IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 11 August 2023, accepted 1 September 2023, date of publication 18 September 2023,
date of current version 25 September 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3316142

RESEARCH ARTICLE

The Effect of Synthetic Voice Data Augmentation on Spoken Language Identification on Indian Languages

A. R. AMBILI [1,2] AND RAJESH CHERIAN ROY [3], (Member, IEEE)
[1]Federal Institute of Science and Technology, Angamaly 683577, India
[2]ASIET, Kalady, APJ Abdul Kalam Technological University, Thiruvananthapuram, Kerala 683574, India
[3]Department of Computer Science and Engineering, Muthoot Institute of Technology and Science, Puthencruz 682308, India

Corresponding author: Rajesh Cherian Roy (rajeshcherian@yahoo.co.in)

ABSTRACT Multilingual based voice activated human computer interaction systems are currently in high demand. The Spoken Language Identification Unit (SPLID) is an inevitable front end unit of such a multilingual system. These systems will be a great boon to a country like India where around 24 official languages are spoken. Deep learning architectures for spoken language identification have progressed to the point that they can now perform well, even in the presence of various background noises. However, a strong phonetic relationship across various Indian languages leads to increased confusion in the SPLID unit. Therefore, the goal of this study is to propose a synthetic voice data augmentation method based on speech synthesis to improve the spoken Indian language identification system. Here the research attempts to determine how well pre-trained computer vision models recognize spoken languages in synthetic and classical audio augmentation environments. The accuracy of the models was compared using bottleneck features extracted from three different pre-trained models VGG16, RESNET50, and Inception-v3 while using an Artificial Neural Network (ANN), Support Vector Machine (SVM), Logistic Regression (LR), Random Forest (RF), Naive Bayes (NB), Decision Tree (DT) and KNN (K-Nearest Neighbors) as classifiers. The proposed system was tested on three Indian language datasets - two comprising seven Indian languages (Hindi, Malayalam, Tamil, Telugu, Marathi, Kannada and Bengali), one containing five Indian languages (Tamil, Hindi, Malayalam, Oria and Assamese), and on a foreign language dataset. It was found that the addition of synthetic audio samples improved the accuracy by 17%. Among the pre-trained models, VGG16 and Inception-v3 combined with PCA and ANN were found to have the maximum accuracy of 97%.

INDEX TERMS Classifiers, data augmentation, inception-v3, RESNET50, speech synthesis, SPLID, VGG16.

I. INTRODUCTION

Voice applications, such as Automatic Speaker Verification Systems (ASV), Interactive Virtual Response (IVR) systems, contactless health care systems, and Google Assistants, are currently in high demand. These systems are adaptable to ordinary people only if they can process voice commands in different languages. Thus multilingual based systems are those systems that should be capable of recognising many languages. Therefore in an Indian setting, such systems are a great boon because India is a country with around 24 official languages spoken by its citizens. Even today end-to-end multilingual systems are receiving attention, and most of the traditional systems require separate Language Dependent (LD) models for processing. In multilingual systems, Spoken Language Identification (SPLID) is a vital front-end unit. There are numerous approaches in the literature, ranging from traditional to deep learning paradigms in the field of spoken language identification. Several features including Mel Frequency Cepstral Coefficients (MFCC), Linear

The associate editor coordinating the review of this manuscript and approving it for publication was Ganesh Naik.

Frequency Cepstral Coefficients (LFCC), SDA (Shifted Delta Add), and fused features are utilized in feature selection whereas several classifiers, such as the Gaussian Mixture Model (GMM), Support Vector Machine (SVM), and Naive Bayes (NB) are employed in spoken language identification. Deep learning architectures such as Deep Neural Network (DNN), Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) are also used in SPLID. Classification accuracy is affected by noise variability, speaker variability, gender variability, accent variations, and restricted availability of training data. Although these models have been proven to have high accuracy at development time, they may yield unexpected results when used in real-time situations. To improve classification model accuracy, the data-centric approach of deep learning is currently receiving more attention than the model-centric approach. The model-centric approach concentrates on the model architectures and parameters for performance improvement. However, the data-centric AI approach focuses on improving data rather than the model for more powerful performance. Augmentation is a subclass of data-centric approaches that can create new data points through interpolation, extrapolation or other means. Augmentation in speech applications can be of non-spectral augmentation, such as speed perturbation, noise, music, and reverb which were used in most of the studies. Spectral augmentation [1] such as time masking (vertical/horizontal) or frequency masking has also been tested. But all these augmentation methods were performed on the same existing utterance samples in the dataset. Such methods may be called classical augmentation. In this paper, we use augmentation methods that can artificially create new utterances using speech synthesis. We refer to such augmentation methods as synthetic augmentation. In this work, we examine the effect of both classical and synthetic augmentation methods in enhancing the performance of SPLID systems in Indian languages.

A. SCOPE AND NOVELTY

Although Indian languages are broadly classified into four language families, the population of India is made up of about 73% speakers of Indo-Aryan languages and 20% speakers of the Dravidian language family. The Indian languages considered in our work are Hindi, Marathi, Oria, Assamese (Indo-Aryan family) and Malayalam, Tamil, Telugu, Kannada (Dravidian family). There are many similar sounds and pronunciations of phonemes in various Indian languages. These common origins among different Indic languages cause inter-dependencies which can lead to misclassification of the SPLID unit. Thus, from an Indian language perspective, language identification can be improved significantly if the model can be trained with a large number of versatile audio samples. However, most datasets contain only a limited number of training audio samples from a single speaker. On the other hand if it is a multi-speaker dataset, the same audio utterances are repeated by other speakers. Hence, the SPLID method requires a large but versatile number of audio samples in the dataset. But manual data collection and data labelling of such larger dataset creation demand extensive time and huge costs. Thus, synthetic data generation is crucial because it can be used to create data to simulate and represent non-encountered audio utterances. Augmentation using synthetic speech synthesis can be used for artificial generation of new audio utterances. We propose a novel approach for enhancing the performance of SPLID systems in Indian languages by employing synthetic voice data augmentation methods. Synthetic augmentation of the audio samples is done using speech synthesis. In addition, this work proves the efficacy of dimensionality reduced pre-trained models initially developed for computer vision applications in the spoken language identification task. In brief, this work aims to contribute to both data-centric and model-centric approaches. The data-centric approach is in terms of classical and synthetic augmentation whereas the model-centric approach is in terms of the use of hybrid models incorporating PCA for the improvement of SPLID systems. These hybrid models include pre-trained neural networks such as VGG16, RESNET50, and Inception-v3 followed by PCA and seven different classifiers like ANN, SVM, LR, RF, NB, DT and KNN. We refer to the combination of a pre-trained model with PCA and ANN as a modified pre-trained model. Since the work concentrates on three pre-trained models, VGG16, RESNET50, and Inception-v3, the corresponding modified pre-trained models are referred to as MOD-VGG16-PCA, MOD-RESNET50-PCA, and MOD-Inception-v3-PCA respectively.

The contribution of the work can be summarised as follows:

- We created and annotated synthetic audio data samples using Indic Text To Speech (TTS) dataset.
- We investigated the impact of the augmentation of synthetic audio samples using speech synthesis and classical augmentation on the models performance accuracy. It is observed that the model performance depends on the quality of the synthetic augmentation.
- We determined the impact of the pre-trained models VGG16, RESNET50, and Inception-v3 as feature extractors combined with different machine learning classifiers (ANN, SVM, KNN, NB, LR, DT and RF) on four standard datasets (IIIT-H, IIT-M, Mozilla Common Voice, and VoxForge voice datasets).
- Next, we analysed the effect of a significant feature selection using PCA on the model.

II. LITERATURE REVIEW

MFCCs [2], SDC [3], i-Vector [4], Linear Discriminant Analysis [LDA] [5], and x-vector [6] are the primary conventional methods used for spoken language identification. A mel-spectrogram combined with a Convolutional Recurrent Neural Network [CRNN], X-vector combined with DNN and Wave2Vec speech representations was tested in [7] and it showed the highest F1-score of 91%. Some of the works carried out in this area include SPLID with

ConvNets [8], Self-Attentive Pooling and deep 1D Time-Channel Separable Convolutions [9], the fusion of L-CNN and RNN [10], MFCC and LSTM [11]. Deep Multilayer Perceptron, Convolutional Neural Networks, and denoising Auto-Encoder type deep learning architectures are currently used for speech enhancement. A comparative analysis of these models was performed using the spectrogram of audio signals in [12]. Augmentation techniques are normally used to improve the generalization capability of systems to various unknown attacks. Speaker variability, gender variability, age variability, room conditions, environmental conditions, and recording conditions all affect the accuracy of the spoken language identification units. Previously, unknown attacks in the case of spoken language identification under noisy conditions were addressed using conventional methods. A compressive technique applied on high dimensional phase and magnitude based features [13] and an adaptive weighting selector for the unknown attack in combination with clustering techniques are some of the approaches considered for unknown attacks without data augmentation [14]. Data augmentation techniques such as room reverberation, noise addition, pitch shifting, time stretching, polarity inversion, time masking, frequency masking, and gain change are the commonly used techniques for audio samples [15]. A CNN as a feature extractor and LSTM as a classifier were tested on a spectral augmented code switched database for SPLID [16] and improved 3-5% accuracy over the baseline model. Spectral augmentation for LID was presented in [17]. A conditional Generative Adversarial Network (c-GAN) was used as a classifier for spoken language identification [18]. A 6-fold augmentation strategy that combines the original ''clean'' training list with five augmented copies, which consists of speed perturbation, noise, music and reverb was used for LID in [19]. Spectral augmentation along with CRNN provides considerable improvement in accuracy in [20], where speech signals are converted into a mel-spectrogram. Pre-trained models such as RESNET50, CRNN-RESNET50, and Inception-v3 were also experimented in [21], [22], and [23]. The datasets mostly considered in SPLID do not include synthetic audio samples. However, our approach uses synthetic augmentation, which generates synthetic audio samples. There exist techniques to discriminate between real and artificial audio samples and they are primarily applied in the spoofing detection tasks. They can also be used to validate our newly generated synthetic audio samples. MFCC [24], LFCC, and CQCC [25] are the main features used for synthetic spoofing detection. Various deep learning architectures like DNN [26], [27], LCNN [28] and LSTM [29] have been employed for spoofing detection. For the replay attack detection, seven augmentation techniques were tested; out of these, dynamic value change and pitch change showed an 8% improvement in base model accuracy [30]. A data augmentation technique using a-law and mu-law based signal companding was explored in [31] for the detection of logical access attacks. For data augmentation, an Auxiliary Classifier Generative Adversarial Network (AC-GAN) was also proposed to generate more speech samples with diverse variants [32] combined with a post-selection quality frame selection based on CNN, giving more accuracy.

