Assign

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Speech Recognition

Spoken language recognition is a task through which we verify or identify the language spoken (e.g. English, Hindi, Odia, Bengali etc.) in a speech sample. Spoken language recognition has numerous application in a wide range of multilingual services. Automatic routing of an incoming call to a human switchboard operator is a popular example of the automatic language recognition system. Apart from this, the spoken language recognition system can also be useful for various applications like spoken language translation, multilingual speech recognition, spoken document retrieval. It is also fruitful in intelligence and security application for information distillation [?].

Human performs the language recognition task most efficiently. With a short period of training, people are able to recognize the language within seconds of hearing an utterance, even if they are not familiar with the language they can give subjective judgment based on the similarity of the language (e.g. sounds like English). Hence, like any other artificial intelligence technologies, spoken language recognition aims to replicate such human's ability through computational means.

In general, human use two broad classes of information, prelexical information and lexical-semantic information to discriminate one language from other [?]. Acoustic phonetics, phonotactics and prosodic information of an utterance of the language, comes under prelexical information, whereas the words and syntax (i.e. grammar, Phases) level information about a language comes under lexical information. It has also investigated that infants, who have not gained a good lexical knowledge can be able to discriminate languages using prelexical information [?].

Similarly, when an adult is dealing with two unfamiliar languages also use the prelexical information for discrimination. But when the infants' and adult's language experience increases, lexical semantic information plays a vital role for discrimination [?]. Hence there is no doubt that both prelexical and lexical information contributes to the human perceptual process for spoken language recognition. While we know that it requires a major effort to get hold of the lexical information of a new language. Therefore, we can also say human rapidly use the prelexical information for the language recognition process. So, the research community paid more attention to capturing the prelexical information for the development of automatic spoken language recognition system.

Prelexical information of a language consists of acoustic phonetics (spectrum, phone inventory), phonotactics (sequences of sounds), prosodic (duration, pitch, intonation) information [?]. From the literature, it is shown that the prosodic cues are less informative than the phonotactic one [? ?]. Therefore, most of the works reported in the literature towards spoken language recognition use acoustic phonetics information or phonotactics information or a combination of both information for discrimination of spoken languages.

The human speech apparatus is capable of producing a wide range of sounds. Speech sounds as concrete acoustic events are referred to as phones. Each language has a different set of phones and even if some phones are common in some languages, but their pronunciation is different [?]. These differences guarantee each language have different acoustic feature distribution. Therefore, people attempt to model the acoustic-phonetic distribution of a language using the acoustic features.

In phonological studies, it can be summarised that each language has its unique phonological rules to govern the combination of different phones. For example, the sequence of phones that occur frequently in one language could be rare in other languages. Such phonotactic constraints can be characterized by a phone n-gram model. In this approach, people try to distinguish languages by comparing the frequency of occurrence of certain reference sounds or sound sequences with that of the target language.

In general acoustic-phonetics based spoken language recognition systems, extract acoustic features (mel frequency cepstral coefficient (MFCC), perceptual linear prediction coefficient (PLP), linear prediction cepstral coefficient (LPCC), shifted delta coefficient (SDC)) from speech utterance of each language [? ? ?], using classification techniques (shallow neural networks (NN), vector quantization (VQ), Gaussian mixture model (GMM), hidden Markov model (HMM), Gaussian mixture model and universal background model (GMM-UBM), I-Vector, Deep neural networks (DNN)) try to model the acoustic features of each language [? ? ? ? ? ?]. At the time of testing the acoustic feature of the test utterance were compared with each of the language models and the language model giving higher likely-hood score is hypothesized as the language of test utterance. In the case of phonotactic approach the phone sequence of the test utterance was compared with a global phone based n-gram model of each language and the model giving higher likelihood value is identified [?]. Though phonotactic approaches give better performance in some cases, and the combination of acoustic-phonetic and phonotactics approaches improves the overall performance, the main drawback of phonotactic approaches is that it requires the phonetic transcriptions of speech utterances which is very difficult to obtain [?].

The impressive performance improvement got by using deep neural network (DNN) for automatic speech recognition motivates the research community to use various DNN architectures to perform spoken language recognition task [? ? ? ? ? ? ?]. DNN architectures can be used for both classification and feature extraction (bottleneck feature (BNF)). Phonetic aware DNN is also used for language recognition where the GMM posteriors are replaced by the DNN Senones for i-vector extraction [?]. As the spoken language discrimination ability depends on both the acoustic and phonotactic information, people try to extract features having both the information from the bottleneck layer by providing acoustic features to the input of a feed-forward DNN and posterior senone probabilities to the output of DNN [?]. Currently, the time delay neural network based X-vector framework is the state-of-the-art technique for spoken language recognition [? ?]. In the X-vector framework, after x-vector computation, the same classification technique like I-vector was followed.

1.2 Language identification, Language verification and Language diarization

Like speaker recognition, spoken language recognition can be also be viewed as:

- Language identification
- Language verification
- Language diarization

1.2.1 Language identification and verification

In language identification, the task is to determine the spoken language of the test utterance by comparison with a set of enrolled language models. The language model with which the test utterance giving higher likely-hood score is hypothesized as the language of test utterance. If the language's of all the test utterance is available with the machine, then it is called Closed-set language identification, otherwise called Open-set language Identification. The performance of the language Identification system is evaluated in terms of identification accuracy.

In the language verification, the task is to validate the test utterance in the form of target (or claimed) language or not (i.e the decision had been taken between two hypotheses). where the decision has to be made between two hypotheses with respect to a decision threshold. The basic block diagram of the language identification/ language verification system is shown in figure **??**.

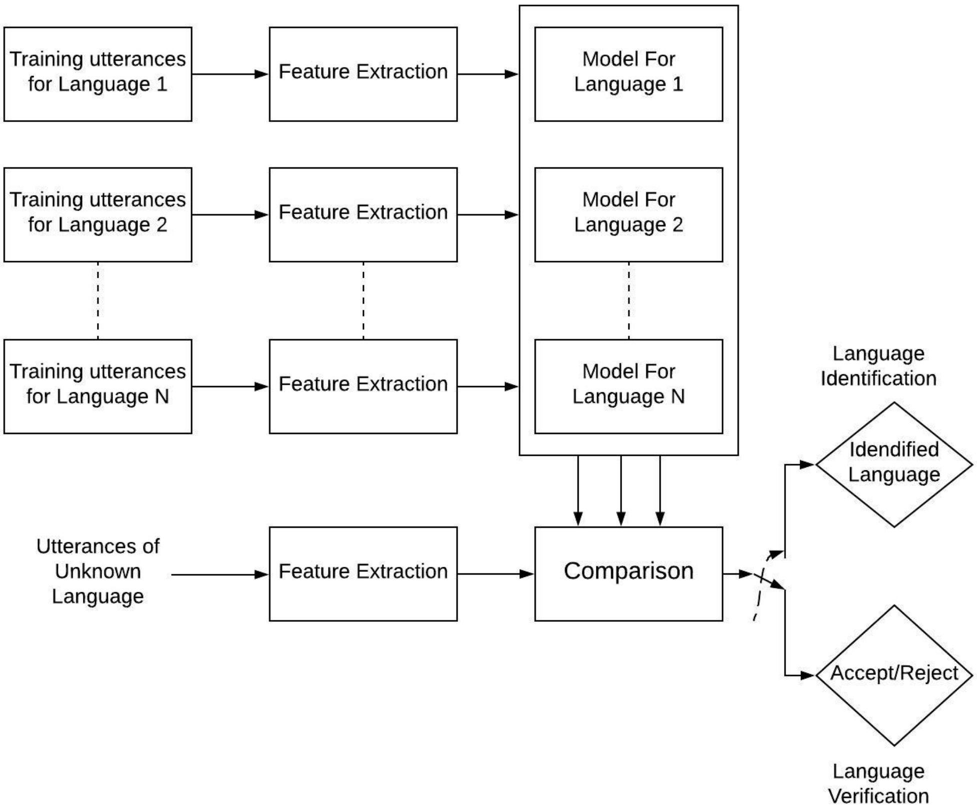

Figure 1.1: Basic block diagram of language identification/verification system.

Language recognition could be best manifested as a multiclass recognition problem, where the input belongs to one from the set of discrete classes. The objective is to recognize the class of the input. Let we have N target classes, $\{L_1, L_2, \ldots, L_N\}$. In the close-set there are N different specified languages, whereas in open set there are $N-1$ specified languages corresponds to $L_1, L_2, \ldots, L_{N-1}$ target classes and L_N class denotes any unseen out of set languages. If O is the spoken utterance

- Language identification: Which of the N language does O belongs to?
- Language verification: Does O belongs to language L_l or to other $N-1$ languages?

1.2.2 Language diarization

Language diarization is a task to perform automatic language segmentation and recognition in a code-switched speech. Language diarization is different from language recognition, as in language recognition the test utterance is a monolingual utterance and the task is to identify the language identity. But in the case of language diarization, the test utterance's both the language identity and boundary of the language transition is unknown and the task is to find the language identity and language boundary. The basic block diagram of the language diarization system is shown in figure **??**. The most challenging part of language diarization is the duration of mono-lingual segments of code-switched speech, which are much shorter than those traditionally studied in the language recognition. The performance of language diarization was measured in-terms of frame error rate (FER) [**?**]. The application of language diarization can improve the performance of multilingual automatic speech recognition (ASR) system by improving the performance of Large vocabulary continuous speech recognition (LVCSR) when the test utterance contains code switch speech [**?**]. Therefore for a country like India, where people use more than two languages while conversation, the language diarization system may play an important role to enhance the ASR performance.

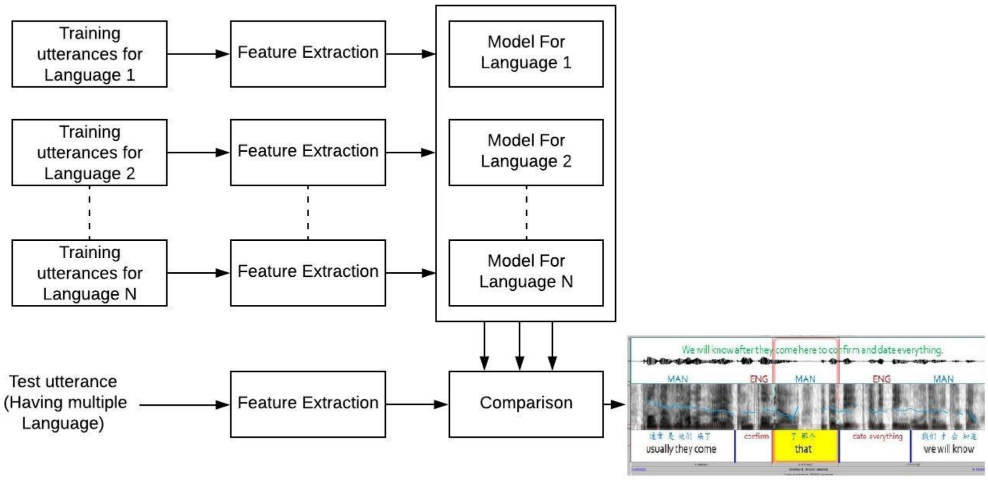

FIGURE 1.2: Basic block diagram of language diarization system [**?**].

As like spoken language recognition, the language diarization can be performed by using both acoustic and phonotactic approach. In [**?**], Lyu et al. used 63-hour conversational South-East-Asia Mandarin/English (SEAME) code-switch corpus to perform language diarization experiment. In SEAME database, average language intervals in monolingual Mandarin and English segments are about 0.81 seconds and 0.67 seconds, respectively. The average number of language changes within a code-switched utterance is about 2.2. They propose a phonotactic approach, where the acoustic features are passed through a phone recognizer and the phone recognizer output with different context was used to train a conditional random field (CRF) classifier. The performance of the system increases with increase in no of context frames. The best performance achieved was 14.4% FER with context length 2. In [?], Yilmaz et al. used an acoustic approach (BNF-UBM-I-vector) to perform language diarization. They use language diarization to perform automatic transcription of bilingual broadcast data. They observed that the use of language diarization with monolingual ASR instead of using bilingual ASR system improve the performance bilingual automatic transcription system.

1.3 Major milestones for language recognition

The first attempt of identifying the spoken language was proposed in [?], where the phonotactic patterns were used to distinguish languages by computing the frequency of occurrence of certain sound units. To perform the language identification task they used five different languages and achieved an overall accuracy of 64%. In the next attempt [?], Markov process was used to model the sequence of sounds having a manual transcription of the utterances of eight different languages. The next attempt was coming on 1980 [?], where they modeled the real speech data using a Markov process with broad phone classes of five different languages and obtained a testing accuracy of 80%.

In 1982 Cimarusti et al. [?] first time extracted acoustic features from the speech signal and modeled using a polynomial classifier. In this work they derived 100 features (i.e 15 area functions, 15 auto-correlation coefficients, 5 bandwidths, 15 Cepstral coefficients, 15 filter coefficients, 5 formant frequencies, 15 log area ratios, and 15 reflection coefficients) from the linear prediction coefficient of the speech utterance having frame size 30 msec with a context shift of 30 msec. They used 8 languages in the study and achieved an accuracy of 84%. In [?], Foil et al. used noisy radio recordings to identify the languages, where they examined two types of language identification systems. In the first approach, they extracted seven prosodic features (based on rhythm and intonation) from pitch and energy contour. In the second they used formant frequencies (in terms of values and locations) to represent the characteristic sound patterns of the language. A k-means clustering algorithm and VQ were used for classification. The language identification performance achieved on three languages was 64% correct with 11% rejection on the test data duration of 4.5 seconds with the signal to noise ratio (SNR) of 5 dB. Motivating by the work of Foil et al., In [?] Goodman et al. attempted to improve the LPC based format extraction algorithm with six language database and test data duration of smaller than 10 seconds. The final performance was reported in terms of, a function of time, SNR and no-decision rate.

