Spoken Language Identification under Emotional Speech Variations

Thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science
in
Electronics and Communication Engineering by Research

by

Priyam Jain
201431198
priyamjain1527@gmail.com

International Institute of Information Technology
Hyderabad - 500 032, INDIA
November 2020
It is certified that the work contained in this thesis, titled “Spoken Language Identification under Emotional Speech Variations” by Priyam Jain, has been carried out under my supervision and is not submitted elsewhere for a degree.

12/11/2020

Adviser: Dr. Anil Kumar Vuppala
To Everyone
Acknowledgments

Our world is an ever changing place, research drives the advancement and developments of it. In the words of my adviser Dr. Anil Kumar Vuppula, "Research is not done in one day by a single person, it involves time, continuous effort and team work". His words gave me constant motivation to work hard along with his technical guidance. I want to offer my sincere gratitude towards him for providing me with this opportunity and to support me throughout these last 3 years. During this time, he always tried to have some kind of lab work assigned to me. The result of this was that I was able to get my publication right after the 3rd year. His constant suggestions have been one of the most important factors leading to this thesis.

I want to thank Krishna Gurugubelli, PHD scholar from my lab, for revising my papers, thesis and providing guidance throughout these years. He was involved with me right from the beginning, helped me learn the basics of speech signal processing. He taught me how to be a researcher, how to understand and appreciate a research work. His help in formulation of this thesis is immense. I also want to thank Ganesh Mirishkar and Tirusha Mandava for helping me learn the kaldi toolkit. A thankful mention has to be given for my other lab mates Rama Krishna sir, Ravi Kumar sir, V Raju, Javid and Ganga Mohan sir.

I want to thank my friend Nikita Chaturvedi, for her constant motivation and self belief that she brought in me. I also want to thank my friends Shreya Jain, Neeraj Battan, Allen Antony, Juniad Ali and Shreyash Shriyam for their support throughout my final year. Lastly, I want to mention the group (or a gang) that stayed with me throughout my stay at IIIT-Hyderabad (and beyond) and made it a fun and a memorable journey, though bumpy at times, the LSD-R. I am sorry I cannot name all you guys.

I am grateful to my parents, Manju Jain and Prabhat Jain, and my younger sister Priya for always believing in me and providing me with all the support. I want to thank them for providing me a privileged and a comforting background.
Identifying language information from speech utterance is referred to as spoken language identification. Language Identification (LID) is essential in multilingual speech systems. There are various conditions under which the performance of LID systems are sub-optimal, such as short duration, background noise, channel variation, and so on. There have been efforts to improve performance under these conditions, but the impact of speaker emotion variation on the performance of LID systems has not been studied. Hence in contrast to the previous studies, for the first time in the literature, the present work investigated the impact of emotional speech on language identification. In this work, different emotional speech databases have been pooled to create the experimental setup. A dataset of this kind was not available for LID, and is a contribution of this thesis. Additionally, state-of-art i-vectors, time-delay neural networks (TDNN), long short term memory (LSTM), and deep neural network (DNN) x-vector systems have been considered to build the LID systems. Performance of the LID system has been evaluated for speech utterances of different emotions in terms of equal error rate and $C_{avg}$. The results of the study indicate that the speech utterances of anger and happy emotions degrades performance of LID systems more compared to the neutral and sad emotions.

To that effect, we investigated adaptation approaches for improving the performance of LID systems by incorporating emotional utterances in the form of adaptation dataset. Hence, we studied a prosody modification technique called Flexible Analysis Synthesis Tool (FAST) to vary the emotional characteristics of an utterance in order to improve the performance, but the results were inconsistent and not satisfactory. Therefore, we propose a combination of Recurrent Convolutional Neural Network (RCNN) based architecture with multi stage training methodology, which outperformed state-of-art LID systems.

Keywords: TDNN, LSTM, DNN x-vector, Flexible Analysis Synthesis Tool, Recurrent Convolutional Neural Network, multistage training
**Contents**

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Language Identification</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Challenges involved with LID</td>
<td>2</td>
</tr>
<tr>
<td>1.3 Objective and scope</td>
<td>2</td>
</tr>
<tr>
<td>1.4 Thesis outline</td>
<td>3</td>
</tr>
<tr>
<td>2 Language Identification</td>
<td>4</td>
</tr>
<tr>
<td>2.1 How humans perform LID</td>
<td>4</td>
</tr>
<tr>
<td>2.2 Basic structure of LID systems</td>
<td>5</td>
</tr>
<tr>
<td>2.2.1 Frontend in LID: Feature Extraction</td>
<td>5</td>
</tr>
<tr>
<td>2.2.2 Backend in LID: Classifier</td>
<td>7</td>
</tr>
<tr>
<td>2.2.2.1 Explicit LID</td>
<td>7</td>
</tr>
<tr>
<td>2.2.2.2 Implicit LID</td>
<td>7</td>
</tr>
<tr>
<td>2.3 Performance metrics used in this work</td>
<td>8</td>
</tr>
<tr>
<td>2.3.1 Equal Error Rate: ERR</td>
<td>8</td>
</tr>
<tr>
<td>2.3.2 C Average: ( C_{\text{avg}} )</td>
<td>8</td>
</tr>
<tr>
<td>2.4 Motivation and recent works on LID: From the literature</td>
<td>9</td>
</tr>
<tr>
<td>2.4.1 Literature review from recent years</td>
<td>9</td>
</tr>
<tr>
<td>2.4.2 Motivation for current work from literature</td>
<td>11</td>
</tr>
<tr>
<td>3 Study of LID systems under Emotional Speech</td>
<td>12</td>
</tr>
<tr>
<td>3.1 Introduction</td>
<td>12</td>
</tr>
<tr>
<td>3.2 Dataset used in this work</td>
<td>12</td>
</tr>
<tr>
<td>3.3 Baseline systems</td>
<td>13</td>
</tr>
<tr>
<td>3.3.1 I-vector</td>
<td>13</td>
</tr>
<tr>
<td>3.3.2 Time Delay Neural Network: TDNN</td>
<td>14</td>
</tr>
<tr>
<td>3.3.3 Long Short Term Memory: LSTM</td>
<td>15</td>
</tr>
<tr>
<td>3.3.4 DNN x-vector</td>
<td>15</td>
</tr>
<tr>
<td>3.4 Experimental setup of the baseline systems</td>
<td>16</td>
</tr>
<tr>
<td>3.4.1 Features</td>
<td>16</td>
</tr>
<tr>
<td>3.4.2 I-vector</td>
<td>16</td>
</tr>
<tr>
<td>3.4.3 TDNN and LSTM</td>
<td>16</td>
</tr>
<tr>
<td>3.4.4 DNN x-vector</td>
<td>18</td>
</tr>
<tr>
<td>3.5 Results and discussion for baseline systems</td>
<td>18</td>
</tr>
<tr>
<td>3.6 Summary</td>
<td>20</td>
</tr>
</tbody>
</table>
## Contents

4 Towards LID systems robust to emotional variation ............................ 22
4.1 Introduction to prosody modification ................................................ 22
4.2 Flexible Analysis Synthesis Tool: FAST ........................................... 22
   4.2.1 Analysis Block .............................................................. 23
   4.2.2 Processing Block .......................................................... 23
   4.2.3 Synthesis Block ............................................................. 23
4.3 Results and discussion for FAST study .............................................. 24
4.4 Summary of prosody modification ..................................................... 25
4.5 Moving away from prosody modification .......................................... 25
4.6 Recurrent Convolutional Neural Network: RCNN ................................. 26
4.7 Multistage Training: MST ............................................................ 27
4.8 Experimental setup of RCNN ........................................................ 27
4.9 Results and discussion for RCNN and MST study ............................... 28
4.10 Summary of RCNN and MST ......................................................... 30

5 Conclusions and Future Scope .......................................................... 31
   5.1 Conclusions ................................................................. 31
   5.2 Future Scope ................................................................. 32

