Indian Language Identification using Deep Neural Networks

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

Master of Science
in
Electronics and Communications Engineering by Research

by

KAMSALI VEERA MOUNIKA
201334011
`mounika.kv@research.iiit.ac.in`

International Institute of Information Technology
Hyderabad - 500 032, INDIA
June 2018

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled "**Indian Language Identification using Deep Neural Networks**" by KAMSALI VEERA MOUNIKA, has been carried out under my supervision and is not submitted elsewhere for a degree.

Date	Adviser: Dr. Anil Kumar Vuppala

To Mom and Dad

Acknowledgments

First and foremost, I would like to express my sincere gratitude to my advisor, Dr. Anil Kumar Vuppala for accepting me as a student under his guidance and devoting his time and effort in advising me. I would like to thank him for constantly pushing my limits and teaching me to always have a grand vision for my work.

My heartfelt thanks to Dr. Suryakanth V Gangasetty, for his constant encouragement and faith in me towards every step I took. I would like to thank him for his patience, motivation and immense support to venture out of my comfort zone and take up challenging tasks.

I thank Prof. Raj Reddy for advising me to take the path that I have taken (DNN). I offer my sincere acknowledgements to Prof. B. Yegnanarayana who inspired me through his passion for teaching. I am grateful to him for making me realize the importance of writing and note taking.

Special thanks to Sivanand Achanta for carrying out elaborate discussions and experiments with me that have shaped my understanding. Thanks for constantly coming up with ideas for improving the LID system, and for your suggestions at various point of time during my research.

I thank Hari Krishna, Harsha, Bhanu Teja, Sudarshan and Sai Krishna for helping me learn the basics when I joined the lab. I thank Lakshmi for being a wonderful mentor cum co-author to my work. I thank Ravi Kumar for his fun-filled co-operation at data collection. I thank Ayushi, Ganga Mohan, Rambabu, Rama Krishna, Vishala, Tejas, Ravi Shankar, Nivedita, A.Raju, V.Raju, and Harika for being wonderful colleagues at work.

Endless thanks to Jahnavi, Tejaswini, and Deepya who have always been there to help me. Special thanks to Vasudha, Vamsi, and Karthik for helping me finish thesis on time. I thank Tarun for proofreading my papers. I cherish my memories with Sushma, Nancy, Pratyusha, Divya, and Sushmita.

Finally, I would like to express my profound gratitude to my family for their love and support. Their immense encouragement and confidence throughout my years of study, drove me to constantly give my best in every endeavor. I am blessed to have them for being with me and supporting every dream of mine. This accomplishment would not have been possible without them. I dedicate this thesis to my parents Brahmam.K.V and Sujatha.K.V. "The apple never falls far from the tree ."

Abstract

The objective of a language identification system is to identify the language in a given spoken utterance. Based on the availability of corresponding text to a given utterance, the approach for language identification can either be implicit or explicit. In an implicit case, we only have access to the acoustic signal and statistical models are built over the features extracted from the acoustic signal. In an explicit case, corresponding text is available for the given utterance and rule-based approaches are used to identify the language. This thesis presents an implicit approach based on deep neural network (DNN) models for language identification.

DNN models have been recently proposed for the task of language identification. In this thesis, DNN models have been explored in the Indian scenario. While DNN based approach is inherently a frame based one, we propose an attention mechanism based DNN architecture for utterance level classification there by efficiently making use of the context at model level. This approach can be viewed as an alternative to the state-of-the-art Gaussian mixture model based iVector baseline system. On contrary to previously published works using DNNs in LID, DNN equipped with attention mechanism has the advantage of discriminative end-to-end training without any generative component or post-averaging of posterior probabilities obtained at frame level.

Several experiments have been performed on a dataset consisting 13 of the official Indian languages with 120 hours of training data. Evaluation of models were performed on 30 hours of testing data with about 2.5 hours for each language. Equal error rate (EER) per language is used as the performance metric. From the results, it is found that the DNN with attention mechanism outperforms the regular DNN and i-vector baseline system indicating the effectiveness of the attention mechanism.

Further, a combination of excitation source and vocal tract system features have been explored for the task of language identification. A better performance gain has been achieved by combining the features using a late fusion mechanism. This result indicates the complementary nature of the excitation source information to that of the widely used vocal tract system information.

Contents

List of Figures

List of Tables

List of Abbreviations

MFCC	Mel Frequency Cepstral Coefficients
LPCC	Linear Prediction Cepstral Coefficients
RCC	Residual Cepstral Coefficients
DNN	Deep Neural Networks
DNN-WA	Deep Neural Network with Attention
GMM	Gaussian Mixture Model
SVM	Support Vector Machine
EER	Equal Error Rate
LID	Language Identification
ASR	Automatic Speech Recognition
SR	Speaker Recognition

Chapter 1

Introduction

Language Identification (LID) refers to the task of identifying the language being spoken by a speaker in a given utterance [1]. Broadly, applications of a language identification system can be divided into two categories, namely front-end for human listeners and front-end for machines. As a front-end for human listeners, a language identification system is required to route an international call to a human operator who is fluent in the caller's language. As a front-end for machines, the LID system could be used as a multilingual voice controlled information retrieval system for machines. As research in automatic speech recognition progresses, language identification system would be necessary for any multi-lingual speech recognition system. In case of a multi-lingual speech recognition task, using a LID system as a front-end to identify the language of the given utterance improves the performance of the speech recognizer by reducing the complexity by directly processing the speech over the identified language rather than running over several languages. Especially, in an Indian scenario, where almost every state has a language of its own and every language having hundreds of dialects, development of a language identification system becomes crucial.

1.1 Issues in Language Identification

While any spoken utterance has underlying lexical, speaker, channel, environment and other such variations, the goal is to recognize the identity of the language invariant to these factors. Any language is a sequence of phones/sound units. The possible differences among different languages are the phoneme inventory, frequency of occurence of phonemes, acoustic signature and the duration of the sound unit in different languages, and intonation patterns at higher levels. The performance of any LID system thus depends on the amount of reliable information extracted called the features that are task-specific. Statistical models are required to model the distribution of such features for a corresponding language. It is clear from the literature that as long as the languages considered are different from each other in a way that, the phoneme set is quite different across the languages, identification becomes easier. But any overlap between phoneme set of two or more languages, makes it challenging to identify a language. One such case is dealing with Indian languages, since most of the Indian languages have same origin, although there are few unique sounds, all the languages share a common phoneme set. Adding to that is the diversity, almost every state has a language of its own and every language has hundreds of dialects. Developing a language identification system that can tolerate such variability becomes extremely challenging.

1.2 Issues addressed in the thesis

In this thesis, we address the issues involved in developing an implicit language identification system using deep neural networks. An Indian language database consisting 13 of the official Indian languages recorded in read mode . As mentioned in the previous section, the performance of a LID system majorly depends on the amount of reliable information captured and the statistical model used to capture the distribution of such language-specific information. Apart from that, a few necessary features as mentioned below have been taken care while building LID system:

1. The system is not biased towards any language or a group of languages
2. Development of the system is simple
    - adding a new language into the existing system should be simple
    - amount of data required to extract the language cues should be small
3. The system performs well irrespective of variations in :
    - channel and environment
    - accent
    - speaker

The following are the major contributions of the thesis:

- An Attention mechanism based deep neural network architecture for utterance level classification is proposed for language identification
- A combination of excitation source information and the vocal tract system information is explored for the task of language identification
- Contributed towards data collection for Indian language database

1.3 Thesis organization

In Chapter 2, an overview of the language identification task is presented. Firstly a brief review on the applications that require language identification are mentioned, then the related works in language identification are discussed. Diferent discriminative features and statistical models that are widely used to solve the task of language identification have been discussed.

In Chapter 3, the baseline approaches are discussed followed by the description of dataset used for the language identification task in this work. Baseline approaches are then analyzed followed by the summary and conclusion from the experiments conducted.

In Chapter 4, the proposed deep neural network architecture called the attention mechanism is explained in detail. Next, the architectural choices and training procedures are mentioned followed by the experiments and results section. Analysis on experimental results is given followed by summary and conclusions.

In Chapter 5, the performance of a langauge identification system by combining the evidences from vocal tract system and excitation source fetaures is examined. Individual systems are built using the vocal tract system features, excitation source features and finally by combining the evidences coming from both the features using a late fusion mechanism another system is built.

Chapter 6 concludes the thesis by providing a summary, important conclusions and future work.

Chapter 2

Approaches to Language Identification

2.1 Overview of Language Identification

Based on the availability of corresponding text to a given utterance, the approach for language identification can either be implicit or explicit. In an explicit case, corresponding text is available for the given utterance and rule-based approaches are used to identify the language. In an implicit case, we only have access to the acoustic signal and statistical models are built over the features extracted from the acoustic signal. This thesis presents an implicit approach towards language identification.

Implicit LID approaches capture the essential differences across languages by modeling the distributions of language-discriminative features extracted from the acoustic speech signal. Usually a language independent set of spectral features will be extracted from a set of training utterances. A statistical classifier is used to identify the language-dependent patterns in these features. There is no mapping to explicit linguistic units such as phones. Except the language-specific audio data, no language-specific knowledge in form of transcriptions is needed for training, making this type of architectures easily expandable to further target languages.The language related information is extracted from a broad phonetic transcription instead of using acoustic features extracted from speech signal.

Approaches using phonotactic and prosodic features derived from the speech signals have been explored for LR in [2][3][4][5]. Lately, following the trend in speaker recognition, i-vector based approaches have proven to be successful for LR [6]. The i-vector based approach uses Gaussian mixture models (GMMs) for acoustic modeling. The use of deep neural networks (DNN) in acoustic modeling for speech recognition instead of traditional GMMs has resulted in significant performance gains [7]. This has led researchers in the LR community to explore i-vector extraction using DNNs for acoustic modeling. Application of DNNs in this way for LR has been recently investigated in [8][9].

Broadly, there are two ways in which neural networks can be used for LR: (1) to get posteriors and use a back-end classifier (i.e., as an acoustic model), and (2) as an end-to-end LR system. In [8], the authors proposed the use of convolutional neural networks for posterior extraction and then trained the i- vector based LR systems. They have reported improvements in noisy conditions over the conventional i-vector based systems.

In [9], a single DNN acoustic model for both speaker and lan- guage recognition tasks has been trained and significant gains have been achieved in both the tasks simultaneously. On the other hand, an end-to-end LR system using 8-layer DNN was explored in [10] and excellent performance improvements over i-vector baseline on the standard NIST LRE 2009 dataset have been reported. In this work, we propose to further explore DNN and a modified DNN architecture for end-to-end LR task.