A. MOTIVATION BEHIND THE RESEARCH: SIGNIFICANCE OF SYNTHETIC VOICE AUGMENTATION ON INDIAN LANGUAGES

The Indian languages with similar roots are likely to be mistaken for one another. Indo-Aryan (IA) languages evolved from the Sanskrit/Prakrit combination. Prakrits have gradually developed into modern IA languages like Marathi, Hindi, Bangla, Gujarati and Punjabi. Dravidian languages have evolved from Proto-Dravidian roots; hence, their origin is entirely independent of Sanskrit [33], [34]. During their growth, they were influenced by Sanskrit and Prakrit and many words were absorbed from these two languages into Dravidian languages. The Indian languages are phonetically correlated. Dravidian languages contain both long and short vowels (e.g. ē, ō and e,o), whereas most Indo-Aryan languages only have long vowels (e.g. ē, ō). The exceptions among Indo-Aryan languages are Konkani, Sinhala and a few others. This indicates the similarity between speaking the words in these languages. Consider the examples of ''Who'' and ''What'' respectively:

Examples in Dravidian languages:

a)Kannada: yāru, ēnu b) Telugu: evaru, ēmi

c) Tulu: ēra, dāne d)Malayalam: ār, ent

Examples in Indo-Aryan languages:

a)Sanskrit: kaḥ, kim b)Hindi: kaun, kyā

c)Konkani: kŏṇ, kasane d)Marathi: kōṇ, kāy

From the above, it is seen that the same word ''Kaun'' (meaning who) exists in Hindi, Konkani and Marathi. Such word similarities are found more among languages from the same language family than among languages from different language families. Therefore, this may result in increased confusion in the SPLID task. Context-dependent algorithms in the SPLID unit may reduce this confusion. Deep neural context-dependent networks such as LSTM and RNN may somewhat fix this problem. But all of these networks are hungry for data to achieve better performance. So one of the essential requirements for developing SPLID models is the set of transcribed audio samples. Moreover, audios have to be phonetically rich and balanced; phonetically rich means that all the phonemes of the language are present, and balanced means that the phonemes should, as far as possible, have the same number of occurrences. But creating such a dataset requires a longer time and a higher cost. Synthetic data augmentation is a preferable solution for this problem. Although speech synthetic augmentation works are common in foreign languages, this proposed research on synthetic augmentation using speech synthesis leads to a new pathway for SPLID work. We experimented with a data augmentation method with the help of synthetic voice

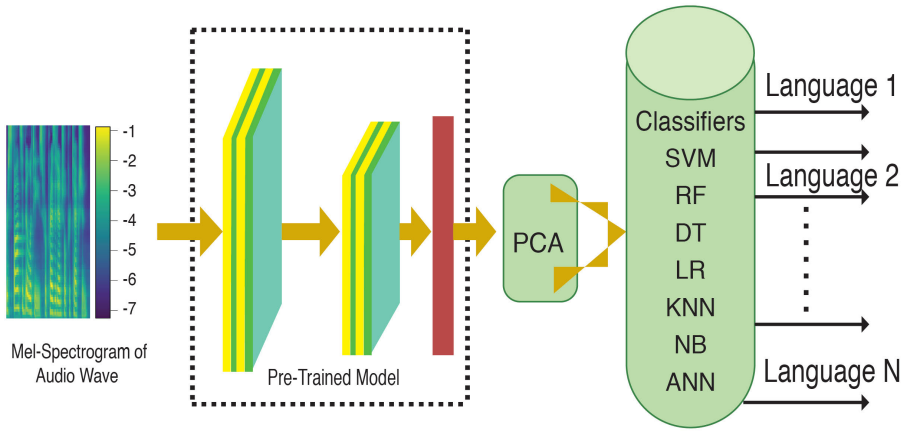


FIGURE 1. Block diagram of the proposed system.

generation in Indian languages. Text sentences from one Indian language dataset (IIIT-H dataset) were selected and their corresponding synthetic voices were generated using the IIT-M speech synthesis model. These newly developed synthetic utterances are then split as training and testing datasets and are used for training the model. These synthetic voice samples help to improve model performance, thereby reducing confusion in LID classification.

III. PROPOSED METHOD

The proposed research aimed to determine the efficacy of synthetically and classically augmented audio samples and to develop an optimum model for spoken language detection in Indian languages. The transfer learning strategy is used here for feature extraction. The best model for the task was selected by comparing the results obtained by conducting the model experimentation in two ways. In the first approach, the feature extraction done by the pre-trained models in combination with two fully connected hidden layers was used for the classification of the languages under study. Here, we retrain the fully connected layers using various hyperparameters such as the optimizer, learning rate, and activation function after freezing the weights of the convolution layers with the values from the respective pre-trained model. The last layer of the pre-trained models was selected as the feature extraction layer. Hence the extracted feature dimensions are different for each model. Since both pre-trained models and ANN are neural networks, this first approach can be called the modified pre-trained model approach. In the second approach, features extracted from pre-trained models were passed to different conventional machine learning classifiers for prediction. Figure 1 depicts the proposed method incorporating both the above mentioned approaches. ANN, SVM, NB, LR, RF, KNN and DT were chosen as the classifiers. The three main pre-trained models VGG16, RESNET50 and Inception-v3 were selected for this study because of their success in SPLID works. But instead of directly feeding these features to the classification stage, the significant features were selected with the help of PCA - a well-known dimensionality reduction technique. This step dramatically reduces the number of classification parameters.

As a pre-processing step, a mel-spectrogram representation of the audio samples was obtained. These spectrogram images were then split into training and testing samples. The size of the image required for each pre-trained model is different. Hence the image size was adjusted accordingly for each pre-trained model. In the first phase of experimentation, the pre-trained model accuracy was tested on classically augmented real data samples of the IIIT-H, IIT-M, VoxForge and Mozilla Common Voice datasets. These datasets contained only real audio samples. Using non-spectral augmentation technique, new data samples were created. This method is referred to as the CLASSICAL-AUG. The different augmentation strategies used here are white noise addition, pitch shifting, gain shifting, and time shifting which are discussed in section V under ''Classical speech augmentation''. In the second phase, synthetic augmentation using speech synthesis was used and this method is called SYNTH-AUG. On both the newly created data samples, three pre-trained models VGG16, RESNET50 and Inception-v3 were tested. These extracted features are given as inputs to the classification stage. The different steps of this proposed method are explained in detail below, starting from feature extraction, model architecture, hyperparameters, classifiers, models with classical augmentation and models with synthetic augmentation. The remainder of this paper is organized as follows: Section IV describes the mel-spectrogram feature extraction. Section V explains the details of datasets, the synthetic and classical augmentation methods, feature selection using PCA, pre-trained model architectures, classifiers and their hyperparameters. Section VI presents the results and discussion and section VII presents the conclusion and future scope.

IV. MEL-SPECTROGRAM GENERATION

Mel-spectrograms are generated using audio clips or utterances of the data. A mel-spectrogram is a spectrogram in which the frequencies are converted to the mel scale. A spectrogram is a bunch of FFTs stacked on top of each other. This is a way to visually represent a signal's loudness, or amplitude, as it varies over time at different frequencies. Figure 2 shows the mel-spectrogram of the real, classical and synthetic audio samples. The conversion of frequencies f (Hz) to Mel scale m (mels) is as follows.

$$m = 2595 * log_{10}(1 + f/700) \quad (1)$$

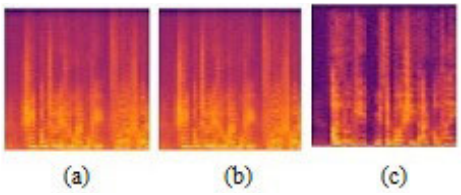


FIGURE 2. Mel-Spectrogram image of a) Original audio sample b) Classically augmented audio sample c) Synthetically augmented audio sample.

The signal has a sampling frequency of 22050 Hz. The number of channels (mel filters) used in this spectrogram was 128. An overlap length of 1024 samples and a window length of 2048 samples were used to obtain the spectrogram. The number of frames obtained is 65.

V. WORKFLOW

The two objectives of this work were to find the effect of the combined pre-trained network and PCA with classical audio augmentation and synthetic audio augmentation on spoken language identification.

A. DATASETS

To accomplish the first objective of language identification with classical augmentation, we need a dataset that contains different Indian languages. The experiment started with four general datasets: IIIT-H, IIT-M, VoxForge and Mozilla Common Voice dataset. VoxForge is a Foreign language dataset. We used it for model comparison.

1) IIIT-H DATASET

The IIIT-Hyderabad Indic speech dataset [35] was developed by Speech and Vision Lab and consists of data in textual and speech format. Seven languages from the IIIT-H dataset were chosen for this study. The same dataset was used for synthetic data augmentation. The languages under consideration are Malayalam, Tamil, Hindi, Telugu, Marathi, Kannada and Bengali. The dataset comprises 7000 audio samples divided approximately equally among the seven language classes.

2) IIT-M DATASET

This dataset results from a project to develop text-to-speech (TTS) synthesis systems [36] for Indian languages and enhance the quality of synthesis. It consists of thirteen languages. Seven languages from the IIT-M dataset were selected for this study. The same dataset was used for synthetic data augmentation. The languages under consideration are Malayalam, Tamil, Hindi, Telugu, Marathi, Kannada and Bengali.

3) VoxFORGE DATASET

The work on VoxForge [37] is based on eight foreign languages. VoxForge is a project set up to collect transcribed speech for use in open source speech recognition engines. This dataset is large and diverse in terms of variety and size. Here, we have considered 1000 audio samples for each language category, thus avoiding class imbalances of any kind. We used eight languages, namely German, French, Russian, Italian, Portuguese, Dutch, English and Spanish to test our model.

4) MOZILLA COMMON VOICE DATASET

Mozilla Common Voice is an audio dataset that consists of a unique MP3 and corresponding text file. This dataset is publicly available. They have prepared this dataset by considering multilingual links between speech segments of different languages. The dataset also includes demographic metadata like age, accent etc. The common_voice_7_0 dataset consists of 11192 validated hours in 76 languages. We used Indian languages, namely Tamil, Hindi, Malayalam, Oria and Assamese from the Mozilla Common Voice dataset [38] for the proposed study. For a balanced dataset, we used approximately 1000 samples from each language.

B. SYNTHETIC AUGMENTATION AND CLASSICAL AUGMENTATION

The augmentation process was classified into classical and synthetic augmentation. Creating synthetic voice samples is a tedious task. This will be explained in the following.