Bibliography ......................................................................................... 34
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Speech signal for &quot;My name is Priyam&quot;</td>
<td>4</td>
</tr>
<tr>
<td>2.2</td>
<td>Basic structure of a LID system</td>
<td>5</td>
</tr>
<tr>
<td>2.3</td>
<td>Source-filter model of speech production</td>
<td>5</td>
</tr>
<tr>
<td>2.4</td>
<td>Process for MFCC and fbank extraction for input speech signal</td>
<td>6</td>
</tr>
<tr>
<td>2.5</td>
<td>A general explicit LID system for N target languages</td>
<td>8</td>
</tr>
<tr>
<td>2.6</td>
<td>FRR &amp; FAR v/s threshold and EER</td>
<td>9</td>
</tr>
<tr>
<td>3.1</td>
<td>TDNN architecture</td>
<td>14</td>
</tr>
<tr>
<td>3.2</td>
<td>A basic LSTM cell</td>
<td>15</td>
</tr>
<tr>
<td>3.3</td>
<td>TDNN architecture used throughout this thesis</td>
<td>17</td>
</tr>
<tr>
<td>3.4</td>
<td>LSTM architecture used throughout this thesis</td>
<td>17</td>
</tr>
<tr>
<td>3.5</td>
<td>DNN x-vector architecture used throughout this thesis for extracting embeddings</td>
<td>17</td>
</tr>
<tr>
<td>3.6</td>
<td>Effect of speech utterance duration of different emotions, on the performance of state-of-the-art language identification systems. (a) i-vector + PLDA, (b) TDNN, (c) LSTM, and (d) DNN x-vector</td>
<td>20</td>
</tr>
<tr>
<td>4.1</td>
<td>Block Diagram depicting the prosody modification algorithm of FAST</td>
<td>23</td>
</tr>
<tr>
<td>4.2</td>
<td>Hidden states of a single RCL with T = 3. The bottom coloured figure represents input whereas forward and recurrent connections are represented by solid and dash lines respectively</td>
<td>26</td>
</tr>
<tr>
<td>4.3</td>
<td>RCNN architecture used in this work</td>
<td>28</td>
</tr>
</tbody>
</table>
# List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Details of the dataset used in this work</td>
<td>13</td>
</tr>
<tr>
<td>3.2</td>
<td>Performance (in terms of EER and $C_{avg}$) comparison between the state-of-the-art systems for language identification for the speech utterance of matched and mismatched conditions due to emotional states</td>
<td>18</td>
</tr>
<tr>
<td>3.3</td>
<td>Performance (in terms of EER and $C_{avg}$) of the state-of-the-art systems for language identification for the speech utterances of different emotional states</td>
<td>19</td>
</tr>
<tr>
<td>3.4</td>
<td>Deviation in performance (in terms of EER) of the state-of-the-art systems obtained using Equation 3.2</td>
<td>20</td>
</tr>
<tr>
<td>4.1</td>
<td>Performance (in terms of EER) of the baseline systems considered under different emotional conditions after prosody modification</td>
<td>24</td>
</tr>
<tr>
<td>4.2</td>
<td>Performance (in terms of EER) of the baseline systems considered under different emotional conditions after prosody modification with adaptation</td>
<td>25</td>
</tr>
<tr>
<td>4.3</td>
<td>Performance (in terms of EER) of the baseline systems and RCNN considered under different emotional conditions</td>
<td>29</td>
</tr>
<tr>
<td>4.4</td>
<td>Performance (in terms of EER) of the baseline systems and RCNN considered under different emotional conditions after prosody modification with adaptation</td>
<td>29</td>
</tr>
<tr>
<td>4.5</td>
<td>Performance (in terms of EER) of the baseline systems and RCNN considered under training with adaptation data and multi stage training conditions</td>
<td>29</td>
</tr>
<tr>
<td>4.6</td>
<td>Deviation in performance of the LID systems from Table 4.5 calculated using Equation 3.2</td>
<td>30</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

1.1 Language Identification

Speech Signal Processing (SSP) is a domain of digital signal processing limited to speech signals. One of the main aims of SSP is to improve human machine interaction by making machines understand and be able to manipulate human speech [75]. Speech signals broadly contain three types of information, namely speaker, prosodic and linguistic. Example of speaker related information is gender, age, and the identity of a particular speaker. Prosody refers to the emotion conveyed in the speech. Linguistic information contains language as well as the spoken content.

Spoken language identification is the process of determining in which language the speech is produced. There are thousands of languages and hundreds of language families in the world. This leads to diversity in the human population. Due to globalisation and interaction of human population from different linguistic backgrounds, multilingual systems become a necessity [91]. Hence, in this technology driven world Language Identification (LID) plays a key role in huge number of applications. Some examples are multilingual speech recognition [84], multilingual communication systems [54], spoken document retrieval [95], spoken language translation [25], among many more.

LID systems can play a role as a frontend or preprocessing component of a speech-enabled application or device. Amazon Alexa, Google Assistant, Microsoft Cortana, Samsung Bixby and Apple Siri are some of the prominent speech-enabled devices that have garnered increased amount of user base over recent years [32]. As these speech enabled smart devices reach our homes and offices, the development and improvement in this domain gathers increased interest.

We have seen challenges such as NIST Language Recognition Evaluation (NIST LRE) [67] and Oriental Language Recognition (OLR) [80] getting good attention and hence providing improved solutions to the society. Associations such as European Language Resources Association (ELRA) [46] and Speechocean [1] are working towards collecting and providing huge amount of resources for promoting the development in speech and language technologies domain. Moreover, the advancement in compu-
tation power and machine learning, coupled with the signal processing knowledge has enabled rapid improvement in LID [66].

1.2 Challenges involved with LID

From the studies in [2, 93, 85], state-of-art systems for LID have shown degradation in performance due to the variations such as short duration speech and multiple dialects of a language family. There have been some attempts to see the effectiveness of LID systems for noise and telephone quality speech [65, 90, 99, 45]. From the studies in [85, 55, 50], state-of-art LID systems have been shown to provide better performance on speech data collected in a neutral emotion state. However, there are not many attempts which study the performance of LID systems in the context of different emotional states. Recent studies on the impact of speakers’ emotional state on speaker verification [69], speaker identification [23], and automatic speech recognition [88, 87] indicate that the variation in acoustic features due to different emotional states results in performance degradation of these systems.

At feature level, acoustic feature modification and prosody modification are done to improve performance, whereas MAP based adaptations can be done at the model level [74]. Prosody modification is achieved by altering the three prosody parameters of an utterance. While these are traditional methods of adaptation in the speech processing literature, modern machine learning architectures and training methodologies have also been used for similar tasks in speech processing, as well as for other domains. All of the above aspects motivated us to investigate the impact of speaker’s emotional state on the performance of state-of-the-art LID systems, and provide robust solution for the problem. To the best of our knowledge this is the first time in literature that a study such as this is done for the task of LID.

1.3 Objective and scope

The main aim of this thesis is to study the effect and improve robustness of LID systems under the influence of emotional speech. Emotion in a spoken utterance is characterized by the three prosody parameters: duration, intonation and intensity [48]. Recognising the emotion of an spoken utterance is in itself a challenging task and an independent domain of SSP [1].

The scope of the thesis is as follows:

1. To aggregate a dataset for training and evaluation of LID systems in the context of the concerned problem.

2. To highlight that performance degradation is possible due to the emotional speech for LID systems.

3. To study the degradation, and use traditional adaptation techniques to improve performance.
4. To introduce the use of Recurrent Convolutional Neural Network (RCNN) for LID, combining them with multi stage training methodology to provide robust solution to the concerned problem.

1.4 Thesis outline

- Chapter 2 provides a detailed description of LID and advancement of the domain through literature review.

- Chapter 3 describes the baseline systems, and their performance analysis. We will also describe the dataset aggregated for this study in this chapter.

- Chapter 4 will demonstrate our attempts towards achieving robust LID systems for emotional variation. We will describe the feature level adaptation by modifying prosody and it’s performance analysis. Followed by this, we describe our proposed RCNN architecture, training methodology and it’s performance analysis.

- Chapter 5 concludes our finding, reports the prominent observations and provides the scope for future development.
Chapter 2

Language Identification

2.1 How humans perform LID

Humans perform LID by mapping words to the vocabulary of a language, consider sentences "My name is Priyam" and "mera naam Priyam hai", the first sentence is in English while the second is in Hindi. A person knowing the phonetic vocabulary of the two languages can easily classify those sentences. The problem narrows down to gathering the complete phonetic space of target languages. Some languages have overlapping vocabulary with two or more languages, in that case the linguistic rules also have to be employed. These could be transition of phones, syllables, words or the differences in semantics or syntactic. Hence apart from the vocabulary, knowledge and structure of language becomes essential prerequisite to performing LID. Apart from these, prosodic cues also help to identify certain languages, even in the absence or lack of vocabulary knowledge. Mori et al. showed this in a study for English and Japanese [51].

In summary there could be various ways to distinguish between different set of languages, but the problem with machines is that the input speech signal is a digital signal, an example shown in Figure 2.1, hence modelling of the digital speech signal has to be done in a way which complements the differences at various levels by making use of the available data [81].

![Figure 2.1 Speech signal for "My name is Priyam"](image-url)
2.2 Basic structure of LID systems

The basic structure of a LID system is described in Figure 2.2. The feature extraction which receives the raw speech waveform as input is oftenly referred as frontend, while the classifier is referred as the backend of a LID system.

![Figure 2.2 Basic structure of a LID system](image)

While most LID systems can be broken down into two components to represent this structure, sometimes this differentiation is not trivial. We will provide an in depth literature review of LID systems in this chapter.

2.2.1 Frontend in LID: Feature Extraction

The important criteria for a set of feature to be valid for or any classification task is that it should be able to represent the input sample with a reduced size, while enhancing (or restoring) the distinguishable properties [83]. As noted in Section 2.1, languages can be identified by using various cues at different levels. Hence the set of target languages influence feature extraction depending on the type of cues which facilitate best differentiation between the target languages.

The process of speech production is considered to be a source-filter model [29] and is depicted in Figure 2.3. Most of the conventionally used features in speech processing (including LID) are based on this model of speech production, such as Mel Frequency Cepstral Coefficient (MFCC), filter bank (fbank) energies, Linear Prediction (LP) residual among many more [53, 63, 33]. Prosody based features have also been used for LID and demonstrated specifically for the case of Indian LID in [64].

![Figure 2.3 Source-filter model of speech production](image)

The process for extraction of MFCC and fbanks is shown in Figure 2.4. It begins by first computing the Fast Fourier Transform (FFT) of the input speech signal, then the frequencies are converted to mel scale by using the formula shown in Equation 2.1

\[ m = 2595 \log_{10} \left( 1 + \frac{f}{700} \right) \quad (2.1) \]
This is followed by applying log to convert the spectral domain to cepstral domain followed by the application of Discrete Cosine Transform (DCT). From the output of DCT, we consider first few coefficients as MFCC. The most used practice is to take first 13 coefficients of the DCT output as MFCC. If the coefficients are taken before applying DCT, they are called fbank energy coefficients.

