This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2941773, IEEE Transactions on Information Forensics and Security

Fusing MFCC and LPC Features using 1D Triplet CNN for Speaker Recognition in Severely Degraded Audio Signals

Anurag Chowdhury, *Student Member, IEEE,* Arun Ross, *Senior Member, IEEE*

***Abstract*—Speaker recognition algorithms are negatively impacted by the quality of the input speech signal. In this work, we approach the problem of speaker recognition from severely degraded audio data by judiciously combining two commonly used features: Mel Frequency Cepstral Coefficients (MFCC) and Linear Predictive Coding (LPC). Our hypothesis rests on the observation that MFCC and LPC capture two distinct aspects of speech, viz., speech perception and speech production. A carefully crafted 1D Triplet Convolutional Neural Network (1D-Triplet-CNN) is used to combine these two features in a novel manner, thereby enhancing the performance of speaker recognition in challenging scenarios. Extensive evaluation on multiple datasets, different types of audio degradations, multi-lingual speech, varying length of audio samples, etc. convey the efficacy of the proposed approach over existing speaker recognition methods, including those based on iVector and xVector.**

***Index Terms*—Speaker Recognition, Degraded Audio, Deep Learning, MFCC, LPC, 1-D CNN, Feature-level Fusion**

I. Introduction

SPEAKER recognition, sometimes known as voice biometrics, refers to the recognition of an individual based on their voice. Speaker recognition systems find applications in a number of different domains including telephone banking [5], E-Commerce [8] and forensics [6]. Human voice is also being incorporated into user interfaces for accessing and controlling smartphones [4] and personal virtual assistants [3] in the form of *voice-controlled user interfaces*. These interfaces use human voice for enabling services and speaker recognition is expected to be a critical component.

The performance of speaker recognition systems is adversely impacted by a number of factors. For example, noisy environments and animated conversations involving multiple subjects can confound a speaker recognition system. Further, the quality of the microphone and distance of the subject from the microphone can also lead to a marked drop in speaker recognition accuracy. In forensic applications, the audio signal may be severely degraded leading to difficulties in recognizing individuals.

In this work we aim to develop a speaker recognition algorithm that is robust to a wide range of audio capture quality, ambience and perturbations. Some of the current voice recognition enabled products [3] already incorporate advanced hardware based measures, such as circular arrays of far-field microphones, for enabling robust audio input interfaces, thereby aiding their speech and speaker recognition capabilities. Our goal, on the other hand, is to develop a software based solution that is not restricted to specific audio interfaces for performing robust speaker recognition.

Both authors are with the Department of Computer Science Engineering, Michigan State University, East Lansing, MI, 48823, USA, e-mail: {chowdh51, rossarun}@cse.msu.edu.

Impact of audio degradation on speaker recognition in real life scenarios

Most speaker recognition enabled products are vulnerable to challenging audio conditions such as low audio SNR, differing dialects and accents, and background noise [7]. The problem is exacerbated in presence of background noise [7]. One of the most challenging daily-life scenario for digital assistants is the babble noise [30] in crowded environments, such as coffee shops. Babble noise typically comprises speech from multiple individuals in the background. In this case, the voice commands from the intended user can be misinterpreted or even go unattended due to lack of usable audio data in the presence of extensive background noise.

In the next section, we will discuss some of the more established speaker recognition algorithms which help define the performance and utility of speaker recognition in real life scenarios. A majority of speaker recognition algorithms rely on some form of short-term spectral speech features like Mel Frequency Cepstral Coefficients (MFCC) and Linear Prediction Cepstral Coefficient (LPCC). However, their reliance on one type of speech feature constrains their performance and utility. For example, while MFCC features are known to represent perceptual speech, they are also unreliable in presence of audio degradations. This reduces the performance and reliability of algorithms based on MFCC features alone.

This is where we position our work in relation to the current existing literature, as detailed in the following sections. We propose a deep learning based algorithm, referred to as 1D-Triplet-CNN, to combine speech *perception* features and speech *production* features, given by MFCC and LPC features, respectively, for learning a joint feature space that efficiently models the entire *speech chain*. Finally, we describe the speaker verification experiments conducted in this work to demonstrate the superior representation capability of the joint feature space, even in the presence of varying types and strength of background noise.

II. Related work

Speaker recognition is often categorized into text-dependent and text-independent methods. While text-dependent speaker

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2941773, IEEE Transactions on Information Forensics and Security

recognition relies on the user to utter a fixed pass-phrase for the system to recognize the user, text-independent speaker recognition does not require any such fixed pass-phrase and can recognize the user from their vocal characteristics alone. In the scope of this work we will be focusing on text-independent speaker recognition as it presents a more general case and has direct use in a myriad of applications.

One popular approach for performing text-independent speaker identification uses gaussian mixture models (GMM) for modeling individual speakers based on the mel-frequency cepstral coefficients (MFCC) of their speech data [41]. Depending on the quality of the speech, speech utterances of length 5 to 15 seconds are required for estimating the GMM based speaker models. However, often in the real world, long speech utterances (> 5secs) are not available for training the GMM model of a speaker.

The GMM based approach was then extended in [39] to the application of speaker verification using background speaker normalization. The authors presented a technique for selecting a set of speakers to form a background speaker model. This model gave an estimate of the spread of speech features across the selected set of speakers. Every test speech utterance was then verified against the background speaker model using a maximal likelihood test. The authors showed their results on both the TIMIT dataset [19] (average SNR 53dB) and NTIMIT dataset [27] (average SNR 36dB). However, the conclusions on efficacy of the algorithm on noisier data ($<$ 20dB) was not established.

In another work by Campbell et al. [12], the approach based on kernelized SVMs was combined with the GMM based approach. The authors form a GMM-supervector from speaker adapted GMM [40]. The GMM-supervector is then used to map an input speech utterance to a high dimensional feature space to derive kernels for the SVM. These high dimensional GMM-supervectors often require sophisticated matchers; therefore, in order to enable the use of simple metrics, such as cosine similarity, for speaker verification, the high-dimensional GMM supervectors were transformed to a lower-dimensional space, called the total variability space, using factor analysis [15]. A given speech utterance in this total variability space is represented by a low-dimensional vector called i-vector [16]. The i-vectors are then used for speaker verification.

