from post-evaluation work comes from merging all in acoustic subsystems to one system, many Gaussians, eigen-channel compensated features and MMI training to produce the best performing acoustic subsystem and also the best performing stand alone subsystem in our post-evaluation work.

Table 8.7: Performance of acoustic subsystems on LRE 2007 data

Submission	30	10	3
GMM256-MMI (15 MMI iterations)	4.15	8.61	18.43
GMM256-MMI-chcf (3 MMI iterations)	3.73	9.81	20.98
GMM2048-eigchan	2.76	7.38	17.14
Post-eval:	30	10	3
GMM2048-ML	8.03	12.89	21.77
GMM2048-chcf	2.94	7.40	17.93
GMM2048-MMI-chcf (3 MMI iterations)	2.41	7.02	16.90

Table 8.8: Different phone recognizers as tokenizers for phonotactic approach on LRE 2007 data

Submission	30	10	3
EN_Tree	4.56	12.32	24.54
HU_Tree	5.58	11.54	23.45
RU_Tree	6.31	12.99	24.51
Post-eval	30	10	3
EN_Tree	3.54	10.68	22.66

SVM-GMM 512

Here, the GMMs with 512 mixture components were MAP adapted from a UBM. Means of all Gaussians are concatenated together and normalized by their standard deviations and square root of their weight. These were used as features for SVM classifier, similarly as in our speaker recognition system [122] and MIT language recognition work [123]. We used libsvm tool for training and testing SVM models [124]. This system is very simple and one of the best stand alone system(see Table 8.6). The system is very complementary to others and fuse very well (see Section 8.3.5).

8.3.3 Phonotactic system

Different phone recognizers

I have shown in Section 5 that performance of the LID system depends on the performance of the phoneme recognizer. MIT showed the same trend for the HMM phoneme recognizer [71]. Upper part of the Table 8.8 shows that the most complex phone recognizer incorporating all kinds of adaptation performs much better. Lower part of the Table 8.8 presents the same system as the upper one with only one difference: amount of training data for training decision trees for all languages. We manage to process only 3 hours of training data for each language before the submission (together 60 hours = 3 hours ×13 languages +20 hours from English). As post evaluation work, we processed the rest (up to 450 hours) of the data to have the results comparable. The GMM English phone recognizer is about 30% relative better then the Hungarian NN based phone recognizer.

Table 8.9: N-gram LM's versus BTs ($100 \times C_{avg}$)

LRE 2007	30	10	3
HU_Tree	5.58	11.54	23.45
HU_strLM (4-gram)	6.35	13.86	27.12
HU_LM Witten Bell (BUT 2005)	5.85	12.63	23.81
HU_LM no smoothing (MIT 2005 and BUT 2005 post-eval)	6.30	12.72	25.06
HU_LM MAP adaptation from UBM	5.54	11.75	23.54

8.3.4 N-gram LM vs. Binary Tree

In our PRLM systems, both the N-gram LMs and BTs were used. In both cases, trigram lattice counts were used. See Table 8.9 for the comparison of these approaches.

The setup for the BT training was setting the minimum data mass criterion to 450, the minimum entropy reduction was set to 0.001, and both the adaptation and smoothing constants r were set to 2 (see [116] for details on these parameters).

Our strategy for scoring the test utterances in the case of N-gram models in the previous years was to choose those N-grams, that appeared in the training data (of any language) certain amount of times to avoid scoring unseen data. Furthermore, the Witten-Bell smoothing was applied (see lines 2 and 3 in Table 8.9). This would however mean, that the set of suitable N-grams for the 2007 evaluations would be very limited as for some languages, very limited data was available. On the other hand, using unseen N-grams would cause severe data sparsity. The adaptation (as described in Section 8.2.2) turned to be a good approach. See Table 8.9 for results. We expected that smoothing the N-grams in the fashion described in Section 8.2.2 would also bring some gain, however no improvement was observed.

Multi-models

We found, that it is beneficial to train multiple models per language, if there is sufficient amount of data for that language. We have chosen those languages, for which large training databases are available. The abbreviation A3E7M5S3G2 denotes the number of models per particular language (e.g. A3 stands for 3 models for Arabic). Better description with exact database division is in Table 8.10. These models could represent different dialects, group of speakers, databases, etc. In our case, the clustering was simply empirical. We end up by producing several scores per language in the scoring phase. Producing final one-score-per-language is again done using LDA backend. Results are presented in Table 8.11.

Factor Analysis - FA

Table 8.12 describes influence of FA (see Section 8.2.2) to the phonotactic system with Binary decision trees. It mainly helps for 30 second condition. We observed little or no improvement in case of 10 and 3 seconds tasks, where little data for model adaptation was available.

The results with multi-models are similar to the ones in Table 8.11, but the system with FA is more complementary to our other acoustic and phonotactic systems.