![Figure 2.4 Process for MFCC and fbank extraction for input speech signal](image)

Linear Prediction Analysis of speech signal leverages the source-filter model of speech production depicted in Figure 2.3. It estimates the vocal tract characteristics, represented by Linear Prediction Coefficients (LPC), and the input source, also called LP residual, by minimizing the distance between actual speech and the predicted one. It uses AutoRegressive Moving Average (ARMA) as a predictor [47]. Apart from LP residual, LPC and Linear Prediction Cepstral Coefficients (LPCC) have also been used [94, 11]. Prosody based features used for LID can be derived in various ways based on the three prosody parameters. One such work is based on segmenting a speech signal using Vowel Onset Points (VOP), then taking the $F_0$ countour, duration and energy to represent the prosodic properties of a language [49].

Since most of the approaches are fourier transform based and speech signal is considered to be quasi-stationary, frames of 20-200 ms are mostly considered for extracting these features. Hence one input signal is divided into several frames, which can then be fed as one input sample to the backend, or several frames stacked to make a segment as input to the backend. This choice depends on the structure of the backend. Commonly these features are used by concatenating their vector delta and double deltas as well. Another variant Shifted Delta Cepstral (SDC) are also used widely [6]. These are more prominent in LID due to their longer context modelling ability compared to vanilla MFCCs (or delta-delta MFCCs). The process of extracting SDC features from cepstral coefficients ($c(t)$) is shown in Equation 2.2 and Equation 2.3. $N$ denotes the number of cepstral coefficients, $d$ denotes delta time shift, $P$ distance between blocks and $K$ number of blocks considered for computing SDC at one time instance.

$$\delta_j(t) = c_j(t + d) - c_j(t - d), \quad 0 \leq j \leq N - 1$$ \hspace{1cm} (2.2)

$$SDC(t) = \delta_j(t + (i - 1)P), \quad 1 \leq i \leq K$$ \hspace{1cm} (2.3)

The problem with almost all of the above stated representations is that the long term temporal context is somewhat lost due to framing of the signal, even with SDC features. This has been addressed in the literature by using a representation learning network which can model the long temporal context. This
is where the distinction between the frontend and backend of a LID system gets compromised, but with better performance. Examples of such system are presented in Section 2.4 as part of the literature review.

2.2.2 Backend in LID: Classifier

The choice of backend classifier is influenced by variety of factors such as number of target languages, size of the data available, computation power, among others. Early stages of LID development was mostly focused on using Hidden Markov Model (HMM) [98], Gaussian Mixture Model (GMM) [37] and Support Vector Machine (SVM) [96] as the backend classifier. Other techniques such as Vector Quantization (VQ) and Dynamic Time Warping (DTW) were also prominent at some point [13, 36]. Recently neural network based classifiers, such as Deep Neural Network (DNN) and Recurrent Neural Network (RNN), have become more prominent [41, 7]. Sometimes the backend also performs representation learning task to improve classification performance. I-vector and x-vector extraction converts variable length MFCC feature frames to fixed dimensional embedding, which are then used to generate language labels using various techniques such as cosine scoring [15, 76].

All LID systems can be broadly classified into two categories, explicit LID and implicit LID, on the basis of how they perform classification.

2.2.2.1 Explicit LID

Explicit LID systems performs the classification by explicitly learning all target languages. The most general kind of a explicit LID system would be to use speech recognizer of all the target languages as shown in Figure 2.5. The token level in explicit LID is mostly phone level by employing a phone recognizer often realised using GMM and HMM [4, 86]. This involved generating a phone inventory of all the target languages. Apart from phone recognition, lexical models have also been studied for LID [30]. Recent works on explicit LID systems involved using a phone discriminating DNN to generate phonetic features [81].

2.2.2.2 Implicit LID

Implicit LID systems perform the classification by learning to differentiate between the set of target languages. The differentiating boundaries are learnt directly from the set of input features. Phone inventory and transcription of speech samples is not required for implicit approach. Early implicit studies identified that even for languages with similar phoneme sets, the frequency of distribution of phoneme varied largely [19]. They focused on extracting language specific prosodic features and formant vectors from voiced speech segments. Sugiyama et al. used LPCs and LPCCs with VQ of different code book sizes to perform LID [79]. A recent work by Bhasker et al. explored vocal tract features to extract language related information [5].
2.3 Performance metrics used in this work

Conventional classification metrics are suitable for evaluating LID system performance, but the two most prominent in the field of LID are Equal Error Rate (ERR) and C Average (C_{avg}) as they are also used in recent LID challenges such as NIST LRE and OLR challenges [67, 80].

2.3.1 Equal Error Rate : ERR

ERR determines the threshold values for false rejection rate (FRR) and false acceptance rate (FAR). The lower the ERR better is the system’s accuracy. For a binary classifier, FRR is the percentage of samples that belong to a particular class but are not classified as that class, while FAR is the percentage of samples not belong to a class but wrongly classified as that class. Figure 2.6 shows the graph for variation of FAR and FRR with respect to threshold value. In the scenario of multiple classes, ERR is calculated by considering all possible binary classifiers in the OnevsAll approach.

2.3.2 C Average: C_{avg}

Another prominent metric used in the literature and major challenges is C_{avg}. In this thesis, we have used it at some points to compare the performance with existing works. The calculation of C_{avg} uses a pairwise loss between target and non-target classes. The pairwise loss is obtained as shown in Equation 2.4

\[ C(L_t, L_n) = P_{Target}P_{Miss}(L_t) + (1 - P_{Target})P_{FA}(L_t, L_n) \]
In the equation, $L_t$ and $L_n$ are target and non-target languages, respectively and $P_{\text{Target}}$, $P_{\text{Miss}}$ and $P_{\text{FA}}$ are the prior probability, miss probability and false alarm probability of the target language. During evaluation, the value of $P_{\text{Target}}$ is set to 0.5. Further, the $C_{\text{avg}}$ is defined as average of this pairwise loss and is shown in Equation 2.5

$$C_{\text{avg}} = \frac{1}{N} \sum_{L_t} \sum_{L_n} C(L_t, L_n)$$ (2.5)

where $N$ is the number of languages.

![Figure 2.6 FRR & FAR v/s threshold and EER](image)

### 2.4 Motivation and recent works on LID: From the literature

#### 2.4.1 Literature review from recent years

In 2014, Ming Li et. al proposed generalised i-vectors for LID and speaker verification [40]. Their proposed i-vector framework was based on phonetic tokenization and tandem features. Their proposed method outperformed the i-vector baseline by 45% EER on NIST LRE 2007 dataset. In the same year, Sriram Ganapathy et. al proposed to use bottleneck features (BN) extracted from Convolution Neural Network (CNN) along with the conventional acoustic features for the LID task in Robust Automatic Transcription of Speech (RATS) program. They reported around 25% improvement in terms of EER compared to the acoustic features [20]. Javier Gonzalez-Dominguez et. al made a huge impact in the field of LID, that same year, by introducing the use of Long Short Term Memory (LSTM) RNN for this task [24]. They highlighted LSTM’s ability to model long temporal sequences and that too with fewer trainable parameters compared to the feed forward DNN.

In 2015, Alicia Lozano-Diez et. al proposed an end-to-end approach for LID by using Convolutional Deep Neural Networks (CDNN). There proposed architecture achieved comparable performance with respect to the baseline i-vector on the NIST LRE 2009 Voice of America (VOA) short utterance (3
seconds) dataset. But the advantage was that it used fewer parameters compared to other deep architectures. On score level fusion with the baseline i-vector, they reported 11% improvement [43]. Athanasios Lykartsis and Stefan Weinzierl presented an approach to LID based on speech rhythm. They extracted rhythm related features by using beat histogram [44]. Yan Song et. al proposed a variation to the Deep BN feature extracted from a DNN architecture to tackle the performance degradation under short duration utterance and dialect recognition. They employed a DNN like architecture Deep bottleneck network (DBN), where two internal layers were acting as feature extractors. The fusion of features from these two layers was demonstrated to give performance gains [78].

In 2016, Mounika K. V. et. al investigated DNN architecture for LID on a dataset of 12 Indian languages. They also proposed to add a attention mechanism to get utterance level classification and thereby improving DNN accuracy for LID task [52]. Saad Irtza et. al proposed a hierarchical framework for LID. They performed hierarchical classification by using language clusters. The final language label was predicted only at the final level. They achieved it by using different feature representations at each level and Gaussian Probabilistic Linear Discriminant Analysis (GPLDA) as backend for all levels [27]. Wang Geng et. al introduced attention based LSTM-RNN end-to-end LID architecture [22]. Their proposed architecture outperformed conventional i-vector systems by 34.33% in terms of EER.

In 2017, Ma Jin et. al proposed an end-to-end LID approach which employed bilinear pooling. They used high order statistics of LID-senones for representation. These LID-senones are analogous to the senones in ASR. They showed that their approach was robust to speaker, noise and channel variations [28]. Saad Irtza et. al addressed the issue of scalability (in terms of number of target languages) in LID systems. They demonstrated better scalability when using a hierarchical LID architecture. There system only required a few select layers to be trained when adding more target languages [26]. Sarith Fernando et. al used bidirectional LSTM (bLSTM) for LID, specifically for improving performance under short duration speech [18]. Peng Shen et. al proposed to employ conditional generative adversarial nets (GANs) in LID. Their motivation was to tackle the below par performance of neural network based LID architectures under small train set [71].