1) SYNTHETIC SPEECH AUGMENTATION

Speech Synthesis (SS) techniques have been developed, and can be utilized to make synthetic speech. Unit selection speech synthesis, Hidden Markov Model (HMM) speech synthesis, and Neural Network (NN) based voice synthesis [9] are techniques commonly used in SS systems. Therefore, a dataset that contains real and synthetic audio utterances is required. So we decided to use INDIC TTS [36], which is primarily concerned with the synthesis of speech from text. This dataset is a collaborative initiative for synthesising diverse Indian languages established by well-known institutes such as IITs. So we prepared a synthetic dataset using the Indic TTS website. The synthetic voice generation method used in the IIT-M dataset was HTS-US. This was used for synthetic sample generation. Each language has a separately trained TTS model. Original voice samples were obtained from the IIT Madras speech lab. Synthetic audio samples were generated from the INDIC TTS site with the help of

Python script and Selenium package [42], [43]. Although the Indic TTS team generated four alternative speech synthesis models, the publicly available speech synthesis model from the INDIC TTS is HTS-STRAIGHT. Programmatic labelling was used for annotations of the dataset. A separate folder was created for each language for the synthetic and authentic sample datasets. The validity of the created data samples was ensured using speech spoofing detection techniques [44]. The synthesised sentences were selected from the IIT-M dataset, which covers a diverse language space. But the creation of the synthesized sample distribution of the Telugu language meets with some unprecedented delay. So for the Telugu language, we selected sentences of short duration. This created a synthetic dataset for this investigation. Table 1 lists the details of the synthetic dataset.

TABLE 1. Details of synthetically augmented dataset.

Text Data	Synth-AUG Utterances	Synth-Classical-AUG Utterances
7000	12000	72000

2) CLASSICAL SPEECH AUGMENTATION

Here the augmentation performed on real samples in the dataset is considered classical augmentation. The sampling rate was set to 22050 Hz. Augmentation techniques like white noise addition, pitch shifting, gain shifting, time stretching and speed change are applied to the dataset for the creation of classically augmented samples. This augmentation is done with the help of the Librosa library. Librosa is a package from Python to analyse audio signals. Dataset details with classical augmentation are given in Table 2. The classical augmentation methods used for making the classically augmented dataset are explained below.

1) Pitch shifting: Pitch shift by a random number in [−4,4] semitones (Semitone Shift Range)
2) Gain changes: The gain of the audio signal is changed by a specific number of decibels (dB) (min factor=0.1, max factor=0.4)
3) White noise: Noisy signal is generated by adding to the clean signal a random noise signal parameterised by a noise percentage factor. The selected values of the noise percentage factor were 0.1, 0.2, 0.3, 0.4 and 0.5
4) Signal speed scaling by a random number in [0.8, 1.2]
5) Time shift in the range [−0.005, 0.005] seconds.

C. FEATURE EXTRACTION

Feature extraction is an essential step in signal analysis. A two phase feature selection method was attempted in this study. In the first phase, we use transfer learning methods that utilize CNNs to extract deep linguistic features to represent the characteristics of a speech utterance. Transfer learning is a widely explored method that can be used to accelerate training and improve the performance of the deep learning model. Three pre-trained networks VGG16, RESNET50 and Inception-v3 were utilized. The output feature map is 25088 in the case of VGG16. This feature map dimension is the flattened output of the last frozen layer (512*7*7) of VGG16. Similarly, the dimensions for RESNET50 and Inception-v3 are 100352 and 131072 respectively. Following the first feature extraction phase, a second dimensionality reduction phase was applied to the extracted pre-trained features. The selection of the most discriminative features will always help to speed up classifier training and system robustness. This method will help the model select the significant features and reduce the feature dimension. Linear and nonlinear dimensionality reduction techniques such as PCA, K-PCA, LDA, ICA, SVD, LLE, Modified LLE and Isomap were tested and are tabulated in Table 3. Among these, PCA and ICA showed the best results. PCA extracts orthogonal dimensions with the maximum variance of the training instances. PCA focuses on the mutual orthogonality of the major components and finds the uncorrelated components in a reduced feature space. PCA is chosen over ICA for discriminative feature selection in the second phase. These reduced feature sets are used to predict language probabilities. The optimal number of features for PCA was fixed at 300. This feature number was iteratively fixed by observing the classification accuracy results versus the number of features. Figure 3 depicts the graph of the number of features versus classification accuracy of the VGG16 model using PCA. We can see the trend of increasing accuracy with the number of components with a limit of 300. The classification accuracy initially has a steep increase, and then it has a zero slope after 300. Hence we selected the optimal number of components for PCA as 300. Thus, the extraction's feature size from phase 1 to phase 2 is reduced from 25088 to 300 in the case of VGG16. Therefore, using

TABLE 2. Details of classically augmented dataset.

Dataset-Name	Real	Noisy-AUG
IIT-M	7000	84000
IIIT-H	7000	84000
Mozilla Common Voice	5000	60000
VoxForge	8000	96000

TABLE 3. Comparison of model accuracy using different dimensionality reduction methods.

Name of the Method	Number of features	Accuracy
Without PCA	25088	93%
PCA	300	97%
K-PCA	300	95%
LDA	6	90%
ICA	300	96.9%
SVD	300	88%
LLE	1000	94.35%
Modified LLE	1000	94.79%
Isomap	1000	72%

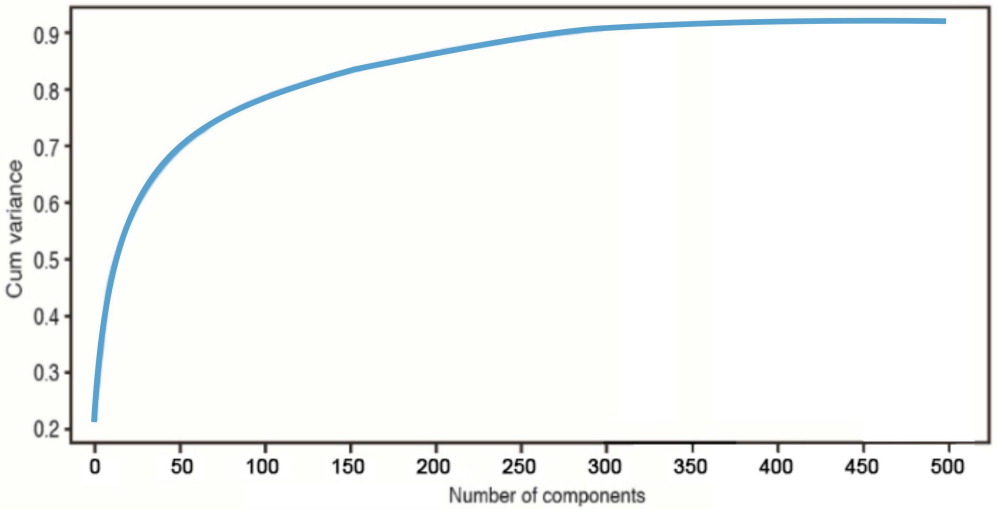


FIGURE 3. Plot of variance versus number of features of the VGG16 model using PCA.

PCA with a feature map of 300, there is an 82.62% decrease in the feature dimension from the pre-trained feature kernel map. This helped in reducing the memory requirements and improving the predictive accuracy.

D. ARCHITECTURES AND HYPERPARAMETERS DETAILS OF THE PRE-TRAINED MODELS

1) ARCHITECTURE OF VGG16

It is a commonly used pre-trained model [39] that is used primarily for images. It is an extensive network with approximately 138 million parameters. It uses 64, 128 and 256 convolutional filters and three layers of 512 convolutional filters and convolutional filters with max pooling layers. It uses two fully connected layers with 4096 nodes. The output layer is a softmax layer of 1000 nodes. Each layer consists of a large set of 3*3 convolutional filters. It has an input size of 224*224*3. Therefore, the mel-spectrogram images of the audio samples in our study were resized into 224*224*3 pixels. We utilized the ''Inter-Linear'' interpolation method for upsampling the dimension to 224. The unique feature of VGG16 is that instead of having many hyperparameters they focused on convolution layers of 3 $\times$ 3 filters with a stride of 1 and always used the same padding and maxpool layer of 2 $\times$ 2 filters of stride 2. It follows this arrangement of convolution and max pool layers consistently throughout the architecture. A general diagram is shown in Figure 4.

2) ARCHITECTURE OF RESNET50

RESNET50 [40] accepts an input of size 224*224*3 pixels. Consequently, the mel-spectrogram image must be scaled to 224*224*3 pixels. The Residual Network (RESNET) is a well-known deep learning pre-trained model. The problem of training very deep networks has been resolved by the introduction of residual blocks. The RESNET model is made up of these blocks. RESNET is derived from VGG19 and has residual connections between the layers. Skip connections are at the heart of a residual block. The problem of the vanishing gradient in deep CNNs is eliminated because of these skip connections. The skipped connections provide regularisation that may hurt the performance of the architecture. There is a 34-layer plain network in the architecture inspired by VGG19 in which the shortcut or skip connections are added. These skip connections or residual blocks convert the architecture into a residual network. The RESNET50 model consists of five stages, each with a convolution and an identity block. Each convolution block has three convolution layers and each identity block also has three convolution layers. RESNET50 has more than 23 million trainable parameters.