In [?], Sugiyama et al. performed VQ classification on LPC derived features. They explored the difference between using one VQ codebook per language vs. one common VQ codebook for all languages, where the languages were classified according to their occurrence probability in histogram patterns. The experiment results show that the recognition rates for the first and second algorithms were 65% and 80%, respectively. The speech database used in this paper contains 20 languages: 16 sentences uttered twice by 4 males and 4 females with a duration of 8 seconds each.

In [?], Nakagawa et al. compared four methods VQ, discrete HMM, continuous density HMM and GMM. Comparative analysis of all mentioned methods was conducted in four languages. In the final performance, it is shown that the results obtained using continuous HMM and GMM (81.1%) were better than vector quantization (77.4%) and discrete HMM (47.6%).

In [?], Zissman et al. perform a comparative study using HMM and GMM as a classifier with standard MFCC features along with their delta coefficients. The algorithms were evaluated on four multi-language speech databases: a three language subset of the spoken language library, a three language subset of a five-language Rome laboratory database, the 20 language CCITT database, and the ten languages OGI telephone speech database [?]. They observed that performance of single state HMM (I.e. GMM) classifier is comparable with the performance of multi-state HMM. The authors mentioned that the observed result may be due to the lack of training data to train the multi-state HMM.

In [?], Zissman et al. compared the performance of four approaches for automatic spoken language identification. The approaches taken by them are GMM for acoustic feature classification, single-language phone recognition followed by language-dependent interpolated n-gram language modeling (PRLM), parallel PRLM (which uses multiple single-language phone recognizer, each trained in a different language), and language dependent parallel phone recognition (PPR). The approaches were evaluated with the OGI multi-language telephone speech corpus and NIST 1994 corpus. They observed the systems have phone recognizer performed better than the GMM classifier. The top-performing system was parallel PRLM, which exhibited an error rate of 2% for 45-s utterances and 5% for 10-s utterances in two-language, closed-set, forced choice classification. The error rate for 11-language, closed-set, forced-choice classification was 11% for 45-s utterances and 21% for 10-second utterances.

In [?], the authors used GMM-universal background model (UBM) classifier in the spoken language recognition task, by motivating the performance of GMM-UBM over standard GMM in speaker recognition [?]. They used 39 dimensions MFCC features along with its velocity and acceleration coefficients with vocal tract length normalization (usually use to suppress speaker dependent information). They used GMM-UBM classifier, to model the acoustic feature vectors. They observed the performance of GMM-UBM is better (i.e 13.4%) than Parallel PRLM approach (i.e 16.6%) in case of 45 sec testing case, but in the case of 10 sec testing case the performance of parallel PRLM (PPRLM) (i.e 24.8%) is better than the GMM-UBM based approach (i.e 27%). The overall fused system performance is 10.2% for 45 sec and 18.4% for the 10-sec testing case. They used OGI Multi-Language Telephone Speech Corpus and NIST 1994 corpus for conducting experiments.

In [?], they were motivated by the performance of multiple-language phone recognition and n-gram language modeling on language identification task, which indicates that the automatic language identification task largely depends on the sequence information of speech utterances. The standard MFCC features are not able to capture sequence information of the utterances. In this paper they used shifted delta coefficient (SDC) which captured sequence information of speech utterances. They observed that the results obtained are comparable with phonotactic based PPRLM method. They used CallFriend and OGI corpora for system evaluation and achieved the best performance of 6.90% in CallFriend corpora.

In [?], they proposed a vector space modeling (VSM) method for automatic spoken language identification (LID) by motivating that the overall sound characteristics of all spoken languages can be covered by a universal collection of acoustic units, which can be characterized by the acoustic segment models (ASMs). A spoken utterance is then decoded into a sequence of ASM units. The ASM framework furthers the idea of language-independent phone models for LID by introducing an unsupervised learning procedure to circumvent the need for phonetic transcription. This is analogous to representing a text document as a term vector. They converted a spoken utterance into a feature vector with its attributes representing the co-occurrence statistics of the acoustic units, then built a vector space classifier for LID. The proposed VSM approach leads to a discriminative classifier at the back-end. The observed result tells that the latter approach gives superior performance over likelihood-based n-gram language modeling (LM) back-end for long utterances. The performance of the proposed VSM framework in-terms of equal error rate (EER) was 2.75% and 4.02% in the 1996 and 2003 NIST language recognition evaluation (NIST-LRE) 30-second tasks, respectively.

In [?], they used an ensemble of binary classifiers with shifted delta coefficient (SDC) to perform spoken language recognition. They adopted a distributed output coding strategy in ensemble classifier design, where they decomposed a multiclass language recognition problem into many binary classification tasks, each of which addresses a language recognition sub-task by using a component classifier. Then, they combined the results of the component classifiers to form an output code as a hypothesized solution to the overall language recognition problem. In this way, they effectively projected the high-dimensional feature vectors into a low-dimensional space by maintaining language discriminating characteristics of the spoken utterances. By fusing the output codes from both phonotactic features and cepstral features, they achieved an equal-error-rates of 1.38% and 3.20% for 30-second trials on the 2003 and 2005 NIST-LRE databases respectively.

In [?], motivated by the improvement in performance of speaker recognition system using I-vector based modeling, the authors use I-vector based modeling approach in spoken language recognition task. They achieved a performance of 2.1% EER on NIST 2009 evaluation protocol.

In [?], i-vector representation based on BNF features is presented for language identification (LID). In the proposed system, the BNF features are extracted from a deep neural network, which can effectively mine the contextual information embedded in speech frames. The i-vector representation of each utterance is then obtained by applying a total variability approach to the BNF features. The resulting performance of the LID has been significantly improved with the proposed BNF feature based i-vector representation. The obtained best result was 1.98%, 3.47%, 9.71% in-terms of EER on the NIST 2009 evaluation set with 30, 10 and 3 seconds test duration respectively.

In [?], they explore the use of Long Short-Term Memory (LSTM) recurrent neural networks (RNNs) for automatic language identification (LID). The use of RNNs is motivated by their better ability in modeling sequences with respect to feedforward networks. From the work, it is observed that long short term memory (LSTM) RNNs can effectively exploit temporal dependencies in acoustic data and learn relevant features for language discrimination. The proposed approach is compared to baseline i-vector and feedforward Deep Neural Network (DNN) systems in the NIST Language Recognition Evaluation 2009 dataset. The work shows LSTM RNNs achieve better performance than feedforward net based DNN system and i-vector system. The performance using LSTM RNN was 8.35% in terms of average EER with 3-second test data, which is better compared to 9.58% in feedforward net with 8 layers and 15.89% in I-vector system.

In [?], they used a convolutional neural network (CNN) for the training of automatic speech recognition (ASR) system, the posterior of the model is used for i-vector extraction instead of UBM posteriors. The task was performed on the RATS database with five target languages (Farsi, Urdu, Pashto, Arabic, Levantine, and Dari) and achieved a performance of 10.95%, 5.24% and 3.53% with 30, 10 and 3 seconds test utterances respectively.

In [?], Richardson et al. compared standard i-vector and various deep learning approaches for spoken language recognition task. Motivated by the gain of performance in ASR by using deep learning frameworks, the authors used deep feedforward network as a classifier, phonetic aware deep feedforward network for i-vector computation and bottleneck features were extracted from a deep feedforward network, which is used later to model the i-vector. Finally, it is observed that the performances of bottleneck feature (BNF) based i-vector system outperform the performances of all other systems. In NIST 2011 language recognition evaluation set the best performance achieved in-terms of $\mathrm{C}_{\mathrm{avg}}$ was 0.304% using DNN as posterior in 30 seconds, 1.24% in 10 seconds and 7.53% in 3 seconds using GMM as posterior with BNF-i-vector system.

In [?], Zhang et al. mentioned that traditional bottleneck feature extraction requires additional transcribed speech information, which is very difficult to get. Hence they proposed alternate unsupervised deep learning methods for bottleneck feature extraction. They used unsupervised cluster Gaussian mixtures for posterior labeling of frames instead of using posterior phone labeling for bottleneck feature extraction. They also used variational autoencoder and adversarial autoencoder for speech feature extraction. Then they used the i-vector framework to classify spoken languages. In this work they use three databases: 1) a four Chinese dialect dataset, 2) a five Arabic dialect corpus, and 3) multigenre broadcast challenge corpus (MGB-3) for language recognition task. The best fusion system performance archived in Chinese dataset was 95.7% accuracy, in Pan-Arabic was 81.3% accuracy and in MGB-3 dataset was 65.4% accuracy.

In [?], Snyder et al. applied x-vectors to the task of spoken language recognition motivated by the performance of x-vector in the speaker recognition task. The x-vector framework consists of a deep neural network that maps sequences of speech features to fixed-dimensional embeddings, called x-vectors. In this framework, long-term language characteristics are captured in the network by a temporal pooling layer that aggregates information across time. Once x-vectors were computed, the same classification procedure like i-vectors was followed. In 2017 NIST-LRE, x-vectors outperformed all state-of-the-art i-vector systems. The best performance achieved using x-vectors (multilingual BNF with augmentation) in terms of c_{avg} was 14.0% in 3 seconds of test data.

The summary of all the works reported above towards spoken language recognition is tabulated in table **??** and table **??**.

1.4 Databases for spoken language recognition

The availability of large dataset has been the major driving factor in the development of speech technology [?]. To design a spoken language recognition system, we need a set of speech data of different languages having variations within the language like intersession variability, speaker variation, device variation and recording environment variation.

There are three standard organizations, who develop standard challenge databases for language recognition study. These standard corpora are Oregon graduate institute telephone speech corpus(OGI-TS), LDC call friend telephone speech corpus and NIST Language recognition corpus.

Table 1.1: A summary on different studies made on language recognition.

LRE Systems	Method used	Performance	Database
Leonard et al. [**?**]	Phonotactic Pattern	Identification accuracy=64%	In house data (five language)
House et al. [**?**]	Markov Process (Manual phone transcription)		In house data (Eight language)
Li et al. [**?**]	Markov Process (Real speech model)	Identification accuracy=80%	In house data (Five language)
Cimarusti et al. [**?**]	100 dimension acoustic feature (Polynomial Classifier)	Identification accuracy=84%	In house data (Eight language)
Foil et. al. [**?**]	Prosodic feature, Frequency of Format location	Identification accuracy=64%	In house data, Noisy data with 5 dB SNR (Three language)
Goodman et al. [**?**]	Improved LPC based Format extraction algorithm		In house data, Noisy data (Six language)
Sugiyama et al. [**?**]	LPC derive feature VQ Code-book	Identification accuracy=80%	In house data, (Twenty language)
Nakagawa et al. [**?**]	VQ, Discrete HMM, Continuous HMM, GMM	Identification accuracy=81.1%	In house data, (Four language)
Zissman et al. [**?**]	HMM, GMM (MFCC+ Delta)	Identification accuracy=80%	OGI-11L Data set
Zissman et al. [**?**]	GMM Parallel PRLM	Error rate 45 sec- 11% 10 sec- 21%	OGI-11L Data set NIST 1994
Wong et al. [**?**]	GMM-UBM	Error rate 45 sec- 10.2% 10 sec- 18.4%	OGI-11L Data set NIST 1994
Carrasquillo et al. [**?**]	SDC-GMM, PPRLM	Error rate 45 sec- 6.90%	OGI Data set CallFriend Corpus
Li et al. [**?**]	Vector Space Modelling (VSM)	EER 30 sec- 2.75% (1996) 30 sec- 4.02% (2003)	NIST-1996 NIST-2003

Table 1.2: A summary on different studies made on language recognition.

LRE Systems	Method used	Performance	Database
Ma et al. [?]	SDC Binary Classification	EER 30 sec- 1.38% (2003) 30 sec- 3.20% (2005)	NIST-2003 NIST-2005
Dehak et al. [?]	I-vector	EER 30 sec-2.1%	NIST-2009
Song et al. [?]	BNF-I vector	EER 30 sec-1.98% 10 sec-3.47% 3 sec-9.71%	NIST-2009
Gonzalez et al. [?]	RNN-LSTM	EER 3 sec- 8.35%	NIST-2009
Lei et al. [?]	CNN-I vector	EER 30 sec- 10.95% 10 sec- 5.24% 3 sec-3.53%	RATS (Five language)
Richardson et al. [?]	DNN DNN-I-Vector BNF-UBM-I-vector BNF-DNN-I-vector	C_{avg} 30 sec-0.304% 10 sec-1.24% 3 sec- 7.53%	NIST-2011
Zhang et al. [?]	Unsupervised BNF Autoencoder Feature	Identification accuracy Chinese - 95.7% Pan Arabic - 81.3% MGB-3 - 65.4%	Chinese Dialect Pan Arabic MGB-3
Snyder et al. [?]	X-vector	C_{avg} 3 sec - 14.0%	NIST-2017

1.4.1 Oregon graduate institute telephone speech corpus (OGI-TS)

The OGI-TS is the first publicly available multi-lingual speech corpus developed to perform spoken Language recognition task [?]. The OGI-TS speech corpus contains the speech from 11 languages: English, Farsi, French, German, Hindi, Japanese, Korean, Mandarin, Chinese, Spanish, Tamil, and Vietnamese. Each language contains the speech from about 80 native speakers, where each speech utterance in the corpus was spoken by a unique speaker over the telephone channel with a sampling frequency of 8kHz. The corpus was collected and developed in 1992, and the latest version was released in 2002 which includes recorded utterances from about 2052 speakers, for a total of about 38.5 hours of speech.

1.4.1.1 OGI-TS 22 language corpus

The current version of the OGI 22 Language corpus consists of telephone speech from 21 languages: Arabic, Cantonese, Czech, English, Farsi, German, Hindi, Hungarian, Japanese, Korean, Indonesian, Mandarin Chinese, Italian, Polish, Portuguese, Russian, Spanish, Swedish, Swahili, Tamil and Vietnamese [?]. The corpus contains fixed vocabulary utterances (e.g. days of the week) as well as fluent continuous speech. Approximately 20,000 utterances in 16 languages have corresponding orthographic transcriptions.