In 2018, Alicia Lozano-Diez et. al proposed a fixed length embedding vector for LID extracted using a DNN like architecture. These representations are learnt at the utterance level and are task specific. They reported 7.3% improvements in terms of EER over the baseline i-vector system [42]. Weicheng Cai et. al proposed a learnable dictionary layer, this layer mimics the GMM training and supervector operations on top of a CNN to result in utterance level fixed length representations. Their proposed method outperformed simple average pooling on the NIST LRE 2007 closed set task [10]. Sarith Fernando et. al proposed a factorized hidden variability subspace (FHVS) technique for improving the robustness of bLSTM LID architectures under speaker, noise and channel variations [16]. Sarith Fernando et. al also proposed use of a novel features based on temporal envelopes, which are extracted using linear prediction and then converted to cepstral features. When used as frontend for a bLSTM based backend, they achieved 38.4% improvement over state-of-art bottleneck features on OLR17 dataset [17].
In 2019, Weicheng Cai et al. proposed an end-to-end attention based CNN-bLSTM architecture for LID. CNN-bLSTM was used at the frontend as a representation learning network, followed by self attentive pooling layers to get utterance level representations. They reported comparable decrease in EER over other state-of-art utterance level approaches on NIST LRE 2007 closed set task [9]. Bharat Padi et al. proposed a hybrid framework based on i-vector and bLSTM to model the sequential information from short segment i-vectors. I-vectors are extracted from short speech segments in an utterance and fed into a bLSTM network, which has attention mechanism. Their proposed system improved performance under short duration speech and noisy conditions [55]. Xiaoxiao Miao et al. proposed a novel two dimensional attention mechanism as well as demonstrated performance gains in using Convolutional and long short term memory (CLSTM) neural network. Their proposed attention mechanism learnt weights for frequency bands in addition to the traditional weights along just the time dimension [50].

2.4.2 Motivation for current work from literature

Many of the above discussed architectures have been shown to perform well for computer vision and natural language processing related tasks as well. B-LSTM and GRUs have been used for semantic analysis on textual data [73] and image captioning [92]. This goes to show that these architectures are independent of the modality of data, rather depends on the modelling task. This is also evident from the fact that Convolutional Neural Networks (CNN) are heavily prominent in the field of computer vision but not much in NLP or speech [38]. Recently introduced Recurrent Convolutional Neural Network (RCNN) for image scene labelling task [57] have also given good results for emotion and phoneme recognition tasks on spoken data [97]. This motivated us to investigate them for the LID task.

Apart from architecture level developments, there has been work on developing new training methodologies to improve performance. Improving performance of LID by using a student teacher network [72] and joint learning by optimizing two or more loss functions simultaneously are evidence of that [60]. This idea of using multiple loss functions in a single network combined with the way a Gaussian Mixture Model (GMM) is initialised using a Universal Background Model (UBM), motivated us to investigate non contemporary methods of initialising and adapting a neural network.
Chapter 3

Study of LID systems under Emotional Speech

3.1 Introduction

As stated in Section 1.3, effect of emotional speech on LID systems is largely an unexplored area and hence the first step towards studying it is to consider a few state-of-art architectures and observe their performance. In this thesis we have considered i-vector, TDNN, LSTM and x-vector as baseline systems, owing to their recognition and performance in the literature over the years. These systems also to some extent cover a large spectrum of types of LID systems. In this chapter, we will start by first describing our dataset and then use our aggregated dataset on the considered systems and study the variation of performance.

3.2 Dataset used in this work

In this thesis, the experiments have been performed by aggregating various datasets for the following 7 languages: Basque, English, German, Hindi, Serbian, Spanish and Telugu. For train data of English, Hindi and Telugu, IIIT-ILSC dataset [89] has been used. German train data has been taken from [61], while Spanish and Basque train data has been taken from Mozilla Common Voice Project[1] Emotional datasets for Basque, English, German, Hindi, and Telugu are taken from [68, 39, 8, 35, 34]. The Spanish[2] and Serbian[3] are collected from ELRA. In this work, four basic emotions anger, happy, sad and neutral are considered.

In total 7 sets are prepared: train, test, test_angry, test_happy, test_neutral, test_sad and adaptation with each containing utterances from the 7 considered languages. Table 3.1 describes the dataset in terms of number of utterances and duration. Utterances from the datasets considered for training are split in two sets, namely train and test datasets. These datasets were originally created with no emphasis on emotion and are largely composed of neutral utterances. Then we have the emotion labelled dataset

Table 3.1 Details of the dataset used in this work

<table>
<thead>
<tr>
<th>Type</th>
<th>Basque</th>
<th>English</th>
<th>Hindi</th>
<th>German</th>
<th>Serbian</th>
<th>Spanish</th>
<th>Telugu</th>
<th>Total Duration</th>
<th>Total Utterances</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train</td>
<td>1014</td>
<td>960</td>
<td>2380</td>
<td>1208</td>
<td>292</td>
<td>780</td>
<td>1520</td>
<td>12.9</td>
<td>8154</td>
</tr>
<tr>
<td>Test</td>
<td>309</td>
<td>540</td>
<td>650</td>
<td>588</td>
<td>36</td>
<td>285</td>
<td>645</td>
<td>5.26</td>
<td>3053</td>
</tr>
<tr>
<td>Test Neutral</td>
<td>140</td>
<td>50</td>
<td>150</td>
<td>90</td>
<td>36</td>
<td>85</td>
<td>150</td>
<td>0.71</td>
<td>701</td>
</tr>
<tr>
<td>Test Angry</td>
<td>140</td>
<td>50</td>
<td>150</td>
<td>110</td>
<td>24</td>
<td>76</td>
<td>150</td>
<td>0.66</td>
<td>700</td>
</tr>
<tr>
<td>Test Happy</td>
<td>140</td>
<td>50</td>
<td>150</td>
<td>70</td>
<td>24</td>
<td>84</td>
<td>150</td>
<td>0.7</td>
<td>668</td>
</tr>
<tr>
<td>Test Sad</td>
<td>140</td>
<td>50</td>
<td>150</td>
<td>70</td>
<td>24</td>
<td>79</td>
<td>150</td>
<td>0.78</td>
<td>663</td>
</tr>
<tr>
<td>Adaptation</td>
<td>1681</td>
<td>175</td>
<td>1455</td>
<td>1180</td>
<td>291</td>
<td>836</td>
<td>2100</td>
<td>8.6</td>
<td>7718</td>
</tr>
</tbody>
</table>

for each language, using these we create 4 test sets corresponding to the 4 emotions. A part of neutral data from the emotion labelled datasets is also used in training to minimize the variance due to channel and recording environments. Apart from train and emotional test datasets, we have created a separate adaptation dataset consisting of emotional utterances from all 7 languages and 4 emotions. The separate adaptation set is created for various adaptation tasks to be done during the training phase in latter part of this thesis.

The emotion labelled dataset for each language contains parallel emotional utterances. For example, if there are 10 different sentences to be recorded for 5 speakers, then each speaker will record each sentence in all 4 emotions. Hence a speaker will record $10 \times 4$ sentence, and in total the dataset will contain $10 \times 4 \times 5 = 200$ utterances. The splitting of utterances into datasets for train phase (train and adaptation sets) and test phase (rest) is based on the number of speakers available from each original dataset. All the speakers in train and adaptation are different from test datasets and so is the spoken content while the recording environment and channel characteristics are same among all emotions of an utterance.

### 3.3 Baseline systems

#### 3.3.1 I-vector

I-vectors are used to learn fixed dimensional embeddings from spoken data [15]. They are low-rank vectors, representing the characteristics of the speaker and linguistic information in an utterance. The Gaussian Mixture Model (GMM) and the Universal Background Model (UBM) are modelled using MFCC feature frames, which are extracted from speech utterances. The idea of i-vector extraction is based on the total variability space modelling technique. Initially a UBM is trained using a small set of training samples, which is then used to adapt a set of train samples through eigenvoice adaptation, which are in turn used to train a GMM. The eigenvoice adaptation makes the assumption that all the important variation is captured in a low rank matrix termed as total variability matrix [14]. All GMM and UBM mean vectors are stacked together to obtain their respective super vectors. I-vector representation for a given utterance is then obtained as shown in Equation 3.1.
\[ M = m + Tw + \epsilon. \] \hspace{1cm} (3.1)

In Equation 3.1, \( M \) and \( m \) are GMM and UBM supervectors respectively, \( T \) is the total variability matrix, \( w \) is the i-vector having normal distribution and \( \epsilon \) is the residual noise. \( \epsilon \) is there to account for any variability not captured in \( T \).

Furthermore, dimensionality reduction techniques such as Linear Discriminant Analysis (LDA) and Probabilistic Linear Discriminant Analysis (PLDA) are used to maximise the intra-class variance while minimising inter-class variance between features vectors [2, 59]. These techniques are applied to the i-vectors before scoring.