2.2 Relation to prior work

Neural networks for LID in Indian languages have been proposed earlier in [11]. Given large amounts of training data per language (<10 hrs), the DNN based approach outperforms the i-vector based approaches as was reported in [10]. However, one drawback of the DNN based systems is that, the decision is taken at every frame and the context used is fixed whilst language id is usually assigned to a whole utterance. To better capture the temporal context and to do utterance wise classification, recently, recurrent neural network (RNN) based LR has been proposed [12]. This technique uses long short-term memory (LSTM) [13] cells as RNN units which have been shown to be effective in memorizing long temporal context. The LSTM-RNNs are trained using back-propagation through time, with targets set for every 5 frames. However, the LSTM-RNNs are computationally intensive to train and also because of the sequential nature of RNNs they are not parallelizable. Although computation can be reduced using simple RNNs [14] instead of advanced RNN units like LSTMs, the sequential nature is still preserved. In this paper, we propose a recently introduced architecture in [15] for alleviating the above problems for LR. In [15], authors proposed a feed-forward deep neural network with attention for solving some memory problems involving pathological long range dependencies. We refer to this architecture as DNN with attention (DNN-WA) in the rest of the paper. The advantage of DNN-WA is that while it is able to memorize, it is also parallelizable because of the strictly feed-forward architecture with no recurrent connections. We have explored this architecture in the context of LR for classifying entire utterance rather than emitting a frame level decision and finally combining the decisions as will be done for a DNN. This also takes into account the problem of using contextual information for LR. In addition, as will be shown later in section 5, this architecture allows us to see what part of the input sequence plays a crucial role in making the decision. The attention mechanism allows us to peep into the input feature frames that are more important for LR. To summarize the contributions of this work, firstly we have explored DNN-WA [15] architecture for utterance level LR and secondly investigation of effect of depth of DNNs for LR has been carried out.

2.3 Features for Language Identification

The fundamental issue of the automatic language identification is to explore the effective discriminative cues for languages.

According to their level of knowledge abstraction, LID features fall into five groups: syntactic, lexical, phonotactic, prosodic and acoustic. Lower level features, such as acoustic spectral feature, are easy to obtain but not robust as speech variations due to the presence of speaker or channel variations. Higher level features, such as lexical/syntactic features, rely on large vocabulary speech recognizer, which is language and domain dependant, and is therefore difficult to generalize across languages and domains. Phonotactic features which refers to the rules that govern the combinations of the different phones in a language, become a trade-off between computational complexity and performance. It is generally agreed that phonotactics, carry more language discriminative information than the phonemes themselves. They are extracted from output of a phoneme recognizer, which is supposed to be more robust against the effects of speaker and channel variations than spectral features. In [16], [17] and [18], researchers have focused on acoustic-prosodic-phonotactic features .

Different levels of information provide complementary language cues.The prosodic features are more effective for shorter utterances while the phonotactic features work better for longer utterances. Features that are derived from different sources of language information present in the speech signal can be grouped into three categories. They are: (1) features related to the anatomical differences in the vocal tract, (2) features related tro the excitation source of the vocal tract, and (3) features related to differences in the style of speaking.

The structural differences in the shape and size of the vocal tract represent the anatomical differences that could hold language information. Speech is produced when varying vocal tract system is excited by the varying excitation source. Both the source and system keep changing with time. But the time-varying vocal tract is assumed to be in quasi-stationary state for a short-duration of about 10-30 ms. The spectral content of the speech segment is used to characterize the vocal tract shape. The periodic oscillation of the vocal folds is a major source of excitation for the vocal tract system. Oscillations of these folds contain some language information along with the speaker information. The differences in speaking habits are due to the manner in which the speakers have learned to use their speech production mechanism and the language constraints. These differences indicate the possible variations from the speech production point of view.

Among the different sources of language information, speaking habit and the style of speaking are considered as higher-level information. These features are hard to quantify with the existing techniques. Language identification systems rely on the features derived from the vocal tract shape and its excitation characteristics to represent the language information[19],[20]. Apart from these, prosodic features like intonation, rhythm and stress help in identifying languages[21]. Following subsections discuss some of these features.

2.3.1 Short-term Spectral Features

Short-term spectral features primarily characterize the vocal tract shape, along with some excitation source information embedded within. They are extracted from the speech signal of duration 10-30 ms. These features are realized in different forms such as Mel-frequency cepstral coefficients, linear prediction coefficients, residual cepstral coefficients etc.

2.3.1.1 Mel-Frequency Cepstral Coefficients

Mel-frequency cepstral coefficients (MFCC) are the most widely used approaches of feature extraction technique in speech processing tasks. The shape of the vocal tract manifests itself in the envelope of the short time power spectrum, and the job of MFCCs is to accurately represent this envelope. Given a speech signal, MFCC feature vectors are obtained by following the steps provided in the Fig 2.1

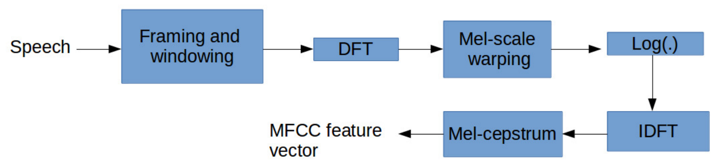

Figure 2.1: Computation of MFCC feature vector for a speech signal

Given spoken utterance is first segmented into frames of about 20-30 ms and within this duration, the vocal tract system is said to be in quasi-stationary state. As it is assumed to be in quasi-periodic state, it satisfies the condition of LTI to apply DFT over such a frame. For each of such frames, a periodogram estimate of the power spectrum is calculated and the mel filterbank is applied to the power spectra. The filterbank energies are computed and the logarithm of all filterbank energies is computed. Then the DCT is applied over all the filterbank energies. We choose to keep 12 coefficients along with the total energy of each filter. These 13 cepstral copefficients form the MFCC feature vector.

2.3.1.2 Linear Prediction Cepstral Coefficients

According to Linear prediction (LP) analysis, it is believed that any particular speech sample may be predicted by a linear combination of its previous samples. The number of previous samples considered to predict the current sample is known as the order of the prediction. The weights applied to each of the previous speech samples are known as the linear prediction coefficients (LPC). They are calculated so as to minimize the prediction error also called a the LP residual. LPC features are known to model the vocal tract system and hence used for language, speaker, spech recognition tasks.

2.3.1.3 Residual Cepstral Coefficients

The language information associated with the excitation of the vocal tract has been a subject of interest in language identification tasks. LP analysis models the vocal tract (system) parameters and hence the information about the excitation (source) of the vocal tract is present in the residual signal. Cepstra obtained from the residual signal called the RCC features are used to represent such an information about the excitation source. The higher order relations among the samples of the speech signal remain in the residual signal that could be used in identifying the language in an utterance.

2.3.2 Long-term Features

Language identification becomes easier as the duration of speech increases. But, for an ideal LID system to be used in real scenario, the test case has to be as small as possible and without degradation in accuracy. Language information exists in longer durations of speech. Inorder to identify a language from shorter duration, without compromising the accuracy of LID system, there is a need for efficient represntation of cepstral dynamic trajectory over some short segments of speech. One such technique is the computation of shifted delta cepstra for the given speech signal. Additional temporal information of few previous and few next frames can be included in current frame by computing the shifted delta ceptra for each frame with a context of n frames where n is typically 5, 7, 11 etc.

2.3.2.1 Shifted Delta Cepstral Features

The most widely used features for the task of LID are mel frequency cepstral coefficients (MFCC). Dynamic information from MFCC could be obtained by including static + delta + delta delta features. Computing the delta cepstra in this way captures temporal information over fewer frames alone i.e less context. An improved LID performance could be obtained using SDC feature vectors created by stacking delta cepstra compounded across multiple speech frames [22] thus capturing the context to a greater extent. The computation of SDC is illustrated in Fig. 2.2

SDC features are specified by 4 parameters, N, d, P, k, where, N is the number of cepstral coefficients considered from each frame, d represents the time delay and time advance for delta computation. P represents the time shift between consecutive delta computed blocks and k represents the number of blocks whose delta coefficients are concatenated together, where i = 0,1,2..k-1.

$$\Delta c(t) = c(t + iP + d) - c(t + iP - d) \tag{2.1}$$

Given a spoken utterance, SDC feature vector is computed for every frame time t considering the context across P*k frames based on the configuration i.e 7-1-3-7 configuration captures the context over 21 frames and 10-1-3-3 captures the context over 9 frames.

2.3.3 Prosodic Features

Prosodic differences among world languages include variations in intonation, rhythm, and stress. These variations are represented using features derived from fundamental frequency (F0) contour, duration, and energy contour. Human perception experiments also suggest that prosodic features are informative language cues [23]. Within the first few days of life, infants are able to differentiate between

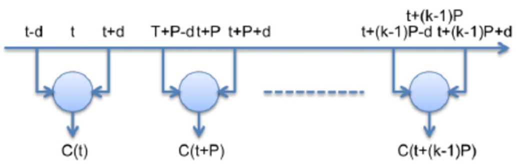

Figure 2.2: Computation of SDC feature vector at frame t for configuration N-d-P-k

the sounds of their native language and other languages, possibly by attending to prosodic properties of speech [24]. These phenomena indicate that the listener in the absence of higher level knowledge of a language, he presumably relies on lower level constraints. However, prosodic feature has not been fully exploited in LID [25].

2.4 Models for Language Identification

Language identification is a classification task and this can be performed using either a genrative model like Gaussian mixture model or discriminative model like support vector machine or neural networks. Generative classifiers learn a model of the joint probability, $p(x, y)$, of the input x and the label y, and make their predictions by using Bayes rules to calculate $p(y|x)$, and then picking the most likely label y. Discriminative classifiers model the posterior $p(y|x)$ directly, or learn a direct map from inputs x to the class labels.

2.4.1 Generative Models

2.4.1.1 Gaussian Mixture Model and Universal Background Model

The most widely used modeling technique for classifying sounds of different languages is the Gaussian mixture model [26] [2]. Under the GMM assumption, each feature vector v_t at frame time t is assumed to be drawn randomly according to a probability density that is a weighted sum of multi-variate Gaussian densities:

$$g\left(\frac{x}{\mu}, \sigma\right) = (2\Pi)^{\frac{D}{2}} \left|\sigma\right|^{\frac{-1}{2}} e^{-} \left(\frac{1}{2} \left(x-\mu\right)^{T} \sigma^{-1} \left(x-\mu\right)\right) \tag{2.2}$$

During identification process, an unknown speech utterance is firstly segmented into smaller frames and feature vectors are extracted from each frame, and then the feature vectors are evaluated by each Gaussian mixture models of the target languages, the log likelihood scores for each target languages are calculated. The unknown speech is classified to the language that generates the highest log likelihood score. Lincoln labs GMM system uses Shifted Delta Cepstra (SDC) feature vectors as input. In their system, a single universal background model (UBM) with 2048 mixtures is firstly trained from the entire train set, then language-dependent model for each target language is adapted from the UBM using the language specific portions of the data. Consider about the duration of the test speech segment, three sets of duration dependent Gaussian classifiers are trained from the training data with specific duration (i.e. 30 seconds,10 seconds and 3 seconds). According to the length of the test speech segment, the test segment is evaluated on one set of these duration based GMM classifiers and language scores are generated, and these language dependent scores are converted to log likelihood ratios by subtracting the scores generated from the UBM model. These log likelihood ratios are compared to make final language identification decision.