1.4.2 LDC callfriend telephone speech corpus

The call friend corpus released by LDC having unscripted conversion speech of 12 languages [? ?]. These 12 languages are Egyptian Arabic, English, Farsi, French, German, Hindi, Japanese, Korean, Mandarin, Spanish, Tamil, and Vietnamese. From the 12 languages, 3 languages each having two dialects: English (American English with southern and non-southern dialect), Mandarin (Mainland and Taiwan dialect), Spanish (Spanish-Caribbean and Spanish-Non Caribbean dialect). Dialect means the variation in utterances of the same language by the speakers belongs to the different geographical region. The database consists of 60 telephone conversations for each language, where 20 are used for the training set, 20 for the development set and 20 for the evaluation set. As per the protocol the training set is used to train the language model and classifier, the development set to train the back-end classifier and the evaluation set for final system evaluation.

1.4.3 NIST Language recognition corpus (NIST LRE)

The National Institute of Standards and Technology (NIST) has conducted a series of evaluations of spoken language recognition in 1996, 2003, 2005, 2007, 2009, 2011, 2015 and 2017 [? ? ?].These evaluations have been designed to foster research progress, with the goals of

1. Exploring promising new ideas in language recognition.
2. Developing advanced technology incorporating these ideas.
3. Measuring the performance of this technology.

It has been seen that more languages are added from year to year. The emphasis of NIST LRE has been on conversational telephone speech (CTS), since most of the likely applications of the technology involve signals recorded from the public telephone system. In order to collect speech data of more languages in a cost-effective way, NIST has adopted broadcast narrow-band speech (BNBS) lately in LRE 2009, LRE 2011, LRE 2015 and LRE 2017. BNBS data are excerpts of call-in telephone speech embedded in broadcast and webcast. The call-in excerpts are used as we could expect them to cover as many speakers as possible. Broadcast entities like the Voice of America (VOA) broadcast are having more than 45 languages. Alternatively, the British Broadcast Company (BBC) also produces and distributes programs in a large number of languages. The number of target languages in NIST 1996 has 12, NIST 2003 has 12, NIST 2005 has 9, NIST 2007 has 24 with 5 open-set languages, NIST 2009 has 23 with 16 open-set languages, NIST 2011 has 24, NIST 2015 has 20 and NIST 2017 has 14. In NIST 2017, unlike previous LREs the evaluation data will be divided into partitions based on the data source, i.e., MLS14 and VAST, for each language, resulting in a total of 28 partitions. The NIST evaluation protocol has three testing conditions: 30 seconds, 10 seconds and 3 seconds.

1.5 Performance measure for language recognition

The spoken language recognition experiments generally evaluated by identification accuracy, average equal error rate, and average detection cost [?].

1.5.1 Average detection cost($\mathrm{C_{avg}}$)

As per NIST LRE, the language recognition system performance is evaluated in terms of average detection cost. In spoken language recognition, the performance is evaluated by presenting the system with a set of trials, each consisting of a test segment and a hypothesized target language. The system has to decide for each trail that whether the target language was spoken in the given segment.

Let $\mathrm{N_T}$ be the number of test segments and N be the number of target languages. By presenting each test segment against all target languages, there are $\mathrm{N_T}$ number of trials for each target and the system under evaluation should produce $\mathrm{N} \times \mathrm{N_T}$ number of true or false decisions. The average detection cost can be written as,

$$\mathrm{C_{avg}} = \frac{1}{\mathrm{N}} \sum_{\mathrm{l}=1}^{\mathrm{N}} \mathrm{C_{DET}}(\mathrm{L_l})$$

where $\mathrm{C_{DET}(L_l)}$ is the detection cost for the subset of $\mathrm{N_T}$ trials for which the target language is $\mathrm{L_l}$.

$$\mathrm{C_{DET}(L_l) = C_{miss}P_{tar}P_{miss}(L_l) + C_{fa}(1 - P_{tar})\frac{1}{N-1}\sum_{m \neq l} P_{fa}(L_l, L_m)}$$

where $\mathrm{C_{miss}}$ and $\mathrm{C_{fa}}$ are the cost of making false rejection (miss probability) and false acceptance. $\mathrm{P_{tar}}$ is the prior probability of the target. In general $\mathrm{C_{miss}}$, $\mathrm{C_{fa}}$ and $\mathrm{P_{tar}}$ are fixed to 1, 1 and 0.5 respectively. $\mathrm{P_{miss}}$ is the false rejection probability and $\mathrm{P_{fa}}$ is the false acceptance probability.

1.6 Summery and Discussion

The various key aspects of Automatic Language Recognition system are discussed in this report. It includes several key aspects of language recognition, covering language characterization and various modeling techniques useful in automatic language recognition. Various system development strategies regarding the task are also discussed. It has been seen from the literature that tremendous development is observed in language recognition in the last few decades. Incorporation of DNN technique in the field of language recognition dramatically change the performance of the system. Though the language recognition is still far from perfect. From the literature, it is observed that SDC features have better language discrimination ability than all the other available features. As SDC feature capture speech dynamics over a wide range of speech frames, which helps to capture some phonotactic information also. Before the incorporation of DNN based frameworks, SDC with i-vector framework was the state-of-art language recognition technique. After neural network based approaches evolved in the field of language recognition BNF based i-vector approach has been used as the state-of-the-art approach. Recently found that the x-vector framework, which was originally developed for speaker recognition, outperformed several state-of-the-art i-vector systems on the NIST LRE 2017 database in the language recognition task. The NIST LRE database series is the most widely used dataset for language recognition study.

It had been shown that acoustic and phonotactic features have been widely used in the language recognition task. Whereas human listening experiments indicated that prosodic and other high-level features are equally informative. This prompts us to develop some techniques for language recognition in the future, which can effectively capture such high-level information to further improve the performance of the system.

Chapter 2

Acoustic feature extraction and classification techniques used for Spoken language recognition

The acoustic information is generally considered as the first level analysis of speech production. Human speech is a longitudinal pressure wave, and different speech events can be distinguished at an acoustic level, according to amplitude and frequency components of the waves. The acoustic information is one of the simplest forms of information, which can be obtained during the speech parameterization process directly from raw speech. The higher level of speech information like phonotactic and word information can also be extracted from the acoustic information [?]. For the spoken language recognition task, the Mel frequency cepstral coefficients (MFCC) are the most widely used feature extraction technique. The language discriminative information largely reflects on the temporal patterns of the speech signal. Thus, once the basic acoustic features have been obtained, additional features are appended to each feature vectors. Some commonly utilized additional features are the delta and acceleration cepstrum (MFCC + delta + acceleration) and the shifted delta cepstrum (SDC).

2.1 Mel frequency cepstral coefficients (MFCC)

The Mel frequency cepstral coefficients (MFCCs) are one of the most commonly used filter-bank based parameterization method for speech processing applications, such as speech recognition, speaker verification/identification and spoken language recognition. The human perception of the frequency content of the sound follows a nonlinear scale called Mel scale [?]. So, the Mel-scale is used to approximate the nonlinear frequency resolution of the human ear. After the magnitude-square of the Fourier Transform is calculated for the input windowed frame of speech, it is passed through a bank of triangular Mel filters and the natural logarithm of the filter bank energies is taken. As the filter bank log-energies are highly correlated, a linear transformation technique (I.e the Discrete Cosine Transform (DCT)) is used to de-correlate the information yielding in Mel frequency cepstral coefficients. The steps to extract MFCC Features from speech sample is depicted in figure **??**. The brief description of all the steps are as follows.

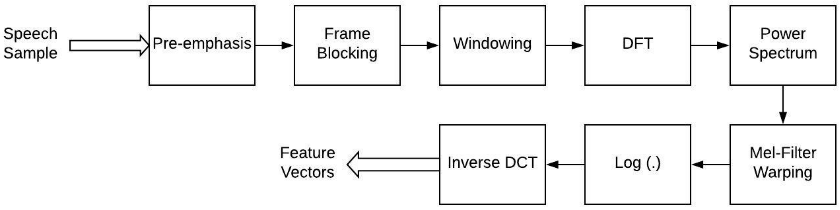

FIGURE 2.1: Block diagram of MFCC extraction.

2.1.1 Pre-emphasis

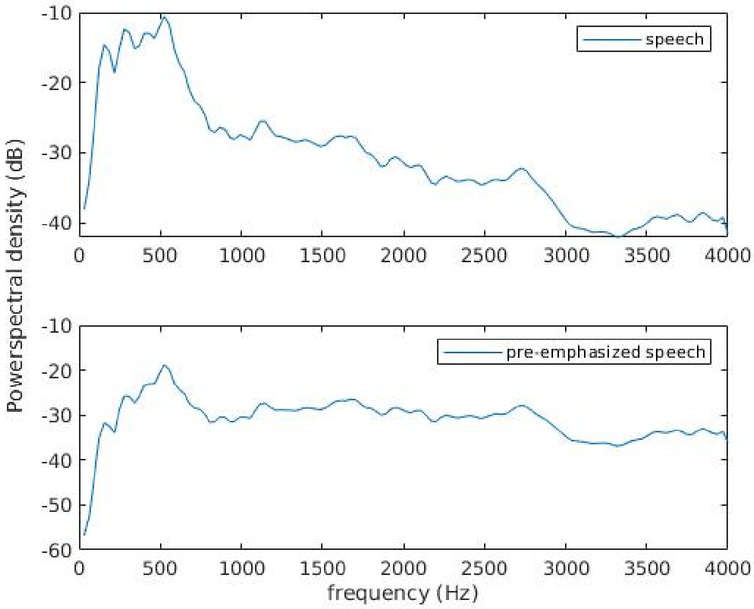

FIGURE 2.2: The power spectral density of the original speech signal and pre-emphasized speech signal sampled at 8000 Hz.

Figure **??** shows the power spectral density of the original speech and its pre-emphasized speech. It has been observed from the figure that the power of the signal falls sharply in high frequency regions. It is estimated that about 80% of the power is contained within frequency components below 1000 Hz [**?**]. To perform verious recognition task human ear's cochlea utilizes a fine-tuning mechanism based on feedback from the brain that amplifies special frequencies. Therefore the human ear can easily recognize these low energy regions. Since our objective is to design a recognition system, we have to do something to enhance the high frequency components, by which high frequency information can also play a role in recognition task. This can be achieved through pre-emphasis. One method which has been used quite often is a differentiator (single zero filter), whose transfer function is given in equation **??**

$$\mathrm{H_p(z)} = 1 - \alpha \mathrm{z}^{-1} \tag{2.1}$$

The most popular range of values for α is between 0.95 and 0.97, although values in the range of 0.9 and just less than 1.0 have also been used in different systems. Figure **??** shows the power spectral density of the pre-emphasized signal using $\alpha = 0.98$. Here we notice that the absolute power for each frequency range has been reduced, but the relative power is better distributed along the different frequencies.

2.1.2 Frame blocking

The speech is slow varying quasi-stationary signal. Therefore, speech analysis must always be carried out on short segments across which the speech signal is assumed to be stationary. Short-term spectral measurements are typically carried out over the range of 10-30 ms frame size and shift [**?**]. In that direction, the speech signal is divided into frames of L samples, with adjacent frames being separated by M samples with the value M less than that of L. The first frame consists of the first L samples. The second frame begins from M samples after the first frame, and overlaps it by L - M samples and so on. This process continues until all the speech samples are taken into account.

2.1.3 Windowing

The next step is to window each individual frame to minimize the signal discontinuities at the beginning and end of each frame. The concept applied here is to minimize the spectral distortion by using the window to taper the signal to zero at the beginning and end of each frame. If we define the window as w(n), $0 \leq \mathrm{n} \leq \mathrm{L}{-}1$ where L is the frame length, then the result of windowed signal can be written as:

$$\mathrm{s_o(n)} = \mathrm{w(n).s_i(n)} \tag{2.2}$$

where w(n) the window function. In general hamming window (to eliminate the problem of spectral leakage and zero offset) is widely used to analyze the speech frames.

$$\mathrm{w(n)} = 0.54 - 0.46\cos\left(\frac{2\pi \mathrm{n}}{\mathrm{L}-1}\right), 0 \leq \mathrm{n} \leq \mathrm{L}-1 \tag{2.3}$$

2.1.4 DFT

The next step in the processing of the speech data to be able to compute its spectral features is to take a Discrete Fourier Transform of the windowed data. The DFT is computed using equation **??**. The DFT length L is always grater then the windowed speech segments length to avoid time aliasing.

$$\mathrm{S(k)} = \sum_{\mathrm{n}=0}^{\mathrm{L}-1} \mathrm{s(n).e}^{\frac{-\mathrm{j}2\pi \mathrm{kn}}{\mathrm{L}}}, 1 \le \mathrm{k} \le \mathrm{L}-1, 1 \le \mathrm{n} \le \mathrm{L}-1 \tag{2.4}$$

2.1.5 Power spectrum

The loudness is related with intensity of the signal, hence our goal is to convert it into a value that would represent loudness so that we may mimic human perception. Therefore, after the calculation of DFT of each windowed speech segment the power spectrum ($|\mathrm{S(K)}|^2$) is calculated.