Table 8.10: Multiple models distribution with abbreviation A3E7M5S3G2

Arabic	CallFriend	Mandarin	Fisher (HKUST)
	Fisher		SRE 2006
	other		CallFriend - mandarin
English	Foreign accented Eng.		CallFriend - Taiwan
	Fisher		other
	CallHome, OGI 22, OGI multilang	Spanish	CallFriend Caribbean
	SRE 2005 - native		CallFriend non-Caribbean
	SRE 2005 - foreign		other
	CallFriend - south	German	CallFriend
	CallFriend - north		other

Table 8.11: Multi-models for binary decision tree on LRE 2007 data (100 × C_{avg})

LRE 2007	30	10	3
HU_Tree	5.58	11.54	23.45
HU_TREE_A3E7M5S3G2	4.54	10.96	23.34

8.3.5 Fusion

To the evaluation, we submitted a fusion of all systems, but we know that all systems are not necessary. Figure 8.3 shows how many systems are important. The saturation is around 5^{th} system, but the most of the gain cams from the first three. Adding more systems than 3 improves results only in short duration tasks. The order of the most contributing systems is:

1. GMM2048-MMI-chcf
2. EN_Tree
3. HU_TREE_A3E7M5S3G2_FA
4. GMM512-SVM
5. RU_LM

Table 8.13 concludes the fusion section with comparing results of submitted system and post-evaluation results. The big improvement comes from the fusion and calibration (F&C) where we used duration-dependent F&C and we trained it on more data (see Section 8.3.1). Further improvement is achieved by merging advantages from all our acoustic systems to single one (see Section 8.3.2) and by training binary decision tries for phonotactic modeling on more data in our best phonotactic system (see Section 8.3.4)

Table 8.12: Binary decision trees with FA (100 × C_{avg})

LRE 2007	30	10	3
HU_Tree	5.58	11.54	23.45
HU_Tree_FA	5.01	11.45	23.83
HU_TREE_A3E7M5S3G2_FA	4.52	10.35	23.66

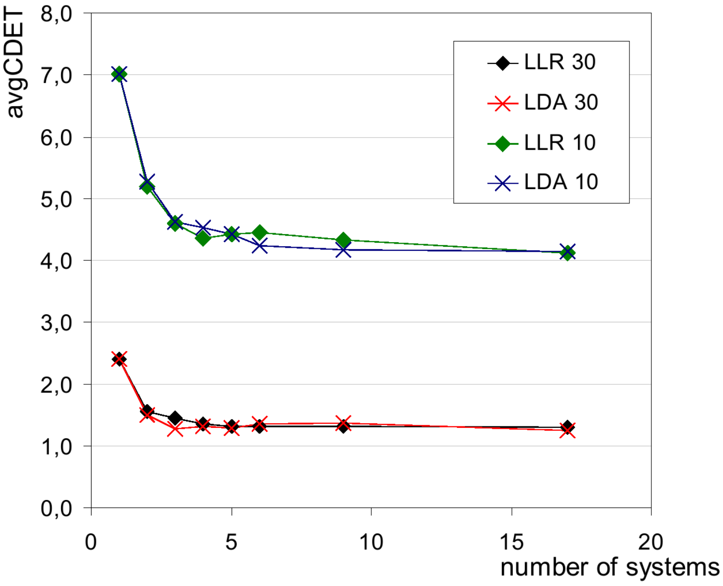

Figure 8.3: Importance of the number of fused systems on LRE 2007 data

Table 8.13: Recapitulation of fused results on LRE 2007 data using LLR multiclass fusion

	100xCavg			Cllr avg			Cllr multiclass		
	30	10	3	30	10	3	30	10	3
Submitted	2.01	4.74	14.20	.075	.184	.761	.284	.663	2.357
New Calibration	1.41	4.43	12.98	.056	.166	.447	.212	.614	1.671
Post-evaluation system	1.30	4.12	12.53	.051	.156	.433	.191	.577	1.615
Best 3 systems (LDA fusion)	1.28	4.63	13.53	.053	.161	.459	.187	.605	1.718

8.4 Dealing with small amount of data for Thai language through public media

Two main scenarios face the issue of the lack of training data. It can be either the situation when we have not enough training data from standard sources or when we do not have any data for particular language at all. In NIST LRE 2007 evaluation we can simulate the first scenario with Thai where we have only 1.5 hour of training data (see Table 8.1), and we can easily simulate the second scenario by removing these data.

There is unlimited source of speech data available from the broadcast media. We can acquire data from several sources, each of which has different channel parameters, quality and number of available languages. The quality of different Internet sources varies a lot and it is important to carefully choose them. We have used an archive[10] of Voice of America Internet radio to obtain the Thai data. This particular data of VoA were obtained in MP3 format, bit rate is 24 Kbit/s, sampling rate 22,050 Hz, 16 bit encoding, mono. Original media data we obtained include a big portion of music and speech with the music in the background. We have to deal with this

[10] FTP server 8475.ftp.storage.akadns.net directory /mp3/voa/eap/thai

Table 8.14: Results for acoustic system with and without eigen-channel compensation for dealing with radio data on LRE 2007 - 10 second condition

LRE 2007 - 10 sec.	No channel comp.		Channel comp.	
	NIST	Radio	NIST	Radio
DCF all lang.	12.83	13.66	7.30	7.56
EER all lang.	13.02	13.80	7.41	7.65
Thai DCF	7.81	11.61	3.93	6.05

problem and select only clean speech segments. Also we should deal with the problem of a low speaker variability in the obtained data, for instance as it is common in news programs, which are moderated by a few speakers.

We downloaded about 250 hours of radio data. We decided to select only phone calls, because they match the target CTS data and we can successfully detect them. The phone calls usually contain no music and we suppose they represent conversational speech. We detected 10 hours of telephone calls in the obtained data which is about 4% of the original recordings [125].