Both the iVector-PLDA [16] and UBM-GMM [40] based speaker recognition techniques use MFCC features for representing the speech data. However, it is important to note that the MFCC feature set is not robust [22] to audio degradations and can, therefore, potentially affect the performance of UBM-GMM and iVector-PLDA in noisy scenarios. Authors in [22] have, hence, used the Linear Predictive Cepstral Coefficients (LPCC) features, which are more robust to audio perturbations, for narrowing down the search space, and then used the MFCC features for performing speaker recognition in the reduced search space. In order to alleviate the problems caused by degraded audio, several spectrum estimation methods and speech enhancement techniques have also been evaluated as front-end processing techniques for developing robust speaker recognition methods. Voice activity detection is one such technique used for detecting parts of the audio with speech activity in them. Sadjadi et al. [44] used it as a front-end processing technique for detecting and removing non-speech parts of the audio, which are typically long noisy audio segments.

Since the audio data captured in unconstrained scenarios is mostly noisy, the focus of the speaker recognition research over the last decade has concentrated on speaker recognition in noisy audio conditions. Additive background noise is one of the most common type of noise encountered in speech signals. A popular technique commonly employed for dealing with effects of background noise is spectral subtraction [10]. One application of such technique is seen in the robust feature estimation method [49] that can capture source and vocal tract related speech properties from spectral-subtracted noisy speech utterances. While pre-processing the noisy speech to obtain near clean speech is one way of dealing with degraded audio signals, another approach [34] relies on training a classifier on noisy data to make it robust to audio degradations, thereby preempting the need for using any front-end processing techniques. Such an approach can work well when the degree and type of audio-degradations are constrained. However, often there is extensive degree of audio degradations, rendering portions of the audio unreliable for performing speaker recognition. The work in [32] combines missing data recognition with UBM-GMM for marginalizing feature values which are deemed to be unreliable, and for performing speaker recognition in noisy speech.

Apart from additive noise, *convolutive reverberations* are also commonly found in audio samples captured in indoor scenarios. The work in [51] addresses this issue in a two staged approach. They first use the noisy speech data to train a DNN classifier to produce a binary time-frequency (T-F) mask. The mask identifies and separates out the unreliable T-F units at each audio frame. The masked output audio is then evaluated using GMM-UBM speaker models, trained in reverberant environments, to perform speaker recognition. Another problem associated with speech production in noisy environments is the Lombard effect [25], where the speakers involuntarily tend to adjust their vocal effort in-order to accommodate the noisy environment, hence also perturbing their natural voice characteristics which can effect speaker recognition adversely. Authors in [25] have further established the dependence of Lombard speech on noise type and noise level using a GMM based Lombard speech type classifier.

Speaker recognition like many other classification tasks has attracted usage of deep learning based techniques [31] for improving the current state-of-art. Richardson et. al [43] used spectral audio features (like MFCC) for performing speaker recognition on the input frame using deep neural networks (DNN). Zhang et al. [50] trained multi-layer perceptrons and Deep Belief Networks for learning discriminative feature transformations, and de-reverberated features from noisy speech data affected with microphone reverberations. Deep learning techniques, however, require large amounts of training data. Since large amounts of data is not always readily available, Richardson et. al [42] mixed synthetically generated noisy data along with the available clean speech data to train

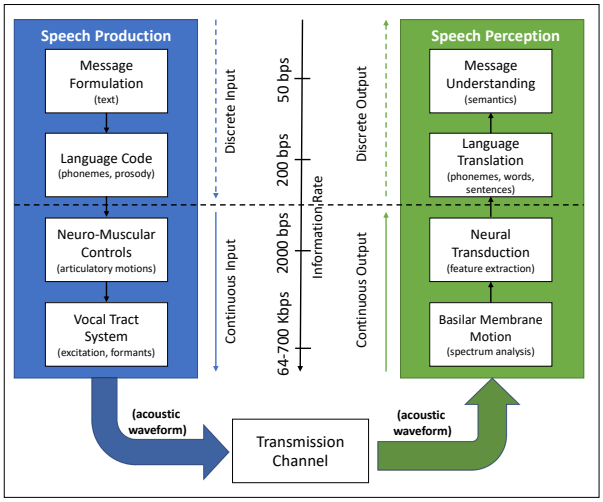


Figure 1. A visual representation of the speech chain as given in [23]

a denoising DNN that could be used as a front-end processing technique for speaker recognition. In one of the recent works [13], authors allude to the possibility of extracting glottal features for performing speaker recognition and have shown results for the same using their proposed 1D CNN based speaker recognition algorithm. In another work [47], a DNN based feature embedding, called xVector, was learnt from MFCC features and combined with PLDA classifier for performing speaker recognition. While the xVector technique is shown to outperform iVector-PLDA baseline on several public datasets, its performance on severely degraded data can not be commented upon. Also, due to the usage of a fully-connected DNN based architecture, the xVector model has almost 4.2 million learnable parameters, which necessitates the availability of a large amount of training data.

In the following sections, we will discuss the proposed technique we have designed for speaker recognition on considerably degraded speech data.

III. Theoretical Foundations

Text-independent speaker recognition can be seen as the process of extracting the speaker dependent characteristics from the human speech, regardless of the textual content within, for uniquely identifying the human speaker at the source. This process can be well described by using the speech chain [17] which expounds the physics and biology of the spoken language in an ordered fashion.

A. Speech Chain

The speech chain, illustrated in Figure 1, is typically used for explaining the physics and biology involved during formulation, articulation, propagation and reception of a message from a speaker to the listener. While the focus in speech chain at its extremities is on the message being transferred through the chain, it is also important to notice the change in information rate of the message as it passes through the chain. The information rate contained in the transmitted spoken message is significantly higher than the base information rate of the text message itself, as also shown in Figure 1. The 'Neuro-Muscular controls' in the speech production process encodes the pronunciation elements of the message as articulations and the 'Vocal Tract System' generates the sound from the articulation hence imparting the spoken language its acoustic properties. Thus, the articulatory and acoustic information in the speech leads to increase in the net information content of the speech. The speech perception process on other hand performs spectral analysis on the audio transmitted through the channel using the 'Basilar Membrane Motion' which is further passed through the 'Neural Transduction' phase to extract speech features essential for performing tasks like speech, speaker and language recognition.