### 3.3.2 Time Delay Neural Network: TDNN

Unlike traditional DNNs, TDNN learns long-term dependencies through variable sized context modelling at each layer. The input can be any acoustic feature frames, such as MFCC and fbank. Layers in the DNN are activated from the full temporal context of the previous layer, whereas in TDNN each neuron is activated by a narrow context, coming from the previous layer, and gets accumulated as the network goes deep. Furthermore, the initial layers in a TDNN also learn from the whole temporal context due to back-propagation of the gradients. Thus lower layer activation becomes translation-invariant [56]. Figure 3.1 depicts the TDNN architecture used in this thesis.

![TDNN Architecture](image)

**Figure 3.1** TDNN architecture

The length of context received by a neuron of each layer is a hyperparameter and depends on the modelling task. For speech, assuming consecutive frames may be very similar, instead of taking consecutive frames as context at each layer we consider taking only 2 or 3 frames from the previous layer.
An example of this can be seen in layer 3 of Figure 3.1 where only current frame minus 3 and plus 3 are considered.

### 3.3.3 Long Short Term Memory: LSTM

LSTM network is a special class of recurrent neural networks, used for sequential modelling tasks. The activation at each time-step is a mapping learnt from the input as well as previous time-steps. Each LSTM cell has three gates associated with it: input gate (for activation of input), output gate (for activating the output) and forget gate (for manipulating the cell state). In practice, several layers with each containing multiple LSTM cells are stacked together to form a LSTM network. Generally, the activation at the last time-step of the last state is considered for classification tasks, as it has been mapped from the complete temporal context [24].

![A basic LSTM cell](image)

**Figure 3.2** A basic LSTM cell

Figure 3.2 depicts a basic LSTM cell with all the operations. In that figure, the upper horizontal arrow represents the temporal context \(c_t\), while the lower one is for the hidden state activation \(h_t\) and the \(x_t\) is the input at time step \(t\).

### 3.3.4 DNN x-vector

Similar to i-vectors, x-vectors learnt using a DNN architecture are fixed dimension embeddings for variable length spoken utterances. Several hidden layers are stacked one after the other as in a traditional DNN, but the layers are not fully connected rather are more similar to that of a TDNN. Hence each layer is activated with a narrow temporal context of the previous layer along with its own temporal context, so it keeps getting accumulated, hence the deeper layers get a wide temporal context [77].

After the dense layers, a stat pooling layer is used to calculate mean and standard deviation of the previous layer output across all time-steps. Hence the stat pooling layer makes sure of the fixed dimensional embedding for variable duration utterances. Generally, we use the concatenated mean and standard deviation vectors from the stat pooling layer as input to subsequent dense layers. These subsequent dense layers are fully connected layers, like that of a traditional DNN and the architecture is
terminated with a softmax layer to facilitate training using backpropagation. Embeddings can be extracted as the output of these latter dense layers, before non linearity has been applied to them. These embeddings thus represent the language related information and can be used separately for various tasks. But since the task here is LID, we use the extraction architecture as it is for classification as well.

3.4 Experimental setup of the baseline systems

In this work, we have used the kaldi toolkit [58] end to end, from feature extraction to classification models.

3.4.1 Features

This work uses two most common and well known features as input to the classifiers: 40 dimensional (40D) filter bank (fbank) and 13 dimensional (13D) Mel Frequency Cepstral Coefficients (MFCC). The cepstral mean variance normalisation and energy based Voice Activity Detection (VAD) have been applied to MFCC features to normalize and remove the features corresponding to non speech frames. For x-vector extraction, the VAD output of MFCC is adapted to remove non speech frames from the fbanks as well. Apart from this, each MFCC frame feature representation is converted to SDC representation, in order to capture longer temporal dependencies.

3.4.2 I-vector

For i-vector extraction, this work uses MFCC features to train the UBM/GMM. 1800 utterances are selected from the train data to train the UBM, and then 3600 utterances are selected to train the GMM using the Expectation Maximization technique. 400 dimensional i-vectors are extracted using 2048 Gaussian mixtures. LDA and PLDA transforms are applied, where LDA reduces the dimensions of the i-vectors to 150. As noted in [31], cosine scoring provides similar performance as SVM for i-vectors. Hence we use cosine scoring to evaluate the performance of the i-vectors. Cosine scoring is done by taking the cosine distance between the test vector and the mean vector of target language from the train set.

3.4.3 TDNN and LSTM

Input to both TDNN and LSTM are 40D fbanks, described in Section 3.4.1. For TDNN, we are using 6 hidden layer architecture similar to [80]. The temporal context of the layers are shown in Figure 3.3. In this work, LSTM network uses the first layer as the affine transformation layer with similar temporal context as the first layer of TDNN architecture. Then a stacked LSTM layer with 512 cell output size is used. The LSTM architecture used in this work is described in Figure 3.4.
Figure 3.3 TDNN architecture used throughout this thesis

Figure 3.4 LSTM architecture used throughout this thesis

Figure 3.5 DNN x-vector architecture used throughout this thesis for extracting embeddings
In Figure 3.3 and Figure 3.4, L denotes the number of target languages. Relu activation is used in each layer with the exception of output layer. The shape of a layer is denoted in the format input × output. In case of LSTM layer, the sizes are denoted for one LSTM cell.

3.4.4 DNN x-vector

The architecture for extracting x-vector embeddings is similar to [76], which is demonstrated in Figure 3.5. The DNN x-vector extraction uses 40D fbank features, as described in Section 3.4.1. In Figure 3.5, F (40) is the input feature dimension, T (depends on the utterance length) is number of frames and C (7) is the number of classes. The x-vector embeddings are taken as the output of segment6, which implies that we are considering 512 dimensional embeddings. These embeddings can be scored in the similar way as i-vectors, but in this work, we are scoring them directly from the softmax layer.

3.5 Results and discussion for baseline systems

This study aims to investigate the effect of emotional state present in the speech utterance, on the performance of a LID system. The main obstruction to such studies is the lack of a standard dataset, which is why we have designed a dataset (discussed in Section 3.2) to study the performance of LID systems in the context of various emotional states such as neutral, anger, happiness, and sadness. In this regard, state-of-the-art systems such as i-vector, TDNN, LSTM and DNN x-vector architectures have been studied to determine how emotional states in speech affect the performance of LID systems. The LID systems have been trained using the speech utterances of the train set. In this chapter, evaluation of LID systems with speech utterances of test set is referred to as a matched test condition, and the evaluation of LID systems with speech utterances of angry, happy, and sad emotional state is referred to as mismatched test condition. The results have been reported in Table 3.2, Table 3.3 and Figure 3.5.

Table 3.2 Performance (in terms of EER and C_avg) comparison between the state-of-the-art systems for language identification for the speech utterance of matched and mismatched conditions due to emotional states.

<table>
<thead>
<tr>
<th>System</th>
<th>Matched</th>
<th></th>
<th>Mismatched</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>EER</td>
<td>C_avg</td>
<td>EER</td>
<td>C_avg</td>
</tr>
<tr>
<td>i-vector</td>
<td>15.29</td>
<td>0.15</td>
<td>23.54</td>
<td>0.25</td>
</tr>
<tr>
<td>i-vector + LDA</td>
<td>13.13</td>
<td>0.13</td>
<td>19.45</td>
<td>0.23</td>
</tr>
<tr>
<td>i-vector + PLDA</td>
<td>11.17</td>
<td>0.12</td>
<td>17.77</td>
<td>0.22</td>
</tr>
<tr>
<td>TDNN</td>
<td>17.38</td>
<td>0.18</td>
<td>16.69</td>
<td>0.2</td>
</tr>
<tr>
<td>LSTM</td>
<td>18.10</td>
<td>0.27</td>
<td>23.69</td>
<td>0.28</td>
</tr>
<tr>
<td>DNN x-vector</td>
<td>8.12</td>
<td>0.08</td>
<td>17.18</td>
<td>0.22</td>
</tr>
</tbody>
</table>

Table 3.2 shows the performance comparison between state-of-art LID systems for matched and mismatched (which is an aggregated set of speech utterances of anger, happy and sad emotional utter-
Table 3.3 Performance (in terms of EER and $C_{avg}$) of the state-of-the-art systems for language identification for the speech utterances of different emotional states.

<table>
<thead>
<tr>
<th>System</th>
<th>i-vector</th>
<th>i-vector + LDA</th>
<th>i-vector + PLDA</th>
<th>TDNN</th>
<th>LSTM</th>
<th>DNN x-vector</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>EER</td>
<td>C$_{avg}$</td>
<td>EER</td>
<td>C$_{avg}$</td>
<td>EER</td>
<td>C$_{avg}$</td>
</tr>
<tr>
<td>Test Neutral</td>
<td>22.11</td>
<td>0.21</td>
<td>17.97</td>
<td>0.19</td>
<td>15.41</td>
<td>0.17</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>7.13</td>
<td>0.12</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Test Anger</td>
<td>25.00</td>
<td>0.26</td>
<td>22.86</td>
<td>0.26</td>
<td>22.00</td>
<td>0.25</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>22.14</td>
<td>0.29</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Test Happy</td>
<td>24.10</td>
<td>0.27</td>
<td>20.51</td>
<td>0.26</td>
<td>18.41</td>
<td>0.24</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>20.96</td>
<td>0.24</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Test Sad</td>
<td>21.42</td>
<td>0.21</td>
<td>16.89</td>
<td>0.18</td>
<td>14.03</td>
<td>0.17</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>5.9</td>
<td>0.22</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

ances) testing conditions. For matched test data DNN x-vectors shows best performance and i-vector with PLDA system shows comparable performance in terms of EER and $C_{avg}$. The LSTM based LID system shows poor performance in both matched and mismatched test conditions. For mismatched test data, TDNN shows best performance, and the DNN x-vector and i-vector PLDA systems show comparable performances. However, the results suggest that there is a degradation in performance of all LID systems except TDNN under mismatch test condition.