3) ARCHITECTURE OF INCEPTION-v3

In comparison to VGG16, Inception Networks (GoogLeNet/Inception-v1) [41] have proven to be more computationally efficient, both in terms of the number of parameters generated by the network and the economical cost incurred. The image input size was set to 299*299*3 in this case, which has specific advantages like factored convolution, smaller convolution (3*3 convolution), asymmetric convolution, and auxiliary classifier. Convolutions, average pooling, max pooling, concatenations, dropouts and fully linked layers are some of the symmetric and asymmetric building components that constitute the model. The model makes considerable use of batch normalisation, which is also applied to the activation inputs. Softmax was used to calculate the loss. The architecture of the Inception-v3 network was developed gradually and methodically, as follows:

1) Factorized Convolutions: This decreases the number of parameters used in the network, which lowers the computational complexity. It also monitors the effectiveness of the network.
2) Smaller convolutions: Training progresses more quickly when smaller convolutions are substituted for larger ones. A 5*5 filter has 25 parameters; two 3*3

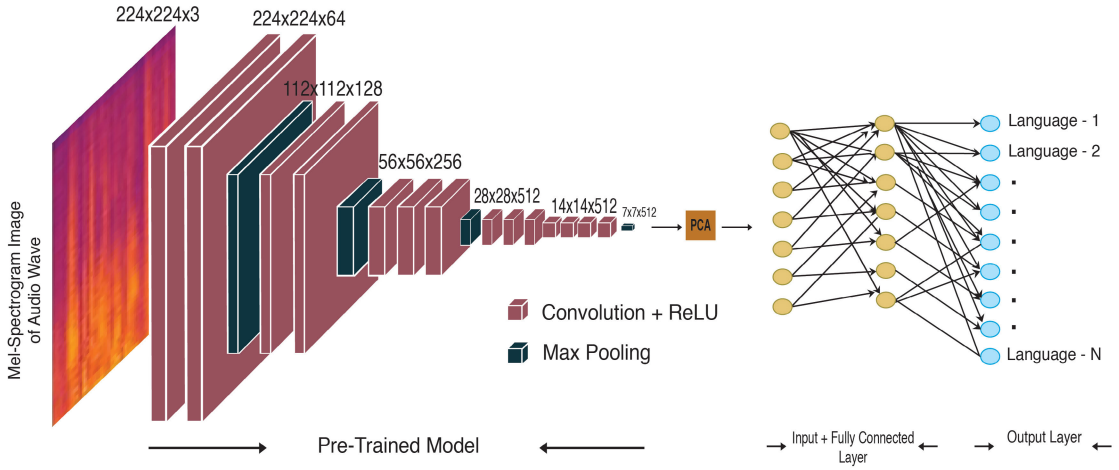


FIGURE 4. General depiction of the proposed model: Feature extraction using pre-trained model and ANN as classifier.

TABLE 4. Hyperparameters selected for the proposed model.

Models	Optimiser	Learning rate	Active function forall fully connected layers except the output layer	Active function for the output layer	Number of neurons in customised layer	
VGG16	RmsProp	0.0001	Relu	Softmax	256	64
RESNET50	Adam	0.0001	Relu	Softmax	256	64
Inception-v3	RrmsProp	0.0001	Relu	Softmax	256	64

filters in place of a 5*5 convolution have only 18 (3*3 + 3*3) parameters.

3) Asymmetric convolutions: A 1*3 convolution followed by a 3*1 convolution can be used instead of a 3*3 convolution. The number of parameters would be slightly larger than that of the suggested asymmetric convolution if a 3*3 convolution was swapped out for a 2*2 convolution.
4) Auxiliary classifier: A small CNN called an auxiliary classifier is placed between layers during training, and the loss it incurs is added to the loss of the primary network. In Inception-v3, an auxillary classifier is used as a regulariser.
5) Grid size reduction: Grid size reduction is usually done by pooling operations.

E. CLASSIFIERS AND HYPERPARAMETER SETTINGS

We utilized different classifiers like ANN, SVM, KNN, LR, DT, NB and RF for the classification task. Since ANN gives the best results for the proposed model, a detailed explanation about the ANN architecture and the selection of hyperparameters is given below. Although the other classifiers are famous machine learning classifiers, their hyperparameters settings are also discussed.

1) TRANSFER LEARNING WITH ANN

The input, hidden and output layers are the three layers of nodes that must be present in a basic ANN. Here the fully connected feed-forward neural network has two fully connected (FC) layers, followed by an output layer. For example, in VGG16 on top of the input layer, the stacked fully connected layers have 256 and 64 neurons respectively. Thus, an input at level j, x_j is mapped to its corresponding activation y_j (input of the layer above) as shown below. The activation function for these two layers is Rectified Linear Unit function (RELU).

$$y_j = RELU(x_j) = max(o, x_j) \tag{2}$$

$$x_j = b_j + \sum_i w_{ij} y_i \tag{3}$$

where i is an index over the units of the layer below and b_j is the bias of unit j. The number of languages under study decides the number of output nodes. The output layer is then configured as a softmax layer, where hidden units map input x_j to a class probability p_j in the form

$$p_j = \frac{exp(x_j)}{\sum_i exp(x_j)} \tag{4}$$

where i is an index over all the classes. As a cost function for back-propagating gradients in the training stage, we used a cross-entropy function. Different numbers of hidden layers with different numbers of neurons were tested. Note that the presented architecture works at the frame level, meaning that each single frame (plus its corresponding context) is fed-forward through the network, obtaining a class posterior probability for all target languages.

2) HYPERPARAMETER SELECTION - PRE-TRAINED MODELS

Hyperparameters are variables that specify the network structure, such as the number of hidden units, dropout, activation function, and weight initialization, and how the network is trained such as learning rate, batch size, and epochs. The process of selecting appropriate hyperparameter values for a learning algorithm is known as hyperparameter tuning. The different parameters used here are the optimizer, learning rate, activation function and number of neurons in the customized layers. The search space for the hyperparameters was selected based on the previous works found in the literature. The specific values for these parameters were chosen by using a grid search. The search space for the optimizers is Adam, RMSProp, and SGD. The model was tested with two learning rates with values 1e-3 and 1e-4. The search space for the activation function comprised ReLu, Tanh and Softmax, and the possible number of neurons in the customized layers were 64, 128, 256, and 512. Table 4 shows the hyperparameter details selected for the proposed model.

3) TRANSFER LEARNING WITH OTHER CLASSIFIERS

The performance of the saved model feature parameters from the pre-trained models was also compared with different classifiers like SVM, KNN, LR, DT, NB, and RF. The hyperparameters of the classifiers were selected after different iterations. A majority voting ensemble method is explored to obtain the best results for each classifier. In an ensemble technique, a few poor learners are combined to perform better than the basic models. The ensemble technique may use weighted voting among the classifiers to forecast the label of occurrences. We fit five different versions of the KNN algorithm, each with a different number of neighbours used when making the predictions. We used one, three, five, seven, and nine neighbours (odd numbers to avoid ties). When k increases from 1 to 9, the error of the KNN classifier subsequently increases. The SVM algorithm does not natively predict probabilities, although it can be configured to predict probability-like scores. We fit five different versions of the SVM algorithm with a polynomial kernel, each with a different polynomial degree, set via the ''degree'' argument. We used degrees 1-5. A value of C=100 and γ=2*10^{-2} is selected for all the polynomial degrees. Our expectation is that by combining the predicted class membership probability scores predicted by each different SVM model, the soft voting ensemble will achieve a better predictive performance than any standalone model used in the ensemble, on average. We used a function called get_voting that creates the SVM models and combines them into a soft voting ensemble. The parameter that significantly affect the performance of the RF classifier is ntree (number of trees). So we set ntree values from 1:9. We can also demonstrate ensemble voting for classification with a decision tree algorithm. Here, each DT with a different maximum depth of the decision tree is set using the maximum depth argument. We used different depth values of 1-9. For the Logistic Regression model, we applied an L2 penalty assigning equal class weights to the output classes. The hyperparameters were obtained in repeated stratified cross validation, with 10-fold cross-validation with a repeat count of 3.

F. EVALUATION METRICS

We have assessed the model performance using a number of criteria including accuracy, precision and recall. Accuracy gives more importance to tp and tn whereas precision and recall give more importance to fp and fn.

1) Accuracy: It is the ratio of correct outputs compared to the total number of outputs.

$$Accuracy = \frac{tp + tn}{tp + fp + tn + fn} \tag{5}$$

2) Precision: It is the ratio of correct positive predictions to the total predictions from the positive class.

$$Precision = \frac{tp}{tp + fp} \tag{6}$$

3) Recall: The recall is used to measure the fraction of positive patterns that are correctly classified.

$$Recall(r) = \frac{tp}{tp + tn} \tag{7}$$

4) F1 Score: F1 score (also known as F-measure or balanced F-score) is an error metric that measures model performance by calculating the harmonic mean of precision and recall for the minority positive class.

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{8}$$

VI. RESULTS AND DISCUSSION

Google Colab was used to implement all the models. The mel-spectrogram images of audio samples were generated, and these images were used to create the training and testing sets. These were fed into the pre-trained models as inputs. Different input sizes are required for other pre-trained models. As a result, the sizes were changed to meet the requirements.

A. RESULTS OF CLASSICAL AUGMENTATION: CLASSICAL-AUG

Since both the pre-trained model and ANN are neural networks, the resulting combined model can be referred to as a modified pre-trained model. Hence, in this study, we represent the modified pre-trained models with PCA as MOD-VGG16-PCA, MOD-RESNET50-PCA and MOD-Inception-v3-PCA. Similarly, the respective modified pre-trained models without PCA were named as MOD-VGG16, MOD-RESNET50 and MOD-Inception-v3.

1) VGG16

Compared to ResNet-50 and Inception-v3, VGG16 has more depth of layers. So it will be definitely helpful for the identification of the most low level features. Table 5 shows

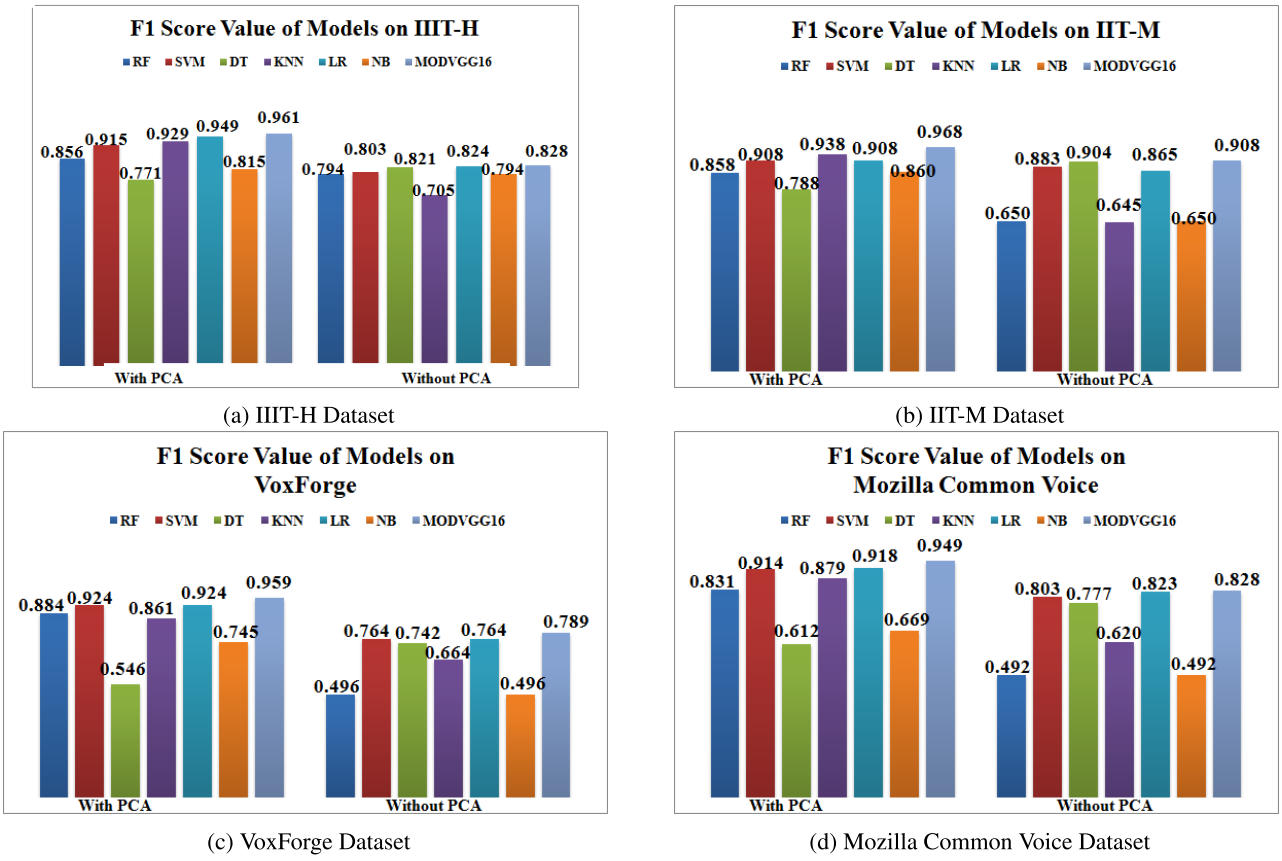
TABLE 5. Accuracy of modified VGG16 with and without PCA using classical augmentation.