2.1.6 Mel filter warping

The human auditory perception is based on a scale which is somewhat linear up to the frequency of 1000 Hz and then becomes close to logarithmic for the higher frequencies. This was the motivation to use the Mel scale. It has been seen from the literature, that the 24 band filter bank is used to model the auditory system [**?**]. The designed 24 Mel filter banks are presented in figure **??**. First the lowest and highest frequency is converted to Mel scale using equation **??**. The unity height triangular filters are uniformly spaced in Mel scale, so in Mel scale the difference between the lowest and highest frequency are uniformly divided into M (no of filter)+1 segments to find the center frequency of each filter. Then the center frequencies which are in Mel scale are converted to Hz scale using equation **??** [**?**]. The triangular filter banks ($\mathrm{H_m(k)}$) can be constructed using equation **??**. $\{\mathrm{k_{c_m}}\}_{\mathrm{m}=1}^{\mathrm{M}}$ are the center frequencies of the filter, $\mathrm{k_{c_0}}$ and $\mathrm{k_{c_{M+1}}}$ are the boundary frequencies in Hz scale (lowest and highest frequency). The output of the Mel filter bank can be computed using equation **??**. Where $\mathrm{H_m(k)}$ is the filter bank function and M is the no of filter bank used.

$$\mathrm{f_{Mel}} = 2595\ .\ \log_{10}\Big(1 + \frac{\mathrm{f_{lin}}}{700}\Big) \tag{2.5}$$

$$\mathrm{f_{lin}} = 700.\Big(10^{\frac{\mathrm{f_{mel}}}{2595}} - 1\Big) \tag{2.6}$$

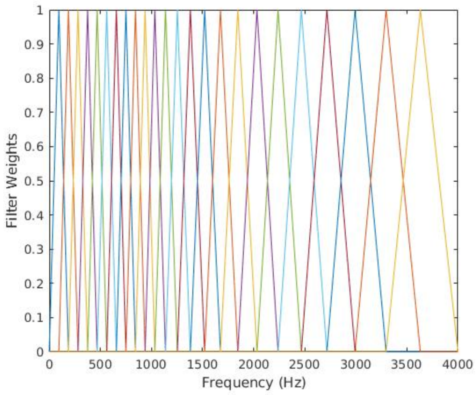

FIGURE 2.3: Shape of Mel filter bank for a 24 filter system with sampling frequency of 8000 Hz .

$$H_m(k) = \begin{cases} 0 & \text{for } k \leq k_{c_{m-1}} \\ \frac{k-k_{c_{m-1}}}{k_{c_m}-k_{c_{m-1}}} & \text{for } k_{c_{m-1}} \leq k \leq k_{c_m} \\ \frac{k_{c_{m+1}}-k}{k_{c_{m+1}}-k_{c_m}} & \text{for } k_{c_m} \leq k \leq k_{c_{m+1}} \\ 0 & \text{for } k \geq k_{c_{m+1}}, \quad 1 \leq m \leq M \end{cases} \tag{2.7}$$

$$S(m) = \sum_{k=1}^{N} |S(k)|^2 H_{m+1}(k), 0 \leq m \leq M-1 \tag{2.8}$$

2.1.7 log(.)

Now we have to warp the power spectra into a logarithmic scale. As we know the dynamic range of the power spectra very high in between different Mel frequency bands, so logarithm with base 10 is used to reduce the dynamic range. It has been seen that in human perception, the vocal tract response information plays a vital role to recognize speaker and language. In time domain the vocal tract response and excitation response are convoluted, therefore in frequency domain they are multiplied. The use of log(.), converts multiplication to addition, latter low time liftering is used to enhance the vocal tract information.

2.1.8 Inverse DCT

In the previous section we computed the log of the power spectral density of the signal in Mel scale. By taking inverse discrete cosine transform (DCT), we will get the cepstrum of the signal. The most attractive features of the cepstrum is its inherent invariance toward linear spectral distortions, which make it a good candidate for usage in speaker and language recognition task. We can also compute the cepstrum using inverse discrete Fourier transform, but in that case the coefficients are complex (s(m) is not symmetric). One possibility is to take square magnitude of the coefficients or to use inverse DCT directly to get real coefficients. The coefficients using inverse DCT is computed using equation **??**.

$$\mathrm{c(n)} = \sum_{\mathrm{m=0}}^{\mathrm{M-1}} \mathrm{a_m log_{10}(S(m))cos}\left(\frac{\pi \mathrm{n(m-0.5)}}{\mathrm{M}}\right), 0 \leq \mathrm{n} \leq \mathrm{C}-1 \tag{2.9}$$

$$\mathrm{a_m} = \begin{cases} \frac{1}{\mathrm{M}} & \text{for } \mathrm{m} = 0 \\ \frac{2}{\mathrm{M}} & \text{for } \mathrm{m} > 0 \end{cases}$$

Where C is the no of cepstral coefficients needs to be computed and DCT length M should be always grater then or equal to the no of filter bank used to compute Mel coefficients.

2.2 Delta and Delta-Delta cepstra (Δ and $\Delta - \Delta$)

The static MFCC feature vectors provides a good estimation of local spectra, but it fails to capture the dynamics of human speech. The dynamics of human speech is very important information for distinguishing different languages [**?**]. Therefore, the performance of the system greatly enhanced by adding time derivative of basic static parameters. The Delta and Delta-Delta provides an estimation of local temporal derivatives of the speech cepstrum, and are implemented as a least square approximation of the local slope and calculated over multiple frames. The first order derivative are computed as delta coefficients and can be computed as per equation **??**.

$$\Delta \mathrm{c_i(n)} = \frac{\sum_{\mathrm{k=-N}}^{\mathrm{k=N}} \mathrm{k c_i(n+k)}}{\sum_{\mathrm{k=-N}}^{\mathrm{k=N}} \mathrm{k}^2} \tag{2.10}$$

Where $\Delta \mathrm{c_i(n)}$ is the delta coefficient of the $\mathrm{i^{th}}$ cepstral stream and $\mathrm{n^{th}}$ frame. The N defines the no of frames to be used for the computation of delta coefficients. Typically the range of N is from 2 to 4. Similarly the $\Delta - \Delta$ coefficients are computed using the same equation on Delta coefficients instead of static coefficients. As the Delta and Delta-Delta coefficients computation needs the coefficients of previous and post frame, needs some modification at beginning and end of speech. This end-effect problem can be solved by using simple first order differences at the start and end of the speech (as per equation **??** and **??**).

$$\Delta \mathrm{c_i(n)} = \mathrm{c_i(n+1)} - \mathrm{c_i(n)}, \quad \mathrm{n} < \mathrm{N} \tag{2.11}$$

$$\Delta \mathrm{c_i(n)} = \mathrm{c_i(n)} - \mathrm{c_i(n-1)}, \quad \mathrm{n} \geq \mathrm{T} - \mathrm{N} \tag{2.12}$$

Where T is the total no of frames. In general the delta and delta-delta cepstrum are concatenated with the static cepstrum to form a single feature vector containing both the static and dynamic information of the speech signal.

2.3 Shifted delta coefficients (SDC)

The delta and delta-delta feature able to capture the temporal information, But it is limited, and not able to capture the higher level temporal dynamics information. If we consider $N = 2$, the delta and delta-delta can able to capture the temporal dynamics information across 5 frames, i.e. across 50 ms (if frameshift = 10ms).

Temporal information has proven useful in distinguishing languages, i.e accessing the likelihood of one phone following another (phonotactic information). Thus, this intuition leads to design a feature, which can capture the temporal dynamics information in a larger window. One way, we can choose larger N value in the delta calculations to include a much longer window in the calculation. But, this will only produce a much longer average of the slope and the finer details will be lost. The SDC feature is a better alternative to capture spectral dynamics in a longer window [**?**]. The SDC feature is obtained by concatenating future and past frames delta cepstra with the current feature vector.

In general, the computation of SDC are specified by four parameters: z, d, p, k. z specifies the number of basic cepstral streams to use in the calculation, i.e. the number of basic cepstral coefficients. Each of z cepstral streams are treated separately and SDC values are computed for each of them prior to concatenation with the original cepstral coefficients. p is the number of frames from one delta calculation to the next and k is the total number of delta values concatenated together to form the SDC. The diagram showing the computation procedure of SDC is shown in figure **??**

The final parameter d is the difference value used in the delta calculation. for all the SDC calculations the delta values were calculated by subtracting the cepstral value at $n - d$ from that at $n + d$. Thus for each of the M cepstral streams, the final vector at time n is given by the concatenation of all the $\Delta c_i(n, z)$ for $0 \leq z \leq k$, where i represents the i^{th} cepstral coefficient. The mathematical equation for the computation of $\Delta c_i(n, z)$ is given in equation **??**.

$$\Delta c_i(n, z) = c_i(n + zp + d) - c_i(n + zp - d) \tag{2.13}$$

It has been seen that, the regression based delta computation is more efficient for better estimate then the method of simple subtracting [**?**]. In this way, the $\Delta c_i(n, z)$ can be computed using

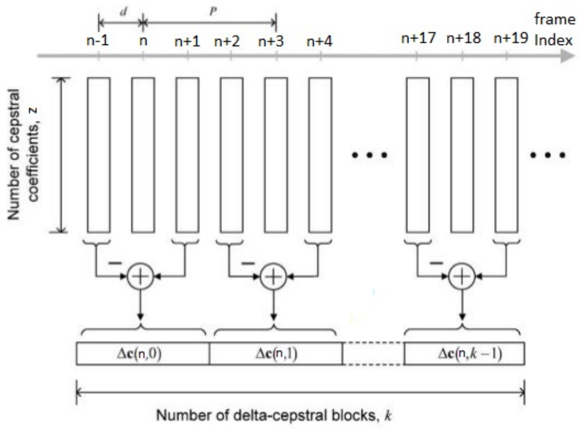
d
p
n-1
n
n+1
n+2
n+3
n+4
n+17
n+18
n+19
frame Index
Number of cepstral coefficients, z
−
Δc(n,0)
Δc(n,1)
Δc(n,k−1)
Number of delta-cepstral blocks, k

FIGURE 2.4: SDC feature extraction at $\mathrm{n^{th}}$ frame, $\mathrm{z-d-p-k = 7-1-3-7}$.

equation **??**.

$$\Delta \mathrm{c_i(n,z)} = \frac{\sum_{\mathrm{l=-d}}^{\mathrm{l=d}} \mathrm{l c_i(n+zp+l)}}{\sum_{\mathrm{l=-d}}^{\mathrm{l=d}} \mathrm{l^2}} \tag{2.14}$$

With either the standard subtraction method, or the modified regression based technique, the SDC capture temporal dynamics information in a much larger window then the standard delta and delta-delta. In general, people use SDC feature for language recognition task with configuration $\mathrm{z-d-p-k = 7-1-3-7}$. The best way to compute the SDC feature is, after computing static MFCC (c(n)) and its delta coefficients (d(n)), the SDC coefficients of the $\mathrm{n^{th}}$ frame is given as:

$$\mathrm{sdc(n)} = \Big[\mathrm{c(n), d(n-p), \ldots, d(n), \ldots (d(n+p))}\Big]$$

2.4 Gaussian Mixture Model

Gaussian Mixture Model (GMM) is a parametric probability density function represented as a weighted sum of Gaussian component densities. GMMs are capable of representing a large class of sample distributions. Thus, GMMs are commonly used as a parametric model of the probability distribution of continuous measurements or features in a speaker and language recognition systems.

The use of GMM may also be motivated by the intuitive notion that the individual component densities may model some underlying set of hidden classes. For example, in speaker recognition, it is reasonable to assume the acoustic space of spectral related features corresponding to a speaker's broad phonetic events, such as vowels, nasals or fricatives. These acoustic classes reflect some general speaker dependent vocal tract configurations that are useful for characterizing speaker identity. The spectral shape of the $\mathrm{i^{th}}$ acoustic class can in turn be represented by the mean μ_i of the $\mathrm{i^{th}}$ component density, and variations of the average spectral shape can be represented by the covariance matrix Σ_i. Because all the features used to train the GMM are unlabeled, the acoustic classes are hidden in that the class of an observation is unknown. GMM parameters are estimated from training data using the iterative expectation-maximization (EM) algorithm or maximum a posteriori (MAP) estimation from a well-trained prior model.

For a D-dimensional feature vector denoted as X, the mixture density for a language S is defined as weighted sum of M_g component Gaussian densities as given by the following equation **??** [**?**].

$$P(X|S) = \sum_{i=1}^{M_g} w_i P(X|\theta_i) \tag{2.15}$$

where w_i are the weights and $P(X|\theta_i)$ are the component densities. Each component density is a D-variate Gaussian function of parameters θ_i ($\theta_i = \begin{bmatrix} \mu_i & \Sigma_i \end{bmatrix}$). The probability density function of $P(X|\theta_i)$ can be written as per equation **??**

$$P(X|\theta_i) = \frac{1}{(2\pi)^{D/2}|\Sigma_i|^{\frac{1}{2}}} \quad e^{-\frac{1}{2}\begin{bmatrix}(X-\mu_i)^T\Sigma_i^{-1}(X-\mu_i)\end{bmatrix}} \tag{2.16}$$

where μ_i is a mean vector and Σ_i covariance matrix for i^{th} component. The mixture weights have to satisfy the constraint **??** [**?**].

$$\sum_{i=1}^{M_g} w_i = 1 \tag{2.17}$$

The complete Gaussian mixture density is parameterized by the mean vectors (μ_i), the covariance matrices (Σi) and the mixture weights (w_i) from all component densities. These parameters are collectively represented by

$$S = \begin{bmatrix} w_i, \mu_i, \Sigma_i \end{bmatrix} \quad i = 1, \ldots, M_g \tag{2.18}$$

There are several variants on the GMM shown in equation **??**. The covariance matrices can be full or diagonal. The parameters can be shared, or tied, among the Gaussian components, such as having a common covariance matrix for all components. The choice of model configuration depends on the amount of data used to train the model and the particular recognition task to be perform. The component Gaussian are acting together to model the overall feature density, full covariance matrices are not necessary even if the features are not statistically independent. The linear combination of diagonal covariance basis Gaussians are capable of modeling the correlations between feature vector elements. Thus, in general people use diagonal elements of the covariance matrix for speaker and language recognition task.