From the perspective of speaker recognition, our interest is focused on two different parts of the speech chain. First being the 'Vocal Tract System' in the 'Speech production' process as it imparts the speech its acoustic properties and can be used for modeling the vocal tract system that gives an individual their unique voice characteristics. We use Linear Predictive Coding (LPC) for accomplishing this task and is elucidated upon in the upcoming sections. Second being the 'Neural Transduction' phase in 'Speech perception' process where the speech features, as perceived by the listener, are extracted from the speech audio and hence can be used for modeling the human auditory system that can discriminate between the speakers in the speech audio. We use Mel-Frequency Cepstral Coefficients (MFCC) for accomplishing this task and is further explained upon in the upcoming sections.

B. Vocal tract modeling using Linear Predictive Coding (LPC)

According to the source-filter model [33] of speech, human voice can be seen as filter (vocal tract) output of excitation from an energy source (lungs). Vocal tract of humans can be modeled as a time-varying digital filter. Furthermore, the all-pole model of filter design is chosen for easier estimation and analysis of the human vocal tract. Thus, the transfer function of the digital filter equivalent of vocal tract can be given by,

$$H(z) = \frac{G}{1 - \sum_{k=1}^{p} \alpha_k z^{-k}}. \tag{1}$$

Speech data is sequential in nature and, for modeling the vocal tract, we assume that the voice acoustics of the n^{th} speech sample ($S[n]$) can be viewed as a combination of p past speech samples. Thus, the n^{th} speech sample ($S[n]$) can be written as:

$$S[n] = \sum_{k=1}^{p} \alpha_k S[n-k] + G.u[n], \tag{2}$$

where, $S[n-k], k = 1, 2, 3 \ldots p$ are the p past speech samples, G is the gain factor, $u[n]$ is the excitation corresponding to the n^{th} speech sample, α_k's are the vocal tract filter coefficients, and "." represents the scalar multiplication operation. *Linear Predictive Coding* (LPC) is often referred to as "inverse filtering" as its aim is to determine the "all zero filter" which is the inverse of the vocal tract model. Just as in the source-filter model of human voice the LPC model also estimates voice acoustics of the n^{th} speech sample $\hat{S}[n]$, conditioned on previous speech samples, given as,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2941773, IEEE Transactions on Information Forensics and Security

$$\hat{S}[n] = \sum_{k=1}^{p} \hat{\alpha}_k S[n-k]. \tag{3}$$

where α_k's are the LPC parameters. The error in prediction is given by,

$$e[n] = S[n] - \hat{S}[n].kl \tag{4}$$

$$E = \sum_{n=1}^{N} (e(n))^2 \tag{5}$$

Minimizing the above energy (E), using auto-regressive modelling [18], will help obtain the Inverse-Vocal Tract filter model. The above energy can be minimized using auto-regressive modelling [18] for obtaining the Inverse-Vocal Tract filter model. The filter coefficients of the Inverse-Vocal Tract filter model are the LPC model parameters (α_k), which provide an estimate of the human vocal tract filter coefficients.

C. Perceptual speech features using Mel-Frequency Cepstral Coefficients (MFCC)

Humans are exceptionally good at identifying other *known* humans from their voice[24], even in presence in audio degradations. In order to model the human auditory perception system, it is important to understand how humans recognize speakers from their voice. Human auditory system is made up of outer ear, middle ear and inner ear. Our interests are more vested in the inner ear because this is where the auditory nerves pick up sound signals from the cochlea and deliver it to the brain. The human cochlea is a part of inner ear and its function is to separate sounds based on their frequency content and transduce the sound waves to electrical signals. This part of the auditory processing is done in 'Basilar Membrane Motion' phase in the speech chain, as given in Figure 1. The auditory nerve fibers then carry these electrical signals to the brain. Our brain then performs complex spectral and temporal processing of the sound signal, as shown in the 'Neural Transduction' phase in the speech chain in Figure 1, for extracting speech features. These features are then used for performing tasks like speech, speaker, and language recognition.

Mel-cepstral frequency coefficients (MFCC) have been used in literature [35], for modeling the human auditory perception system. MFCC feature is also a very popular choice for performing speaker recognition. Therefore, we also chose MFCC for representing the perceptual speech features in a given audio sample for our work. In the coming sections, we will discuss the MFCC feature extraction process in detail and will also elucidate upon the MFCC and LPC feature fusion and the deep learning architecture that we propose for speaker verification.

D. Rationale behind fusing LPC and MFCC features for speaker recognition

As discussed in the previous sections both LPC and MFCC model different characteristics of a speaker's voice which could be used separately for speaker recognition. However, the nature of the voice features being captured by LPC and MFCC are complimentary, as the MFCC features describe the perceptual speech features while the LPC features describe the vocal tract model for the speaker in a speech audio. The combination of MFCC and LPC features, therefore, uniquely represent the voice characteristics of a speaker. In this work, we devise a CNN based feature level fusion algorithm that combines and projects the voice characteristics in the MFCC and LPC feature spaces into a d-dimensional joint feature space. Here, the value of d depends on the CNN architecture. The joint feature space is learnt in such a way (as explained in later sections) that the joint feature representation captures highly discriminative speaker dependent voice characteristics for improving speaker recognition performance.

IV. Proposed 1D-Triplet-CNN for performing Speaker Recognition

We aim to do speaker verification from a fusion of LPC and MFCC features and therefore propose our own *1D-Triplet-CNN* architecture, given in Figure 3, for the task. In our network design we create three clones of a 1D-CNN model, forming the proposed 1D-Triplet-CNN. The weight matrices of the three clones share the same memory space and are called 'shared weights'. During training, an update made to any of the three 1D-CNN clones updates the 'shared weights' and is, therefore, reflected across the entire triplet of CNNs. For training the network, we provide a data triplet D_t as input to the CNN, given by $D_t = (S_a, S_p, S_n)$. Here, S_a and S_p, called *anchor* and *positive* samples respectively, are two different speech samples taken from a subject 'A', whereas, S_n, the negative sample, is a speech sample from another subject 'B', such that $A \neq B$.The task of the loss function in the training phase, described in section IV-5, is to help the network learn the similarity between the anchor sample and the positive sample and the dissimilarity between the anchor sample and the negative sample.