CLASSICAL-AUG-WITH PCA													
		IIIT-H			IIT-M			Mozilla Common Voice			VoxForge		
Slno	Classifier	Accuracy	Precision	Recall	Accuracy	Precision	Recall	Accuracy	Precision	Recall	Accuracy	Precision	Recall
1	RF-PCA	0.854	0.857	0.855	0.860	0.865	0.852	0.849	0.831	0.832	0.884	0.885	0.884
2	SVM-PCA	0.914	0.915	0.915	0.914	0.911	0.906	0.907	0.916	0.912	0.924	0.925	0.924
3	DT-PCA	0.769	0.772	0.770	0.799	0.788	0.788	0.617	0.615	0.610	0.547	0.545	0.547
4	KNN-PCA	0.929	0.929	0.929	0.943	0.939	0.938	0.875	0.873	0.885	0.860	0.862	0.860
5	LR-PCA	0.948	0.949	0.949	0.914	0.911	0.906	0.927	0.913	0.923	0.924	0.925	0.924
6	NB-PCA	0.832	0.817	0.813	0.849	0.867	0.853	0.653	0.687	0.652	0.730	0.761	0.730
7	MOD-VGG16-PCA	0.971	0.960	0.963	0.969	0.968	0.969	0.956	0.949	0.949	0.959	0.959	0.959
CLASSICAL-AUG-WITHOUT-PCA													
Slno	Classifier	Accuracy	Precision	Recall	Accuracy	Precision	Recall	Accuracy	Precision	Recall	Accuracy	Precision	Recall
1	RF	0.794	0.794	0.794	0.655	0.651	0.649	0.497	0.494	0.491	0.523	0.513	0.480
2	SVM	0.801	0.805	0.801	0.884	0.883	0.884	0.815	0.805	0.802	0.773	0.766	0.762
3	DT	0.819	0.822	0.820	0.910	0.906	0.903	0.761	0.777	0.778	0.758	0.741	0.744
4	KNN	0.704	0.707	0.704	0.620	0.695	0.602	0.613	0.621	0.620	0.679	0.669	0.660
5	LR	0.815	0.825	0.824	0.854	0.863	0.867	0.825	0.821	0.826	0.773	0.766	0.762
6	NB	0.794	0.794	0.794	0.655	0.651	0.649	0.497	0.494	0.491	0.523	0.513	0.480
7	MOD-VGG16	0.826	0.830	0.826	0.909	0.908	0.909	0.840	0.830	0.827	0.798	0.791	0.787

the various accuracies produced by the different classifiers with the modified VGG16 trained model using the classical augmentation data. Here the model's accuracy, precision, recall and F1 for different classifiers with and without PCA are tabulated. Compared to models using other machine learning classifiers, the MOD-VGG16-PCA model yields good results. As found in many papers, it is observed here also that ANN gives better accuracy and performance compared to other M/L classifiers. From the results, it is evident that all classifiers with PCA outperformed those without PCA. For the IIIT-H dataset the MOD-VGG16-PCA model gives the highest accuracy of 97.1%. The reduced accuracy of 82% for the MOD-VGG16 (without PCA) indicates the importance of significant feature selection for classification accuracy. So using this MOD-VGG16-PCA method an improvement of 17.55 % was obtained in the VGG16 model performance. The precision and recall for the best model were 0.960 and 0.963 respectively. This high precision value indicates the reliability of this model in predicting the true class. Based on the results, only LR-PCA, KNN-PCA and SVM-PCA with their best parameters can, to some extent, compete with the proposed method. The accuracies of the LR-PCA, KNN-PCA and SVM-PCA were almost 94.8%, 92.9% and 91.4% respectively. Generally it is found that LR will give better results with proper selection of features. So the VGG16-PCA in combination with LR is able to give good results here too. It is known that KNN gives better results than SVM when the training data size is much larger than the no. of features ($m \gg n$). This could be why KNN performs well in our experiments. But the DT-PCA is showing the lowest accuracy of 76.9%. For the other three datasets IIT-M, Mozilla Common Voice and VoxForge, the MOD-VGG16-PCA gives the best results of 96.9%, 95.6% and 95.9% respectively. The drop in model performance for these datasets in comparison with IIIT-H was around 1%. An F1 score comparison is shown in Figure 5. F1 scores of 0.961, 0.968, 0.949, and 0.959 were obtained by the model (MOD-VGG16-PCA) on IIIT-H, IIT-M, Mozilla Common Voice, and VoxForge datasets respectively. All these high F1 scores proves the proposed model's (MOD-VGG16-PCA) ability to both capture positive cases (recall) and be accurate with the cases it does capture (precision).

2) RESNET50

RESNET50 model with the help of residual connections is trying to fit a residual mapping. These residual mappings may not be able to capture all the low level features from the voice samples. This may be one of the reasons for the accuracy drop in the MOD-RESNET50-PCA compared to with other modified pre-trained models. The best results of all the models come from the MOD-RESNET50-PCA models. Table 6 and Figure 6 show the results of RESNET50 for different datasets. The classification accuracy of this model was 93.33%. The accuracy obtained in comparison with VGG16 and Inception-v3, dropped by 5%. The MOD-RESNET50-PCA method provides the highest accuracy for all four datasets. The accuracies for VoxForge and Mozila



(a) IIIT-H Dataset

(b) IIT-M Dataset

(c) VoxForge Dataset

(d) Mozilla Common Voice Dataset

FIGURE 5. F1 comparison of modified VGG16 with different classifiers.

Common Voice datasets are only 87% and 86% respectively. For the IIIT-H dataset, RF-PCA is showing somewhat better results of 90.7% as compared to other machine classifiers. Since RF is an ensemble learning technique, it can keep the moderate variance, eventually improving accuracy. For the IIT-M, Mozilla Common Voice and Voxforge, RF gives accuracies of 71.8%, 80.4% and 85.6% respectively. All other classifiers do not yield promising results with the RESNET50 architecture. The models were trained using the Adam optimizer with a learning rate of 0.0001. To reduce overfitting within the models, the feed forward and convolutional layers contain an L2 penalty. All the models used a batch size of 32. The F1 scores of the model based on RESNet50 (MOD-RESNET50-PCA) are 0.950, 0.907, 0.847, and 0.879 on IIIT-H, IIT-M, Mozilla Common Voice, and VoxForge datasets, respectively.

3) INCEPTION-v3

Since Inception-v3 is known to be more competent for identifying the low level features with fewer layers, it also shows the best results in our experiment, comparable to that given by VGG16. The best results of the Inception-v3 model came from MOD-Inception-v3-PCA on the IIIT-H dataset. Here MOD-Inception-v3 gives the best results in comparison with other machine learning classifiers. Even though KNN-PCA is also competent and gives the same results, its precision and accuracy are lesser than those of the MOD-Inception-v3-PCA method. On the IIT-M, Mozilla Common Voice and VoxForge datasets, the accuracies were 96.11%, 95.4% and 95.3% respectively. So KNN also gives the best results in line with the results obtained with VGG16. Precision and recall also provided consistent results. Table 7 and Figure 7 show the results of Inception-v3 for different datasets. On the IIIT-H, IIT-M, VoxForge, and Mozilla Common Voice datasets, the model (MOD-Inception-v3-PCA) achieved F1 scores of 0.975, 0.963, 0.953, and 0.963 respectively.

B. RESULTS OF SYNTHETIC AUGMENTATION: SYNTH-CLASSICAL-AUG

As a baseline, we used classical data augmentation (see Section B in Section V). We refer to this network as PRETRAINED-CLASSICAL-AUG. The two best models were found to be the MOD-VGG16-PCA model and the MOD-Inception-v3-PCA model, with both models trained using classical augmentation. Although these models achieved accuracies of 97% and 96% respectively, their accuracies reduced to 87.5% and 84.2% respectively when the models were tested on unseen sentences from other datasets. The RESNET50 model could only achieve an accuracy

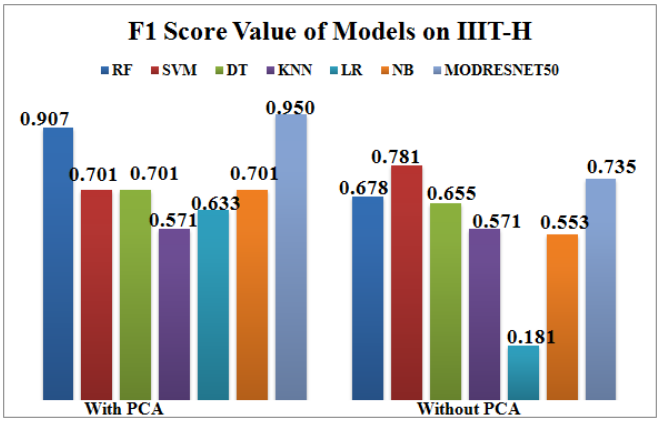
TABLE 6. Accuracy of modified RESNET50 with and without PCA using classical augmentation.