2.4.1 Training of GMMs

Given training feature vectors and a configuration of GMM, we wish to estimate the parameters of the GMM 'S'. The parameters of the GMM should be estimated in such a way, that the models are best matches with the distribution of training feature vectors. There are several techniques available for estimating the parameters of GMM. In general people use maximum likelihood (ML) estimation to find the model parameter. The objective of the ML estimator, is to maximize the likelihood of the training data with respect to the model parameters. If assume a sequence of independent training vectors $\mathrm{X} = \{\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_\mathrm{T}\}$, the likelihood of the GMM is given as,

$$\begin{aligned}
\mathrm{L(S|X)} &= \prod_{\mathrm{j=1}}^{\mathrm{T}} \mathrm{P(S|X_j)} \\
&= \prod_{\mathrm{j=1}}^{\mathrm{T}} \frac{\mathrm{P(X_j|S)P(S)}}{\mathrm{P(X)}} \\
&\approx \max_{\mathrm{S}} \quad \prod_{\mathrm{j=1}}^{\mathrm{T}} \mathrm{P(X_j|S)} \quad \mathrm{(ML \quad estimation)} \\
&\approx \max_{\mathrm{S}} \quad \mathrm{L(X|S)} \\
&\approx \max_{\mathrm{w_i},\mu_\mathrm{i},\Sigma_\mathrm{i}} \quad \prod_{\mathrm{j=1}}^{\mathrm{T}} \sum_{\mathrm{i=1}}^{\mathrm{M_g}} \mathrm{w_i P(X_j|}\theta_\mathrm{i}) \\
&\approx \max_{\mathrm{w_i},\mu_\mathrm{i},\Sigma_\mathrm{i}} \quad \sum_{\mathrm{j=1}}^{\mathrm{T}} \ln \sum_{\mathrm{i=1}}^{\mathrm{M_g}} \mathrm{w_i P(X_j|}\theta_\mathrm{i})
\end{aligned} \tag{2.19}$$

Unfortunately, the expression in equation **??** is nonlinear (due to the presence of log operation over sum operation). Therefore, the parameters can't be directly estimated using ML, a special case of iterative ML estimation called expectation maximization (EM) is generally used to estimate the parameters. The block diagram of GMM training is shown in figure **??**. The basic idea of EM is, begin with an initial model 'S', and estimate a new model '$\tilde{\mathrm{S}}$' such that $\mathrm{P(X|\tilde{S})} \geq \mathrm{P(X|S)}$. The new model become the initial model for the next iteration and the process is repeated until some convergence.

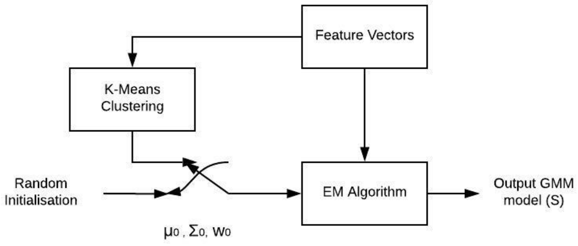

FIGURE 2.5: Basic block diagram of GMM training.

2.4.1.1 Expectation Maximization Algorithm

We have to maximize the likelihood $L(X|S)$ with respect to the model parameters $S = \{w_i, \mu_i, \Sigma_i\}$, then the partial derivative with respect to μ_i and Σ_i can be written as:

$$\begin{aligned}\frac{\partial L(X|S)}{\partial \mu_i} &= \frac{\sum_{j=1}^{T} w_i P(x_j|\theta_i)\left(\frac{x_j-\mu_i}{\Sigma_i}\right)}{\sum_{i=1}^{M_g} w_i P(x_j|\theta_i)} \\ \frac{\partial L(X|S)}{\partial \Sigma_i} &= \frac{\sum_{j=1}^{T} w_i P(x_j|\theta_i)\left(\frac{\Sigma_i^{-1}(x_j-\mu_i)(x_j-\mu_i)^T\Sigma_i^{-1}}{2} - \frac{1}{2\Sigma_i}\right)}{\sum_{i=1}^{M_g} w_i P(x_j|\theta_i)}, \quad i = 1, 2, \ldots, M_g\end{aligned} \tag{2.20}$$

Equating the partial derivatives in the equation **??** to zero and by assuming,

$$\gamma_{ij} = \frac{w_i P(x_j|\theta_i)}{\sum_{i=1}^{M_g} w_i P(x_j|\theta_i)} \tag{2.21}$$

the μ_i and Σ_i can be written as:

$$\begin{aligned}\mu_i &= \frac{\sum_{j=1}^{T} \gamma_{ij} x_j}{\sum_{j=1}^{T} \gamma_{ij}} \\ \Sigma_i &= \frac{\sum_{j=1}^{T} \gamma_{ij}(x_j - \mu_i)(x_j - \mu_i)^T}{\sum_{j=1}^{T} \gamma_{ij}}, \quad i = 1, 2, \ldots, M_g\end{aligned} \tag{2.22}$$

There is a constraint of $\sum_{i=1}^{M_g} w_i = 1$, so to find the optimal value, we have to use Lagrange constraint optimization, the modified objective function can be written as:

$$\max_{w_i} \quad L(X|S) + \eta\left(\sum_{i=1}^{M_g} w_i - 1\right) \tag{2.23}$$

The partial derivative of equation **??** with respect to w_i can be written as:

$$\begin{aligned}\frac{\partial\left(L(X|S) + \eta\left(\sum_{i=1}^{M_g} w_i - 1\right)\right)}{\partial w_i} &= \sum_{j=1}^{T} \frac{P(x_j|\theta_i)}{\sum_{i=1}^{M_g} w_i P(x_j|\theta_i)} \\ &= \sum_{j=1}^{T} \frac{\gamma_{ij}}{w_i} + \eta\end{aligned} \tag{2.24}$$

Equating the partial derivatives in the equation **??** to zero,

$$\begin{aligned}\sum_{\mathrm{j=1}}^{\mathrm{T}}\gamma_{\mathrm{ij}} &= -\,\eta(\mathrm{w_i})\\ -\eta(\mathrm{w_1}+\mathrm{w_2}+\ldots+\mathrm{w_{M_g}}) &= \sum_{\mathrm{i=1}}^{\mathrm{M_g}}\sum_{\mathrm{j=1}}^{\mathrm{T}}\gamma_{\mathrm{ij}}\\ -\sum_{\mathrm{i=1}}^{\mathrm{M_g}}\sum_{\mathrm{j=1}}^{\mathrm{T}}\gamma_{\mathrm{ij}} &= \eta \qquad \left(\text{as} \quad \sum_{\mathrm{i=1}}^{\mathrm{M_g}}\mathrm{w_i}=1\right)\end{aligned} \tag{2.25}$$

From the definition of γ_{ij} (as per equation **??**) we can say, γ_{ij} is the probability of $\mathrm{j^{th}}$ vector corresponds to $\mathrm{i^{th}}$ Gaussian. So, $\sum_{\mathrm{i=1}}^{\mathrm{M_g}}\gamma_{\mathrm{ij}}=1$, and $\sum_{\mathrm{j=1}}^{\mathrm{T}}\sum_{\mathrm{i=1}}^{\mathrm{M_g}}\gamma_{\mathrm{ij}}=\mathrm{T}$.

From equation **??**, we get $\eta=-\mathrm{T}$, and $\mathrm{w_i}$ can be written as:

$$\mathrm{w_i}=\frac{1}{\mathrm{T}}\sum_{\mathrm{j=1}}^{\mathrm{T}}\gamma_{\mathrm{ij}} \tag{2.26}$$

Putting all together, we can write the estimated values of $\mathrm{w_i}, \mu_{\mathrm{i}}$, and Σ_{i} as:

$$\begin{aligned} \mu_{\mathrm{i}} &= \frac{\sum_{\mathrm{j=1}}^{\mathrm{T}}\gamma_{\mathrm{ij}}\mathrm{x_j}}{\sum_{\mathrm{j=1}}^{\mathrm{T}}\gamma_{\mathrm{ij}}}\\ \Sigma_{\mathrm{i}} &= \frac{\sum_{\mathrm{j=1}}^{\mathrm{T}}\gamma_{\mathrm{ij}}(\mathrm{x_j}-\mu_{\mathrm{i}})(\mathrm{x_j}-\mu_{\mathrm{i}})^{\mathrm{T}}}{\sum_{\mathrm{j=1}}^{\mathrm{T}}\gamma_{\mathrm{ij}}}\\ \mathrm{w_i} &= \frac{1}{\mathrm{T}}\sum_{\mathrm{j=1}}^{\mathrm{T}}\gamma_{\mathrm{ij}}, \quad \mathrm{i}=1,2,\ldots,\mathrm{M_g}\\ \text{where,}\quad \gamma_{\mathrm{ij}} &= \frac{\mathrm{w_i P(x_j|\theta_i)}}{\sum_{\mathrm{i=1}}^{\mathrm{M_g}}\mathrm{w_i P(x_j|\theta_i)}} \end{aligned} \tag{2.27}$$

From the above equation **??**, It is worth to note that the results don't constitute a close form solution. The parameters of the GMM depends on γ_{ij} value, which again depends on the model parameters. This suggests, solve for a analytic solution is not possible. The solution can be obtain by using a simple iterative procedure. Which is well known as EM algorithm. We first choose some initial values for the means, covariances, and weights. Then by considering the initial values, the γ_{ij} value is computed (known as expectation step) and then using the γ_{ij} value the means, covariances, and weights values are computed (known as maximization step). The equation **??** can be modified as:

$$
\begin{aligned}
\gamma_{\mathrm{ij}}^{\mathrm{k+1}} &= \frac{\mathrm{w}_{\mathrm{i}}^{\mathrm{k}} \mathrm{P}(\mathrm{x_j}|\theta_{\mathrm{i}}^{\mathrm{k}})}{\sum_{\mathrm{i=1}}^{\mathrm{M_g}} \mathrm{w}_{\mathrm{i}}^{\mathrm{k}} \mathrm{P}(\mathrm{x_j}|\theta_{\mathrm{i}}^{\mathrm{k}})} \\
\mu_{\mathrm{i}}^{\mathrm{k+1}} &= \frac{\sum_{\mathrm{j=1}}^{\mathrm{T}} \gamma_{\mathrm{ij}}^{\mathrm{k+1}} \mathrm{x_j}}{\sum_{\mathrm{j=1}}^{\mathrm{T}} \gamma_{\mathrm{ij}}^{\mathrm{k+1}}} \\
\Sigma_{\mathrm{i}}^{\mathrm{k+1}} &= \frac{\sum_{\mathrm{j=1}}^{\mathrm{T}} \gamma_{\mathrm{ij}}^{\mathrm{k+1}} (\mathrm{x_j} - \mu_{\mathrm{i}}^{\mathrm{k+1}})(\mathrm{x_j} - \mu_{\mathrm{i}}^{\mathrm{k+1}})^{\mathrm{T}}}{\sum_{\mathrm{j=1}}^{\mathrm{T}} \gamma_{\mathrm{ij}}^{\mathrm{k+1}}} \\
\mathrm{w}_{\mathrm{i}}^{\mathrm{k+1}} &= \frac{1}{\mathrm{T}} \sum_{\mathrm{j=1}}^{\mathrm{T}} \gamma_{\mathrm{ij}}^{\mathrm{k+1}}, \quad \mathrm{i} = 1, 2, \ldots, \mathrm{M_g}
\end{aligned}
\tag{2.28}
$$

The process continues until $\mathrm{L}(\mathrm{X}|\mathrm{S}^{\mathrm{k+1}}) \geq \mathrm{L}(\mathrm{X}|\mathrm{S}^{\mathrm{k}})$. In general the EM algorithm takes many more iteration for convergence, thus K-means algorithm is used to initialize the model parameters. For each observation X, if the γ_{ij} value is given, than its known as complete data, in such cases the analytic solution can be obtained using equation **??**. But, in practice only observations are given known as incomplete data. In such situation, EM algorithm is used to estimate the model parameters.

2.4.1.2 Maximum a posteriori (MAP) Parameter estimation

The parameters of the GMM can also be estimated using MAP estimation. This approach is generally used in the pattern recognition tasks where the available labeled training data is limited. In speaker and language recognition task this approach is used to derive the adapted speaker/language model by adapting the training vectors with the universal background model (UBM) [**?**].

Like EM algorithm, the MAP estimation is also a two step process.

1. Step 1: The sufficient statistics of the training data are computed for each mixture in the prior model (The model computed using ML estimation).
2. Step 2: The new estimated sufficient statistics are combined with the old sufficient statistics sufficient statistics from the prior mixture parameters using a data-dependent mixing coefficient.

The data-dependent mixing coefficient is designed so that mixtures with high counts of new data rely more on the new sufficient for final parameter estimation and mixtures with low counts of new data rely more on the old sufficient statistics for final parameter estimation. The prior model is a large GMM (known as UBM) that is trained using ML estimation with large amount of data which encompasses the different kinds of speech that may be encountered by the system during training. These different kinds may include different channel conditions, composition of speakers/languages, acoustic conditions, etc.