2.4.2 Discriminative Models

2.4.2.1 Support Vector Machines

Support Vector Machines are based on the concept of decision planes that define decision boundaries. A decision plane is one that separates between a set of objects having different class memberships. Classification tasks based on drawing separating lines to distinguish between objects of different class memberships are known as hyper plane classifiers. Support Vector Machines are particularly suited to handle such tasks. The original objects(data) mapped are rearranged, using a set of mathematical functions, known as kernels. The process of rearranging the objects is known as mapping (transformation). Note that in this new setting, the mapped objects are linearly separable and, thus, instead of constructing the complex curve, all we have to do is to find an optimal line that can separate the two objects.

Support vector machines are supervised binary classifiers. Proposed in [27], they are based on the idea of finding, from a set of learning examples X=$\{(w_1, y_1), (w_2, y_2), ...(w_N, y_N\}$, the best linear separator H for distinguishing between the positive examples and negative examples. When the kernel function is used the SVM separator is defined as follows.

$$f(x) = \sum \alpha_i t_i K \underset{i=1}{\overset{N}{(x, x_i)}} + b_i \tag{2.3}$$

where a and b are the SVM parameters set during the training step. Several kernel functions with the best set of results obtained by using the cosine kernel function are considered. This kernel is linear kernel and computed as follows :

$$K(x, y) = b(x)^t, b(y) \tag{2.4}$$

where w_1 and w_2 correspond to two i-vectors. There are two strategies to extend the SVM approach to a multi-class problem. The first strategy is the one Vs one and the second one is the one Vs all strategy. In the first approach, we estimate the separator between the target class (language) and every other class considering one at a time. In the second approach, we estimate the separator between the target class and other classes considering all classes at a time.

2.4.2.2 Artificial Neural Network Models

Neuron is the basic unit of a neural network model. Artificial neural network models (ANN) consist of interconnected neurons, and the interconnection between any two neurons has a weigh associated with it. The capability of a neural network model to discriminate between patterns of different classes is exploited for language identification task recently[8,9]. Broadly, there are two ways in which neural networks can be used for LID: (1) to get posteriors and use a back-end classifier (i.e., as an acoustic model), and (2) as an end-to-end LR system. In [8], the authors proposed the use of convolutional neural networks for posterior extraction and then trained the i-vector based LR systems. They have reported improvements in noisy conditions over the conventional i-vector based systems. In [9], a single DNN acoustic model for both speaker and language recognition tasks has been trained and significant gains have been achieved in both the tasks simultaneously. On the other hand, an end-to-end LR system using 8-layer DNN was explored in [10] and excellent performance improvements over i-vector baseline on the standard NIST LRE 2009 dataset have been reported. In this work, we propose to further explore DNN and a modified DNN architecture for end-to-end LID task.

2.5 Performance Measurement

Identification is the process of determining which language has been spoken in a given speech utterance. The system performs a 1:N classification, where, N is the total number of classes i.e languages. Generally it is assumed that all the signals come from a fixed set of known languages, thus the task is often referred to as closed-set identification. By adding a none of the above option to closed-set identification makes it becomes an open-set identification task. The performance of the LID system is evaluated with a parametric called equal error rate (EER). A brief description of EER is given in the below paragraph.

2.5.1 EER

In a classification task when there is a data imbalance across classes measuring the performance in terms of accuracy can be misleading, i.e, the class with more number of examples influences the accuracy. To avoid this we use equal error rate to measure the accuracy. Equal error rate treats identification task as multiple binary classification tasks. The classification performance of binary classification task is measured by

- True positive (TP) is the correct positive prediction

- False positive (FP) is the incorrect positive prediction
- True negative (TN) is the correct negative prediction
- False negative (FN) is the incorrect negative prediction

The measurements False Acceptance ratio (FAR) and False Rejection ratio (FRR) are give by:

$$FAR = FP/(FP + TN) \tag{2.5}$$

$$FRR = FN/(FN + TP) \tag{2.6}$$

FAR is the percentage of the test utterances that are not in certain languages but are classified as true for those language. It also called false alarm rate. FRR is the percentage of the test utterances that in certain languages but classified as false for those languages. Sometimes we also call it as miss classification rate. Both FAR and FRR depend on the threshold value considered, the point at which both these error measure tend to be equal is termed as the equal error rate (EER). The value indicates that the proportion of false acceptances is equal to the proportion of false rejections. The lower the equal error rate value, the higher the accuracy of a LID system.

Chapter 3

Baseline Approach : Generative Model

3.1 Gaussian Mixture Model (GMM)

Gaussian mixture can be viewed as overlapping Gaussian functions with an ability to reflect the short-term spectral density of a language. The spectral density of a speech sample from multivariate Gaussian distribution can be given by :

$$g\left(x|\mu,\sigma\right) = \left(2\Pi\right)^{-\frac{D}{2}} \left|\sigma\right|^{\frac{-1}{2}} e^{-}\left(\frac{1}{2}\left(x-\mu\right)^{T}\sigma^{-1}\left(x-\mu\right)\right) \tag{3.1}$$

with the distributed means μ, covariance defined by σ and T represents the number of samples in an utterance.

The information of a language captured by a feature vector is fitted using a number of Gaussian distributions. A number of Gaussian components are needed due to varying nature of the speech signal. The mixture of all the Gaussians used to fit the data of a language is termed as a Gaussian mixture model. The basic assumption made in using a GMM is that, each feature vector extracted from the test speech segment was produced by using GMM components and each component is given a weight. The number of components used to fit the language feature vectors is generally found using the trial and error method. Due to their ability to scale and train large dataset, high performance and their probability framework, GMMs are better modeling techniques for LID [26]. GMM uses maximum likelihood and maximum aposteriori approaches to estimate the parameters like means, variances and Gaussian weights.

3.1.1 Maximization Algorithm

A model M out of all the N registered language models (13 here), that matches best to the distribution of current feature vector is estimated. This mode if estimation is termed as maximum likelihood estimation(MLE). Through the process, the model parameters that maximize the likelihood of the GMM are estimated given the training data. The GMM likelihood for a given sequence of vectors X is given in the following equation :

$$p(X \mid \lambda) = \prod_{t=1}^{T} \sum_{c=1}^{C} w_c g\left(x_t \mid \mu_c, \sigma_c\right) \tag{3.2}$$

$$Mixtureweights : \bar{p_i} = \frac{1}{T} \sum_{t=1}^{T} p\left(i \mid \vec{x_t}, \lambda\right) \tag{3.3}$$

$$Means : \vec{\mu_i} = \frac{\sum_{t=1}^{T} p\left(i \mid \vec{x_t}, \lambda\right) \vec{x_t}}{\sum_{t=1}^{T} p\left(i \mid \vec{x_t}, \lambda\right)} \tag{3.4}$$

$$Variance : \vec{\sigma_i^2} = \frac{\sum_{t=1}^{T} p\left(i \mid \vec{x_t}, \lambda\right) \vec{x_t}^2}{\sum_{t=1}^{T} p\left(i \mid \vec{x_t}, \lambda\right)} - \vec{\mu_i^2} \tag{3.5}$$

This process of estimation is done iteratively using an algorithm called expectation maximization. The GMM parameters i.e means ($\hat{\mu_i}$), variances(σ_i^2) and mixture weights (p_i)get updated in each iteration. Once the parameters are estimated, the final step of maximum likelihood estimation is to obtain the aposteriori probability for every feature vector. Bayes aposteriori rule is used to estimate the aposteriori. In order to obtain a better performance from GMM, an extension to the basic GMM model has been introduced in [29] called the universal background model(UBM). More details about a UBM has been given in the next section. The performance of GMM based LID systems is give by

Table 3.1: LID Accuracy using GMM

# Gaussian Components	Accuracy(in %)
64	43.16
128	44.50
256	47.33
512	48.00
1024	35.66

3.2 Universal Background Model (UBM)

The UBM was first introduced as an alternate GMM model. Later, in [29], the UBM was used as an initial model for the enrollment phase. In this approach, a language-specific GMM is adapted or derived from the UBM using Bayesian adaptation [30]. In contrast to performing maximum likelihood training of the GMM for an enrollment language, this model is obtained by updating the well-trained UBM parameters. This relation between the language model and the background model provides better performance than independently trained GMMs and also lays the foundation for the language model adaptation techniques that were developed later. Formulations of this approach has been mentioned below.

Let X = x_n | n = 1...T denote the set of acoustic feature vectors obtained from the enrollment language l. Given a UBM as in (1) and the enrollment speakers data X, at first the probabilistic alignment of the feature vectors with respect the UBM components is calculated as :

$$p\left(g \mid x_n, \lambda_0\right)=\frac{\prod_g p\left(x_n \mid g, \lambda_0\right)}{\sum_{g=1}^M \prod_g p\left(x_n \mid g, \lambda_0\right)}=\gamma_n(g) \tag{3.6}$$

Next, the values of values are used to calculate the sufficient statistics for the weight, mean, and covariance parameter as

$$N_s(g)=\sum_{n=1}^T \gamma_n(g) \tag{3.7}$$

$$F_s(g)=\sum_{n=1}^T \gamma_n(g)\, x_n \tag{3.8}$$

$$S_s(g)=\sum_{n=1}^T \gamma_n(g)\, x_n x_n^T \tag{3.9}$$

These quantities are known as the zero, first, and second-order BaumWelch statistics, respectively. Using these parameters, the posterior mean and covariance matrix of the features given the data vectors X can be found as

$$E_g\left[x_n \mid X\right]=\frac{F_s(g)}{N_s(g)} \tag{3.10}$$

$$E_g\left[x_n x_n^T \mid X\right]=\frac{S_s(g)}{N_s(g)} \tag{3.11}$$

The maximum a posteriori (MAP) [31] adaptation update equations for weight, mean, and covariance, (3), (4), and (5), respectively, are given below.

$$\hat{\prod_g}=\left[\alpha_g N_s(g)/T+\left(1-\alpha_g\right)\prod_g\right]\beta \tag{3.12}$$

$$\hat{\mu_g}=\alpha_g E_g\left[x_n / X\right]+\left(1-\alpha_g\right)\mu_g \tag{3.13}$$

$$\hat{\sigma_g}=\alpha_g E_g\left[x_n x_n^T / X\right]+\left(1-\alpha_g\right)\left(\sigma_g+\mu_g \mu_g^T\right)-\left(\hat{\mu_g}\hat{\mu_g}^T\right) \tag{3.14}$$

The scaling factor b in (3) is computed from all the adapted mixture weights to ensure that they sum to unity. Thus, the new GMM parameters are a weighted summation of the UBM parameters and the sufficient statistics obtained from the observed data. The variable is defined as

$$\alpha_g = \frac{N_s\,(g)}{N_s\,(g) + \gamma} \tag{3.15}$$

Here, r is known as the relevance factor. This parameter controls how the adapted GMM parameter will be affected by the observed language data. In the original study [29], this parameter was defined differently for the model weight, mean, and covariance. However, since only adaptation of the mean vectors turned out to be the most effective, we only use one relevance factor in our discussion here. The perfomance of UBM based lID systems is presented in the below Table

Table 3.2: LID Accuracy using GMM-UBM

# Gaussian Components	Accuracy(in %)
64	46.83
128	49.16
256	49.83
512	49.83
1024	48.833

3.3 I-Vector

The State-of-the-art iVector based LID system was first introduced in [32]. The iVector based system operates on the assumption that all the pertinent variability is captured by a low rank rectangular matrix T named the total variability matrix. The GMM supervector (vector created by stacking all mean vectors from the GMM) for a given utterance can be modeled as given in Equation 3.16

$$M = m + Tw + \epsilon \tag{3.16}$$

where, m is the UBM supervector, w is the iVector and ϵ is the residual noise term. Conceptually, an iVector captures the sequence summary given an utterance but is computationally intensive. Once the iVector is computed, a dimension reduction technique like principal compound analysis (PCA) is used to reduce the length of the sequence vector (context summarization). The performance of I-vectior based LID systems is presented in the below Table.