Our phone call detector is based on the fact, that a telephone channel acts like a bandpass filter, which passes energy between approximately 400 Hz and 3.4 KHz. On the other hand, regular wideband speech contains significant energy up to around 5 KHz. For the detection, we first re-sample the signal to commonly used 16 kHz. The signal is divided into frames of 512 samples with no overlap and Fast Fourier Transform is done on each frame. To detect boundary between wide-band and telephone speech, we concentrate on the frequency range between 2350 and 4600 Hz. The energy function in this range was used. At first it was normalized to zero mean and unit variance. Then values in the first half and values in the the second half of the function were summed. A ratio between these two sums was compared with a threshold and the decision was made. If the sum from higher frequencies is bigger than the sum from lower frequencies, there is more energy and it is a wide band speech.

Since there were some English calls, in our detected phone calls, the phonotactic LID (from LRE2005 BUT system) was used to detect Thai versus English. Consequently we listened to all samples with low recognition confidence to verify they are not English.

Several experiments were run using the original telephone data provided by NIST and the telephone data acquired from radio. All experiments were done on 10 second segments. Two Thai models were trained on both sets respectively. Other 13 language-dependent GMMs were shared by both systems. The results are presented in Table 8.4 for the Thai language only (Thai DCF row) and for the complete systems containing all 14 languages. The 'NIST' column stands for a system with Thai trained on 1.45 hour of CTS data available from LDC, the 'Radio' denotes systems trained on telephone data obtained from broadcasts.

First, both systems were trained without channel compensation. Then, eigen- channel adaptation was applied, using a matrix containing 50 eigen-channels computed without radio data.

Although the recognizer trained on radio data does not bring as good results as the recognizer trained on CTS data, the results show (in comparison to the average DCF) that in case of language there are no available data for, radio data can be used.

8.5 Conclusions

There are several statements we can make from the results in this section. The first one is about back-ends and fusion. It is necessary to have lots of calibration data as closed as possible to the target data. This is at least as important as having good systems that are calibrated.

In the acoustic system we showed that both eigen-channel adaptation and MMI training are greatly beneficial in language recognition task. It was shown that, the approximation of the standard eigen-channel adaptation – eigen-channel adaptation in feature domain – is almost as accurate as the standard approach. Moreover, it has a great advantage, as it allows to apply MMI parameter re-estimation without modifying the MMI training algorithm, which was proved by the results in this chapter.

In the phonotactic system, I have shown in Section 5 and [18] that the performance of the LID system depends on the performance of the phoneme recognizer. Later, MIT showed the same trend for the HMM phoneme recognizer [71] with improving it by adding different adaptations. Results in this chapter confirm this statement. I have shown, that the data sparsity problem of the N-gram LMs can be solved by using the binary-tree adaptation scheme. Our experiments show, that it is the **adaptation from UBM** that solves the problem, rather than the BT structural context clustering. We proposed the technique of multi-models and we have shown that it is beneficial to split the training data of each class to several subsets, train separate models on these subsets, and have the backend do their linear combination.

Chapter 9

Conclusion

Partial conclusions were presented at the end of each chapter, this chapter therefore summarizes only the most important findings and draws some directions of future work in LRE.

In the **phonotactic approach**, we have confirmed, that the accuracy of the phone recognizer is crucial for good performance of LRE. It is probably better to train only a couple of good phone recognizers on languages from which we have enough data with annotations that we can trust, than to target an extensive amount of languages. The languages on which the phone recognizers are trained do not have to correspond to the target languages the LRE system should detect.

Using lattices instead of 1-best phone recognizer outputs proved to be helpful in all experiments we have performed. On the other hand, the good results obtained with anti-models on 2003 data did not generalize on 2005 and 2007 data: anti-models worked, but the improvements obtained were far beyond expectations. For the work done in 2007, this can be probably explained by proper calibration of the systems, so that the anti-models had less chances to correct erroneous results.

We have shown that the data sparsity problem of the N-gram LMs can be solved by using the binary-tree adaptation scheme. Our experiments however show, that it is the adaptation from UBM that solves the problem, rather than the structural context clustering in trees. We proposed the technique of multi-models and we have shown that it is beneficial to split the training data of each class to several subsets, train separate models on these subsets, and have the backend do their linear combination.

Also, we found that the adaptation from UBM is advantageous both for classical LM and tree-based approaches. We have also presented a concept of factor analysis in PRLM and we have shown its gain in LRE. Note that the notion of adaptation from UBM and of channel normalization have been common in speaker recognition community so that we could take advantage of our group's activities in this domain (see our last NIST speaker recognition system description in [126]). The UBM approach seems very promising in cases where the target languages have very different amounts of training data.

In the **acoustic approach**, we pioneered the use of discriminatively trained models in LRE, which became the state-of-the-art in this domain.

The eigen-channel adaptation, which allows to compensate for intra-language variability (and is also in common use in speaker recognition), was successfully ported to the LRE domain and it is nice to see the links between this compensation in acoustic and phonotactic approaches. It was shown that the approximation of the standard eigenchannel adaptation — eigenchannel adaptation in feature domain — is almost as accurate as the standard approach. Moreover, it has a great advantage, that it allows to apply MMI parameter re-estimation without modifying the MMI training algorithm: we showed that when eigenchannel adaptation is applied in feature domain, further improvement of the result can be achieved by subsequent re-estimating of the parameter of GMM by using MMI training.