In the testing phase, as shown in Figure 3, we arrange the trained CNN into a Siamese network instead of a triplet network. Unlike the training phase, here we only need two copies of the trained CNN for matching a data pair $D_p = (S_1, S_2)$. Here,S_1 and S_2 are audio samples from two different recordings. The two copies of the trained CNN are then used to extract embeddings for S_1 and S_2, individually. Cosine similarity metric is used to compare the extracted embeddings and provide a corresponding match score. In an ideal case, embedding of a sample pair belonging to same subject should give a match score close to 1, while embeddings of a sample pair belonging to two different subjects should result in a match score close to -1.

In the following sections we will discuss our network design choices and the thought process behind it.

1) Speech Parametrization and Data Organization: Similar to [13], we split the input audio clips into smaller patches, called *audio frames*, using a sliding window of length $\lfloor 0.02 * fs \rfloor$ and stride $0.5 * \lfloor 0.02 * fs \rfloor$. Here, fs is the sampling frequency of the input audio. Same window sizes and strides were used for both LPC and MFCC extraction process. We use voice activity detection (VAD), prior to feature extraction, to remove unvoiced portions from the input audio. VOICEBOX [11] toolbox is used for extracting 40 dimensional LPC and MFCC *feature frames* from the audio frames.

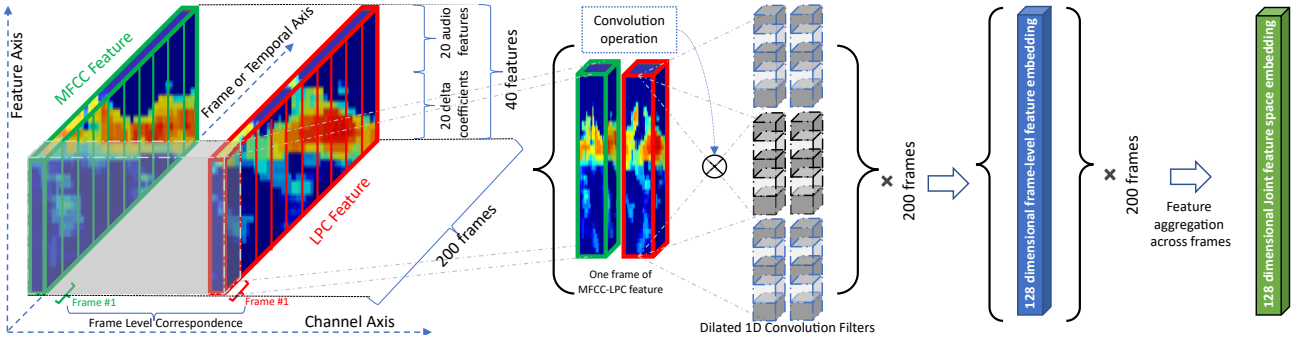


Figure 2. A visual representation of feature fusion in the proposed 1D-Triplet-CNN architecture

Each LPC feature frame comprises of 20 LPC coefficients concatenated with 20 first order delta coefficients. Similarly, MFCC feature frame comprises of 20 mel-cepstral coefficients(including zeroth order coefficient) and 20 first order delta coefficients. The extracted features were also normalized using cepstral mean variance normalization (CMVN). In our experiments, we are able to achieve better generalizability by using CMVN and hence is an important part of the algorithm. The number of audio frames that can be extracted from an audio depends on its sampling frequency and duration. Therefore, for training the CNN on input of fixed dimensionality, we randomly sample 200 contiguous feature frames, called *feature patch*, from every audio in every batch. Hence, each feature patch is of size 40×200. Doing this, we also achieve a form of data augmentation similar to 'random cropping' operation used in image based CNNs.

2) Feature Level Fusion of LPC and MFCC features: We stack the MFCC and LPC feature patches along the third dimension to create a, $40 \times 200 \times 2$ dimensional, two-channel feature patch referred to as *MFCC-LPC*. Here, the first channel corresponds to the MFCC feature patch and the second to the LPC patch. The two-channels of the MFCC-LPC feature are then combined in the proposed CNN architecture, as illustrated in Figure 2, using dilated 1D convolution filters. The proposed CNN architecture is designed to transform the 2-channel, 40-dimensional representation of each MFCC-LPC feature frame into a 128-channel, 1-dimensional frame-level feature embedding. Here, the 128 output channels represent a vector in the, hence learnt, 128-dimensional *Joint Feature Space*. The frame-level embeddings are aggregated across the 200 input frames, using average pooling, to output a 128 dimensional joint-feature space representing the speaker dependent information present in the input MFCC-LPC features.

3) Dilated 1D Convolutions: The design of convolutional layers in a CNN plays a fundamental role in determining its learning capability and efficiency. Each convolutional layer along the depth of a CNN learns different "concepts" from the data and transforms the data for layers further deeper in the CNN. As also mentioned by authors in [13], unlike images, speech data does not exhibit some of key properties that a CNN leverages for learning from image data. For example, pixels in images bear spatial relationships in a local neighborhood, in form of a semantic structure, which can be learnt by using 2-D convolutional filters in the CNN .

Speech data on other hand, when represented in form of two dimensional feature patches (e.g MFCC and LPC), does not exhibit similar semantic structures in a 2-D local neighborhood. The main reason behind this, as pointed in [13], is that the pixel values along Y axis correspond to the feature values (MFCC/LPC), which in case of MFCC features are placed on a logarithmic scale, while the pixel values along X axis vary with time on a linear scale. Therefore any semantic relationships that might exist should be constrained along 1D neighborhoods in the X and Y axes individually.