Given a prior model ($\mathrm{S^{prior}}$) and training vectors from the desired class ($\mathrm{X} = \{\mathrm{x_1, x_2, \ldots, x_T}\}$), the sufficient statistics with respect to the prior model can be computed using equation **??**.

$$\begin{aligned}
\gamma_{\mathrm{ij}} =& \frac{\mathrm{w_i^{prior} P(x_j|\theta_i^{prior})}}{\sum_{\mathrm{i=1}}^{\mathrm{M_g}} \mathrm{w_i^{prior} P(x_j|\theta_i^{prior})}} \\
\mathrm{n_i} =& \sum_{\mathrm{j=1}}^{\mathrm{T}} \gamma_{\mathrm{ij}} \\
\mathrm{E_i(X)} =& \frac{1}{\mathrm{n_i}} \sum_{\mathrm{j=1}}^{\mathrm{T}} \gamma_{\mathrm{ij}} \mathrm{x_j} \\
\mathrm{E_i(X^2)} =& \frac{1}{\mathrm{n_i}} \sum_{\mathrm{j=1}}^{\mathrm{T}} \gamma_{\mathrm{ij}} \ \mathrm{Diag(x_j x_j^T)}
\end{aligned} \tag{2.29}$$

The new sufficient statistics computed from the training data then used to update the prior sufficient statistics for mixture i to create the adapted parameters for mixture i by using the equation **??**

$$\begin{aligned}
\hat{\mathrm{w}}_{\mathrm{i}} =& \left[\frac{\alpha_{\mathrm{i}}^{\mathrm{w}} \mathrm{n_i}}{\mathrm{T}} + (1 - \alpha_{\mathrm{i}}^{\mathrm{w}}) \mathrm{w_i} \right] \Gamma \\
\hat{\mu}_{\mathrm{i}} =& \alpha_{\mathrm{i}}^{\mathrm{m}} \mathrm{E_i(x_t)} + (1 - \alpha_{\mathrm{i}}^{\mathrm{m}}) \mu_{\mathrm{i}} \\
\hat{\Sigma}_{\mathrm{i}} =& \alpha_{\mathrm{i}}^{\mathrm{v}} \mathrm{E_i(x_t^2)} + (1 - \alpha_{\mathrm{i}}^{\mathrm{v}})(\mathrm{Diag}\ (\Sigma_{\mathrm{i}}) + \mu_{\mathrm{i}}^2) - \hat{\mu}_{\mathrm{i}}{}^2
\end{aligned} \tag{2.30}$$

A scale factor Γ is used, which ensures that all the new mixture weights sum to 1. $\alpha_{\mathrm{i}}^{\mathrm{w}}, \alpha_{\mathrm{i}}^{\mathrm{m}}$ and $\alpha_{\mathrm{i}}^{\mathrm{v}}$ are the adaptation coefficient which controls the balance between the old and new model parameter estimates. α_{i} is defined as:

$$\alpha_{\mathrm{i}}^{\mathrm{w}} = \alpha_{\mathrm{i}}^{\mathrm{m}} = \alpha_{\mathrm{i}}^{\mathrm{v}} = \alpha_{\mathrm{i}} = \frac{\mathrm{n_i}}{\mathrm{n_i + r}} \tag{2.31}$$

where r is a fixed relevance factor, which determines the extent of mixing of the old and new estimates of the parameters. If a mixture component has a low probabilistic count ($\mathrm{n_i}$) of new data, then $\alpha_{\mathrm{i}} \Rightarrow 0$ causing the de-emphasis of the new (potentially under-trained) parameters and the emphasis of the old (better trained) parameters. For mixture components with high probabilistic counts, $\alpha_{\mathrm{i}} \Rightarrow 1$ causing the use of the new class-dependent parameters. The pictorial representation of the same is presented in figure **??**. The relevance factor is a way of controlling how much new data should be observed in a mixture before the new parameters begin replacing the old parameters. Thus, this approach is robust to limited training data.

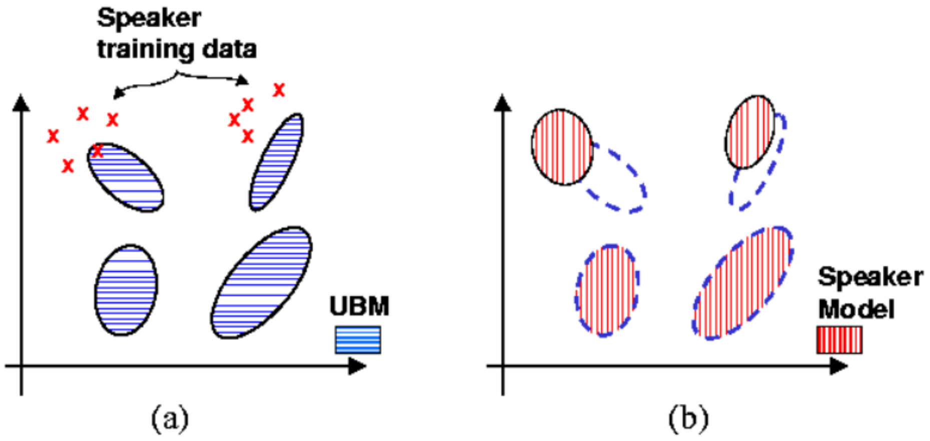

Figure 2.6: (a) The training vectors (x's) are probabilistically mapped into the UBM (prior) mixtures. (b) The adapted mixture parameters are derived using the statistics of the new data and the UBM (prior) mixture parameters. The adaptation is data dependent, so UBM (prior) mixture parameters are adapted by different amounts [?].

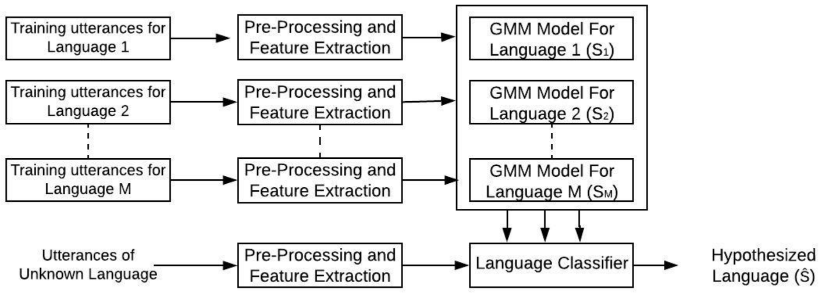
Training utterances for Language 1
Training utterances for Language 2
Training utterances for Language M
Pre-Processing and Feature Extraction
Pre-Processing and Feature Extraction
Pre-Processing and Feature Extraction
GMM Model For Language 1 (S1)
GMM Model For Language 2 (S2)
GMM Model For Language M (SM)
Utterances of Unknown Language
Pre-Processing and Feature Extraction
Language Classifier
Hypothesized Language (Ŝ)

Figure 2.7: Basic block diagram of language identification using GMM.

2.4.2 Testing of GMMs

In the testing phase the objective is to identify the hypothesized model from a set of models $\{\mathrm{S_1, S_2, ..., S_M}\}$ given a set of testing vectors $\mathrm{X} = \{\mathrm{x_1, x_2, .., x_T}\}$. The identified model $(\hat{\mathrm{S}})$ can be evaluated using equation ??.

$$\begin{aligned}\hat{\mathrm{S}} &= \arg \max_{1\leq i\leq M} \mathrm{P(S_i|X)} \\ &= \arg \max_{1\leq i\leq M} \frac{\mathrm{P(X|S_i)}}{\mathrm{P(X)}}\mathrm{P(S_i)}\end{aligned} \tag{2.32}$$

Using ML detection criteria (i.e Assuming equal probability occurrence of all the models) and the statistical independence of the testing vectors, the decision rule for the most probable model can be redefined as:

$$\hat{\mathrm{S}} = \arg \max_{1\leq i\leq M} \sum_{j=1}^{T} \log(\mathrm{P(x_j|S_i)}) \tag{2.33}$$

The basic block diagram of Language identification task using GMM based training and testing are presented in figure ??.

2.5 Gaussian Mixture Model and Universal Background Model (GMM-UBM)

The GMM-UBM approach is a vary successful technique for speaker recognition applications. Later by inspiring from the performance of GMM-UBM based speaker recognition systems, people start using GMM-UBM technique in language identification task. After GMM-UBM technique dominantly used in language identification task.

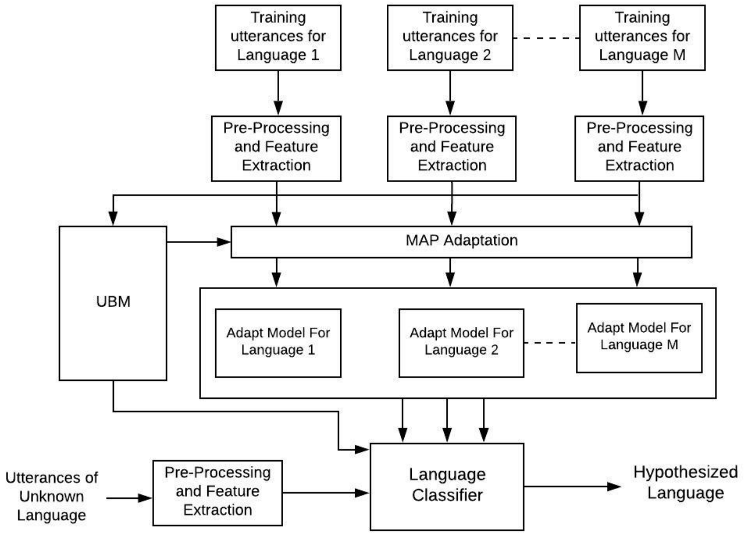
Training utterances for Language 1
Training utterances for Language 2
Training utterances for Language M
Pre-Processing and Feature Extraction
Pre-Processing and Feature Extraction
Pre-Processing and Feature Extraction
UBM
MAP Adaptation
Adapt Model For Language 1
Adapt Model For Language 2
Adapt Model For Language M
Utterances of Unknown Language
Pre-Processing and Feature Extraction
Language Classifier
Hypothesized Language

Figure 2.8: Basic block diagram of language identification using GMM-UBM.

The basic block diagram of GMM-UBM based language identification system is shown in figure ??. The training phase of GMM-UBM technique is divided into two distinct stages. First a set of feature vector from a number of different languages (typically data from all possible languages used for testing) are taken to train a single GMM (known as UBM). The UBM is considered to represent the characteristics of all different languages. In the second stage the UBM is adapted for each of the languages in the system using MAP adaptation. The detail of MAP adaptation training procedure is described in section ??. The idea behind MAP adaptation is that, the parameters of the Gaussian which show higher probability towards the language training data will tends towards the parameters of the training data, and the parameters of the mixture show lower probability to the language training data will remain fairly close to the original UBM. The MAP adaptation of the GMM parameters is often applied to the means of the mixture components rather than means, variances and weights (i.e. $\alpha^{\mathrm{v}}, \alpha^{\mathrm{w}} = 0$) [?].

In the testing phase, given a set of testing vectors $\{x_1, x_2, \ldots, x_T\}$, UBM model S^{ubm} and the adapt models of each language $\{S_i^{adapt}\}_{i=1}^{M}$, the identified language model can be evaluated using equation **??**. Where M is the total no of languages used in the system.

$$\hat{S} = \arg \max_{1 \le i \le M} \sum_{j=1}^{T} \left[\log(P(x_j|S_i^{adapt})) - \log(P(x_j|S^{ubm}) \right] \tag{2.34}$$

2.6 I-vector based language identification system

Language recognition and speaker recognition has many similarities in terms of technical formulation. Thus, in literature most of the descriptors and modeling techniques used to perform language identification task are borrowed from speaker recognition. In case of speaker recognition, in GMM-UBM based statistical modeling, the MAP adaptation not only adapt the speaker specific information but also adapts the channel and session variations. Thus to improve the speaker recognition performance, needs a modeling technique which can model the speaker specific information and other variability separately or to model all the information, suppress the other variability and retain the speaker specific information. In [**?**], the authors propose a method to model all the variations separately. They represented a speaker utterance by a supervector (M), that consists of additive components of speaker and channel/session subspace. This technique is well known as joint factor analysis (JFA). The speaker dependent supervector is defined as:

$$M = m + Vy + Ux + Dz \tag{2.35}$$

where m is a speaker and session independent supervector (generally mean vector of UBM having $M_g D \times 1$ dimension). V and D define a speaker subspace (eigenvoice matrix and diagonal residual respectively), and U defines a session or channel subspace (eigen channel matrix). The vector y, z and x are the speaker and channel/session dependent factors in their respective subspace and each is assumed to be a random variable with a standard normal distribution $(N(0, I))$. In JFA, first the subspaces (i.e V, D and U) have to be estimated from the appropriate labeled corpora, than the speaker and session factors (i.e y, z and x) have to be computed for the target utterances. The scoring is done by computing the likelihood of the test utterance vectors against the session compensated speaker model $(M - Ux)$. In [**?**], the authors slightly modify the modeling technique to reduce the computation and enhance the recognition performance. The authors proposed a single subspace modeling instead of two subspace (speaker and channel separately). This single subspace is known as total variability subspace. The total variability subspace contains both the speaker and channel variability information. The new speaker and channel dependent supervector can be defined as:

$$\mathrm{M} = \mathrm{m} + \mathrm{Tw} \tag{2.36}$$

where T is the total variability subspace and w is the speaker and channel dependent vector (known as identity vector/ i-vector). T is a rectangular matrix of low rank and w is a vector having standard normal distribution. After i-vector extraction some channel compensation techniques (like linear discriminative analysis (LDA), within class covariance normalization (WCCN) and nuisance attribute projection (NCA)) are used to suppress the channel variability. A cosine kernel scoring technique is used to find the recognition performance. The block diagram of i-vector based language recognition system is shown in figure **??**.

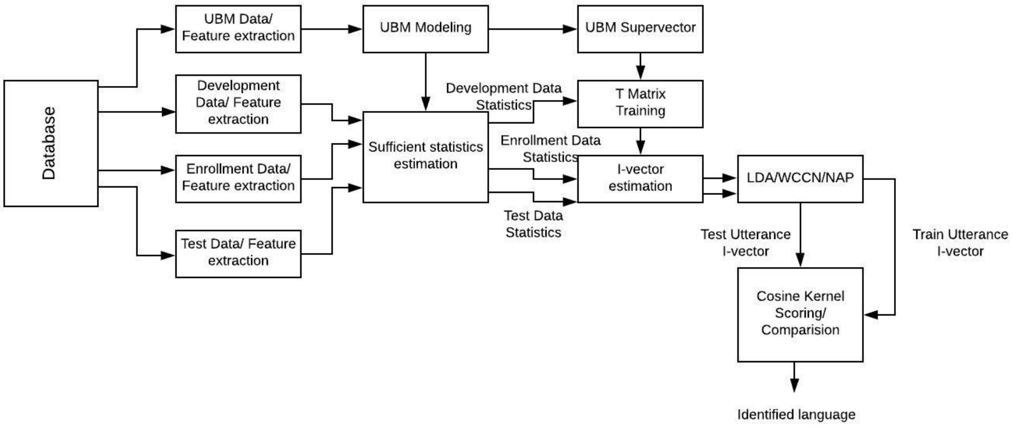
UBM Data/ Feature extraction
UBM Modeling
UBM Supervector
Database
Development Data/ Feature extraction
Development Data Statistics
T Matrix Training
Sufficient statistics estimation
Enrollment Data Statistics
Enrollment Data/ Feature extraction
I-vector estimation
LDA/WCCN/NAP
Test Data Statistics
Test Data/ Feature extraction
Test Utterance I-vector
Train Utterance I-vector
Cosine Kernel Scoring/ Comparision
Identified language

Figure 2.9: Basic block diagram of language identification using i-vector.