Almost as important as the performance of the actual LRE systems is the proper setting of **back-ends and fusion**. It is necessary to have lots of calibration data as closed as possible to the target data. We have investigated into LLR and LDA calibrations and fusions (and their combinations) and we owe great thank to Niko Br umer that has pushed this work forward by his theoretical work and FoCal toolkit.

9.1 Future work

We are aware that, during the work on this thesis, we have significantly pushed forward the state-of-the-art in LRE but there are still many open issues that require theoretical development as well as massive experimental work. To cite just the ones that could be investigated in short- to medium-time:

- First works were done on discriminative training of n-grams and distributions in trees. The results so far were not convincing, but this idea definitely deserves more work as we feel that discriminative training would make the system itself better calibrated with less post-processing required.
- Still in the phonotactic approach, we feel that the n-grams and binary decision trees trained on symbols (either stored in 1-best strings or in lattices) could be replaced by "softer" representation, for example by phone or phone-state posteriors. This would allow us to virtually get rid of the decoder, save space required for lattices. This approach was pioneered by Niko Brümer [127], did not work for him, but we feel a great potential in it. We have investigated it in speaker recognition, where the first results were very optimistic.
- In the acoustic approaches, there is a great potential for factor analysis [120] that is currently the "top" technique in speaker recognition. Similarly as in SRE, we will be able to define several nuisance factors (speaker, channel) that all have to be compensated for.
- The biggest challenge in LRE nowadays is the availability and quality of training data. Resources such as Fisher do not exist for most languages and we have to recur to collection of data from other sources. Initial experiments in acquisition of telephone data from broadcasts showed promising results and we continue work in this direction.

Bibliography

[1] Boner A., *Spracherkennung mit Computer*, AT Verlag, Aarau, Switzerland, 1992.

[2] Y.K. Muthusamy, N. Jain, and R.A. Cole, "Perceptual benchmarks for automatic language identification," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Apr. 1994, vol. 1, pp. 333–336.

[3] D.A Leeuwen, M. Boer, and R. Orr, "A Human Benchmark for the NIST Language Recognition Evaluation 2005," in *IEEE Odyssey: The Speaker and Language Recognition Workshop*, Stellenbosch, South Africa, Jan. 2008.

[4] H.P. Combrinck and E.C. Botha, "Automatic language identification: Performance vs. complexity," in *Proceedings of the 8th Annual South African Workshop on Pattern Recognition*, Rhodes University, Grahamstad, Nov. 1997.

[5] Y.K. Muthusamy, *A Segmental Approach to Automatic Language Identification*, Ph.D. thesis, Center for Spoken Language Understanding, Oregon Graduate Institue of Science and Technology, Portland, Oregon, Oct. 1993.

[6] J.C. Catford, *A Practical Introduction to Phonetics*, Oxford University Press, Great Britain, 1988.

[7] M.A. Zissman, "Overview of current techniques for automatic language identification of speech," in *Proc. IEEE International Workshop on Automatic Speech Recognition and Understanding (ASRU)*, Dec. 1995, pp. 60–62.

[8] Y.K. Muthusamy, E. Bernard, and R.A. Cole, "Automatic language identification: A review/tutorial," *IEEE Trans. Signal Processing*, vol. 11, no. 4, pp. 33–41, 1994.

[9] P.A. Torres-Carrasquillo, E. Singer, M.A. Kohler, R.J. Greene, D.A. Reynolds, and J.R. Deller Jr., "Approaches to language identification using gaussian mixture models and shifted delta cepstral features," in *Proc. International Conferences on Spoken Language Processing (ICSLP)*, Sept. 2002, pp. 89–92.

[10] S. Nakagawa, Y. Ueda, and T. Seino, "Speaker-independent, text-independent language identification by hmm," in *Proc. International Conferences on Spoken Language Processing (ICSLP)*, Oct. 1992, pp. 1011–1014.

[11] M.A. Zissman, "Automatic language identification using gaussian mixture and hidden markov models," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Apr. 1993, pp. 399–402.

[12] E. Singer, P.A. Torres-Carrasquillo, T.P. Gleason, W.M. Campbell, and D.A. Reynolds, “Acoustic,phonetic,and discriminative approaches to automatic language identification,” in *Proc. Eurospeech*, Sept. 2003, pp. 1345–1348.

[13] W. Campbell, T. Gleason, J. Navratil, D. Reynolds, W. Shen, E. Singer, and P. Torres-Carrasquillo, “Advanced language recognition using cepstra and phonotactics: MITLL system performance on the NIST 2005 language recognition evaluation,” in *IEEE Odyssey: The Speaker and Language Recognition Workshop*, San Juan, Puerto Rico, June 2006.

[14] B. Burget, P. Matějka, and J. Černocký, “Discriminative training techniques for acoustic language identification,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Toulouse, France, May 2006, pp. 209–212.

[15] Y.K. et al. Muthusamy, “A comparison of approaches to automatic language identification using telephone speech,” in *Proc. Eurospeech*, Sept. 1993, vol. 2, pp. 1307–1310.

[16] M.A. Zissman and E. Singer, “Automatic language identification of telephone speech messages using phoneme recognition and n-gram modeling,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Apr. 1994, pp. 305–308.

[17] Y. Yan and E. Barnard, “An approach to automatic language identification based on language-dependent phone recognition,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, May 1995, pp. 3511–3514.

[18] P. Matějka, P. Schwarz, J. Černocký, and P. Chytil, “Phonotactic language identification using high quality phoneme recognition,” in *Proc. Eurospeech*, Sept. 2005, pp. 2237–2241.