Table 3.3 shows the performance of LID systems for speech utterances of different emotional states. The results show that the performance of LID systems for speech of sad emotional state is comparable to that of neutral emotional state. The effect of sadness on the performance of LID systems is less compared to anger and happiness. Though the DNN x-vector system showed the best performance under matched conditions, it shows huge degradation in performance under mismatched emotional conditions.

From Table 3.2 and Table 3.3 we can note that the two metrics (EER and $C_{avg}$) are parallel. The baseline systems considered have also been considered in recent OLR challenges, i-vector, TDNN, LSTM in AP18-OLR [80] and DNN x-vector in AP19-OLR [82]. I-vector+PLDA was the best performance baseline system in AP18-OLR, it reported 0.0596 and 5.86 as $C_{avg}$ and EER respectively on full length AP18-OLR dataset. On our dataset, i-vector+PLDA is best among the AP18-OLR systems, reported 0.12 and 11.17 as $C_{avg}$ and EER respectively. DNN x-vector reported 0.1257 and 12.22 as $C_{avg}$ and EER respectively on the AP19-OLR dataset, while it reported 0.08 and 8.12 as $C_{avg}$ and EER respectively on our dataset. This shows that these systems reported parallel magnitude of performance on our dataset, keeping in mind that the OLR challenge datasets are much larger in terms of duration and number of utterances. This validates our parameter selection for the baseline systems. We will be omitting $C_{avg}$ from further reporting and analysis, to keep the discussion simple.

The exception noted in Table 3.2 for performance degradation of TDNN under mismatched test conditions can be attributed to it’s excellent performance on sad utterances, as observed through Table 3.3. But it’s performance under anger and happy emotions is worse compared to other systems. Hence we study the deviation in performance of these systems with respect to the performance under neutral emotion, see Table 3.4. This deviation has been calculated between EER for anger, happy and sad test utterances ($E_A$, $E_H$ and $E_S$) with respect to the EER for test neutral set ($E_N$). From Table 3.4 we can note that the 2 best performing systems under mismatched condition (from Table 3.2) show the highest deviation in performance. This further strengthens the notion that these LID systems are over-fitting towards neutral utterances, slightly towards sad utterances in turn as well.
\[
\text{deviation} = \sqrt{\frac{(E_A - E_N)^2 + (E_H - E_N)^2 + (E_S - E_N)^2}{3}}
\] (3.2)

**Table 3.4** Deviation in performance (in terms of EER) of the state-of-the-art systems obtained using Equation 3.2

<table>
<thead>
<tr>
<th>System</th>
<th>deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>i-vector</td>
<td>2.06</td>
</tr>
<tr>
<td>i-vector + LDA</td>
<td>3.24</td>
</tr>
<tr>
<td>i-vector + PLDA</td>
<td>4.26</td>
</tr>
<tr>
<td>TDNN</td>
<td>11.80</td>
</tr>
<tr>
<td>LSTM</td>
<td>7.01</td>
</tr>
<tr>
<td>DNN x-vector</td>
<td>10.02</td>
</tr>
</tbody>
</table>

**Figure 3.6** Effect of speech utterance duration of different emotions, on the performance of state-of-the-art language identification systems. (a) i-vector + PLDA, (b) TDNN, (c) LSTM, and (d) DNN x-vector.

Figure 3.5 shows the performance under 4 emotions for different test utterance duration. Each utterance, from all 4 emotions, is split into 1 second, 2 second and 3 second utterances to study the effect of duration on the trained LID systems. For a typical LID system, the duration v/s EER would be a monotonically decreasing curve. Similar behavior can be observed, from Figure 3.5, for the mismatched conditions discussed in this chapter.

### 3.6 Summary

In this chapter, we introduced our dataset which has been aggregated from various datasets of the 7 target languages. Following this we studied the effect of emotional speech on the performance of state-of-art systems for language identification (LID). The results shows the similarity in performance for neutral-sad and happy-angry emotion pairs. Further, for high arousal speech (anger and happy) the LID systems shows poor performance. On the other hand, the LID systems showed better performance for low arousal speech (neutral and sad). Though the linguistic information in speech utterance is not varying with the speakers’ emotional state, the state-of-art systems for LID shows poor performance.
for high arousal speech. In this case, normalization of acoustic features may improve the performance of LID systems for high arousal speech. Hence, in further chapters, we will be investigating different adaptation methods to improve the performance of LID systems for the mismatch due to speakers’ emotional state.
Chapter 4

Towards LID systems robust to emotional variation

In this chapter, we will start by providing a brief overview of prosody modification and FAST algorithm as well as use it to perform adaptation in order to improve performance under emotional speech utterances. After performing this study and reviewing the results, we will describe the Recurrent Convolution Neural Network (RCNN) and propose a training methodology for adaptation along with it. All results obtained will be evaluated and we will provide concluding remarks on the performance.

4.1 Introduction to prosody modification

From the results obtained in Chapter [5] it can be observed that there is a clear degradation in performance under emotional mismatch between train and test conditions. Prosody modification is the process of modifying emotion of an utterance by altering the three prosody parameters: duration, intonation and intensity. It is an adaptation technique and has been used in various speech processing tasks, such as in ASR to make them robust to emotion variations [62] or as a data augmentation technique to make them robust to speaker related acoustic variations [70]. There are various algorithms to achieve prosody modification, and one prominent algorithm is Flexible Analysis Synthesis Tool (FAST), created by Gangamohan Paidi [21].

4.2 Flexible Analysis Synthesis Tool: FAST

FAST algorithm is shown in Figure [4.2] it depicts the algorithm as a series of 3 separate components acting on the input source speech to synthesize output speech with the modified prosody according to that of the input target speech. The input source speech can be neutral speech and input target speech can be in any one of the 3 emotions (anger, happy and sad), or vice versa.
4.2.1 Analysis Block

In the analysis block, LP analysis and epoch extraction is done for both (source and target) utterances. LP analysis is used to get the vocal tract and excitation source components. Epoch extraction is used to get the instants of significant excitation and instantaneous frequency contour.

4.2.2 Processing Block

The processing block helps in the alignment of frames from input source speech to get the desired synthesized output. If \((x_1, x_2, ..., x_M)\) and \((y_1, y_2, ..., y_N)\) represent the frame-wise vocal tract shape of the input source speech \((M\) frames) and input target speech \((N\) frames) respectively, then DTW is used to get the optimal path to reach point \((M, N)\), where optimal path for a point \((i, j)\) is given by Equation 4.1:

\[
D(i, j) = d(i, j) + min\{D(i-1, j), D(i-1, j-1), D(i, j-1)\} \tag{4.1}
\]

In Equation 4.1, \(d(i, j)\) is the distance between \(i^{th}\) frame of input source and \(j^{th}\) frame of input target speech. The optimal path helps to get the pitch period of each frame in input source speech as it will be in the target emotional speech. This in turn performs the required duration modification. The features to be changed, by how much and where are decided by using a set of pre defined parameters.

4.2.3 Synthesis Block

Once the modified frame-wise features are obtained for the vocal tract, it is then used as a system with the LP residual as input (acting as the excitation source) to obtain the desired prosody modified speech. An in-depth explanation on each of the component can be found in [21].


4.3 Results and discussion for FAST study

As evident from Table 3.2, degradation is due to the emotional mismatch, since other characteristics of speech are kept same. Change in emotion of an utterance can be linked to it’s prosody. Hence we considered modifying the prosody using FAST to alter the emotion of an utterance.

Prosody modification can also be done with an emotional utterance as source and neutral emotion as target, but in this study we want to improve performance for emotional utterance during evaluation phase. Therefore we considered modifying prosody of an emotional utterance by using their parallel neutral speech as target to synthesize speech utterances to be used during evaluation phase. Hence the utterances from all 3 emotional test sets (anger, happy and sad) are prosody modified and results are shown in Table 4.1. The same configuration and architecture from Chapter 3 has been used for the baseline systems. In this table, the all emotions column shows the EER obtained when utterances from all the 4 emotional test sets are considered together.

Table 4.1 Performance (in terms of EER) of the baseline systems considered under different emotional conditions after prosody modification

<table>
<thead>
<tr>
<th>System</th>
<th>Neutral</th>
<th>Anger</th>
<th>Happy</th>
<th>Sad</th>
<th>All Emotions</th>
</tr>
</thead>
<tbody>
<tr>
<td>i-vector</td>
<td>22.42</td>
<td>41.06</td>
<td>38.61</td>
<td>43.02</td>
<td>36.27</td>
</tr>
<tr>
<td>i-vector + LDA</td>
<td>17.34</td>
<td>39.63</td>
<td>38.03</td>
<td>39.15</td>
<td>33.55</td>
</tr>
<tr>
<td>i-vector + PLDA</td>
<td>15.57</td>
<td>38.72</td>
<td>36.64</td>
<td>38.54</td>
<td>32.40</td>
</tr>
<tr>
<td>TDNN</td>
<td>7.21</td>
<td>40.37</td>
<td>38.42</td>
<td>35.17</td>
<td>30.30</td>
</tr>
<tr>
<td>LSTM</td>
<td>17.84</td>
<td>42.21</td>
<td>37.81</td>
<td>46.33</td>
<td>35.98</td>
</tr>
<tr>
<td>x-vector</td>
<td>8.35</td>
<td>35.83</td>
<td>33.27</td>
<td>30.16</td>
<td>26.92</td>
</tr>
</tbody>
</table>

From Table 4.1 we can observe that the performance under the mismatch scenario has been heavily affected. Performance under neutral utterances, which are not subjected to FAST, has stayed similar to that obtained in Table 3.3. This suggests that there are some unnecessary changes in the speech environment due to prosody modification, hence we included some prosody modified utterances during training as well. Emotional utterances from the adaptation set was used to generate prosody modified utterances for training. These result are shown in Table 4.2.