Table 3.3: Equal error rate (EER in %) of I-Vector LID system

Language	Ass	Ben	Guj	Hin	Kan	Mal	Man	Mar	Odi	Pun	Tam	Tel	Urd	Average
$Static_{3s}$	7.34	16.4	19.38	26.72	16.13	11.94	6.05	14.49	13.38	14.94	29.14	7.08	7.61	14.66
$Static_{5s}$	6.05	14.1	18.1	26.1	16.2	10.6	5.25	13.8	13	13.5	26.9	5.79	6.94	13.5918
$Static_{7s}$	5.94	13.21	16.08	28.07	18.12	10.57	5.00	12.57	11.45	11.89	25.17	5.13	6.35	13.04
$Dynamic_{3s}$	4.95	13.58	14.25	18.23	13.24	7.85	4.80	11.51	6.78	7.98	21.04	5.21	6.01	10.41
$Dynamic_{5s}$	5.12	13	13.2	19.9	14.6	7.51	4.40	10.1	6.49	7.16	19.9	4.99	5.73	10.1818
$Dynamic_{7s}$	4.61	12.33	12.18	21.2	16.41	7.26	4.27	8.77	6.44	7.48	20.04	4.63	5.04	10.05

3.4 Indian Language Database

Most of the Indian languages are originated from Devanagari, the script of the Sanskrit language. Owing to the similarities in origin Indian languages have overlapping phoneme sets. The presence of similarities in the phonesets makes the LID a challenging task. Despite the similarities in phoneme sets, every language has its own influence on the phonotactic constraints which can be used for developing language identification systems. Even the regional proximity influences one language.

Building a language identification system requires a dataset with multiple languages that includes speaker variability and multiple utterances per speaker to nullify the effect of speaker on language. So far, there have not been such Indian language datasets that have enough data. With this constraint, even data collection has been explored as a part of this thesis.

The Indian language database consists of 13 official Indian languages. The speech data is collected in read speech mode at 16 kHz sampling rate. Every language has significant amount of male and female speakers to account for the gender and speaker variability. In our experiments, each speech file is sliced into 5 seconds chunk both in the training and testing datasets. The details of number of speakers (#Male,#Female) per language and the amount of data available (#Hours) per language is given in Table 3.4.

Table 3.4: Description of the Indian language database used [51]

Language	Train			Test		
	#Hrs	#M	#F	#Hrs	#M	#F
Assamese	11.79	19	10	1.94	3	3
Bengali	9.54	24	35	1.36	15	14
Gujarati	9.64	115	75	2.17	35	36
Hindi	10.03	40	26	3.16	15	17
Kannada	10.04	21	16	0.97	10	4
Malayalam	7.04	7	6	2.86	9	7
Manipuri	3.81	5	6	1.45	3	2
Marathi	7.82	72	31	2.47	17	15
Odiya	8.55	31	30	2.18	9	9
Punjabi	14.06	2	9	3.74	2	1
Tamil	4.07	12	8	0.87	5	4
Telugu	10.30	57	21	3.08	4	4
Urdu	7.63	55	18	3.03	16	5

3.5 Summary and Conclusions

In this chapter, we have discussed Gaussian Mixture Model and how to estimate the mean , variance and weight matrices for each gaussian and expectation maximization algorithm. To obtain a better performance from gmm, we move for an extension of basic gmm called the UBM model and its interpretation has been discussed. Further, the state-of-the-art I-Vector based technique was introduced and discussed in the context of LID.

Chapter 4

Deep Neural Network based Language Identification : Discriminative Model

4.1 Introduction to Neural Networks

The characteristics of a neural network can be essentially described by it's architecture and functional propoerties. The architecture describes the structure of the neural network, that is, the number of layers and the number of neurons in each layer. A neural network comprises of numerous interconnected neurons, each possessing a few characteristics such as input, synaptic weights, biases, activation functions and an output. The functional properties include the method of learning, recalling, associating and continuously comparing new data and existing learning. Using these properties, a neural network arranges new data and grows new classification if important.

4.1.1 Neuron

Neuron is the basic unit of a neural network. It is also called as the processing element of a neural network. Here, the term is used with the understanding that it is in no way describing the biological neuron. Figure gives a description of a basic model. Any neuron has an arrangement of N inputs where x indicates the source of the data signal. Every input x is weighted by a weight element. Besides, it has a bias term w_0 and a threshold value b that has to be met to produce a signal. The produced signal i sthen passed over a nonlinear function F to generate the output. The essential neuron model is delineated in Fig. where, x_i,w_i,b,F and y represent the input, weights, bias term, activation function and final output respectively. The transfer function of the neuron is described by the relation given below

$$y_i = F\left(\sum_{i=0}^{n} w_i x_i\right) \tag{4.1}$$

where, x_0 =1 and the firing condition of the neuron is

$$\sum_{i=0}^{n} w_i x_i \geq b_i \tag{4.2}$$

The nonlinearity function F could be anything from a simple sigmoid to a more significant ReLU [33]. The purpose of the nonlinearity function is to ensure that the neuron's response is bounded. Most commonly used activation functions are linear, nonlinear, sigmoid, ReLU and softmax functions.

4.1.2 Models of Neuron

There are three classical models for an artificial neuron or processing unit as described below :

1. McCulloch-Pitts Model

   In McCulloch-Pitts model [34], the activation a_c is given by a weighted sum of it's N input values x_i and a bias term Θ. The output signal y is typically a nonlinear function $f(a_c)$ of the activation value a_c. The operation of a McCulloch-Pitts model is described using the following equation :

$$Activationx = \sum_{i=0}^{n} w_i a_i - \theta \tag{4.3}$$

$$Outputsignals = f(x) \tag{4.4}$$

   The weights assigned for the inputs remain fixed in a McCulloch-Pitts model. Hence, a network using this model is not capable of learning. moreover, the model only allows binary states, operating at discrete time steps.

2. Perceptron Model

   The perceptron model was introduced by Rosenblatt [35]. It consists of outputs from sensory units to a fixed set of association units. The outputs of these units are fed to an McCulloch-Pitts neuron. Predetermined manipulations are performed on the inputs of association units. A learning component is present in a perceptron model which makes Perceptron different from the basic McCulloch-Pitts model. This learning is incorporated during the operation of the limit. The desired or target output b is compared with the actual binary output s, and the difference between the two, i.e the error is used to adjust the weights. The equation below describes the operation of the perceptron model of a neuron.

$$Activationx = \sum_{i=0}^{n} w_i a_i - \theta \tag{4.5}$$

$$Outputsignals = f\left(x\right) \tag{4.6}$$

$$Error\delta = b - s \tag{4.7}$$

$$Weightchangew_i = \eta\delta a_i \tag{4.8}$$

where, η is the learning parameter. Based on the nature of the desired input and output pairs to be represented by the model, the weight adjustment either converges or does not. The perceptron convergence theorem enables us to determine whether the given pattern pairs are representable or not. The corresponding task is said to be representable by a perceptron model only if the weights are converging.

3. Adaline Model

   ADAptive LINear Element (ADALINE) is a model proposed by Widrow [36]. The Adaline model differs from the Rosenblatt's perceptron model in a way that, in the adaline model, the activation value a_c is compared with the target output. The output is thus a linear function of the activation value a_c. The following equation describes the operation of an Adaline.

$$Activationx = \sum x = \sum_{i=0}^{n} w_i a_i - \theta \tag{4.9}$$

$$Outputsignals = f(x) = x \tag{4.10}$$

$$Error\delta = b - s = b - x \tag{4.11}$$

$$Weightchangew_i = \eta\delta x_i \tag{4.12}$$

Hence, the weight update rule generally minimizes the mean squared error δ^2, averaged over all the given inputs. Hence it is called least mean squared (LMS) error learning law.

4.2 Activation Functions

A function used to transform the activation level of a neural unit into an output signal. The neurons of a particular layer gets the same type of activation function. In most of the cases non-linear activation functions are used from which comes the transformation power is drawn by a neural network. Various types of activation functions used in a neural network are sigmoid, tanh, ReLU and softmax function.

4.3 Weight Initialization Schemes

The initial weights choisen determine the starting point in the error landscape. This further controls whether the learning process shall end up at a global minimum or a local minimum. Hence, the weight initialzation schema is as important as the learning algorith itself is. Choice of the weights is crucial as the larger values cause the derivative of activation function to saturate and smaller value causes the learning process to be very slow and end up not converging. Different types of weight initialization techniques exist such as : Normalized initialization , Random walk initialization etc.

1. Normalized Initialization

   The Normalized initialization schema was proposed in [37] for training the deep neural networks. Equation 4.3 below describes the initialization schema

$$\sigma = \sqrt{\frac{6}{(N_{in} + N_{out})}} \tag{4.13}$$

$$W = U\left(-\sigma, +\sigma\right) \tag{4.14}$$

   where, N_{in} and N_{out} are the number of nodes(units) in the current layer and next layer respectively and U(-σ,+σ) is the uniform distribution in the interval (-σ,+σ).

2. Random Walk Initialization

   The Random walk initialization schema was proposed in [38], and according to this schema, weights are drawn randomly from a zero-mean Gaussian distribution and to prevent the gradients from exploding, weights are to be scaled by a factor g at depth D. The value of g differs for each non-linearity.

$$g_{linear} = exp\left(\frac{1}{2N}\right) \tag{4.15}$$

$$g_{ReLu} = \left(\sqrt{2}\right) exp\left(\frac{1.2}{(max\left(N, 6\right) - 2.4)}\right) \tag{4.16}$$

$$W = g * N\left(0, \frac{1}{N}\right) \tag{4.17}$$

   where, N is the number of nodes in the current layer.