[19] P.A. Torres-Carrasquillo, D.A. Reynolds, and J.R. Deller Jr., “Language identification using gaussian mixture model tokenization,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Sept. 2002, pp. 1375–1379.

[20] P. Matějka, I. Szoke, P. Schwarz, and J. Černocký, “Automatic language identification using phoneme and automatically derived unit strings,” in *Proc. International Conference on Text, Speech and Dialogue*, Brno, Czech Republic, Sept. 2004, pp. 147–153.

[21] R.C.F. Tucker, M.J. Carey, and E.S. Parris, “Automatic language identification using subword models,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Apr. 1994, pp. 301–304.

[22] L.F. Lamel and J.L. Gauvain, “Language identification using phone-based acoustic likelihoods,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Apr. 1994, pp. 293–296.

[23] S. Mendoza, L. Gillick, Y. Ito, S. Lowe, and M. Newman, “Automatic language identification using large vocabulary continuous speech recognition,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, May 1996, pp. 785–788.

[24] Schultz T., Rogina I., and Weibel A., “LVCSR-based language identification,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Sept. 1996, pp. 781–784, http://www-2.cs.cmu.edu/˜tanja/.

[25] A.F. Martin and M.A. Przybocki, “NIST 2003 Language Recognition Evaluation,” in *Proc. Eurospeech*, Sept. 2003, pp. 1341–1344.

[26] "National Institute of Standard and Technology," http://www.nist.gov.

[27] A. Martin, T. Doddington, M. Ordowski, and M. Przybocki, "The DET curve in assessment of detection task performance," in *Proc. Eurospeech*, 1997, vol. 4, pp. 1895–1898.

[28] "OGI multi language telephone speech," http://www.cslu.ogi.edu/corpora/mlts/.

[29] "Callfriend corpus, telephone speech of 15 different languages or dialects," www.ldc.upenn.edu/Catalog.

[30] P. Schwarz, P. Matějka, and J. Černocký, "Recognition of phoneme strings using TRAP technique," in *Proc. Eurospeech*, Geneva, Switzerland, Sept. 2003, pp. 825–828.

[31] Y.K. Muthusamy, E. Bernard, and R.A. Cole, "Reviewing automatic language identification," *IEEE Trans. Signal Processing*, vol. 11, pp. 33–41, 1994.

[32] R.G. Leonard and G.R. Doddington, "Automatic language identification," Tech. Rep. Technical Report RADC-TR-74-200, Air Force Rome Air Development Center, Aug. 1974.

[33] R.G. Leonard, "Language recognition test and evaluation," Tech. Rep. Technical Report RADC-TR-80-83, Air Force Rome Air Development Center, Mar. 1980.

[34] D. Cimarusti and R.B. Ives, "Development of an automatic identification system of spoken languages: Phase 1," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, May 1982, pp. 1661–1664.

[35] J.T. Foil, "Language identification using noisy speech," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Apr. 1986, pp. 861–864.

[36] F.J. Goodman, A.F. Martin, and R.E. Wohlford, "Improved automatic language identification in noisy speech," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, May 1989, pp. 528–531.

[37] M. Sugiyama, "Automatic language recognition using acoustic features," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, May 1991, pp. 813–816.

[38] A.S. House and E.P. Neuberg, "Toward automatic identification of the languages of an utterance: Priliminary methodological considerations," *Journal of the Acoustical Society of America*, vol. 62(3), pp. 708–713, 1977.

[39] M. Savic, E. Acosta, and S.K. Gupta, "An automatic language identification system," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, May 1991, pp. 817–820.

[40] K.P. Li, "Automatic language identification using syllabic spectral features," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Apr. 1994, pp. 297–300.

[41] T.J. Hazen and V.W. Zue, "Automatic language identification uing segment-based approach," in *Proc. Eurospeech*, Sept. 1993, vol. 1, pp. 1303–1306.

[42] M.A. Zissman, "Language identification using phoneme recognition and phonotactic language modeling," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, May 1995, pp. 3503–3506.

[43] Y. Yan, *Development of An Approach to Language Identification based on Language-dependent Phone Recognition*, Ph.D. thesis, Center for Spoken Language Understanding, Oregon Graduate Institue of Science and Technology, Portland, Oregon, Oct. 1995.

[44] A.K.V.Sai Jayram, V. Ramasubramanian, and T.V. Sreenivas, "Automatic language recognition using acoustic sub-word units," in *Proc. International Conferences on Spoken Language Processing (ICSLP)*, 2002, pp. 81–84.

[45] O. Andersen, P. Dalsgaard, and W. Barry, "On the use of data-driven clustering technique for identification of poly- and mono-phonemes for four european languages," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Apr. 1994, pp. 121–124.

[46] F. Cummins, F. Gers, and J. Schmidhuber, "Language identification from prosody without explicit features," in *Proc. Eurospeech*, Sept. 1999, vol. 1, pp. 371–374.

[47] A.G. Adami and H. Hermansky, "Segmentation of speech for speaker and language recognition," in *Proc. Eurospeech*, Sept. 2003, pp. 841–844.

[48] K. Berkling, D. Reynolds, and M.A. Zissman, "Evaluation of confidence measures for language identification," in *Proc. Eurospeech*, Sept. 1999, vol. 1, pp. 363–366.