CLASSICAL-AUG-WITH PCA												
		IIIT-H			IIT-M			Mozilla Common Voice			VoxForge	
Slno	Classifier	Accuracy	Precision	Recall	Accuracy	Precision	Recall	Accuracy	Precision	Recall	Accuracy	Precision	Recall
1	RF-PCA	0.907	0.908	0.907	0.718	0.722	0.688	0.804	0.811	0.804	0.856	0.857	0.856
2	SVM-PCA	0.698	0.703	0.699	0.271	0.269	0.263	0.298	0.392	0.210	0.446	0.449	0.445
3	DT-PCA	0.698	0.703	0.699	0.496	0.484	0.482	0.285	0.185	0.205	0.553	0.559	0.552
4	KNN-PCA	0.567	0.575	0.567	0.505	0.496	0.491	0.255	0.230	0.223	0.456	0.459	0.456
5	LR-PCA	0.627	0.639	0.628	0.133	0.002	0.081	0.298	0.392	0.210	0.445	0.449	0.445
6	NB	0.698	0.703	0.699	0.684	0.688	0.679	0.325	0.481	0.350	0.472	0.582	0.472
7	MOD-RESNET50-PCA	0.933	0.949	0.952	0.914	0.906	0.908	0.861	0.847	0.848	0.873	0.885	0.873
CLASSICAL-AUG-WITHOUT-PCA												
Slno	Classifier	Accuracy	Precision	Recall	Accuracy	Precision	Recall	Accuracy	Precision	Recall	Accuracy	Precision	Recall
1	RF	0.678	0.678	0.679	0.703	0.744	0.656	0.643	0.644	0.649	0.750	0.753	0.750
2	SVM	0.793	0.772	0.791	0.242	0.129	0.183	0.243	0.253	0.206	0.185	0.178	0.185
3	DT	0.652	0.657	0.653	0.412	0.401	0.400	0.168	0.039	0.132	0.591	0.593	0.591
4	KNN	0.567	0.575	0.567	0.455	0.446	0.423	0.283	0.281	0.256	0.456	0.459	0.456
5	LR	0.193	0.172	0.192	0.124	0.121	0.124	0.279	0.345	0.300	0.185	0.178	0.185
6	NB	0.525	0.591	0.520	0.307	0.298	0.302	0.279	0.345	0.300	0.221	0.203	0.221
7	MOD-RESNET50	0.780	0.734	0.736	0.523	0.478	0.483	0.568	0.573	0.574	0.516	0.531	0.518

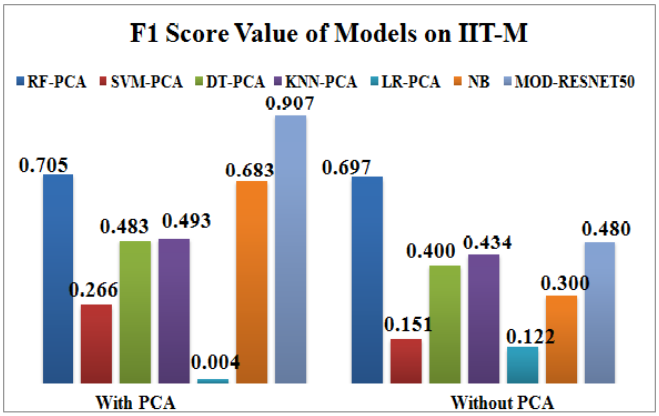
TABLE 7. Accuracy of modified Inception-v3 with and without PCA using classical augmentation.

CLASSICAL-AUG-WITH PCA												
		IIIT-H			IIT-M			Mozilla Common Voice			VoxForge	
Slno	Classifier	Accuracy	Precision	Recall	Accuracy	Precision	Recall	Accuracy	Precision	Recall	Accuracy	Precision	Recall
1	RF-PCA	0.963	0.914	0.913	0.907	0.907	0.907	0.857	0.862	0.882	0.732	0.738	0.963
2	SVM-PCA	0.950	0.942	0.941	0.931	0.937	0.935	0.904	0.894	0.897	0.872	0.874	0.990
3	DT-PCA	0.740	0.690	0.690	0.685	0.686	0.685	0.574	0.405	0.499	0.463	0.465	0.740
4	KNN-PCA	0.970	0.917	0.917	0.892	0.897	0.895	0.863	0.855	0.846	0.858	0.861	0.967
5	LR-PCA	0.940	0.945	0.942	0.901	0.905	0.902	0.905	0.894	0.897	0.872	0.874	0.990
6	NB	0.849	0.804	0.800	0.821	0.829	0.825	0.701	0.732	0.669	0.558	0.682	0.849
7	MOD-Inception-v3-PCA	0.970	0.978	0.973	0.961	0.964	0.962	0.954	0.962	0.965	0.953	0.954	0.953
CLASSICAL-AUG-WITHOUT-PCA												
Slno	Classifier	Accuracy	Precision	Recall	Accuracy	Precision	Recall	Accuracy	Precision	Recall	Accuracy	Precision	Recall
1	RF	0.831	0.822	0.833	0.777	0.774	0.774	0.837	0.815	0.805	0.768	0.770	0.831
2	SVM	0.828	0.827	0.822	0.831	0.834	0.832	0.907	0.916	0.913	0.794	0.797	0.828
3	DT	0.659	0.662	0.665	0.665	0.662	0.663	0.242	0.079	0.204	0.841	0.843	0.659
4	KNN	0.897	0.896	0.895	0.881	0.887	0.883	0.803	0.817	0.804	0.781	0.787	0.897
5	LR	0.901	0.904	0.905	0.868	0.870	0.871	0.863	0.864	0.866	0.794	0.797	0.901
6	NB	0.762	0.764	0.773	0.746	0.744	0.741	0.654	0.687	0.652	0.616	0.632	0.762
7	MOD-Inception-v3	0.967	0.966	0.991	0.941	0.946	0.945	0.932	0.935	0.953	0.919	0.914	0.919

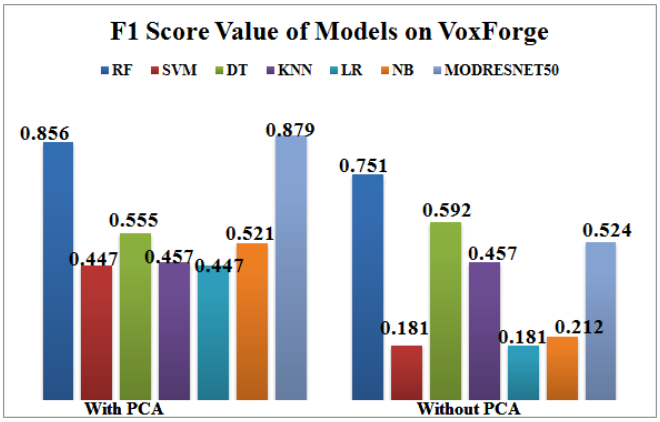
of 83.7%. This indicates an overfitting even though we utilised a classical data augmentation to enlarge the dataset. The second step of the experiment involved generating synthetic voice samples for data augmentation using the TTS model. Synthetic audio data augmentation can be utilized to increase the diversity and expand the dataset. Here a two-step



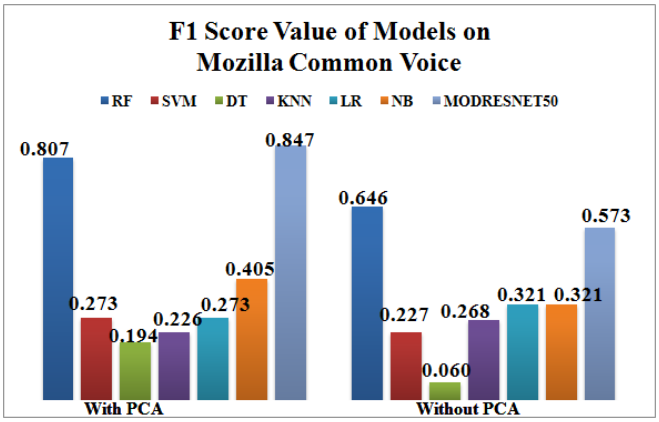
(a) IIIT-H Dataset



(b) IIT-M Dataset



(c) VoxForge Dataset



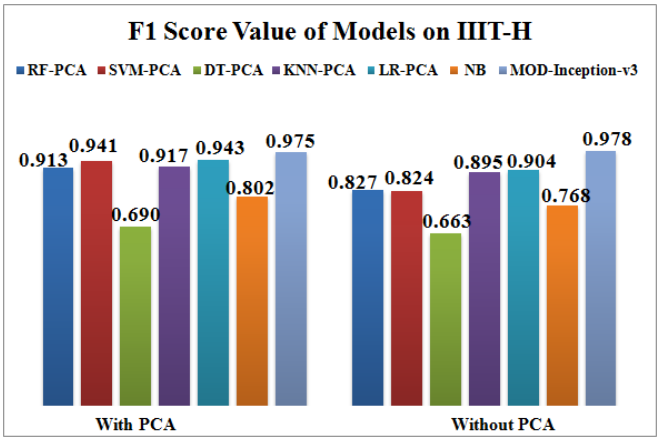
(d) Mozilla Common Voice Dataset

FIGURE 6. F1 comparison of modified RESNET50 with different classifiers.

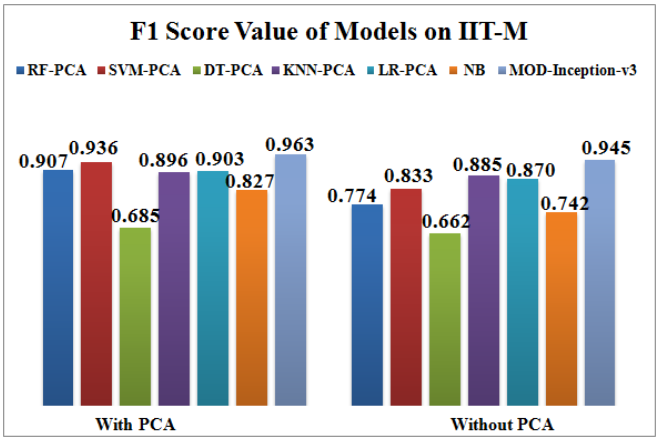
procedure was used to generate the new enlarged dataset. In the first step, new audio samples were created using the TTS model and added to the training samples. A detailed explanation of the creation of the synthetic voice samples is provided in Section V. The embeddings of these new synthetically augmented data were also used for training. In the second step, these synthetic samples created by the TTS model now act as an additional resource for generating new instances of audio samples. Classical augmentations like pitch shifting, noise addition, gain shifting, speed scaling and time shifting were performed on the synthetic voice samples created by the TTS model, thus generating more voice samples for training. In this way, we made many synthetic voice samples for all classes and used these for training the model. The training dataset now contains the original voice samples and the newly generated voice samples. This is referred to as the SYNTH-CLASSICAL-AUG. The synthetic augmentation results again indicate that the MOD-VGG16-PCA method provides a better understanding of the languages. We recorded the classification accuracies for SYNTH-CLASSICAL-AUG in Table 8. The model MOD-VGG16-PCA gives an accuracy of 94.3%. With synthetic data augmentation, the RESNET50 accuracy increased from 83% to 87%. Similarly, Inception-v3 shows a 6.1% increase in classification accuracy. It is evident that the results improved with synthetic augmentation. An F1 comparison and an accuracy comparison of the modified pre-trained models with PCA using classical augmentation and synthetic augmentation are shown in Figures 8 and 9.