4.4 Network Architecture

In this work, we use feed forward neural networks for an end-to-end classification task. A feed forward neural network allows the input signals to travel in one way reaching towards the output. There is no feedback/recurrent connections i.e the output of any layer does not affect tyhe same layer. They are extensively used in pattern recognition tasks and have been successfull.

4.4.1 Single Layer Perceptron

A simple example of a feed forward neural network is a single layer perceptron. This network consists of two layers, namely an input and an output layer. The input layer generally receives the input sample with the dimension being equal to the number of feature vector size. Output layer recieves the output signals. The synaptic links carrying the weights connect every input neuron to the output neuron. Such a network is called a feed forward neutral network.

4.4.2 Multilayer Perceptron

This class of networks consists of multiple layers of computational units, interconnected in a feed forward manner. Each neuron in one layer is connected to all the neurons of the next layer. Multilayer networks use a variety of learning techniques, most popular one is the back propagation algorithm [39]. Through this algorithm, the outputs are compared with the desired output to compute the value of some predefined error function. The error is then fed back and weights are updated in such a way that the error is minimized. This process is repeated for sufficiently large number of training cycles until the convergence happens to global minimum on an error surface. To properly adjust the weights, a general method for non-linear optimization called the gradient descent method is used. According to this algorithm, the derivative of the error function with respect to the network weights is calculated, and the weights are then changed such that the error is decreased. For this reason, back propagation algorithm can only be applied on networks with differentiable activation functions.

For a neural network to perform well on the test set, the learning process has to be good and it also depends on the amount of training data. If the training examples are less/few, the network over fits the data and fails to capture the true statistical process generating the data. A sample heuristic, called the early stopping is often used to ensure that the network will generalize well to unseen examples(test case).

4.4.3 Deep Neural Network Architecture

The details of the proposed method for language identification is presented in this section. A neural network with more than 2 hidden layers is termed as a deep neural network.The basic architecture of deep neural network for language identification is shown in Fig. A closed set LID task of 13 languages of an Indian language database are considered. Based on the similarity measure, the goal of LID is to assign the given spoken utterance to one of the 13 languages. Thus making it a multi class classification problem with 13 classes. It is difficult to train and optimize a deep network. The difficulty in training is most likely due to the prior initialization schemes used earlier[40]. Recently, better initialization techniques have been introduced especially to train a deep network [38],[41],[42]. On the other hand, a lot of modifications to the non-linearity have been introduced and have been shown to improve the performance of deep networks in various application domains[43],[44]. Better first-order gradient descent techniques as in [45],[46], and [40] instead of naive stochastic gradient descent with classical momentum (SGD-CM) have led to better optimum and faster training of the network.

4.4.4 Training

The process of learning that a neural network undergoes is called training. During the training phase, the connection weights get adjusted and the most appropriate set of weights are found to represent the model. Initially, each of the weights is randomly chosen and set nonzero. In order to attain the best fit, the model undergoes a set of iterations called as epochs, using a predefined learning algorithm. Epochs define the number of times the network gets to see the entire data. Based on the hyper parameters, data and the task, sometimes it takes thousands of epochs to train a neural network adequately. A learning algorithm is used and the error is minimized. One such algorithm is back propagation.

A learning algorithm is used to compute the error and to adjust the weights according to the error. One such algorithm is the back propagation algorithm [47] which computes the error and changes the weights according to minimize the error. This algorithm was responsible in large part of the re-emergence of neural networks in the mid 1980s. It is powerful but also expensive in terms of computational requirements for training.

Back propagation is based on a relatively simple form of optimization known as gradient descent. The basic back propagation algorithm follows three steps as given below:

1. The feature vector is presented to the input layer of the network that is further propagated forward through the hidden layers until it reaches the output layer. The actual or predicted output pattern is thus produced through this forward pass.

2. A back propagation is a supervised learning algorithm, the desired outputs are given as part if the training vector(feature vector). The predicted outputs are subtracted from the desired outputs and an error signal is thus produced.

3. The error is passed back through the neural network by computing the contribution of each hidden unit and deriving the corresponding adjustment needed to produce the correct output.

The training process of a neural network can be controlled majorly by two learning parameters i.e learning rate and momentum factor. The learning rate decided the step size of moving towards the minimum on an error surface. It specifies whether the network is going to make a minor/major adjustment of weights for each trial. Momentum is used to control possible oscillations in the weights. While there are many other hyper parameters to set a neural network on training, these two parameters produce a higher impact on training performance and time compared to others.

4.4.4.1 Learning Rate

Learning rate η determines the size of the weight change. If η value is set low, the learning process gets slow. If it is set too large, although the learning takes place fast, the process might end up at some local minimum rather than reaching a global minimum of an error surface. η is generally set between 0 to 1 depending on the problem. An improved technique is to use an adaptive η where, η is initially set a bit high in order to escape local minima and later η is reduced to prevent the learning from overshooting the reached minimum.

4.4.4.2 Momentum Factor

Momentum generally reduces the dependency on one single input pattern while taking the direction decision. It causes the weight changes to be based on more than one input sample. The change is a linear combination of the current gradient and the previous gradient. The range of μ is between 0 and 1. Using the momentum factor prevents the process getting stuck at local minimum by guiding the weight updation in right direction.

Apart from these parameters, the optimization techniques used play a key role towards the convergence. Most widely used optimization techniques include stochastic gradient descent with classical momentum (SGD-CM), ADAM and ADADELTA.

4.4.5 SGD-CM

There are a few variants of the basic gradient descent based on the number of samples to consider for the updation of weights. There are three ways to use a gradient descent method i.e updation of weights can be done by computing the error obtained over single sample, a batch of samples or based on the entire training sample set. In this work, SGDCM with batch-wise analysis. The equation given below describes the optimization technique :

$$\Delta\theta_t = -\eta g_t + \mu\Delta\theta_{t-1} \tag{4.18}$$

$$\theta_{t+1} = \theta_t + \Delta\theta_t \tag{4.19}$$

where, θ_t refers to the parameter(weights), g_t is the gradient. η and μ are the learning rate and momentum respectively and are generally set based on trial and error method (task-specific). Although η could be changed, in the current work, η is fixed throughout the learning process.

4.4.6 ADAM

Adaptive moments is a newly proposed update rule [46], which reduces the problem of having to manually fine tune the learning rate hyper parameter. Using the running averages of first and second-order moment of gradients, a simple per-parameter η is used. The authors propose default values of 0.9

for $\beta 1$, 0.999 for $\beta 2$, and 10 to 8 for ϵ. They show empirically that Adam works well in practice and compares favorably to other adaptive learning method algorithms.

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1)\, g_t \tag{4.20}$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_1)\, g_t^2 \tag{4.21}$$

$$m_t = \frac{\hat{}\ m_t}{1 - \beta_1^t} \tag{4.22}$$

$$v_t = \frac{\hat{}\ v_t}{1 - \beta_2^t} \tag{4.23}$$

$$\theta_t = \theta_{t-1} - \alpha \frac{\hat{m}}{\left(\sqrt{\hat{v}_t}\right) + \epsilon} \tag{4.24}$$

Keeping all these various improvements in view, a DNN architecture has been investigated for the LID task. The effect of depth and width of the neutral network architecture, hyper-parameter optimization techniques and initialization techniques have been explored in this work.

4.5 Deep Neural Network with Attention Mechanism

In this section we describe the architecture of DNN-WA (see Fig. 4.1). This is a simple DNN equipped with attention mechanism. Attention mechanism has been inspired from the one proposed in [51] for neural machine translation. The modification was done such that the attention is computed just by using the input feature vectors as opposed to using both input and output feature vectors as in [48].

Given an input sequence, $X = \{x_1, x_2, \cdots, x_T\}$, a hidden layer representation, $H = \{h_1, h_2, \cdots, h_T\}$, is computed by forward pass through regular DNN and attention is computed on this hidden features.

The attention mechanism $a(h_t)$ shown in Fig. 4.1, is computed using a single layer perceptron and then a *softmax* operation is performed to normalize the values between zero and one.

$$\begin{aligned} H &= [h_1 h_2 \cdots h_T] \\ \gamma &= tanh(W_a H + b_a) \\ \alpha &= softmax(\gamma) \end{aligned} \tag{4.25}$$

In the above equations, α is referred to as *attention vector*, and W_a, b_a are the parameters of the attention network optimized along with other parameters of the network using back-propagation algorithm.

The context vector is computed from the attention vector as

$$c = H\alpha \tag{4.26}$$

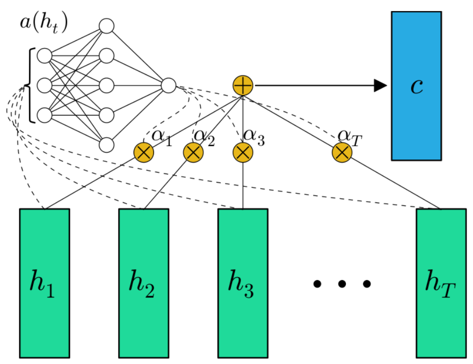

Figure 4.1: DNN-WA [49]

The output is computed by transforming the context vector c using output layer weights U followed by *softmax* operation.

$$y = softmax(Uc + b_o) \tag{4.27}$$

Where b_o is the output layer bias. Note that for the entire input utterance X only a single decision vector y is predicted.

The depth of neural network before/after the attention mechanism can be varied. In this work we have used 1 and 3 hidden layers before the attention to understand the effect of depth. The total number of layers for DNN-WA is the number of hidden layers before the context plus the additional output layer.

4.6 Experiments and Results

We extract 39-dimensional MFCCs (13 static + Δ +$\Delta\Delta$) from each of the 5s chunks. DNNs were trained using a mini-batch stochastic gradient descent (SGD) with classical momentum. Normalized initialization proposed in [37] was used to initialize our networks. We adjust the hyper-parameters using a validation set. In order for the comparison between DNN and DNN-WA to be fair, we have randomized only the sequences presented to the networks while the frames within a given sequence were not randomized. The mini-batch size was equal to the length of the sequence given as input to the network. The input layer has 39 linear units, while the output layer is *softmax* with 13 units. Rectified linear units (ReLU/R) [33] were used as the activation functions in the hidden layers.

Table 4.1: DNN architectures

# layers	Architecture
2	700R 500R
4	700R 500R 200R 100R
6	700R 500R 200R 100R 50R 25R

A grid-search has been performed on various hyper-parameters such as learning rate, number of layers and number of units in each hidden layer. We have also tried different optimizers such as SGD and ADAM and best performing architecture is reported in the above Table. All the networks are trained to minimize the cross-entropy loss over the entire training set. For DNN, the frame based output were averaged before taking the final decision for the utterance while for DNN-WA there was no necessity to do this as can be seen from Eq. 4.27. We use equal error rate (EER) as performance metric, when considering only scores of each individual language.