[49] F. Metze, T. Kemp, T. Schaaf, and H. Schultz, T.and Soltau, "Confidence measure based language identification," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, June 2000, pp. 1827–1830.

[50] J.M. Hombert and I. Maddieson, "The use of 'rare' segments for language identification," in *Proc. Eurospeech*, Sept. 1999, vol. 1, pp. 379–382.

[51] J. Navrátil and W. Z uhlke, "Double bigram-decoding in phonotactic language identification," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Munich, Germany, Apr. 1997, vol. 1, pp. 1115–1118.

[52] S. Harbeck and U. Ohler, "Multigrams for language identification," in *Proc. Eurospeech*, Sept. 1999, vol. 1, pp. 375–378.

[53] E. Wong and S. Sridharan, "Methods to improve gaussian mixture model based language identification system," in *Proc. International Conferences on Spoken Language Processing (ICSLP)*, 2002, pp. 93–96.

[54] Q. Dan and W. Bingxi, "Discriminative training of GMM for language identification," in *ISCA & IEEE Workshop on Spontaneous Speech Processing and Recognition (SSPR)*, Tokyo,Japan, Apr. 2003, p. MAP8.

[55] W.M. Campbell, P.A. Singer, E. andTorres-Carrasquillo, and D.A. Reynolds, "Language recognition with support vector machines," in *IEEE Odyssey: The Speaker and Language Recognition Workshop*, Toledo, Spain, June 2004, pp. 41–44.

[56] S. Parandekar and K. Kirchhoff, "Multi-stream language identification using data-driven dependency selection," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, apr 2003, pp. 28–32.

[57] J.L. Gauvain, A. Messaoudi, and H Schwenk, "Language recognition using phone lattices," in *Proc. International Conferences on Spoken Language Processing (ICSLP)*, Sept. 2004, pp. 1283–1286.

[58] P. Matějka, L. Burget, P. Schwarz, and J. Černocký, "Brno University of Technology System for NIST 2005 Language Recognition Evaluation," in *IEEE Odyssey: The Speaker and Language Recognition Workshop*, San Juan, Puerto Rico, June 2006, pp. 57–64.

[59] J. Navrátil, "Spoken language recognition–a step toward multilinguality in speech processing," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 9, pp. 678–685, Sept. 2001.

[60] S.A. SantoshKumar and V. Ramasubramanian, "Automatic language identification using ergodic hmm," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Mar. 2005, vol. 1, pp. 609–612.

[61] K. Wong and M. Siu, "Language identification using discrete hidden markov model," in *Proc. International Conferences on Spoken Language Processing (ICSLP)*, Sept. 2004, pp. 794–797.

[62] D. A. Leeuwen and N. Brummer, "Channel-dependent gmm and multi-class logistic regression models for language recognition," in *IEEE Odyssey: The Speaker and Language Recognition Workshop*, San Juan, Puerto Rico, June 2006.

[63] P. Matějka, P. Schwarz, B. Burget, and J. Černocký, "Use of Anti-Models to Further Improve State-of-the-art PRLM Language Recognition System," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Toulouse, France, May 2006, pp. 197–200.

[64] Ch. White, I. Shafran, and J.L. Gauvain, "Discriminative classifiers for language recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Toulouse, France, May 2006, vol. I, pp. 213–216.

[65] W.M. Campbell, F. Richardson, and D.A. Reynolds, "Language recognition with word lattices and support vector machines," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Honolulu, Hawaii, USA, Oct. 2007, vol. 4, pp. 989–992.

[66] F. Castaldo, E. Dalmasso, P. Laface, D. Colibro, and C. Vair, "Language identification using acoustic models and speaker compensated cepstral-time matrices," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Honolulu, Hawaii, USA, Oct. 2007, vol. 4, pp. 1013–1016.

[67] D.A. Leeuwen and N. Brummer, "Building language detectors using small amounts of training data," in *IEEE Odyssey: The Speaker and Language Recognition Workshop*, Stellenbosch, South Africa, Jan. 2008.

[68] D. Farris, C. White, and S. Khudanpur, "Sample selection for automatic language identification," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Las Vegas, USA, May 2008, pp. 4225–4229.

[69] D. Caseiro and I. Trancoso, "Spoken Language Identification using the SpeechDat Coprus," in *Proc. International Conferences on Spoken Language Processing (ICSLP)*, Sydney, Australia, Dec. 1998, pp. 1093–1096.

[70] P. Schwarz, P. Matějka, and J. Černocký, "Hierarchical structures of neural networks for phoneme recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Toulouse, France, May 2006, pp. 325–328.

[71] W. Shen and R. Reynolds, "Improving phonotactic language recognition with acoustic adaptation," in *Proc. International Conferences on Spoken Language Processing (ICSLP)*, Antwerp, Belgium, Aug. 2007, pp. 358–361.

[72] D. Povey, *Discriminative Training for Large Vocabulary Speech Recognition*, Ph.D. thesis, Cambridge University, July 2004.

[73] H. Bourlard and N. Morgan., *Connectionist speech recognition: A hybrid approach.*, Kluwer Academic Publishers, Boston, USA, 1994.

[74] H. Hermansky and S. Sharma, "Temporal patterns (TRAPS) in ASR of noisy speech," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Phoenix, Arizona, Mar. 1999, pp. 2427–2431.

[75] P. Schwarz, P. Matějka, and J. Černocký, "Towards lower error rates in phoneme recognition," in *Proc. International Conference on Text, Speech and Dialogue*, Brno, Czech Republic, Sept. 2004, pp. 465–472.