1) INVESTIGATION OF THE QUALITY OF SYNTHETIC AUGMENTED DATA

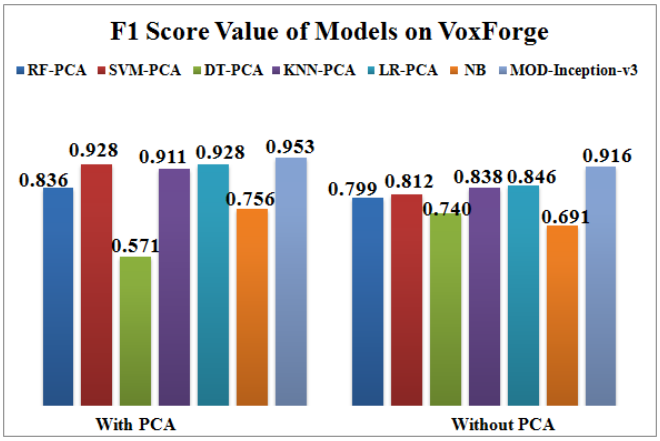
SPLID models require time-consuming and expensive operations for data collection and labelling. But this can be solved using synthetic augmentation. It can be observed from the earlier discussions that the model provides promising results with synthetic augmentation by speech synthesis. Similarly, by adopting speech synthesis techniques to augment audio utterances in the datasets, the operating costs for dataset creation can also be reduced. But if the synthetic data do not have sufficient quality, then it will not accurately reflect the original voice samples. Hence the model can yield unpredictable results. This section of the study attempts to investigate the model performance by adding synthetic augmentation data with varying qualities.



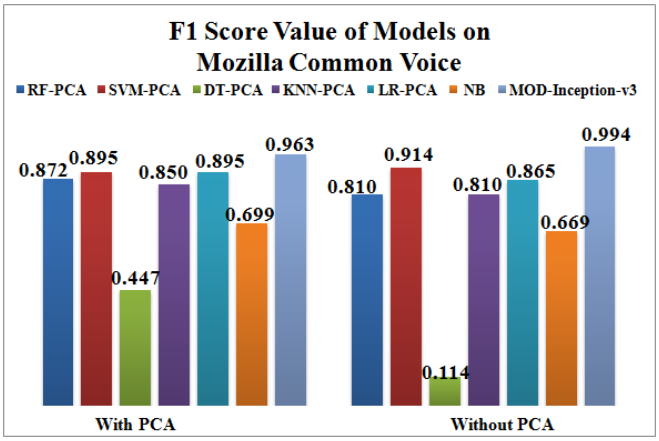
(a) IIIT-H Dataset



(b) IIT-M Dataset



(c) VoxForge Dataset



(d) Mozilla Common Voice Dataset

FIGURE 7. F1 comparison of modified Inception-v3 with different classifiers.

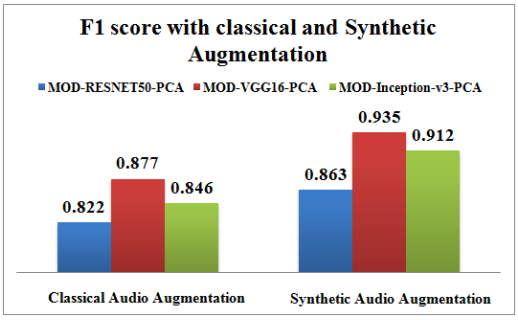


FIGURE 8. F1 Comparison of the modified pre-trained models with PCA using classical augmentation and synthetic augmentation.

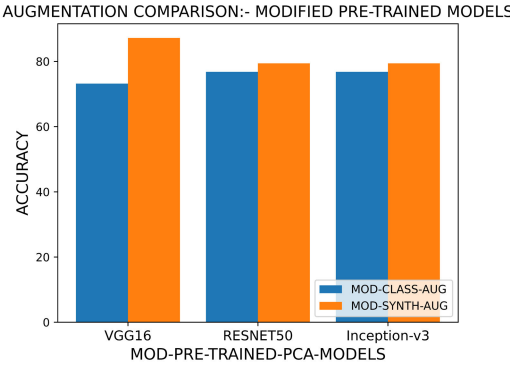


FIGURE 9. Accuracy comparison of the modified pre-trained models with PCA using classical augmentation and synthetic augmentation.

To generate synthesized utterances of varying quality, we utilised a speciality of the Indian language TTS model which was created by exploiting the properties of the Indian languages. In this model, a uniform HMM framework is exploited for building speech synthesizers. A language-independent phone set was then derived in this model. These common phone and common question sets were used to build HTS based systems. This speciality allows us to synthesise a language by utilising other language models. Among the utterances generated in this way, some were of high quality and some were of low quality. In our experiment, we have generated only Malayalam language utterances. Malayalam sentences are given to voice models of Tamil, Telugu, Kannada, Marathi and Hindi for the creation of synthesized

TABLE 8. Accuracy of different modified Pre-Trained models VGG16, RESNET50, Inception-v3 with PCA using classical and synthetic Augmentation.

VGG16							
Slno	Classifier	IIT-M with Classical Voice Augumenation			IIT-M with synthetic Voice Augumentation		
		CLASSICAL-AUG			SYNTH-CLASSICAL-AUG		
		Accuracy	Precision	Recall	Accuracy	Precision	Recall
1	RF-PCA	0.81	0.822	0.825	0.933	0.918	0.916
2	SVM-PCA	0.848	0.858	0.857	0.931	0.927	0.929
3	DT-PCA	0.222	0.098	0.21	0.545	0.332	0.442
4	KNN-PCA	0.807	0.812	0.862	0.916	0.899	0.899
5	LR-PCA	0.523	0.477	0.475	0.576	0.566	0.564
6	NB-PCA	0.546	0.646	0.493	0.683	0.706	0.672
7	MOD-VGG16-PCA	0.875	0.876	0.878	0.943	0.932	0.938
RESNET50							
Slno	Classifier	IIT-M with Classical Voice Augumenation			IIT-M with synthetic Voice Augumentation		
		CLASSICAL-AUG			SYNTH-CLASSICAL-AUG		
		Accuracy	Precision	Recall	Accuracy	Precision	Recall
1	RF-PCA	0.674	0.669	0.662	0.809	0.803	0.801
2	SVM-PCA	0.318	0.412	0.396	0.834	0.835	0.822
3	DT-PCA	0.305	0.204	0.225	0.735	0.721	0.722
4	KNN-PCA	0.275	0.251	0.243	0.840	0.812	0.756
5	LR-PCA	0.318	0.412	0.396	0.872	0.861	0.893
6	NB-PCA	0.345	0.501	0.416	0.562	0.541	0.538
7	MOD-RESNET50-PCA	0.837	0.822	0.823	0.871	0.864	0.862
Inception-v3							
Slno	Classifier	IIT-M with Classical Voice Augumenation			IIT-M with synthetic Voice Augumentation		
		CLASSICAL-AUG			SYNTH-CLASSICAL-AUG		
		Accuracy	Precision	Recall	Accuracy	Precision	Recall
1	RF-PCA	0.837	0.821	0.839	0.877	0.873	0.876
2	SVM-PCA	0.842	0.846	0.847	0.858	0.866	0.855
3	DT-PCA	0.242	0.079	0.204	0.594	0.425	0.519
4	KNN-PCA	0.803	0.817	0.804	0.883	0.874	0.865
5	LR-PCA	0.863	0.864	0.865	0.903	0.893	0.894
6	NB-PCA	0.654	0.686	0.652	0.72	0.752	0.689
7	MOD-Inception-v3-PCA	0.842	0.846	0.847	0.905	0.911	0.914

TABLE 9. Model accuracy with synthetically augmented voice samples of varying quality.

Language-Combination	Accuracy
Malayalam-Malayalam	0.938
Malayalm-Tamil	0.881
Malayalam-Telugu	0.843
Malayalam-Kannada	0.868
Malayalam-Hindi	0.813
Malyalam-Marathi	0.788

Malayalam utterances. Out of all the synthesized utterances, only intelligible utterances were chosen for training. The accuracy delivered by MOD-VGG16-PCA acting as a binary classifier for Malayalam is shown in Table 9. ''Malayalam-Malayalam'' indicates that the Malayalam utterances are synthesised using the Malayalam voice model. Similarly, ''Malayalam-Tamil'' indicates that the Malayalam voice samples are synthesised using the Tamil voice model. The use of low-quality synthesized audio samples has resulted in a decrease in accuracy. Here, all experiments, except ''Malayalam-Malayalam'' show a decrease in accuracy. This reveals that modelling based on unrealistic synthetic speech utterances cannot generate valuable insights. It also proves that the quality of the added synthetic data can also degrade the model performance.

VII. CONCLUSION AND FUTURE SCOPE

Currently ASV, IVR and Google Assistance systems are in high demand. Countries such as India require robust multi-lingual voice application systems. In this context, our study has evaluated the efficacy of synthetic augmentation on the task of spoken language identification of Indian languages. We have shown that a combination of a feature extraction stage composed of a prominent pre-trained computer vision model, a feature dimensionality reduction stage and a suitable classification stage yields excellent performance in the Indian SPLID task. The pre-trained models used in the study were VGG16, RESNET50, and Inception-v3. On the IIT-M dataset, VGG16 and Inception-v3 combined with PCA and ANN yielded a maximum accuracy of 97%. Synthetic augmentation proved to be the best method for increasing the model's accuracy. The scope of this work can be expanded in various directions. The effect of data variabilities like speaker variability, age variability etc, on the performance of SPLID systems can be further investigated. The use of pre-trained models like WaveNet and WaveGlow for speech synthesis and synthetic augmentation and their effect on system performance can be studied. Although the accuracy approaches that of the baseline, deploying pre-trained models with a classifier necessitates a larger RAM and a high-speed processing unit. As a result, further investigation into the reduction of model parameters by making use of only a subset of layers of the pre-trained computer vision models in the feature extraction stage can be pursued.