All the networks are trained to minimize the cross-entropy loss over the entire training set. For DNN, the frame based output were averaged before taking the final decision for the utterance while for DNN-WA there was no necessity to do this as can be seen from Eq. 4.27. We use equal error rate (EER) as performance metric, when considering only scores of each individual language.

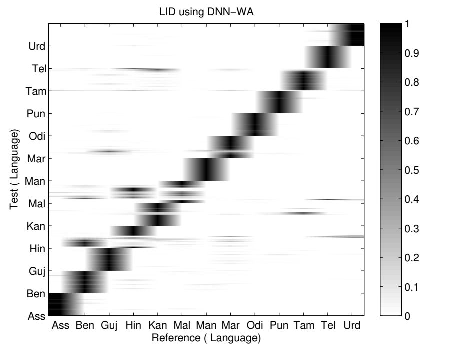

Figure 4.2: Confusion matrix for DNN-WA model with 4 layers

Table 4.2: Equal error rate (EER in %)

Language	Ass	Ben	Guj	Hin	Kan	Mal	Man	Mar	Odi	Pun	Tam	Tel	Urd	Average
$IVector(Baseline)$	5.12	13	13.2	19.9	14.6	7.51	4.40	10.1	6.49	7.16	19.9	4.99	5.73	10.1818
DNN_{2l}	4.11	6.87	7.28	20.8	13.6	16.29	5.29	10.92	4.69	4.83	49.8	3.55	6.67	11.90
DNN_{4l}	4.30	6.59	8.2	22	15.2	18.61	5.6	12.2	4.62	4.99	20	3.04	6.6	10.15
DNN_{6l}	4	6.34	6.66	21.8	12.8	19.71	5.57	9.6	4.52	5.24	20	3.13	6.65	9.69
$DNN-WA_{2l}$	5.66	6.07	7.07	20.7	7.18	17.4	6.00	6.69	6.52	6.50	11.6	4.98	6.20	8.66
$DNN-WA_{4l}$	6.10	6.42	6.56	25.4	7.8	15.2	4.8	7.4	5.84	4.80	8.80	4.60	6.57	**8.48**

The first set of experiments were performed to determine the depth of the architecture that is best suited for LR. The number of units used in each layer are presented in Table 4.1 for various depths. We can see from the Table 4.2 that architecture with depth of 4 layers performs significantly better than architecture with 2 layers.

Next we examine the DNN-WA architecture performance. The number of hidden layers before the attention layer was 1 and 3. We can clearly see form the Table 4.2 that the DNN-WA outperforms the regular DNN (rows 1 vs. 4 and 2 vs. 5) in most of the languages and on the average (last column). This could be attributed to the fact that implicitly model captures context and integrates information before taking the final decision. In DNN-WA also having more number of hidden layers before the attention mechanism helps as can be seen from the bottom two rows of Table 4.2. A plot of confusion matrix obtained using the best architecture (DNN-WA with 4 layers) is shown in Fig. 4.2.

4.7 Analysis of Deep Neural Network with Attention Mechanism

One feature of attention mechanism in neural networks is that it helps us analyze how the input is being attended to while making the decision. A plot of speech signal spectrum (from Assamese language) and the attention vector (α in Eq. 4.25) weights are shown in Fig 5.3. The spectrum is plotted only till 4KHz for clarity. The black regions in the attention plot above the spectrum implies higher values (or *attention*) for the respective frames in the input. It is interesting to note from the figure that the attention is largely towards sections of the signal where the transitions take place as is indicated at three places by the red dashed rectangles.

4.8 Summary and Conclusions

In this work, we have introduced the DNN-WA for performing the utterance level LR. Our results indicate that the proposed attention architecture is well suited for the task of LR. The integration of hidden layer features using the attention mechanism has resulted in effective usage of context. A preliminary qualitative analysis of attention mechanism revealed that transition regions in the signals have more discriminative information for LR.

Further, excitation source information using the RCC features and the vocal tract system information using the MFCC features are used for the LID task. Recently proposed DNN-WA mechanism is used for an utterance level classification. By combining the evidences from RCC based LID system with the conventional MFCC based LID system, an improved EER of 5.7665% is achieved, which is better than the individual EER of 6.2522% and 9.9367% respectively. This result shows that the source information indeed helps in improving the performance of the LID by providing language information that is complementary to the vocal tract system information. Confusion matrix indicates greater difficulty in identifying the languages Hindi and Malayalam compared to others.

Chapter 5

Combining Excitation Source and Vocal Tract System Features for Language Identification

Language information mostly exists in longer durations. Information from shorter durations (sub-segmental information) corresponds to excitation cues. We use the RCC features extracted from the LP Residual signal to represent the excitation source information and MFCC features to represent the vocal tract system features. When DNN based classifier is used to model short durational cues, it fails to capture the long-term dependencies that are reqired for LID. To use the complementary language discriminative information from excitation source, the modeling mechanism should be capable of modeling the long-term dependencies. A regular DNN is a frame based classifier where the decision from every frame is given equal importance. The average over all frames would represent the final decision to the utterance. Unlike DNN, DNN-WA is an utterance based decision mechanism which accounts for the long-term relations by implicitly making use of the context within the utterance.

A set of experiments have been conducted to select the best dimension and type of feature i.e. static Vs dynamic Vs SDC. SDC outperformes the other features and hence we proceed with the SDC features applied on MFCC as well as RCC. From the literature, the SDC configuration 7-1-3-7 is choosen for MFCCs. Further, we found that 10-1-3-3 SDC configuration to work better for RCC. Thus 56D MFCC based and 40D RCC based LID systems are considered for late fusion using DNN-WA.

In this section results are reported for 3 systems MFCC, RCC and MFCC + RCC based LID system across different dimensions. The results given in Table 5.1, Table 5.2, and Table 5.3 are obtained over a testing set consisting of 500 utterances per language each of 5 seconds duration. For every language 500 utterances were held out for validation purpose.

5.1 LID system built using SDC over vocal tract features (MFCC-SDC)

The block diagram of LID system built using MFCC-SDC features is shown in Fig. 5.1. Given a spoken utterance, 13 dimensional MFCCs (static) are extracted from each frame of 20ms with 10ms frame shift. The configuration 7-1-3-7 has been used to compute MFCC-SDC feature vector. Resulting

49 dimensional vector is stacked with the current frame's static features (7 dimensional) making it a 56 dimensional feature vector. DNNs were trained using mini-batch stochastic gradient descent with classical momentum (SGDCM). The size of mini-batch was equal to the length of the sequence given as input to the network. Normalized initialization is used for intializing the network. The dataset used for the experiments consists of 10 hours and 2.5 hours of data per language for training and testing respectively. Validation set was used for hyper-parameter tuning. DNN was trained using mini-batch SGD, where, the size was equal to the length of the sequence given as input to the network.

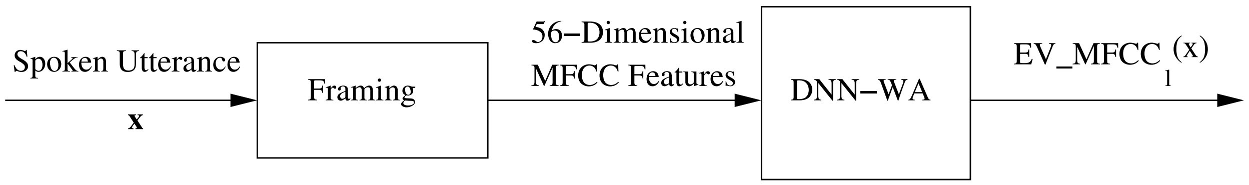

Figure 5.1: Block diagram of the LID sytem using MFCC features

The input and output layers of the DNN and DNN-WA are 56 dimensional and 13 dimensional respectively. Hidden layer dimensions are variable and rectified linear units (ReLU) units were used as the hidden layer activation function. While regular DNN is frame based, the average over all frames of an utterance represents the final decision. In DNN-WA, the attention mechanism helps in direct utterance level classification. Equal error rate (EER) per language is used as the performance metric.

Experiments were performed to determine the depth and breadth of the network for the LID task. It was optimal to use 4 layers with ReLU as activation function for the hidden layers. Table 5.1 describes the results obtained with different architectures. The first column of the Table 5.1 represents the model used with the subscript representing the dimension of the input. Static, dynamic (static + delta + delta delta), and MFCC-SDC features are represented as 13D, 39D and 56D respectively. Column 2-13 represent the EER of the corresponding languages. Final column represents the average EER. Row 2 and Row 3 represent the state-of-the-art iVector based LID results. Row 4 onwards are the DNN based LID results. From Table 5.1 it is clear that the performance of DNN-WA is superior to the regular frame-based DNN model. Hence, we use the end-to-end DNN-WA mechanism in our further experiments.

5.2 LID system built using SDC over excitation source features (RCC-SDC)

Given a speech signal, LP residual is the error obtained when the signal is processed using tenth-order LP analysis. LP analysis removes the second-order relations from the speech signal. The higher order relations among the samples of the speech signal remain in the residual signal. The LP residual mostly contains information about the excitation source. RCC features are used to represent such an information about the excitation source.

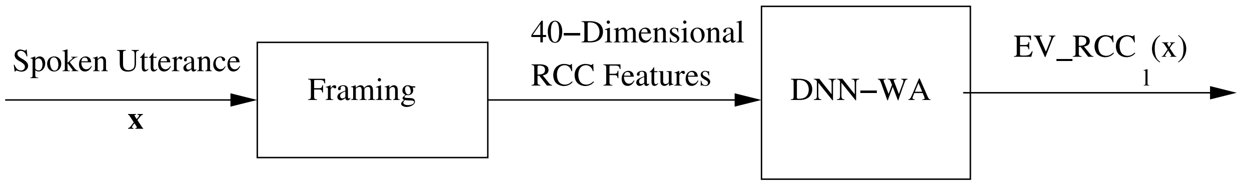

Figure 5.2: Block diagram of the LID sytem using RCC features

Table 5.1: Equal error rate (EER in %) of MFCC based LID system

Language	Ass	Ben	Guj	Hin	Kan	Mal	Man	Mar	Odi	Pun	Tam	Tel	Urd	Average
$iVector_{13D}$	6.05	14.1	18.1	26.1	16.2	10.6	5.25	13.8	13	13.5	26.9	5.79	6.94	13.5918
$iVector_{39D}$	5.12	13	13.2	19.9	14.6	7.51	4.40	10.1	6.49	7.16	19.9	4.99	5.73	10.1818
DNN_{13D}	3.78	7.59	10.71	25	16	23.97	7.18	15.40	7.53	5.60	27.78	3.72	8.06	12.49
DNN_{39D}	4.30	6.59	8.2	22	15.2	18.61	5.6	12.2	4.62	4.99	20	3.04	6.6	10.1512
$DNN-WA_{39D}$	6.10	6.32	6.56	25.4	7.8	15.2	4.8	7.4	5.84	4.8	8.8	4.6	6.57	8.4866
DNN_{56D}	8.97	3.97	8.68	6.91	5.71	5.93	5.6	12.31	5.96	6.2	11.29	2.91	6.83	7.0243
$DNN-WA_{56D}$	6.32	6.21	7.10	6.41	5.78	5.83	6.33	7.1	6.29	5.2	6.82	5.36	6.43	$\mathbf{6.2522}$

The block diagram for LID system built using RCC-SDC features is shown in Fig. 5.2. We use 14 dimensional RCC features coming from each frame of 20ms with a frame shift of 10ms. The configuration 10-1-3-3 has been used to compute SDC feature vector. This configuration has been choosen based on experiments. SDC obtained as a 30 dimensional vector is then stacked with the current frame's static features (10 dimensional) making it a 40 dimensional feature vector. An end-to-end DNN-WA is used for utterance level classification. DNN-WA mechanism shows a significant improvement in performance compared to the regular DNN. DNN-WA has the ability to make use of the context by modeling the long-term dependencies within the utterance. The result of a 4-layer DNN-WA using RCC-SDC features is given in Table 5.2 with an EER of 9.9367%.