Another important point, to be noted, is that even though an audio signal is constantly changing, the speaker dependent voice characteristics are assumed to be stable only within short time scales (25ms). Therefore, we work on short term audio segments, called *audio frames*, as explained in the feature extraction process in section IV-1. The MFCC or LPC feature corresponding to that audio frame is called a *feature frame*. Thus, a feature frame extracted from an audio frame represents the audio properties of only that particular audio frame and does not bear any relationship with its neighboring frames in the context of speaker recognition. Putting these domain specific constraints together we decide to use 1D convolutional filters along the feature dimension (Y axis), as introduced in [13], in our CNN for learning speaker dependent speech characteristics. We have further used *dilated* 1D convolutional layers instead of conventional 1D convolutional layers with 1D pooling as done in [13]. We have chosen to replace pooling operation with dilated convolutions, firstly, as an effort to minimize the data loss within the network due to pooling layers. Secondly, dilated convolutions also help in increasing the receptive field rapidly without greatly increasing the computational cost, as also done in [36]. This is because dilated convolutions, also known as convolution with holes, learn sparse filters. For example, a 1D-convolution filter of kernel size 5×1 with a unit dilation factor actually learns a sparse filter of size 9×1 with alternating indices populated. Such a filter, therefore, spans a larger receptive field than a traditional 1D-convolution filter of size 5×1, while using same number of parameters. Dilated 1D-convolutions can, therefore, replace pooling layers for increasing the receptive field in the network without sustaining any data loss introduced by the latter.

In our work, dilated convolutions are specifically beneficial over conventional convolutions followed by pooling for another reason: it allows the network to learn sparse relationships between the feature values within a feature frame. Learning

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2941773, IEEE Transactions on Information Forensics and Security

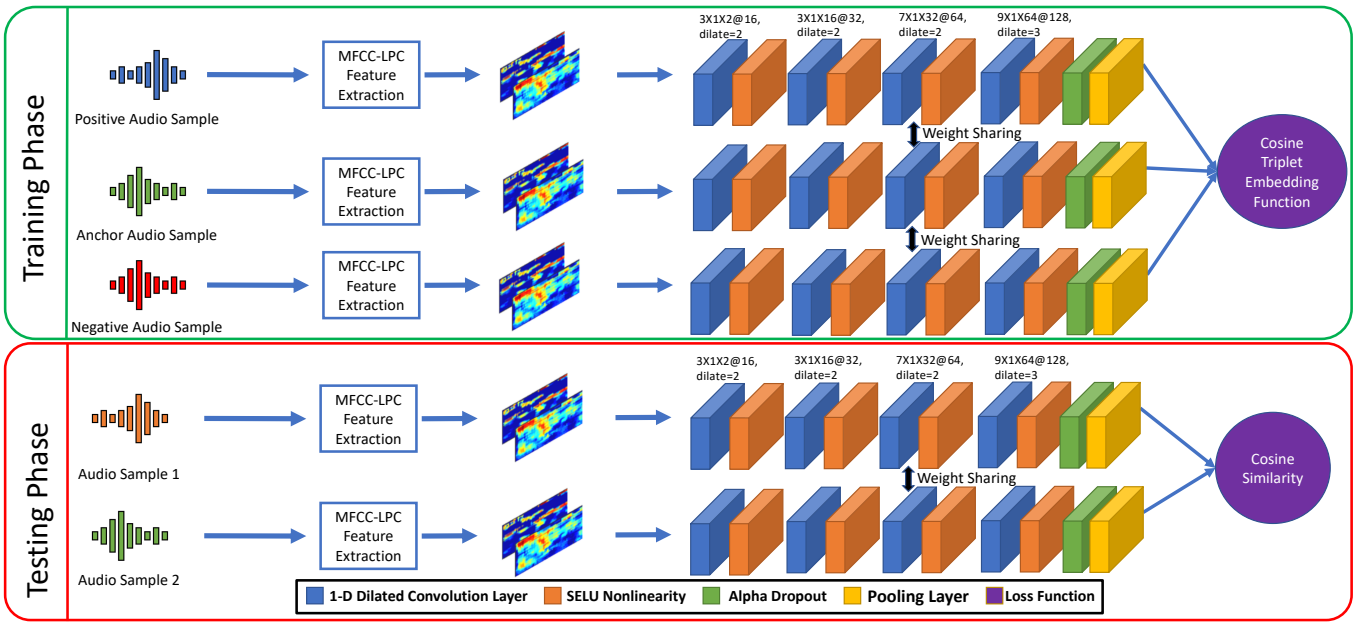


Figure 3. A visual representation of the proposed 1D-Triplet-CNN for performing speaker verification from degraded audio samples

such sparse relationships is particularly useful for learning speaker dependent features in degraded audios. Our intuition behind this is that in case of degraded audios containing structured background noise, the frequency bands closer to that of the noise are degraded uniformly in dense local regions. Hence, learning sparse features using *dilated* 1D convolutions prevents the network to learn from local dense regions and is, therefore, more robust to such audio degradations.

4) SELU Non-Linearity and Alpha Dropout: Similar to spectral subtraction, the authors in [13] subtracted data-mean from the training and validation datasets for zero centering their data. In our experiments, we found this to generalize poorly across datasets degraded with unknown types of noise profiles. This is primarily due to the fact that spectral subtraction based data normalization methods only work for *known* types of noise profiles. Therefore, normalizing the data with respect to a single type of noise, present in the training set, does not generalize across the validation and testing sets degraded with unknown noise profiles. We solve this problem by replacing spectral substraction with input-normalization for every activation layer in our proposed CNN. This ensures a uniform normalization of the data being processed in the CNN at every layer, independent of the type of noise added to the input data. For this purpose, we compared the performance of 'Batch Normalization with ReLU activation' to SELU [29] activation, a recently proposed alternative to the former approach, and found significant performance benefits over the former. We have also used the 'alpha dropout layer', as suggested in [29], for maintaining the self-normalizing property of the SELU activations. Therefore, SELU activation layer coupled with alpha dropout helps in improving the generalizablity, as seen in section VI, of our proposed CNN architecture across different types of audio degradations.

5) Cosine Triplet Embedding Loss: The main aim of triplet based CNNs, as introduced in [46], is to learn an embedding $f(x) \in \Re^d$. Where x is a data sample and $f(x)$ is its embedding in a d-dimensional euclidean space. The embedding is so learnt such that data samples belonging to same class are embedded closer to each other in the d-dimensional space while embedding of samples from different classes are pushed farther apart. Similar to the work in [21], we use cosine similarity metric for learning the embeddings as it provides for better learning dynamics over euclidean metric in our case.