ACKNOWLEDGMENT

The authors would like to thank the IIIT-H TTS developers and INDIC TTS developers, especially IIT Madras and Mozilla Common Voice developers and VoxForge for the datasets.

REFERENCES

[1] P. Rangan, S. Teki, and H. Misra, "Exploiting spectral augmentation for code-switched spoken language identification," 2020, *arXiv:2010.07130*.

[2] M. Biswas, S. Rahaman, S. Kundu, P. K. Singh, and R. Sarkar, "Spoken language identification of Indian languages using MFCC features," in *Machine Learning for Intelligent Multimedia Analytics*. Singapore: Springer, 2021, pp. 249–272.

[3] S. Jothilakshmi, V. Ramalingam, and S. Palanivel, "A hierarchical language identification system for Indian languages," *Digit. Signal Process.*, vol. 22, no. 3, pp. 544–553, 2012.

[4] N. Dehak, P. A. Torres-Carrasquillo, D. Reynolds, and R. Dehak, "Language recognition via i-vectors and dimensionality reduction," in *Proc. 12th Annu. Conf. Int. Speech Commun. Assoc.*, Aug. 2011, pp. 1–4.

[5] I. Lopez-Moreno, J. Gonzalez-Dominguez, O. Plchot, D. Martinez, J. Gonzalez-Rodriguez, and P. Moreno, "Automatic language identification using deep neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2014, pp. 5337–5341.

[6] A. I. Abdurrahman and A. Zahra, "Spoken language identification using i-vectors, x-vectors, PLDA and logistic regression," *Bull. Electr. Eng. Informat.*, vol. 10, no. 4, pp. 2237–2244, Aug. 2021.

[7] G. Singh, S. Sharma, V. Kumar, M. Kaur, M. Baz, and M. Masud, "Spoken language identification using deep learning," *Comput. Intell. Neurosci.*, vol. 2021, pp. 1–12, Sep. 2021.

[8] S. Shukla and G. Mittal, "Spoken language identification using ConvNets," in *Proc. Eur. Conf. Ambient Intell.* Cham, Switzerland: Springer, Nov. 2019, pp. 252–265.

[9] R. Bedyakin and N. Mikhaylovskiy, "Low-resource spoken language identification using self-attentive pooling and deep 1D time-channel separable convolutions," 2021, *arXiv:2106.00052*.

[10] X. Wu, R. He, Z. Sun, and T. Tan, "A light CNN for deep face representation with noisy labels," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 11, pp. 2884–2896, Nov. 2018.

[11] B. Paul, S. Phadikar, and S. Bera, "Indian regional spoken language identification using deep learning approach," in *Proc. 6th Int. Conf. Math. Comput.* Singapore: Springer, 2021, pp. 263–274.

[12] S. A. Nossier, J. Wall, M. Moniri, C. Glackin, and N. Cannings, "An experimental analysis of deep learning architectures for supervised speech enhancement," *Electronics*, vol. 10, no. 1, p. 17, Dec. 2020.

[13] Y. Zhao, R. Togneri, and V. Sreeram, "Compressed high dimensional features for speaker spoofing detection," in *Proc. Asia–Pacific Signal Inf. Process. Assoc. Annu. Summit Conf. (APSIPA ASC)*, Dec. 2017, pp. 569–572.

[14] Y. Zhao, R. Togneri, and V. Sreeram, "Spoofing detection using adaptive weighting framework and clustering analysis," in *Proc. Interspeech*, Sep. 2018, pp. 626–630.

[15] A. K. Kumar, D. Paul, M. Pal, M. Sahidullah, and G. Saha, "Speech frame selection for spoofing detection with an application to partially spoofed audio-data," *Int. J. Speech Technol.*, vol. 24, no. 1, pp. 193–203, Mar. 2021.

[16] P. Rangan, S. Teki, and H. Misra, "Exploiting spectral augmentation for code-switched spoken language identification," in *Proc. WSTCSMC*, Oct. 2020, p. 36.

[17] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A simple data augmentation method for automatic speech recognition," in *Proc. Interspeech*, Sep. 2019, pp. 2613–2617.

[18] P. Shen, X. Lu, S. Li, and H. Kawai, "Conditional generative adversarial nets classifier for spoken language identification," in *Proc. Interspeech*, Aug. 2017, pp. 2814–2818.

[19] D. Snyder, D. Garcia-Romero, A. McCree, G. Sell, D. Povey, and S. Khudanpur, "Spoken language recognition using X-vectors," in *Proc. Odyssey*, Jun. 2018, pp. 105–111.

[20] C. Korkut, A. Haznedaroglu, and L. Arslan, "Comparison of deep learning methods for spoken language identification," in *Proc. Int. Conf. Speech Comput.* Cham, Switzerland: Springer, Oct. 2020, pp. 223–231.

[21] R. Duroselle, M. Sahidullah, D. Jouvet, and I. Illina, "Modeling and training strategies for language recognition systems," in *Proc. Interspeech*, Aug. 2021, pp. 1–6.

[22] C. Bartz, T. Herold, H. Yang, and C. Meinel, "Language identification using deep convolutional recurrent neural networks," in *Neural Information Processing*. Guangzhou, China: Springer, Nov. 2017, pp. 880–889.

[23] J. Valk and T. Alumäe, "VOXLINGUA107: A dataset for spoken language recognition," in *Proc. IEEE Spoken Lang. Technol. Workshop (SLT)*, Jan. 2021, pp. 652–658.

[24] S. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-28, no. 4, pp. 357–366, Aug. 1980.

[25] M. Todisco, H. Delgado, and N. Evans, "A new feature for automatic speaker verification anti-spoofing: Constant Q cepstral coefficients," in *Proc. Odyssey*, Jun. 2016, pp. 283–290.

[26] G. Hua, A. B. J. Teoh, and H. Zhang, "Towards end-to-end synthetic speech detection," *IEEE Signal Process. Lett.*, vol. 28, pp. 1265–1269, 2021.

[27] R. K. Das, J. Yang, and H. Li, "Long range acoustic features for spoofed speech detection," in *Proc. Interspeech*, Sep. 2019, pp. 1058–1062.

[28] R. Hemavathi and R. Kumaraswamy, "Voice conversion spoofing detection by exploring artifacts estimates," *Multimedia Tools Appl.*, vol. 80, no. 15, pp. 23561–23580, Jun. 2021.

[29] L. Huang and J. Zhao, "Audio replay spoofing attack detection using deep learning feature and long-short-term memory recurrent neural network," in *Proc. 2nd Int. Conf. Artif. Intell., Inf. Process. Cloud Comput. (AIIPCC)*, Jun. 2021, pp. 1–5.

[30] H. J. Choi and I. Y. Kwak, "Data augmentation in voice spoofing problem," *Korean J. Appl. Statist.*, vol. 34, no. 3, pp. 449–460, 2021.

[31] R. K. Das, J. Yang, and H. Li, "Data augmentation with signal companding for detection of logical access attacks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2021, pp. 6349–6353.

[32] Y. Zhao, R. Togneri, and V. Sreeram, "Replay anti-spoofing countermeasure based on data augmentation with post selection," *Comput. Speech Lang.*, vol. 64, Nov. 2020, Art. no. 101115.
[33] P. Bhaskararao, "Salient phonetic features of Indian languages in speech technology," *Sadhana*, vol. 36, no. 5, pp. 587–599, Oct. 2011.
[34] A. Baby, N. L. Nishanthi, A. L. Thomas, and H. A. Murthy, "A unified parser for developing Indian language text to speech synthesizers," in *Proc. Int. Conf. Text, Speech, Dialogue*. Cham, Switzerland: Springer, Sep. 2016, pp. 514–521.
[35] K. Prahallad, E. N. Kumar, V. Keri, S. Rajendran, and A. W. Black, "The IIIT-H Indic speech databases," in *Proc. 13th Annu. Conf. Int. Speech Commun. Assoc.*, Sep. 2012, pp. 1–4.
[36] *Indic TTS*. Accessed: Sep. 20, 2023. [Online]. Available: https://iitm.ac.in/donlab/tts
[37] VoxForge. *Free Speech Recognition*. Accessed: Sep. 20, 2023. [Online]. Available: https://www.voxforge.org
[38] R. Ardila, M. Branson, K. Davis, M. Kohler, J. Meyer, M. Henretty, R. Morais, L. Saunders, F. Tyers, and G. Weber, "Common voice: A massively-multilingual speech corpus," in *Proc. 12th Lang. Resour. Eval. Conf.*, 2020, pp. 4218–4222.
[39] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*. San Diego, CA, USA: Computational and Biological Learning Society, Apr. 2015, pp. 1–14.
[40] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[41] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2818–2826.
[42] *Python Package Index-PyPI*. Accessed: Sep. 20, 2023. [Online]. Available: https://pypi.org/project/selenium/
[43] *Selenium Clients and WebDriver Language Bindings*. Accessed: Sep. 20, 2023. [Online]. Available: https://www.selenium.dev/downloads/
[44] A. R. Ambili and R. Cherian Roy, "Multi tasking synthetic speech detection on Indian languages," in *Proc. Int. Conf. Innov. Trends Inf. Technol. (ICITIIT)*, Kottayam, India, Feb. 2022, pp. 1–6, doi: 10.1109/ICITIIT54346.2022.9744221.

A. R. AMBILI received the B.Tech. degree in electronics and communication engineering from the Cochin University of Science and Technology, in 2005, and the M.Tech. degree in electrical engineering with specialization in control and instrumentation from IIT Madras, in 2014. She is currently pursuing the Ph.D. degree in speech processing (under electronics and communication) with APJ Abdul Kalam Technological University, Kerala, India. Since 2006, she has been an Assistant Professor with the Department of Electronics and Communication, Federal Institute of Science and Technology, Kerala. She had almost 15 years of teaching experience. Her research interests include speech processing, image processing, natural language processing, deep learning, machine learning, and embedded systems. She is also a Rashtrapathy Guide Award Winner.

RAJESH CHERIAN ROY (Member, IEEE) received the B.E. degree in electronics and communication engineering from MREC, Jaipur, in 1996, the M.Tech. degree in electronics and communication engineering from the Cochin University of Engineering and Technology, in 2001, and the Ph.D. degree in signal processing from CUSAT, India, in 2010. He is currently a Professor in artificial intelligence with the Muthoot Institute of Science and Technology, Kerala. He has 20 years of teaching experience. He is an invited resource person for many of the conferences and workshops. His research interests include signal transforms, speech processing, and applications of machine learning. He is an active Professional Body Member of ISTE. He had various publications in reputed conferences and journals. He got funding from AICTE, India, to develop the Hardware Architecture for MRT Transform.