Table 5.2 describes the results obtained with different architectures. The first column of the Table 5.2 represents the model used with the subscript representing the dimension of the input. Static and RCC-SDC features are represented as 13D and 40D, 56D. Based on the choice of configuration for SDC computation, 40D and 56D come from 10-1-3-1 and 7-1-3-7 respectively. Column 2-13 represent the EER of the corresponding languages. Final column represents the average EER. Row 2 and Row 3 represent the state-of-the-art iVector based LID results. Row 4 onwards are the DNN based LID results. From Table 5.2 it is clear that the DNN-WA outperforms the regular frame-based DNN model for an RCC based LID system.

In the LP residual signal, the region around the glottal closure instant (GCI) within each pitch period corresponds to high signal-to-noise-ratio (SNR) called the sonority regions. These regions tend to be more robust and are of great use in identifying a language [50]. This could mean that the sonority regions play a key role in decision making given an utterance. Plotting a speech signal spectrum and the attention vector weights helps in visualizing which frame or region of the signal has a major role in

Table 5.2: Equal error rate (EER in %) of RCC based LID system

Language	Ass	Ben	Guj	Hin	Kan	Mal	Man	Mar	Odi	Pun	Tam	Tel	Urd	Average
$IVector_{14D}$	6.59	22.95	19.32	25.64	17.03	14.80	10.60	13.05	16.02	10.66	30.42	11.62	12.57	16.25
DNN_{14D}	0.726	12.06	12.61	32.4	13.15	30.18	17.6	18.19	10.18	6.741	44.41	4.85	16.82	16.9208
DNN_{56D}	2.45	7.25	9.4	36.2	10	14.51	12.14	15.29	9.88	4.64	29.34	5.06	19.4	13.5084
$DNN-WA_{14D}$	5.05	7.2	7.4	24.6	10.6	19.6	5.90	8.47	9.2	5	19.4	5	6	10.2782
$DNN-WA_{56D}$	7.2	7.6	7.83	44.1	12.8	16.2	8.01	9.2	7.30	6.75	27.4	5	7.05	12.816
$DNN-WA_{40D}$	7.22	7	7.51	24.8	9.41	14.1	4.6	8.8	5.79	5.4	23.9	4.28	6.19	**9.9367**

decision making. Interestingly, from Fig. 5.3 it is the transition regions in an utterance that contribute majorly towards identifying a language from a spoken utterance.

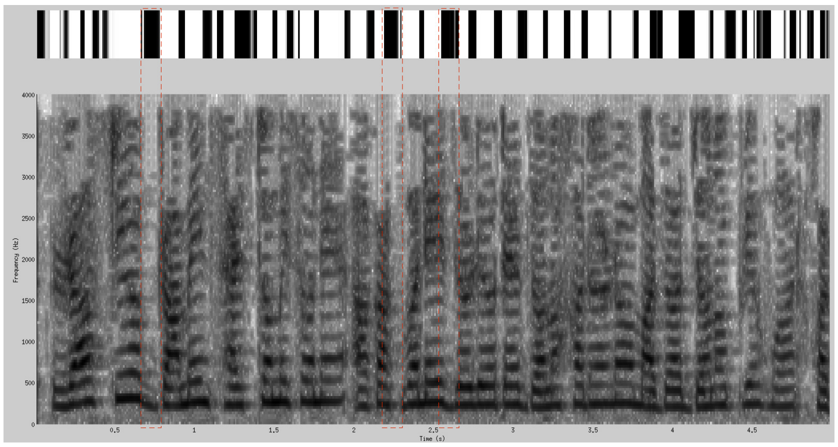

Figure 5.3: An example spectrogram with attention

Table 5.3: Equal error rate (EER in %) of the proposed LID system built by integration of MFCC and RCC based LID systems

Language	Ass	Ben	Guj	Hin	Kan	Mal	Man	Mar	Odi	Pun	Tam	Tel	Urd	Average
$DNN-WA_{MFCC-SDC}$	6.32	6.21	7.10	6.41	5.78	5.83	6.33	7.1	6.29	5.2	6.82	5.36	6.43	6.2522
$DNN-WA_{RCC-SDC}$	7.22	7	7.51	24.8	9.41	14.1	4.6	8.8	5.79	5.4	23.9	4.28	6.19	9.9367
$DNN-WA_{CombinedEvidence}$	5.68	6.4	5.76	7.28	6.02	6.73	4.4	6.32	4.96	4.81	7.57	3.71	5.26	**5.7665**

5.3 LID system built using combined evidence from MFCC-SDC and RCC-SDC features

Block diagram of the LID system built by combining the MFCC-SDC and RCC-SDC based systems is shown in Fig. 5.4 . Result of Combined evidence from excitation source and vocal tract system features is given in Table 5.3. The DNN scores obtained using the MFCC-SDC based LID system and the RCC-SDC based LID system are combined using late fusion mechanism, where the sum rule is used for integration. In the decision logic, the language label associated with the highest combined evidence is hypothesised as the language $L(\mathbf{x})$ of the test utterance x. That is,

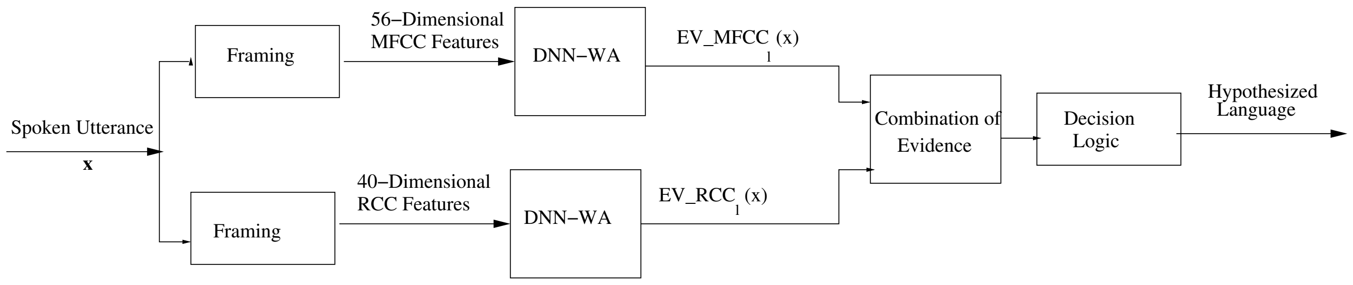

Figure 5.4: Block diagram of the proposed LID system built by integration of MFCC and RCC features

$$L(\mathbf{x}) = \arg\max_{l}\{\log\left(EM_l(\mathbf{x})\right) + \log\left(ER_l(\mathbf{x})\right)\} \tag{5.1}$$

where, $EM_l(\mathbf{x})$ and $ER_l(\mathbf{x})$ corresponds to the normalised DNN evidence for language l obtained using MFCC and RCC features respectively, for the input test utterance x.

5.4 Summary and Conclusions

In this work, excitation source information using the RCC features and the vocal tract system information using the MFCC features are used for the LID task. Recently proposed DNN-WA mechanism is used for an utterance level classification. By combining the evidences from RCC based LID system with the conventional MFCC based LID system, an improved EER of 5.7665% is achieved, which is better than the individual EER of 6.2522% and 9.9367% respectively. This result shows that the source information indeed helps in improving the performance of the LID by providing language information that is complementary to the vocal tract system information. Confusion matrix indicates greater difficulty in identifying the languages Hindi and Malayalam compared to others. As a part of future work, other DNN architectures that can deal with such challenges efficiently can be explored. New activation functions can also be tried.

Chapter 6

Conclusions and Future Work

6.1 Summary and Conclusions

In this thesis, the deep neural network with attention mechanism has been explored for performing the utterance level language identification. Results indicate that the proposed attention architecture is well suited for the task of LID. The integration of hidden layer features using the attention mechanism has resulted in effective usage of context. A preliminary qualitative analysis of attention mechanism revealed that transition regions in the signals have more discriminative information for LID. In the conventional DNN based LID, we have performed experiments to determine optimal depth for the dataset.Our results are consistent with previous findings.

Further, this research covers the utilization of two different aspects of speech information for the task of language identification. This involved the use of vocal tract system and excitation source information. Shifted delta cepstra has been computed over the features to include higher context (frames).

6.2 Future Work

The analysis of varying the utterance length during test time and its effect on the performance of DNN-WA architecture has to be studied. In addition to directly using the context vector from the DNN-WA model for classification, the utterance level representation can be stacked to the regular frame-wise feature vector. This is similar to the way bottleneck features are used. However the key difference is that while bottleneck features change from frame to frame, the proposed context vector remains same throughout the utterance. This approach can be explored as a part of future work.

Related Publications

- **Mounika.K.V**, Ravi Kumar Vuddagiri, Suryakanth V Gangashetty, and Anil Kumar Vuppala, "Combining Evidences from Excitation Source and Vocal Tract System Features for Indian Language Identification Using Deep Neural Networks," accepted to be published in *International Journal of Speech Technology (IJST)*, July 2017.

- **Mounika.K.V**, Sivanand Achanta, Lakshmi HR, Suryakanth V Gangashetty, and Anil Kumar Vuppala, "An investigation of deep neural network architectures for language recognition in Indian languages," in *Proc. INTERSPEECH*, San Francisco, USA, pp. 2930-2933, Sept. 2016.

- Anil Kumar Vuppala, **Mounika K V**, and Hari Krishna Vydana, "Significance of Speech Enhancement and Sonorant Regions of Speech for Robust Language Identification," in *Proc. IEEE International Conference on Signal Processing, Informatics, Communication and Energy Systems (SPICES),*, Calicut, India, Feb. 2015.