The cosine triplet embedding loss used for training the 1D-Triplet-CNN model is given by:

$$L(S_a, S_p, S_n) = \sum_{a,p,n}^{N} \cos(f(S_a, S_n)) - \cos(f(S_a, S_p)) + \alpha_{margin} \quad (6)$$

Here, $L()$ is the cosine triplet embedding loss function. S_a, the anchor sample, and S_p, the positive sample, are speech samples from a subject 'A'. S_n, the negative sample, is a speech sample from another subject 'B', such that $A \neq B$. α_{margin} is the margin of minimum distance between positive and negative samples and is a user tunable hyper-parameter.

V. Datasets and Experiments

A. Experiments

Throughout the paper we perform speaker verification experiments on a variety of datasets and protocols. For evaluating our proposed algorithm we use:

1) TIMIT [19] Acoustic-Phonetic Continuous Speech Corpus
2) Fisher English Training Speech Part 1 Speech [14] dataset
3) NIST SRE 2008 [1] dataset
4) NIST SRE 2010 [2] dataset

We further use noise data, as detailed in section V-B, from NOISEX-92 [48] dataset under varying levels (0 to 20 dB) of Signal to Noise Ratio (SNR) and reverberations to degrade the speech data in above listed datasets. This is done to demonstrate the performance of our algorithm under degraded audio conditions.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2941773, IEEE Transactions on Information Forensics and Security

Table I
VERIFICATION RESULTS ON THE DEGRADED TIMIT SPEECH DATASET.

Exp. #	Training set / Testing set	MFCC / LPC / MFCC-LPC							
		TMR@FMR=10%				minDCF($P_{tar} = 0.01$)			
		1D-Triplet-CNN	UBM-GMM	iVector-PLDA	xVector-PLDA	1D-Triplet-CNN	UBM-GMM	iVector-PLDA	xVector-PLDA
1	S1 / S2	61 / 34 / **75**	27 / 18 / 7	32 / 20 / 18	63 / 22 / 56	8.55 / 10 / **7.95**	9.57 / 9.46 / 9.94	8.63 / 10 / 9.16	**7.95** / 9.52 / 9.27
2	S2 / S1	67 / 42 / **79**	18 / 23 / 14	25 / 13 / 22	**79** / 58 / 60	6.54 / 9.7 / 7.48	9.82 / 9.52 / 9.82	9.34 / 9.82 / 9.4	**6.52** / 8.72 / 8.69
3	S3 / S4	65 / 27 / **76**	35 / 17 / 19	53 / 17 / 32	60 / 37 / 69	8.92 / 9.75 / 8.33	9.04 / 9.64 / 9.94	8.21 / 9.52 / 9.58	**7.19** / 9.4 / 8.14
4	S4 / S3	61 / 19 / 67	26 / 26 / 22	29 / 12 / 19	50 / 35 / **73**	8.38 / 9.64 / 7.18	9.64 / 9.16 / 9.16	8.92 / 10 / 9.88	8.86 / 9.93 / **7.07**
5	S5 / S6	56 / 24 / **71**	22 / 23 / 10	30 / 15 / 17	40 / 42 / 53	9.1 / 9.64 / 8.8	9.94 / 9.88 / 9.94	9.69 / 10 / 9.94	8.98 / 9.94 / **8.37**
6	S6 / S5	66 / 43 / **80**	22 / 23 / 17	36 / 27 / 29	73 / 50 / 68	7.77 / 10 / 7.6	9.04 / 9.58 / 9.46	9.28 / 9.87 / 9.04	**7.24** / 9.28 / 9.22

Exp. #	Training set / Testing set	MFCC / LPC / MFCC-LPC			
		Equal Error Rate (EER, in %)			
		1D-Triplet-CNN	UBM-GMM	iVector-PLDA	xVector-PLDA
1	S1 / S2	17 / 33 / **16**	41 / 42 / 50	38 / 42 / 49	17 / 45 / 17
2	S2 / S1	16 / 25 / **13**	45 / 44 / 48	36 / 43 / 36	14 / 21 / 18
3	S3 / S4	20 / 29 / **17**	32 / 48 / 41	23 / 47 / 32	21 / 22 / 19
4	S4 / S3	19 / 48 / **17**	47 / 44 / 50	44 / 46 / 47	23 / 29 / **17**
5	S5 / S6	23 / 37 / **14**	45 / 45 / 52	40 / 42 / 47	29 / 25 / 17
6	S6 / S5	18 / 26 / **14**	55/ 38 / 49	51/ 42 / 37	15 / 26 / 17

Data Subset	S1	S2	S3	S4	S5	S6
Noise Characteristics	Babble, F16, R1,V1	Car, Factory, R2, V2	Babble, Car, R2, V2	F16, Factory, R1, V1	Car, F16, R1, V1	Babble, Factory, R2, V2

Table II
VERIFICATION RESULTS ON THE DEGRADED FISHER SPEECH DATASET.

Exp. #	Training set / Testing set	MFCC / LPC / MFCC-LPC							
		TMR@FMR=10%				minDCF($P_{tar} = 0.01$)			
		1D-Triplet-CNN	UBM-GMM	iVector-PLDA	xVector-PLDA	1D-Triplet-CNN	UBM-GMM	iVector-PLDA	xVector-PLDA
7	F1 / F1	74 / 73 / **85**	54 / 42 / 55	67 / 18 / 70	57 / 63 / 72	7.69 / 7.19 / **5.67**	9.95 / 9.86 / 9.95	8.33 / 9.97 / 7.8	8.53 / 8.8 / 8
8	F1 / F2	55 / 46 / **74**	39 / 40 / 43	46 / 18 / 53	25 / 42 / 54	9.31 / 9.82 / **7.4**	9.92 / 9.81 / 9.9	9.55 / 9.99 / 9.59	9.94 / 9.7 / 9.37
9	F2 / F2	77 / 76 / **84**	56 / 42 / 57	68 / 20 / 72	56 / 56 / 73	6.96 / 7.37 / **5.53**	9.52 / 9.45 / 9.48	8.19 / 9.96 / 8.07	9.25 / 8.96 / 7.62
10	F2 / F1	39 / 36 / **62**	29 / 42 / 32	38 / 22 / 41	29 / 28 / 44	9.84 / 9.88 / **8.66**	9.95 / 9.89 / 9.95	9.54 / 9.96 / 9.59	9.89 / 9.91 / 9.74