[76] K. Lee and H. Hon, "Speaker-independent phone recognition using hidden markov models," in *IEEE Trans. Acoust., Speech, Signal Processing*, nov 1989, vol. 37(11), pp. 1641–1648.

[77] J.S. Garofolo, L.F.Lamel, W.M. Fisher, J.G. Fiscus, D.S. Pallett, and N.L. Dahlgren, "DARPA–TIMIT acoustic–phonetic speech corpus," Tech. Rep. NISTIR 4930, U.S. Department of Commerce, National Institute of Standards and Technology, Computer Systems Laboratory, Feb. 1993.

[78] "HTK toolkit," htk.eng.cam.ac.uk.

[79] P. Jain and H. Hermansky, "Beyond a single critical-band in TRAP based ASR," in *Proc. Eurospeech*, Geneva, Switzerland, sep 2003, pp. 437–440.

[80] L. Lamel and J. Gauvaion, "High performance speaker-independent phone recognition using cdhmm," in *Proc. International Conferences on Spoken Language Processing (ICSLP)*, Sept. 1993, pp. 121–124.

[81] J. Ming and F.J. Smith, "Improved phone recognition using bayesian triphone models," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Seattle,WA,USA, May 1998, pp. 409–412.

[82] L. Deng, L. Xiang, Y. Dong, and A. Acero, "A hidden trajectory model with bi-directional target-filtering: Cascades vs. integrated implementaion for phonetic recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Philadelphia,PA,USA, Mar. 2005, pp. 337–340.

[83] J.W. Chang, *Near-Miss Modeling: A Segmental Based Approach to Speech Recognition*, Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, June 1998.

[84] A. Robinson, "An application of recurrent nets to phone probability estimation," *IEEE Transactions on Neural Networks*, vol. 5, no. 3, pp. 298–305, 1994.

[85] A.K. Halberstadt, *Heterogenous Acoustic Measurements and Multiple Classifiers for Speech Recognition*, Ph.D. thesis, Massachusetts Institute of Technology, Nov. 1998.

[86] Y.K. Muthusamy, "The OGI Multi-language Telephone Speech Corpus," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Oct. 1992, pp. 892–897.

[87] H. van den Heuvel et al., "SpeechDat-East: Five multilingual speech databases for voice-operated teleservices completed," in *Proc. Eurospeech*, Aalborg, Denmark, Sept. 2001, vol. 3, pp. 2059–2062.

[88] "SpeechDat-East Project," http://www.fee.vutbr.cz/SPEECHDAT-E.

[89] G.B. Dantzig, *Linear Programming and Extensions*, Princeton University Press, Princeton, NJ, 1963.

[90] P. Matějka, "Tuning phonotactic language identificaion system," Tech. Rep., Brno University of Technology, Department of Computer Graphics and Multimedia, Feb. 2005.

[91] "OGI - Oregon Graduate Institute," http://www.cslu.ogi.edu/.

[92] R. Schluter, W. Macherey, B. Muller, and H. Ney, "Comparison of discriminative training criteria and optimization methods for speech recognition," in *Speech Communication*, 2001, vol. 34, pp. 287–310.

[93] T. Hain, L. Burget, J. Dines, G. Garau, M. Karafiat, M. Lincoln, I. McCowan, D. Moore, V. Wan, R. Ordelman, and S. Renals, "The 2005 AMI system for the transcription of speech in meetings," in *Proc. NIST Rich Transcription 2005 Spring Meeting Recognition Evaluation*, Edinburgh,UK, July 2005.

[94] M.A. Zissman, "Comparison of four approaches to automatic language identification of telephone speech.," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 4, no. 1, pp. 31–44, 1996.

[95] J. Cohen, T. Kamm, and A.G. Andreou, "Vocal tract normalization in speech recognition: Compensating for systematic speaker variability," *J. Acoust. Soc. Am.*, vol. 1, no. 97, pp. 2346, 1995.

[96] S. Young et al., *The HTK Book*, University of Cambridge, 2005.

[97] H. Hermansky and N. Morgan, "RASTA processing of speech," *Trans. on Speech & Audio Processing*, vol. 2, no. 97, pp. 578–589, 1994.

[98] R. Schluter and W. Macherey, "Comparison of discriminative training criteria," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Seattle, WA, USA, May 1998, pp. 493–496.

[99] N. Kumar, *Investigation of Silicon-Auditory Models and Generalization of Linear Discriminant Analysis for Improved Speech Recognition*, Ph.D. thesis, John Hopkins University, Baltimore, 1997.

[100] M.J.F. Gales, "Semi-tied covariance matrices for hidden markov models," *IEEE Trans. Speech and Audio Processing*, vol. 7, pp. 272–281, 1999.

[101] L. Burget, *Complementarity of Speech Recognition Systems and System Combination*, Ph.D. thesis, Brno University of Technology, Czech Republic, 2004.

[102] L. Burget, "Combination of speech features using smoothed heteroscedastic linear discriminant analysis," in *Proc. International Conferences on Spoken Language Processing (ICSLP)*, Jeju Island, Korea, Oct. 2004, pp. 971–974.

[103] A. Stolcke, "SRILM - an extensible language modeling toolkit," in *Proc. International Conferences on Spoken Language Processing (ICSLP)*, Denver, Colorado, Sept. 2002, pp. 901–904.