From the figure **??**, it has been seen that, the whole dataset is segmented into four parts: UBM data, development data, enrollment data, test data. UBM data consists of data of all the languages, used to build UBM models as described in section **??**. The development data segment also consists of utterances from all the languages, used to estimate the total variability matrix. After the estimation of total variability matrix (T matrix), the hidden variable w, known as i-vector is estimated from each utterance of the enrollment and test data. The i-vector w, which can be defined by its posterior distribution conditioned to the Baum-Welch statistics for a given utterance. The posterior distribution is a Gaussian distribution and the Baum-Welch sufficient statistics will be estimated from the UBM. Suppose we have T frames $\{\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_\mathrm{T}\}$ of an utterance and the UBM composed of $\mathrm{M_g}$ mixture components defined in some feature space of dimension D. The Sufficient statistics for a given utterance is given by:

$$\begin{aligned} \mathrm{N_i} =& \sum_{\mathrm{j=1}}^{\mathrm{T}} \gamma_{\mathrm{ij}} \quad \text{(zeroth order statistics)} \\ \mathrm{F_i} =& \sum_{\mathrm{j=1}}^{\mathrm{T}} \gamma_{\mathrm{ij}} \mathrm{x_j} \quad \text{(first order statistics)} \\ \mathrm{S_i} =& \mathrm{diag}\left[\sum_{\mathrm{j=1}}^{\mathrm{T}} \gamma_{\mathrm{ij}} \mathrm{x_j} \mathrm{x_j^*}\right], \quad \mathrm{i} = 1, 2, \ldots, \mathrm{M_g} \quad \text{(second order statistics)} \end{aligned} \tag{2.37}$$

where $\mathrm{x_j^*}$ is the Hermitian transpose of vector $\mathrm{x_j}$ and $\gamma_{\mathrm{ij}} = \frac{\mathrm{w_i P(x_j|\theta_i)}}{\sum_{\mathrm{i=1}}^{\mathrm{M_g}} \mathrm{w_i P(x_j|\theta_i)}}$ (from equation **??**). In order to estimate the i-vector, we need to centralize the first and second order sufficient statistics. The centralized statistics is given by:

$$\begin{aligned} \tilde{\mathrm{F}}_{\mathrm{i}} =& \sum_{\mathrm{j=1}}^{\mathrm{T}} \gamma_{\mathrm{ij}} (\mathrm{x_j} - \mathrm{m_i}) \\ =& \sum_{\mathrm{j=1}}^{\mathrm{T}} \gamma_{\mathrm{ij}} \mathrm{x_j} - \sum_{\mathrm{j=1}}^{\mathrm{T}} \gamma_{\mathrm{ij}} \mathrm{m_i} \\ =& \mathrm{F_i} - \mathrm{N_i m_i} \\ \tilde{\mathrm{S}}_{\mathrm{i}} =& \mathrm{diag}\left[\sum_{\mathrm{j=1}}^{\mathrm{T}} \gamma_{\mathrm{ij}} (\mathrm{x_j} - \mathrm{m_i})(\mathrm{x_j} - \mathrm{m_i})^*\right], \quad \mathrm{i} = 1, 2, \ldots, \mathrm{M_g} \\ =& \mathrm{S_i} - \mathrm{diag}(\mathrm{F_i m_i^*} + \mathrm{m_i F_i^*} - \mathrm{N_i m_i m_i^*}) \end{aligned} \tag{2.38}$$

where $\mathrm{m_i}$ is the mean vector of the $\mathrm{i^{th}}$ mixture. We can write the statistics in the matrix format as:

$$\begin{aligned} \mathrm{N_m} =& \begin{bmatrix} \mathrm{N_1} * \mathrm{I} & & \\ & \ddots & \\ & & \mathrm{N_{M_g}} * \mathrm{I} \end{bmatrix} \\ \mathrm{F_m} =& \begin{bmatrix} \tilde{\mathrm{F}}_1 \\ \vdots \\ \tilde{\mathrm{F}}_{\mathrm{M_g}} \end{bmatrix} \\ \mathrm{S_m} =& \begin{bmatrix} \tilde{\mathrm{S}}_1 & & \\ & \ddots & \\ & & \tilde{\mathrm{S}}_{\mathrm{M_g}} \end{bmatrix} \end{aligned} \tag{2.39}$$

where $\mathrm{N_m}$ is a matrix of dimension $\mathrm{M_gD} \times \mathrm{M_gD}$, I is a identity matrix of dimension $\mathrm{D} \times \mathrm{D}$, $\mathrm{F_m}$ is a vector of dimension $\mathrm{M_gD} \times 1$ and $\mathrm{S_m}$ is a matrix of dimension $\mathrm{M_gD} \times \mathrm{M_gD}$. In [**?**] from Theorem 2 we can write the distribution of the hidden variable (w) is $\mathrm{N}(\mathrm{L_T^{-1}} * \mathrm{T^*} * \Sigma^{-1} * \mathrm{F_m}, \mathrm{L_T^{-1}})$. The i-vector can be computed as $\mathrm{E(w)} = \mathrm{L_T^{-1}} * \mathrm{T^*} * \Sigma^{-1} * \mathrm{F_m}$. where E(w) is the expectation of vector w, Σ is the variance of the UBM, $L_T = I + T^*\Sigma^{-1}N_mT$ and T is the total variability matrix.

2.6.1 T Matrix training

1. Step 1: Initialize the T matrix randomly having dimension $M_gD \times \mathrm{Dimension\ of\ i-vector}$
2. Step 2: Compute: $L_T = I + T^*\Sigma^{-1} * N_m * T$
3. Step 3: Accumulate the statistics across utterances:

$$\begin{aligned} {}^{a}N_i &= \sum_u N_i(u) \\ {}^{a}A_i &= \sum_u N_i(u)L_T^{-1} \\ {}^{a}C &= \sum_u F_m(u)(L_T^{-1}(u) * T^* * \Sigma^{-1} * F_m(u)) \\ {}^{a}N_m &= \sum_u N_m(u) \end{aligned} \tag{2.40}$$

4. Step 4: Compute T matrix:

$$T = \begin{bmatrix} T_1 \\ \vdots \\ T_{M_g} \end{bmatrix} = \begin{bmatrix} A_1^{-1} * C_1 \\ \vdots \\ A_{M_g^{-1} * C_{M_g}} \end{bmatrix}, \quad \text{where} \quad C = \begin{bmatrix} C_1 \\ \vdots \\ C_{M_g} \end{bmatrix} \tag{2.41}$$

5. Step 5: Compute covariance update (optional):

$$\Sigma = ({}^{a}N_m)^{-1}\left[\sum_u S_m(u) - \mathrm{diag}(C * T^*)\right] \tag{2.42}$$

6. Step 6: Iterate step 2 to step 4 (or step 5) approximately 20 times to get the estimate of the T matrix.

In the literature, people have not used the second order statistics(S_m) for speaker and language recognition task, in such cases step 5 is not required.

2.6.2 I-vector estimation

After the estimation of T matrix the i-vectors from each utterances of the enrollment set and test set is computed using equation ??.

$$w(u) = L_T^{-1}(u) * T^* * \Sigma^{-1} * F_m(u) \tag{2.43}$$

2.6.3 Cosine kernel scoring

Cosine kernel scoring is technique to find the cosine kernel between the train language i-vectors and test i-vector. Suppose there are L languages having each $\mathrm{K_1, K_2, \ldots, K_L}$ utterances and lets denote the train language i-vector as $\mathrm{w_{lk}}$ and the test utterance i-vector as $\mathrm{w_{test}}$. The identified language ($\mathrm{\hat{S}}$) can be written as

$$\mathrm{\hat{S} = arg \max_{1 \le l \le L} \sum_{k=1}^{K_l} Score_{lk} = arg \max_{1 \le l \le L} \sum_{k=1}^{K_l} \frac{< w_{lk}, w_{test} >}{||w_{lk}||||w_{test}||}} \tag{2.44}$$

2.6.4 Intersession compensation techniques

The intersession compensation techniques are used to enhance the inter-class variations and to suppress the intra-class variations. Generally there are three dominantly used intersession compensation technique to enhance the recognition performance, while the classes are modeled in total variability space. These three techniques are: linear discriminant analysis (LDA), within class covariance normalization (WCCN), nuisance attribute projection (NAP).

2.6.4.1 Linear discriminant analysis (LDA)

LDA is a technique of dimensionality reduction, widely used in the task of pattern recognition. The motivation of using LDA in i-vector based language recognition task is that, as all the utterances of a given language are assume to represent one class, LDA attempts to define new special axes that minimize the intra-class variance caused by channel and speaker effects, and to maximize the variance between languages. This approach searches a new orthogonal axes to better discriminate between classes. The LDA optimization problem can be defined according to the ratio given in equation **??**.

$$\mathrm{J(v) = \frac{v^t S_b v}{v^t S_w v}} \tag{2.45}$$

The ratio is often referred as the Rayleigh coefficient for space direction v. $\mathrm{S_b}$ and $\mathrm{S_w}$ are the between class and the within-class variance between matrix and are calculated as follows:

$$\begin{aligned} \mathrm{S_b} &= \mathrm{\sum_{l=1}^{L} (w_l - \bar{w})(w_l - \bar{w})^t} \\ \mathrm{S_w} &= \mathrm{\sum_{l=1}^{L} \frac{1}{K_L} \sum_{k=1}^{K_L} (w_k^l - \bar{w}_l)(w_k^l - \bar{w}_l)^t} \end{aligned} \tag{2.46}$$

where $\mathrm{\bar{w}_l = \frac{1}{K_L} \sum_{k=1}^{K_L} w_k^l}$ is the mean of i-vectors for each language. $\mathrm{\bar{w}}$ is the mean of i-vector for all the utterances. The maximization is used to define a projection matrix A composed by the best eigen vectors (those with highest eigen values) of the general eigen value equation:

$$\begin{aligned} \mathrm{S_b v} &= \lambda \mathrm{S_w v} \\ \mathrm{S_b S_w^{-1} v} &= \lambda \mathrm{v} \end{aligned} \tag{2.47}$$

where λ is the diagonal matrix of eigenvalues. The the projection matrix is obtained by taking the eigen vectors of the larger eigenvalues (i.e A = V(:, 1 : no of top eigen values)).The identified language ($\hat{\mathrm{S}}$) can be written as:

$$\hat{\mathrm{S}} = \arg \max_{1 \le l \le L} \sum_{k=1}^{K_l} \mathrm{Score}_{lk} = \arg \max_{1 \le l \le L} \sum_{k=1}^{K_l} \frac{< \mathrm{A^t w_{lk}}, \mathrm{A^t w_{test}} >}{||\mathrm{A^t w_{lk}}||||\mathrm{A^t w_{test}}||} \tag{2.48}$$

2.6.4.2 Within class covariance normalization (WCCN)

This technique was first introduced in [?]. In the study, they applied this approach in SVM modeling based on linear separation between target speaker and imposter using one verses all decision. WCCN techniques uses inverse of within-class covariance to normalize the cosine kernel. The resulting solution by a generalized linear kernel form can be written as given in [?]:

$$\mathrm{k(w_1, w_2)} = \mathrm{w_1^t R w_2} \tag{2.49}$$

where R is a symmetric positive semi-definite matrix. The optimal normalized kernel matrix is given by $\mathrm{R} = \mathrm{W}^{-1}$. W is a within class covariance matrix, calculated as:

$$\mathrm{W} = \frac{1}{\mathrm{L}} \sum_{l=1}^{\mathrm{L}} \frac{1}{\mathrm{K_L}} \sum_{k=1}^{\mathrm{K_L}} (\mathrm{w_k^l} - \bar{\mathrm{w}}_l)(\mathrm{w_k^l} - \bar{\mathrm{w}}_l)^t \tag{2.50}$$

In order to preserve the inner product form of the cosine kernel, a feature-mapping function φ can be defined as:

$$\varphi(\mathrm{w}) = \mathrm{B^t w} \tag{2.51}$$

where B is obtained through Cholesky decomposition of matrix $\mathrm{W}^{-1} = \mathrm{BB^t}$. The WCCN algorithm uses the within-class covariance matrix to normalize the cosine kernel function to normalize the cosine kernel functions in order to compensate for intersession variability, while guaranteeing conservation of directions in space. The identified language ($\hat{\mathrm{S}}$) can be written as:

$$\hat{\mathrm{S}} = \arg \max_{1 \le l \le L} \sum_{k=1}^{K_l} \mathrm{Score}_{lk} = \arg \max_{1 \le l \le L} \sum_{k=1}^{K_l} \frac{< \mathrm{B^t w_{lk}}, \mathrm{B^t w_{test}} >}{||\mathrm{B^t w_{lk}}||||\mathrm{B^t w_{test}}||} \tag{2.52}$$

2.6.4.3 Nuisance attribute projection (NAP)

The nuisance attribute projection algorithm is presented in [?]. It is based on finding an appropriate projection matrix intended to remove the nuisance direction. The projection matrix carries out an orthogonal projection in channel's and speaker's complimentary space, which depends only on language class information. The projection matrix is formulated as:

$$\mathrm{P} = \mathrm{I} - \mathrm{RR}^{\mathrm{t}} \tag{2.53}$$

where R is a rectangular matrix of low rank, whose columns are the eigen vectors ($\mathrm{Wr} = \lambda \mathrm{r}$) having the best eigen values of the within-class covariance matrix (W) computed as:

$$\mathrm{W} = \frac{1}{\mathrm{L}}\sum_{\mathrm{l}=1}^{\mathrm{L}}\frac{1}{\mathrm{K_L}}\sum_{\mathrm{k}=1}^{\mathrm{K_L}}(\mathrm{w}_\mathrm{k}^\mathrm{l} - \bar{\mathrm{w}}_\mathrm{l})(\mathrm{w}_\mathrm{k}^\mathrm{l} - \bar{\mathrm{w}}_\mathrm{l})^\mathrm{t} \tag{2.54}$$

The identified language ($\hat{\mathrm{S}}$) using NAP can be written as:

$$\hat{\mathrm{S}} = \arg\max_{1\leq \mathrm{l}\leq \mathrm{L}}\sum_{\mathrm{k}=1}^{\mathrm{K_l}}\mathrm{Score}_{\mathrm{lk}} = \arg\max_{1\leq \mathrm{l}\leq \mathrm{L}}\sum_{\mathrm{k}=1}^{\mathrm{K_l}}\frac{<\mathrm{P}^\mathrm{t}\mathrm{w}_{\mathrm{lk}}, \mathrm{P}^\mathrm{t}\mathrm{w}_{\mathrm{test}}>}{||\mathrm{P}^\mathrm{t}\mathrm{w}_{\mathrm{lk}}||||\mathrm{P}^\mathrm{t}\mathrm{w}_{\mathrm{test}}||} \tag{2.55}$$

2.7 Feedforward network based model

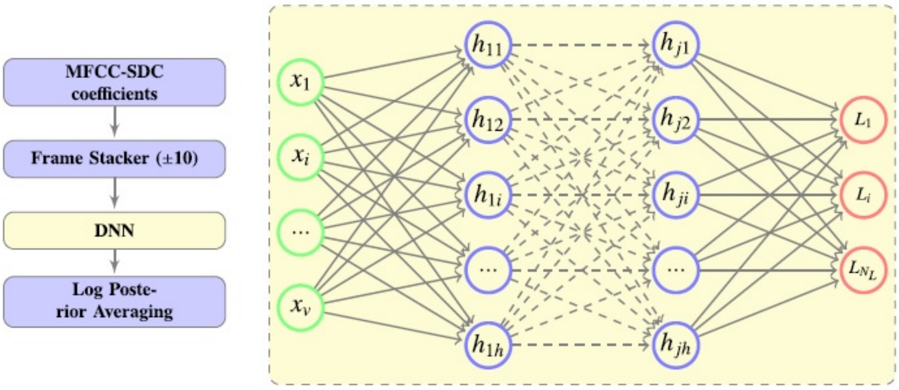

FIGURE 2.10: Feedforward network based classifier [?].