Exp. #	Training set / Testing set	MFCC / LPC / MFCC-LPC			
		Equal Error Rate (EER, in %)			
		1D-Triplet-CNN	UBM-GMM	iVector-PLDA	xVector-PLDA
7	F1 / F1	16 / 17 / **12**	24 / 27 / 23	18 / 43 / 18	22 / 20 / 17
8	F1 / F2	23 / 25 / **17**	29 / 31 / 26	26 / 44 / 24	37 / 30 / 23
9	F2 / F2	16 / 16 / **13**	25 / 31 / 24	19 / 41 / 17	24 / 23 / 16
10	F2 / F1	30 / 31 / **22**	29 / 28 / 27	31 / 41 / 29	37 / 31 / 27

Data Subset	F1	F2
Noise Characteristics	Babble, R1,V1	F16, R1, V1

Each of the datasets and corresponding protocols used in the experiments have been designed for evaluating certain aspects of the speaker verification algorithm. The speaker verification experiments on the degraded TIMIT dataset aim at evaluating the generalizability of the algorithms under a variety of audio perturbations. While the experiments on degraded Fisher dataset aim at evaluating the performance of the algorithms in presence of a large number of speakers, hence testing the modeling capacity of the algorithms. The experiments on the NIST SRE experiments aim at comparing the performance of the algorithms on a multilingual speech dataset containing speech data from varying speech types and conditions. We also perform speaker verification experiment on speech samples of varying audio lengths, as explained in section V-D5, for studying the effect of variation in the length of test audio samples on the performance of speaker verification algorithms. One important point to note is that throughout all our experiments we work with the assumption that only one subject speaks in any given speech sample and there is no overlapping speech from multiple speakers in any audio in the training or testing sets.

B. Datasets

1) TIMIT Dataset: The TIMIT dataset provides clean speech recordings of 630 speakers. There are 462 speakers in the training set and 168 speakers in the testing set. The dataset consists of eight major dialects of American English. There are ten sessions of 3 seconds each per speaker in the dataset. The text spoken by the speakers in the training set and test set are disjoint, making the speaker recognition experiments text-independent.

In our experiments, TIMIT dataset was perturbed [9], [26] with different types (given below) of synthetic noise from the NOISEX-92 [48] noise dataset. We refer to this dataset as 'degraded TIMIT' dataset. The audio degradations were added to the TIMIT dataset in simulated room environments with different acoustic properties and reverberation levels, thereby introducing both additive and convolutive noise into the audio data. The synthetically generated noisy datasets have the following noise characteristics:

1) Noise Type: Following four types of noises were added to the TIMIT dataset:

a)F-16: Noise generated by engine of F-16 fighter aircraft.
b)Babble: Noise generated by rapid and continuous background human speech.
c)Car: Noise generated by engine of a car.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2941773, IEEE Transactions on Information Forensics and Security

Table III
VERIFICATION RESULTS ON THE ORIGINAL AND DEGRADED, NIST SRE 2008 AND 2010 DATASETS.

Exp. #	Training set / Testing set	MFCC / LPC / MFCC-LPC							
		TMR@FMR=10%				minDCF($P_{tar} = 0.01$)			
		1D-Triplet-CNN	UBM-GMM	iVector-PLDA	xVector-PLDA	1D-Triplet-CNN	UBM-GMM	iVector-PLDA	xVector-PLDA
11	P1 / P1	89 / 86 / **93**	51 / 47 / 62	85 / 78 / 88	78 / 76 / 85	5.45 / 5.68 / **4.72**	9.14 / 9.87 / 9.59	5.68 / 7.45 / 5.84	8 / 8.18 / 7.19
12	P1 / P2	21 / 18 / **25**	15 / 11 / 11	14 / 17 / 10	19 / 15 / 17	9.95 / 9.98 / 9.96	**9.9** / 9.99 / 9.99	9.91 / 9.94 / 9.98	9.97 / 9.94 / 9.95
13	P3 / P3	84 / 80 / **89**	58 / 44 / 17	82 / 72 / 41	75 / 65 / 72	6.39 / 6.89 / **5.36**	8.83 / 9.67 / 9.99	6.36 / 7.51 / 9.58	8.35 / 8.76 / 8.25
14	P4 / P4	75 / 73 / **84**	44 / 34 / 11	60 / 28 / 22	58 / 54 / 66	7.24 / 7.77 / **6.62**	9.15 / 9.85 / 9.99	8.5 / 9.84 / 9.97	9 / 9.16 / 8.58
15	P3 / P4	49 / 47 / **56**	43 / 34 / 15	28 / 20 / 15	31 / 35 / 52	9.4 / 9.35 / **9**	9.7 / 9.87 / 10	9.93 / 9.94 / 9.98	9.72 / 9.92 / 9.28
16	P4 / P3	37 / 27 / **56**	31 / 28 / 14	51 / 21 / 20	45 / 52 / 47	9.57 / 9.98 / **9.01**	9.99 / 9.99 / 9.99	9.23 / 9.95 / 9.99	9.55 / 9.65 / 9.5

Exp. #	Training set / Testing set	MFCC / LPC / MFCC-LPC			
		Equal Error Rate (EER), in %			
		1D-Triplet-CNN	UBM-GMM	iVector-PLDA	xVector-PLDA
11	P1 / P1	10 / 11 / **8**	29 / 26 / 23	12 / 15 / 10	14 / 15 / 11
12	P1 / P2	45 / 44 / **39**	44 / 44 / 49	45 / 44 / 47	43 / 46 / 45
13	P3 / P3	12 / 14 / **10**	24 / 28 / 40	13 / 17 / 28	15 / 19 / 16
14	P4 / P4	16 / 17 / **12**	31 / 37 / 46	22 / 36 / 37	20 / 22 / 19
15	P3 / P4	26 / 28 / 23	31 / 34 / 42	35 / 41 / 43	34 / 31 / **22**
16	P4 / P3	32 / 37 / **23**	36 / 37 / 44	24 / 40 / 41	27 / 22 / 27