From Table 4.2, very little improvements can be seen for LSTM, while performance under TDNN and all i-vector systems suffered degradation compared to Table 3.3. The same pattern for i-vector results can be observed here, decreasing order in the performance from i-vector+PLDA to i-vector+LDA to i-vector. Considering the best performing i-vector model (i-vector+PLDA), we can observe that the performance under neutral, happy and sad degraded, while that under anger showed a little improvement. Interestingly for LSTM, performance under neutral and sad suffered little degradation, while that under anger and happy improved by moderate amounts. For TDNN, we can see that the performance under all 4 emotions suffered degradation. Degradation under neutral is minimal, while that under anger and happy is moderate but the performance under sad utterances suffered huge degradation. Only x-vectors...
were able to model the prosody modified utterances properly and a huge improvement in performance is obtained for them under all 4 emotions.

Table 4.2 Performance (in terms of EER) of the baseline systems considered under different emotional conditions after prosody modification with adaptation

<table>
<thead>
<tr>
<th>System</th>
<th>Neutral</th>
<th>Anger</th>
<th>Happy</th>
<th>Sad</th>
<th>All Emotions</th>
</tr>
</thead>
<tbody>
<tr>
<td>i-vector</td>
<td>22.82</td>
<td>24.37</td>
<td>27.04</td>
<td>23.38</td>
<td>24.25</td>
</tr>
<tr>
<td>i-vector + LDA</td>
<td>19.83</td>
<td>22.29</td>
<td>24.21</td>
<td>23.38</td>
<td>22.70</td>
</tr>
<tr>
<td>i-vector + PLDA</td>
<td>18.26</td>
<td>20.80</td>
<td>22.48</td>
<td>19.91</td>
<td>20.32</td>
</tr>
<tr>
<td>TDNN</td>
<td>8.42</td>
<td>24.81</td>
<td>26.57</td>
<td>21.17</td>
<td>20.51</td>
</tr>
<tr>
<td>LSTM</td>
<td>18.97</td>
<td>19.02</td>
<td>22.17</td>
<td>20.70</td>
<td>21.00</td>
</tr>
</tbody>
</table>

4.4 Summary of prosody modification

The results obtained in Table 4.1 and Table 4.2 asserts that the prosody modification has to be done along with adaptation. After introducing the adaptation data along with prosody modification, DNN x-vector became the best performing architecture. The emotions under which we obtained optimal performance (neutral and sad) earlier, was observed to have degraded after prosody modification while that under the high arousal emotions (anger and happy), which was sub-optimal earlier, was observed to have improved in most cases. This suggests that further approaches have to be developed in order to achieve better overall performance.

4.5 Moving away from prosody modification

The studies undertaken until now suggest that there is a degradation in performance under emotional mismatch between train and test conditions. Modification of features did not provide much improvement when done in isolation. This suggested us to employ use of an adaptation set, consisting of emotional utterances, to be used during training. In Section 4.3, we noted improvement in performance after the introduction of this set. Going forward in this chapter we will use this set more extensively for experiments.

From the literature, we found Recurrent Convolution Neural Network (RCNN) have performed well for a few related tasks [97], hence we investigate their use for our application in the rest of this chapter. We also introduce another adaptation technique and demonstrate it’s effectiveness by improving performance under emotional mismatch.
4.6 Recurrent Convolutional Neural Network: RCNN

The RCNN uses a Recurrent convolutional layer (RCL), which applies convolution operation to the input feature map over discrete time steps [97]. The notion of time step in RCL layers is different from that in the RNNs. In RNNs, the notion of time step relates to the actual notion of time in a sequential input whereas, in a RCL layer, it denotes the number of times convolution operation is applied iteratively.

The activation of a hidden layer $h^{(t)}$ at position $(i, j)$ due to the input $x^{(t)}$ is given by Equation \[4.2\]

$$h^{(t)}(i, j) = \sigma\left( \sum_{i'=-s}^{s} \sum_{j'=-s}^{s} w^f_{k}(i', j')x^{(t)}(i - i', j - j') + \sum_{i'=-s}^{s} \sum_{j'=-s}^{s} w^r_{k}(i', j')h^{(t-1)}(i - i', j - j') + b \right) \quad (4.2)$$

where $w^f_k$, $w^r_k$, and $b$ denote the forward and recurrent convolution kernels and the added bias respectively. $\sigma(x)$ denotes the ReLU activation function.

The weights in a RCL are shared across time steps, which is evident from Equation \[4.2\]. The number of time steps $T$ in RCL layer is a hyperparameter, which controls the context received by an activated unit. For example, higher values of $T$ will mean a wider context to the activated unit but also with the same number of parameters as the weights are shared across time step. An example of a RCL with $T = 3$ is shown in Figure \[4.2\].

![Figure 4.2](image)

**Figure 4.2** Hidden states of a single RCL with $T = 3$. The bottom coloured figure represents input whereas forward and recurrent connections are represented by solid and dash lines respectively.

Input to the RCNN is usually a spectrogram, which has time on x-axis and frequency on y-axis. Hence, there is a dependency along both the axis. The operations applied by the RCL layer leverages
on these multidimensional relations. This characteristic of RCL is in my opinion makes them a strong candidate for LID. A complete RCNN architecture consists of several RCL layers, with max pooling layers in between, followed by a few fully connected layers. This architectural design is similar to a conventional CNN.

4.7 Multistage Training: MST

The task of identifying language from spoken utterance is a classification task, frequently learnt by optimizing over cross entropy loss function by using language labels as ground truth. Cross entropy is calculated individually for a sample, and hence does not take into account the overall representation of a class. On the other hand triplet loss function, shown in Equation 4.3 forms a triplet with respect to the input sample by taking a positive and a negative sample. It’s objective is to minimise the distance between input and positive sample, while maximising the same between input and negative sample. So it is often employed in representation learning tasks [12].

\[ L(A, P, N) = \max(||f(A) - f(P)||^d - ||f(A) - f(N)||^d + \alpha, 0) \] (4.3)

In Equation 4.3, \( f(A) \), \( f(P) \) and \( f(N) \) represent the feature vectors of the input, positive and negative samples, \( ||.||^d \) denotes the distance metric.

A network trained to classify languages can suffer from variations if they are not accounted for during training. Emotion is identified as one such variation which affects the performance of a LID system. It can be understood that degradation is observed due to emotion, causing the representation of utterances to deviate from their representation under neutral emotion, hence deviating from the sample space on which model was trained. The RCL layers can be thought of as feature extractors operating on the raw acoustic features. Hence we can use the triplet loss function over these representations to account for the variability due to emotion. Therefore we first train our network to learn emotion invariant representations, then add the final dense layer with softmax activation for classifying languages. This approach differs slightly from transfer learning as we are training the network on same task but in two stages.

4.8 Experimental setup of RCNN

We have extracted 20D MFCC from 25ms frames with 10 ms shift. A sample input to the RCNN is of dimension 50 x 20, obtained by stacking 50 MFCC frames. We consider the output softmax activation as segment level class probabilities for a segment of 50 frames and average that of all segments of an utterance to get utterance level class probabilities and in turn utterance level predictions.

We empirically found that using convolution filters with small kernel size, such as 3, gave better results. Hence for RCL, we use kernels of size 3 and stride 1 with 32 filters. We use the same kernel size for pooling layers but with stride 2. Also we keep the number of time steps in every RCL as 3.
Each RCL is followed by a max pooling layer. The output of last RCL is flattened and fed into the fully connected (FC) layers. Each FC layer is followed by a dropout layer with training drop probability of 0.7. The idea behind using a higher drop probability is to achieve as little over-fitting as possible while not reaching under-fitting, as the difference between training and testing condition is large. The complete architecture is shown in Figure 4.3, the output of fc_2 layer is l2 normalised and the network till this layer is optimised using the triplet loss function with language labels as target. After this, we add another fully connected layer (fc_3) of dimension equal to the number of classes (7) with softmax activation. Apart from the fc_3 layer, all RCL and FC layers have leaky relu activation. The total number of trainable parameters in this architecture are roughly 1.3 million which is ∼10 times the number of training segments.

4.9 Results and discussion for RCNN and MST study

Degradation in performance of LID systems under emotional mismatch has been identified in the previous chapters. Feature modification, as a result of prosody modification, was applied and it resulted in improvement under some degrading scenarios but overall improvement was not obtained. As noted in Section 4.5 and observed through their description given in Section 4.6, we will use them for the task of LID on our dataset, described in Section 3.2. Results for the RCNN along with the comparison to other baseline systems are shown in Table 4.3. As noted in previous chapters that i-vector+PLDA system always outperform i-vector and i-vector+LDA systems, we will omit their results from rest of the discussion.