[104] L. Burget, P. Matejka, P. Schwarz, O. Glembek, and J. Cernocky, "Analysis of feature extraction and channel compensation in GMM speaker recognition system," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 15, no. 7, pp. 1979–1986, Sept. 2007.

[105] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted gaussian mixture models," *Digital Signal Processing*, vol. 10, no. 1–3, pp. 19–41, 2000.

[106] D. A. Reynolds, "Comparison of background normalization methods for text-independent speaker verification," in *Proc. Eurospeech*, Rhodes, Greece, Sept. 1997, pp. 963–966.

[107] Niko Brummer, "Spescom DataVoice NIST 2004 system description," in *Proc. NIST Speaker Recognition Evaluation 2004*, Toledo, Spain, June 2004.

[108] P. Kenny and P. Dumouchel, "Disentangling speaker and channel effects in speaker verification," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Montreal, Canada, May 2004, vol. 1, pp. 47–40.

[109] M. Mason, R. Vogt, B. Baker, and S. Sridharan, "Data-driven clustering for blind feature mapping in speaker verification," in *Proc. Eurospeech*, Lisbon, Portugal, Sept. 2005, pp. 3109–3112.

[110] R. Vogt and S. Sridharan, "Experiments in session variability modelling for speaker verification," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Toulouse, France, May 2006, vol. 1, pp. 897–900.

[111] V. Hubeika, L. Burget, P. Matejka, and J. Cernocky, "Channel compensation for speaker recognition," in *Proc. Joint Workshop on Multimodal Interaction and Related Machine Learning Algorithms*, June 2007.

[112] W. Campbell, D. Sturim, and D. Reynolds, "Support vector machines using GMM supervectors for speaker verification," *IEEE Signal Processing Letters*, vol. 13, no. 5, pp. 308–311, 2006.

[113] W.M. Campbell, "A SVM/HMM system for speaker recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Hong Kong, Apr. 2003, pp. 156–159.

[114] T. Hain, V. Wan, L. Burget, M. Karafiát, J. Dines, J. Vepa, G. Garau, and M. Lincoln, "The AMI System for the Transcription of Speech in Meetings," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Honolulu, Hawaii, USA, Oct. 2007, vol. 4, pp. 357–400.

[115] J. Navratil, "Recent advances in phonotactic language recognition using binary-decision trees," in *Proc. International Conferences on Spoken Language Processing (ICSLP)*, Pittsburgh, PA, USA, Oct. 2006, pp. 421–424.

[116] J. Navratil, Qin Jin, W.D. Andrews, and J.P. Campbell, "Phonetic speaker recognition using maximum-likelihood binary-decision tree models," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Hong Kong, China, Apr. 2003, vol. 4, pp. 796–799.

[117] L.R. Bahl, P.F. Brown, P.V. DeSouza, and R.L. Mercer, "A tree-based statistical language model for natural language speech recognition," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 37, no. 7, pp. 1001–8, July 1989.

[118] Peng Xu, *Random Forests and the Data Sparseness Problem in Language Modeling*, Ph.D. thesis, John Hopkins University, Baltimore, Maryland, 2005.

[119] D.E. Sturim, D.A. Reynolds, E. Singer, and J. P. Campbell, "Speaker indexing in large audio databases using anchor models," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, May 2001, vol. 1, pp. 429–432.

[120] P. Kenny, "Joint factor analysis of speaker and session variability : Theory and algorithms - technical report CRIM-06/08-13. Montreal, CRIM, 2005," Tech. Rep., Centre de recherche informatique de Montréal (CRIM), Motreal, Canada, 2005.

[121] M. Riedmiller and H. Braun, "RPROP – a fast adaptive learning algorithm," 1992.

[122] N. Brümmer, L. Burget, J. Cernocky, O. Glembek, F. Grezl, M. Karafiát, D. van Leeuwen, P. Matejka, P. Schwarz, and A. Strasheim, "Fusion of heterogeneous speaker recognition systems in the STBU submission for the NIST Speaker Recognition Evaluation 2006," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 15, no. 7, pp. 2072–2084, Sept. 2007.

[123] W. Campbell, D.E. Sturim, D.A. Reynolds, and A. Solomonoff, "SVM Based Speaker Verification using a GMM Supervector Kernel and NAP Variability Compensation," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Toulouse, France, May 2006, vol. I, pp. 97–100.

[124] Chih-Chung Chang and Chih-Jen Lin, "Libsvm: a library for support vector machines," http://www.csie.ntu.edu.tw/∼cjlin/libsvm, 2001.

[125] O. Plchot, V. Hubeika, L. Burget, P. Schwarz, and P. Matějka, "Acquisition of telephone data from radio broadcasts with applications to language recognition," in *Proc. International Conference on Text, Speech and Dialogue*, Brno, Czech Republic, Sept. 2008.

[126] L. Burget, M. Fapšo, V. Hubeika, O. Glembek, M. Karafiát, M. Kockmann, P. Matějka, P. Schwarz, and Černocký, "BUT system description for NIST SRE 2008," in *Proc. NIST Speaker Recognition Evaluation 2008*, Montreal, Canada, June 2008.

[127] D.A. Leeuwen, N. Br ummer, and A. Strasheim, "TSS system description for NIST LRE 2007," in *Proc. NIST Language Recognition Evaluation 2007*, Orlando, Florida, USA, Dec. 2007.