The impressive gain in performance obtained using deep neural networks (DNNs) for automatic speech recognition (ASR) have motivated the application of DNNs in other speech technologies such as language recognition and speaker recognition. Feedforward neural network is a fully connected network, which shows the direct use of DNN in the recognition task. In this approach, a fully connected network with multiple hidden layers is trained using back-propagation algorithm to perform recognition task. The output layer of this network has the number of neurons (nodes) same as the number of classes in the recognition task, whereas the input layer has the number of nodes same as the dimension of input feature vector (refer figure **??**). In general, the MFCC/SDC features are computed and then stacked with the feature vectors of the neighboring frames and then given input to the network. After training is over, testing is performed using the feature vectors of the speech utterances. For a test utterance, the output score is computed as:

$$\mathrm{score_l} = \frac{1}{\mathrm{N}} \sum_{\mathrm{t}=1}^{\mathrm{N}} \mathrm{logp}(\mathrm{L_l}|\mathrm{x_t}, \Theta) \tag{2.56}$$

where $\mathrm{p}(\mathrm{L_l}|\mathrm{x_t}, \Theta)$ represents the class probability output for language l corresponding to the input example at time t, $\mathrm{x_t}$ is the feature vector and Θ is the parameter of the DNN.

For a test utterance, the output node giving the highest score is termed as the recognition class.

2.8 Bottleneck feature I-vector model

DNN architectures can be used for feature extraction. From the human language recognition ability, it has been seen that both acoustic phonetic and phonotactic information are used for recognition. Thus people try to extract the features, which have both the information. This type of feature can be extracted by training a DNN, by providing the feature vector to the input and its corresponding posterior senone probabilities (computed from a DNN based ASR system) to the output of the DNN. After the network is trained, activation value of a hidden layer near to output layer can be taken as a feature vector (known as bottleneck feature (BNF)) by providing acoustic feature vectors as input(refer figure **??**). After the BNF feature is extracted, I-vector framework is used to identify the language(refer to figure **??**).

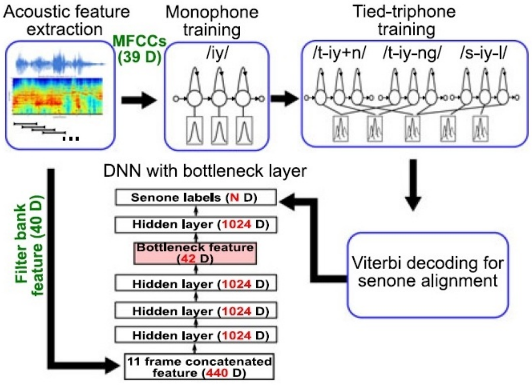

FIGURE 2.11: Bottleneck feature extraction [?].

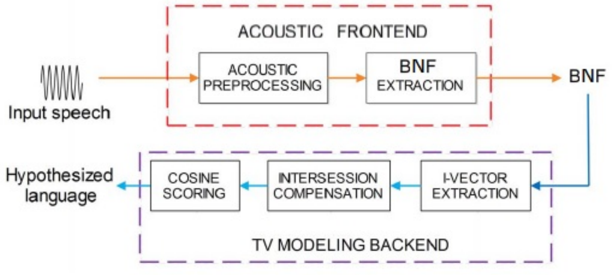

FIGURE 2.12: Bottleneck feature-I-vector based language identification system [?].

2.9 X-vector based modeling

In this approach, the X-vectors are extracted from a trained DNN and then used like I-vectors to perform recognition task. The DNN is trained to discriminate between languages/speakers. The X-vector is a fixed dimensional embedding extracted from the variable length utterance, using a trained time delay neural network (TDNN) based DNN. TDNN is a older approach proposed in [?] on 1989. TDNN works like convolutional neural network (CNN) work on the image to capture local information. The disadvantage of TDNN is, it requires a high amount of data and training time. Thus, in this approach, a subsampling method is used on TDNN to reduce the training time and computational complexity. After sub-sampling as the no of parameters to estimate is reduced, so the training data requirement also reduced. The sub-sampling structure of TDNN is presented in figure ??.

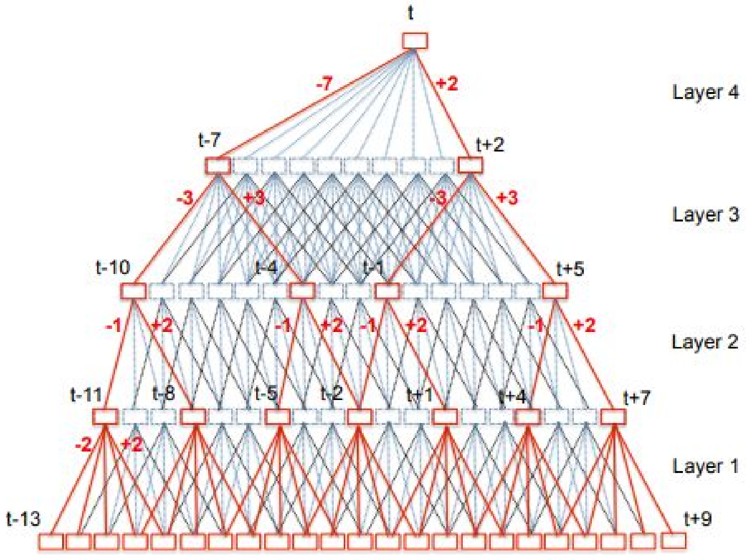

FIGURE 2.13: Computation in TDNN with sub-sampling (red) and without sub-sampling (blue+red) [?].

The architecture of the sub-sampling based TDNN is given in table ??. Suppose an input utterance have T feature vectors of F dimension. The input to the network is the concatenation

Table 2.1: The embedding DNN architecture [?].

Layer	Layer Context	Total Context	Input × Output
layer 1	[t-2,t+2]	5	5F × 512
layer 2	{t-2,t,t+2}	9	1536 × 512
layer 3	{t-3,t,t+3}	15	1536 × 512
layer 4	{t}	15	512 × 512
layer 5	{t}	15	512 × 1500
Stats Pooling	[0,T)	T	1500T × 3000
Segment 6	{0}	T	3000 × 512
Segment 7	{0}	T	512 × 512
Softmax	{0}	T	512 × L

of acoustic features of five frames (2 past, 2 future, 1 current), i.e. a $5 * F$ dimension vector. The first five layers operate on speech frames with a small temporal context centered at the current frame t. The statistical pooling layer aggregates all T frame-level outputs from layer 5 and computes it mean and standard deviation (1500 dimension), then concatenate the mean and standard deviation to form 3000 dimension vector (for the whole utterance) and give input to the next layer. The output layer is a softmax layer having the language labels. From segment 6 or segment 7, the embedding vectors are extracted, and then, PLDA and cosine kernel scoring techniques are used to perform language recognition.

Chapter 3

Delivered System Description

3.1 Database Description

The whole dataset is subdivided into two parts, i.e. training data and testing data. The training data consists of 11 different language speech data. These languages are American English, CREOL, Cantonese-Chinese, Farsi, Georgian, Korean, Mandarin, Russia, Spanish, Turkey, and Vietnamese. The amount of speech data available in each language is given in table **??**. The training data is extracted from multiple corpora, these are OGI-multilingual, NIST 1996, 2003, 2005 LREs, NIST 2004, 2005, 2006, 2008 speaker recognition evaluations (SREs) and NIST 2007 LRE supplementary training data. The test data are extracted from NIST 2009 Evaluation set, which has 3 evaluation conditions, i.e 30 sec, 10 sec and 3 sec.

Table 3.1: Speech data available in each language

Language name	**Approx. duration of speech data**
American English	26 hours
Creol	27 hours
Cantonese-Chinese	26 hours
Farsi	20 hours
Gorgian	26 hours
Korean	10 hours
Mandarin	15 hours
Russia	20 hours
Spanish	14 hours
Turkey	24 hours
Vietnamese	18 hours

3.2 System Description

3.2.1 GMM Based System

As discussed in section **??** of chapter 1, the same procedure is followed to develop the GMM based system. MATLAB environment is used to develop the system. First, the SDC feature vectors of 56 dimensions (including C_0, performed CMVN language-wise) each, are extracted from the speech signal, then the GMM of 512 clusters are used to model the feature vectors of each language. The performance obtained using all three testing conditions is tabulated in table ??.

Table 3.2: GMM based system performance

Performance Measure	30 sec	10 sec	3 sec
$\mathrm{EER_{avg}}$	32.1	39.2	43.3
$\mathrm{C_{avg}}$	30.5	37.9	42.1
Identification Accuracy	56	42.1	23

3.2.2 GMM-UBM Based System

As discussed in section ?? of chapter 1, the same procedure is followed to develop the GMM-UBM based system. MATLAB environment is used to develop the system. First, the SDC feature vectors of 56 dimensions (including C_0, performed CMVN language-wise) each, are extracted from the speech signal, then the UBM (cluster size 512) was built using the feature vectors of approx. 2-hour data of each language. Then the training feature vectors of each language were adapted with the UBM to obtain adapt Models for each language. The performance obtained using all three testing conditions is tabulated in table ??.

Table 3.3: GMM UBM based system performance

Performance Measure	30 sec	10 sec	3 sec
$\mathrm{EER_{avg}}$	28.3	32.5	39.9
$\mathrm{C_{avg}}$	27.9	32.1	38.8
Identification Accuracy	64	49.1	38.1

3.2.3 I-vector based System

As discussed in section ?? of chapter 1, the same procedure is followed to develop the I-vector based system. MATLAB environment is used to develop the system. First, the SDC feature vectors of 56 dimensions (including C_0, performed CMVN language-wise) each, are extracted from the speech signal, then the UBM (cluster size 512) was built using the feature vectors of approx. 2-hour data of each language. Then all the training feature vectors of each language were used to train the T-matrix, after that 400 dimension I-vectors for each utterance are extracted, LDA and WCCN are used to minimize within-class variance and maximize the between-class variance. The performance obtained using all three testing conditions is tabulated in table ??.

Table 3.4: I-vector based system performance

Performance Measure	30 sec	10 sec	3 sec
$\mathrm{EER_{avg}}$	24.2	29.2	33.3
$\mathrm{C_{avg}}$	23.9	28.9	32.6
Identification Accuracy	70.1	53	42

3.2.4 Feedforward Network Based System

As discussed in section **??** of chapter 1, the same procedure is followed to develop the Feedforward network-based system. Python environment with Keras and TensorFlow libraries was used to develop the system. First, the SDC feature vectors of 56 dimensions (including C_0, performed CMVN language-wise) each, are extracted from the speech signal, then contexting of previous 10 and future 10 frames were used to provide 1176 dimension input to the neural network. 5 hidden layers with 2048 neurons were used with the Relu activation function. The output layer had 11 neurons with the softmax activation function. The network was trained using stochastic gradient descent with 0.001 learning rate. The performance obtained using all three testing conditions is tabulated in table **??**.

Table 3.5: Feedforward neural network based system performance

Performance Measure	30 sec	10 sec	3 sec
EER_{avg}	25.3	30.2	36.4
C_{avg}	24.8	30.09	35.9
Identification Accuracy	67	48	39

3.2.5 X-vector Based System

As discussed in section **??** of chapter 1, the same procedure is followed to develop the X-vector based system. Kaldi speaker recognition 2016 recipe was used to develop the system. First, the MFCC feature vectors of 20 dimensions (including C_0, performed CMVN language-wise) each, are extracted from the speech signal. The architecture given in table **??** of section **??** was used with dropout of each hidden layer 0,0,0.0001,0.0001,0.0001,0 respectively. The network is trained with 40 epochs. After X-vector is extracted PLDA classifier was used for classification. The performance obtained using all three testing conditions are tabulated in table **??**.

Table 3.6: X-vector based system performance

Performance Measure	30 sec	10 sec	3 sec
EER_{avg}	18.2	23.1	30.1
C_{avg}	17.92	22.39	29.56
Identification Accuracy	79	61	51

3.3 Summary and Discussion

From the performance of all the systems, it has been observed that the X-vector system outperforms all the other systems. But still, the performance is far from perfect. The performance of the system degrades with the decrease in test utterance duration. Therefore in the future, there is a need for the development of efficient feature extraction and modeling techniques, which will provide a stable Spoken language recognition performance, even if the test duration is small.