Table 4.3 highlights the degradation in performance under anger, happy and sad utterances compared to the neutral ones in case of RCNN as well. It is second in terms of performance under all emotions, after TDNN. Deviation, calculated using Equation 3.2, is 7.68, which is significantly better than that of TDNN and x-vector, the two systems closest to it in terms of performance. This provides an useful in-
Table 4.3 Performance (in terms of EER) of the baseline systems and RCNN considered under different emotional conditions

<table>
<thead>
<tr>
<th>System</th>
<th>Neutral</th>
<th>Anger</th>
<th>Happy</th>
<th>Sad</th>
<th>All Emotions</th>
</tr>
</thead>
<tbody>
<tr>
<td>TDNN</td>
<td>7.13</td>
<td>22.14</td>
<td>20.96</td>
<td>5.9</td>
<td>16.69</td>
</tr>
<tr>
<td>LSTM</td>
<td>17.26</td>
<td>23.86</td>
<td>27.10</td>
<td>19.91</td>
<td>23.68</td>
</tr>
<tr>
<td>i-vector + PLDA</td>
<td>15.41</td>
<td>22.00</td>
<td>18.41</td>
<td>14.03</td>
<td>17.77</td>
</tr>
<tr>
<td>x-vector</td>
<td>8.42</td>
<td>19.29</td>
<td>21.41</td>
<td>12.22</td>
<td>17.18</td>
</tr>
<tr>
<td>RCNN</td>
<td>11.55</td>
<td>22.86</td>
<td>17.99</td>
<td>14.33</td>
<td>16.92</td>
</tr>
</tbody>
</table>

...sight that the performance under RCNN, though degraded under mismatch scenario, has lesser influence of over-fitting while giving optimal performance compared to other systems.

In order to counter the degradation due to emotional mismatch, we had used prosody modification previously in this chapter. We use the same modified utterances described before with RCNN as well. The modified adaptation set utterances are also used for training. Table 4.4 contains the results obtained for this experiment. We can observe, for RCNN, the similar behaviour of degradation in overall performance compared to non modified utterances as that for TDNN and i-vector, though degradation for RCNN is moderately lesser than that for TDNN and i-vector. For RCNN, large degradation can be observed for neutral utterances, while it is negligible for sad utterances. Moderate performance improvements can be observed under anger and happy utterances.

Table 4.4 Performance (in terms of EER) of the baseline systems and RCNN considered under different emotional conditions after prosody modification with adaptation

<table>
<thead>
<tr>
<th>System</th>
<th>Neutral</th>
<th>Anger</th>
<th>Happy</th>
<th>Sad</th>
<th>All Emotions</th>
</tr>
</thead>
<tbody>
<tr>
<td>TDNN</td>
<td>8.42</td>
<td>24.81</td>
<td>26.57</td>
<td>21.17</td>
<td>20.51</td>
</tr>
<tr>
<td>LSTM</td>
<td>18.97</td>
<td>19.02</td>
<td>22.17</td>
<td>20.70</td>
<td>21.00</td>
</tr>
<tr>
<td>i-vector + PLDA</td>
<td>18.26</td>
<td>20.80</td>
<td>22.48</td>
<td>19.91</td>
<td>20.32</td>
</tr>
<tr>
<td>RCNN</td>
<td>20.92</td>
<td>19.13</td>
<td>16.76</td>
<td>14.69</td>
<td>18.49</td>
</tr>
</tbody>
</table>

Table 4.5 Performance (in terms of EER) of the baseline systems and RCNN considered under training with adaptation data and multi stage training conditions

<table>
<thead>
<tr>
<th>System</th>
<th>Neutral</th>
<th>Anger</th>
<th>Happy</th>
<th>Sad</th>
<th>All Emotions</th>
</tr>
</thead>
<tbody>
<tr>
<td>TDNN</td>
<td>6.85</td>
<td>8.86</td>
<td>11.68</td>
<td>3.02</td>
<td>7.60</td>
</tr>
<tr>
<td>LSTM</td>
<td>13.02</td>
<td>12.67</td>
<td>20.43</td>
<td>10.36</td>
<td>14.12</td>
</tr>
<tr>
<td>i-vector + PLDA</td>
<td>11.27</td>
<td>7.86</td>
<td>10.78</td>
<td>7.69</td>
<td>9.33</td>
</tr>
<tr>
<td>x-vector</td>
<td>6.28</td>
<td>10.57</td>
<td>10.78</td>
<td>7.69</td>
<td>8.78</td>
</tr>
<tr>
<td>RCNN</td>
<td>10.27</td>
<td>11.14</td>
<td>11.24</td>
<td>5.73</td>
<td>9.74</td>
</tr>
<tr>
<td>RCNN + MST</td>
<td>9.58</td>
<td>5.87</td>
<td>9.14</td>
<td>5.43</td>
<td>7.47</td>
</tr>
</tbody>
</table>

We train the baseline systems and the RCNN network by including the adaptation set utterances along with train set utterances, and the results are shown in Table 4.5. From that table, we can observe
huge improvement in performance for all the systems. This highlights that some emotional data must be present during training to account for the variation. Except LSTM, all other systems are able to outperform best performance until here (x-vector from Table 4.4) when adaptation set is used. Interesting to note the performance improvement when the multi stage training method is used. We first train the RCNN network by optimising the triplet loss function on adaptation set. The triplet loss network has been restricted to train for only 10-15 epochs, as we do not want the embeddings to overfit the adaptation data. Then the final training on combination of train and adaptation set, with the softmax layer is done and it converges in around 85-100 epochs. Our proposed RCNN when subjected to multi stage training has given the best performance for anger, happy as well as best overall performance.

Table 4.6 shows the deviation of EER for emotional utterances (anger, happy and sad) with respect to the EER under neutral utterances. We can see that the deviation has decreased for all systems because of the use of adaptation set. Interesting to observe that among the two best performing systems noted in Table 4.5, RCNN when done with multi stage training gives lesser deviation compared to TDNN.

Table 4.6 Deviation in performance of the LID systems from Table 4.5 calculated using Equation 3.2

<table>
<thead>
<tr>
<th>System</th>
<th>deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>TDNN</td>
<td>3.74</td>
</tr>
<tr>
<td>LSTM</td>
<td>4.54</td>
</tr>
<tr>
<td>i-vector + PLDA</td>
<td>2.86</td>
</tr>
<tr>
<td>x-vector</td>
<td>3.68</td>
</tr>
<tr>
<td>RCNN</td>
<td>2.72</td>
</tr>
<tr>
<td>RCNN + MST</td>
<td>3.22</td>
</tr>
</tbody>
</table>

### 4.10 Summary of RCNN and MST

From Table 4.5 and Table 4.6, RCNN when done with multi stage training gives best performance and lesser deviation in performance compared to the other systems with comparable performance. In our view RCNN seems to perform well for LID due to being able to model the multidimensional relations in the input spectrogram and the initial training with the adaptation data made it robust to emotional variations.
Chapter 5

Conclusions and Future Scope

5.1 Conclusions

In this thesis, we addressed the degradation in performance of LID due to mismatch in terms of emotional utterances during train and test. To accomplish this, database was pooled from various sources. Additionally, this study used the state-of-art systems such as i-vector, TDNN, LSTM, and DNN x-vector architectures to build the baseline LID systems.

The conclusions drawn from this thesis are as follows:

- Degradation in performance is present under emotional mismatch for all considered baseline systems.

- The results show the similarity in performance for neutral-sad and happy-angry emotion pairs. Further, for high arousal speech (anger and happy) the LID systems show poor performance under mismatch. On the other hand, the LID systems showed better performance for low arousal speech (neutral and sad) under the same mismatch.

- We demonstrated the usage of prosody modification for this problem. Prosody modification had to be done with adaptation set but still with little to no improvements.

- A drawback of prosody modification is that both neutral and emotional utterances are required. This may not be possible every time or in a real world scenario.

- RCNN showed promising performance under the mismatch scenario, slightly behind only TDNN in terms of EER. But RCNN showed lesser deviation of EER compared to TDNN and x-vector, hinting at better generalisation.

- Degradation in performance for emotional mismatch can be reduced by including emotional utterances (adaptation set) while training, which is a form of adaptation.

- The size of adaptation set was smaller compared to the size of train set, in terms of both duration and number of utterances. Considering adaptation set included utterances from all the emotions,
number of utterances for each emotion is therefore very less compared to neutral utterances. This suggests that the emotional variations can be learnt easily but if not handled can lead to large degradation in performance.

- Introducing multistage training during the training of RCNN improved the performance and in turn achieved the best performance overall. This train method can be considered as a type of adaptation as well.

- Care must be taken while preparing datasets for LID by accounting for the emotional variation as well.

5.2 Future Scope

Following are the directions for future work:

- Availability of a standard dataset for this study was the biggest limitation, and hence train data for future LID development should lay emphasis on it.

- Further study needs to be done by studying the feature representation of an utterance under different emotions, to understand the degradation and possible overlap among languages.

- A drawback of prosody modification is that it requires both neutral and emotional utterances, this can be handled by training a GAN like architecture to perform prosody modification.

- The RCNN architecture is one of the many unconventional CNN architectures introduced recently, another example is Gated-CNN. These architectures should be explored in other areas of speech processing. Advantages of Convolution based architectures is that they require lesser parameters and give better generalisation.

- Training methodologies such as the one described in this thesis can be introduced to other areas of speech processing as well. It can also be applied to other similar architectures.
Publications

Related Publications


Other Publications


Bibliography