- Hari Krishna Vydana, **Mounika K V**, and Anil Kumar Vuppala, "Improved Syllable Nuclei Detection Using Formant Energy in Glottal Closure Regions," in *Proc. IEEE International Conference on Devices, Circuits and Communications (ICDCCom)*, Ranchi, India, pp. 1-6, Sept. 2014.

Bibliography

[1] Y. K. Muthusamy, E. Barnard, and R. A. Cole, “Reviewing automatic language identification, ” IEEE Signal Processing Magazine, vol. 11, no. 4, pp. 33-41, Oct 1994.

[2] M. A. Zissman and K. M. Berkling, “Automatic language identification,” Speech Communication, vol 35, no.1, pp. 115-124, 2001.

[3] M. A. Zissman , "Comparision of four approaches to automatic language identification of telephone speech," IEEE Transactions on Speech and Audio Processing, vol. 4, no.1, p. 31,1996.

[4] A.E. Thyme-Gobbel ans S. E. Hutchins, "On using prosodic cues in automatic language identification," in Proc. ICSLP, 1996.

[5] J. L. Rouas, J. Farinas, F. Pellegrino, and R. Andre-Obrecht, "Modeling Prosody for language identification on read and spontaneous speech." in Proc. ICASSP, 2003, pp. 1-40.

[6] N. Dehak, P. A. Torres-Carrasquillo and D. A. Reynolds, and R. Dehak, “Language recognition via i-vectors and dimensionality reduction,” in Proc. INTERSPEECH, 2011, pp. 857-860.

[7] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, and T. N. Sainath, ”Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups,“ IEEE Signal Processing Magazine, vol. 29, no. 6, pp. 82-97, 2012.

[8] Y. Lei, L. Ferrer, A. Lawson, M. McLaren, and N. Scheffer, ”Application of convolutional neural networks to language identification in noisy conditions,“ in Proc. Odyssey-14, Joensuu, Finland, 2014.

[9] F. Richardson, D. Reynolds, and N. Dehak, ”A unified deep neural network for speaker and language recognition,” in Proc. INTERSPEECH, 2015, pp. 1146-1150.

[10] I. Lopez-Moreno, J. Gonzalez-Dominguez, O. Plchot, D. Martinez, J. Gonzalez-Rodriguez, and P. J. Moreno, “Automatic language identification using deep neural networks,” in Proc. ICASSP, 2014, pp. 5337-5341.

[11] Leena Mary, K.Sreenivasa Rao, and B.Yegnanarayana, “Neural network classifiers for language identification using phonotactic and prosodic features,” in Proc. International Conference on Intelligent Sensing and Information Processing, 2005, pp. 404-408.

[12] J. Gonzalez-Dominguez, I. Lopez-Moreno, H. Sak, J. Gonzalez Rodriguez, and P. J. Moreno, “Automatic language identification using long short-term memory recurrent neural networks.” in Proc. INTERSPEECH, 2014, pp. 2155-2159.

[13] S. Hochreiter and J. Schmidhuber, “Long short-term memory,“ Neural Computation, vol. 9, no. 8, pp. 1735-1780, 1997.

[14] S. Achanta, T. Godambe, and S. V. Gangashetty, ”An investigation of recurrent neural network architectures for statistical parametric speech synthesis,“ in Proc. INTERSPEECH, 2015, pp. 2524-2528.

[15] C. Raffel and D. P. W. Ellis, ”Feed-forward networks with attention can solve some long-term memory problems,“ CoRR, vol. abs/1512.08756, 2015. [Online]. Available: http://arxiv.org/abs/1512.08756.

[16] E. Singer, P.A. Torres-Carrasquillo, T.P. Gleason, W.M. Campbell, and D.A. Reynolds, ”Acoustic, Phonetic, and Discriminative approaches to Automatic Language Identification,“ in Proc. Eurospeech 2003, pp. 1345-1348, Geneva, Switzerland, Sept. 2003.

[17] Rong Tong, Bin Ma, Donglai Zhu, Haizhou Li and Eng Siong Chng ”Integrating Acoustic, Prosodic and Phonotactic features for Spoken langauge identification,” ICASSP , May 14-19 Toulous, France, 2006.

[18] Rong Tong, Bin Ma, Donglai Zhu, Haizhou Li and Eng Siong Chng “Integrating Acoustic, Prosodic and Phonotactic features for Spoken langauge identification,” ICASSP , May 14-19 Toulous, France, 2006.

[19] K.Sreenivasa Rao and Dipanjan Nandi, “Implicit excitation source features for language identification,“ in Language Identification Using Excitation Source Features. Springer, 2015, pp. 31-51.

[20] Debadatta Pati and S.R.M.Prasana, ”Subsegmental, segmental and suprasegmental processing of linear prediction residual for speaker information,“ International Journal of Speech Technology, vol. 14, no. 1, pp. 49-64, 2011.

[21] Leena Mary and B.Yegnanarayana, ”Extraction and representation of prosodic features for language and speaker recognition,” Speech communication, vol. 50, no. 10, pp. 782-796, 2008.

[22] P. A. Torres-Carrasquillo, E. Singer, M. A. Kohler, R. J. Greene, D. A. Reynolds, and J. R. Deller Jr, “Approaches to language identification using gaussian mixture models and shifted delta cepstral features.” in Proc. INTERSPEECH, 2002.

[23] Muthusamy, Y.K., Jain,N., Cole R.A. “Perceptual Benchmarks for Automatic Lan- guage Identification.“ in Proc. ICASSP 94, vol.1, pp. 333-336, Adelaide, Australia,Apr. 1994.

[24] Mehler, P. Jusczyk, G. Lambertz, N. Halsted, J. Bertoncini, and C. AmielTison, ”A precursor to language acquisition in young infants,“ Cognition, 29:143178, 1988.

[25] T. J. Hazen and V.W. Zue. ”Recent improvements in an approach to segment-based automatic language identification,” ICSLP, Yokohama, Japan, 1994.

[26] P.A. Torres-Carrasquillo, E. Singer, M. A. Kohler, R. J. Greene, D. A. Reynolds, and J.R.Deller, Jr., “Approaches to language identification using Gaussian mixture models and shifted delta cepstral features,” in ICSLP, 2002, pp.89-92, Denver, USA, 2002.

[27] Hearst, Marti A, Susan T. Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf, "Support vector machines." IEEE Intelligent Systems and their applications 13.4 (1998): 18-28.

[28] Alvin Martin, George Doddington, Terri Kamm, Mark Ordowski, Mark Przybocki, “The DET Curve in Assessment of Detection Task Performance,“ In Eurospeech, pages 1895-1898, Rhodes, Greece, 1997.

[29] Aaron E Rosenberg, “Automatic speaker verification: A review,” Proceedings of the IEEE,64(4):475-487, 1976.

[30] Reynolds, Douglas A., Thomas F. Quatieri, and Robert B. Dunn, "Speaker verification using adapted Gaussian mixture models,." Digital signal processing 10.1-3 (2000): 19-41.

[31] Rok Gajsek, Janez Zibert, Tadej Justin, Vitomir Struc, Bostjan Vesnicer and France Mihelic, "Gender and affect recognition based on GMM and GMM-UBM modeling with relevance MAP estimation." Eleventh Annual Conference of the International Speech Communication Association. 2010.

[32] N. Dehak, P. A. Torres-Carrasquillo, D. A. Reynolds, and R. Dehak, "Language recognition via i-vectors and dimensionality reduction." in Proc. INTERSPEECH, 2011, pp. 857 - 860.

[33] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun,“ Delving deep into rectifiers: Surpassing human-level performance on imagenet classification,” arXiv preprintarXiv:1502.01852, 2015.

[34] Zurada, Jacek M, “Introduction to Artificial Neural System,” West Publishing Company, St. Paul, MN, 1992.

[35] Bose, N. K. and Liang, P , “Neural Network Fundamentals with Graphs, Algorithms, and Applications,” McGraw-Hill, New York, NY, 1996.

[36] B. Widrow, “An Adaptive ‘Adaline’ Neuron Using Chemical ‘Memistors’,” Stanford Electronics Laboratories Technical Report 1553-2, October 1960.

[37] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feed forward neural networks, " in International conference on artificial intelligence and statistics, 2010, pp. 249 - 256.

[38] David Sussillo, "Random walks: Training very deep nonlinear feed-forward networks with smart initialization," arXiv preprint arXiv:1412.6558, 2014.

[39] Hagan, Martin T., and Mohammad B. Menhaj, "Training feedforward networks with the Marquardt algorithm," IEEE transactions on Neural Networks 5.6 (1994): 989-993.

[40] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton, "On the importance of initialization and momentum in deep learning," in Proc. ICML, pages 1139-1147, 2013.

[41] Andrew M Saxe, James L McClelland, and Surya Ganguli, "Exact solutions to the nonlinear dynamics of learning in deep linear neural networks," arXiv preprint arXiv:1312.6120, 2013.

[42] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun," Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," arXiv preprint arXiv:1502.01852, 2015.

[43] Vinod Nair and Geoffrey E Hinton," Rectified linear units improve restricted boltzmann machines," In Proc. International Conference on Machine Learning (ICML-10), pages 807âĂŞ814, 2010.

[44] Zeiler, M.D., Ranzato, M., Monga, R., Mao, M., Yang, K., Le, Q.V., Nguyen, P., Senior, A., Vanhoucke, V., Dean, J. and Hinton, G.E, "On rectified linear units for speech processing," in Proc. ICASSP, pages 3517-3521, 2013.

[45] Matthew D Zeiler, "ADADELTA: an adaptive learning rate method," arXiv preprint arXiv:1212.5701, 2012.

[46] Diederik P. Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," in Proc. International Conference on Learning Representations (ICLR), 2014.

[47] Yoshua Bengio, "Learning deep architectures for ai. Foundations and trends in Machine Learning," 2(1) : 1-127, 2009.

[48] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in Proc. International Conference on Learning Representations(ICLR), 2014.

[49] C. Raffel and D. P. W. Ellis, "Feed-forward networks with attention can solve some long-term memory problems," CoRR, vol. abs/1512.08756, 2015. [Online]. Available: http://arxiv.org/abs/1512.08756

[50] Anil Kumar Vuppala, Mounika.K.V, and Vydana Hari Krishna, "Significance of speech enhancement and sonorant regions of speech for robust language identification," in Proc. IEEE Signal Processing, Informatics, Communication and Energy Systems (SPICES), 2015, pp. 1-5.

[51] Mounika.K.V, Sivanand Achanta Lakshmi.H.R, Suryakanth V Gangasetty, and Anil Kumar Vuppala, "An Investigation of Deep Neural Network Architectures for Language Recognition in Indian Languages," in Proc. INTERSPEECH (pp. 2930-2933) 2016.