Data Subset	P1	P2	P3	P4
Noise Characteristics	NIST SRE 08	NIST SRE 10	NIST SRE 08 + Babble	NIST SRE 08 + F16

Table IV
VERIFICATION RESULTS UNDER VARYING AUDIO LENGTH ON THE NIST SRE 2008 DATASET

Length of Audio (in seconds)	MFCC / LPC / MFCC-LPC							
	TMR@FMR=10%				minDCF($P_{tar} = 0.01$))			
	1D-Triplet-CNN	UBM-GMM	iVector-PLDA	xVector-PLDA	1D-Triplet-CNN	UBM-GMM	iVector-PLDA	xVector-PLDA
3.5	90 / 88 / **94**	53 / 46 / 61	78 / 75 / 86	78 / 74 / 81	4.98 / 5.25 / **4.3**	8.95 / 9.91 / 9.76	6.25 / 7.8 / 6.05	7.61 / 8.31 / 7.41
3	90 / 88 / **94**	52 / 45 / 60	77 / 71 / 84	76 / 71 / 79	5.06 / 5.39 / **4.26**	9.05 / 9.95 / 9.8	6.62 / 7.94 / 6.43	8.19 / 8.74 / 7.74
2.5	89 / 88 / **94**	50 / 43 / 58	70 / 67 / 82	69 / 66 / 75	5.25 / 7.07 / **4.15**	9.15 / 9.94 / 9.79	6.94 / 8.34 / 6.75	8.61 / 8.9 / 8.17
2	87 / 85 / **93**	48 / 43 / 55	66 / 59 / 78	61 / 57 / 66	5.17 / 5.71 / **4.51**	9.59 / 9.95 / 9.79	7.66 / 8.8 / 7.25	9.08 / 9.48 / 8.84
1.5	86 / 84 / **91**	44 / 40 / 50	58 / 48 / 68	52 / 47 / 57	6.11 / 5.89 / **4.84**	9.79 / 9.95 / 9.86	8.9 / 9.49 / 8.41	9.28 / 9.73 / 9.16
1	80 / 79 / **87**	38 / 34 / 46	40 / 33 / 54	37 / 34 / 41	6.98 / 6.95 / **5.76**	9.87 / 9.97 / 9.84	9.4 / 9.81 / 9.25	9.72 / 9.96 / 9.65
0.5	65 / 63 / **76**	27 / 26 / 32	22 / 19 / 31	19 / 20 / 20	8.44 / 8.41 / **7.3**	9.9 / 9.97 / 9.9	9.95 / 9.94 / 9.82	9.92 / 9.96 / 9.89

Length of Audio(in seconds)	MFCC / LPC / MFCC-LPC			
	Equal Error Rate (EER, in %)			
	1D-Triplet-CNN	UBM-GMM	iVector-PLDA	xVector-PLDA
3.5	9 / 10 / **7**	28 / 27 / 22	14 / 16 / 11	14 / 15 / 12
3	9 / 10 / **7**	29 / 26 / 23	16 / 17 / 12	15 / 17 / 14
2.5	10 / 11 / **7**	30 / 27 / 24	17 / 19 / 13	17 / 18 / 15
2	10 / 12 / **8**	31 / 28 / 25	20 / 22 / 15	20 / 21 / 18
1.5	11 / 12 / **9**	33 / 31 / 28	24 / 27 / 18	24 / 26 / 22
1	13 / 14 / **11**	36 / 33 / 32	30 / 33 / 24	31 / 32 / 29
0.5	20 / 20 / **15**	43 / 39 / 38	41 / 44 / 35	43 / 42 / 40

Table V
VERIFICATION RESULTS ON DEGRADED TIMIT DATASET FOR COMPARING THE PERFORMANCE OF 1D-DILATED CNN ARCHITECTURE WITH ALTERNATE 1D CNN AND 2D CNN ARCHITECTURES.

Method	Performance on Testing Set		Performance on Training Set		Number of Model Parameters
	TMR@FMR=10%	minDCF($P_{tar} = 0.01$)	TMR@FMR=10%	minDCF($P_{tar} = 0.01$)	
1D-CNN (with dilation, along feature dimension)	**75.79**	**8.97**	**98.57**	**3.95**	89,696
1D-CNN (with dilation, along time dimension)	19.04	9.64	40.17	9.89	89,696
1D-CNN (with pooling, along feature dimension)	61.3	9.33	88.54	6.95	89,696
2D-CNN (with pooling)	47.02	9.52	93.8	6.17	768,800

d)Factory: Noise generated by heavy machinery operating in a factory environment.

2) Signal to Noise Ratio (SNR): The resultant noisy datasets were each generated at three different SNR levels, viz., 20 dB, 10dB and 0dB.

3) Room Size: The noisy dataset were generated in a simulated room environment with two different room sizes (4m and 20m, side length of cube), referred to as R1 and R2 in the protocol.

4) Reverberation: Two different reverberation coefficients were used to introduce additional noise in the data, referred to as V1 and V2 in the protocol.

2) Fisher English Training Speech Part 1 dataset: The Fisher English Training Speech Part 1 Speech dataset contains conversational speech data collected over telephone channels between pairs of over $12,000$ speakers. Conversations pertaining to a subset of $6,991$ speakers from the dataset were used in this work. A random subset of 4500 speakers (out of $6,991$ total speakers) was chosen for the training set and remaining speakers were reserved for the testing set. Since the speech audios in Fisher dataset contains speech from multi-speaker conversations, speech audio pertaining to each speaker in the conversation is segmented out and processed with voice activity detection to remove empty audio segments. The audio of each speaker was then split into smaller audio snippets of 5-second duration each. We extract 50 audio snippets for each speaker. We have also perturbed the Fisher dataset with the F-16 and Babble noise from the NOISEX-92 [48] noise dataset,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2941773, IEEE Transactions on Information Forensics and Security

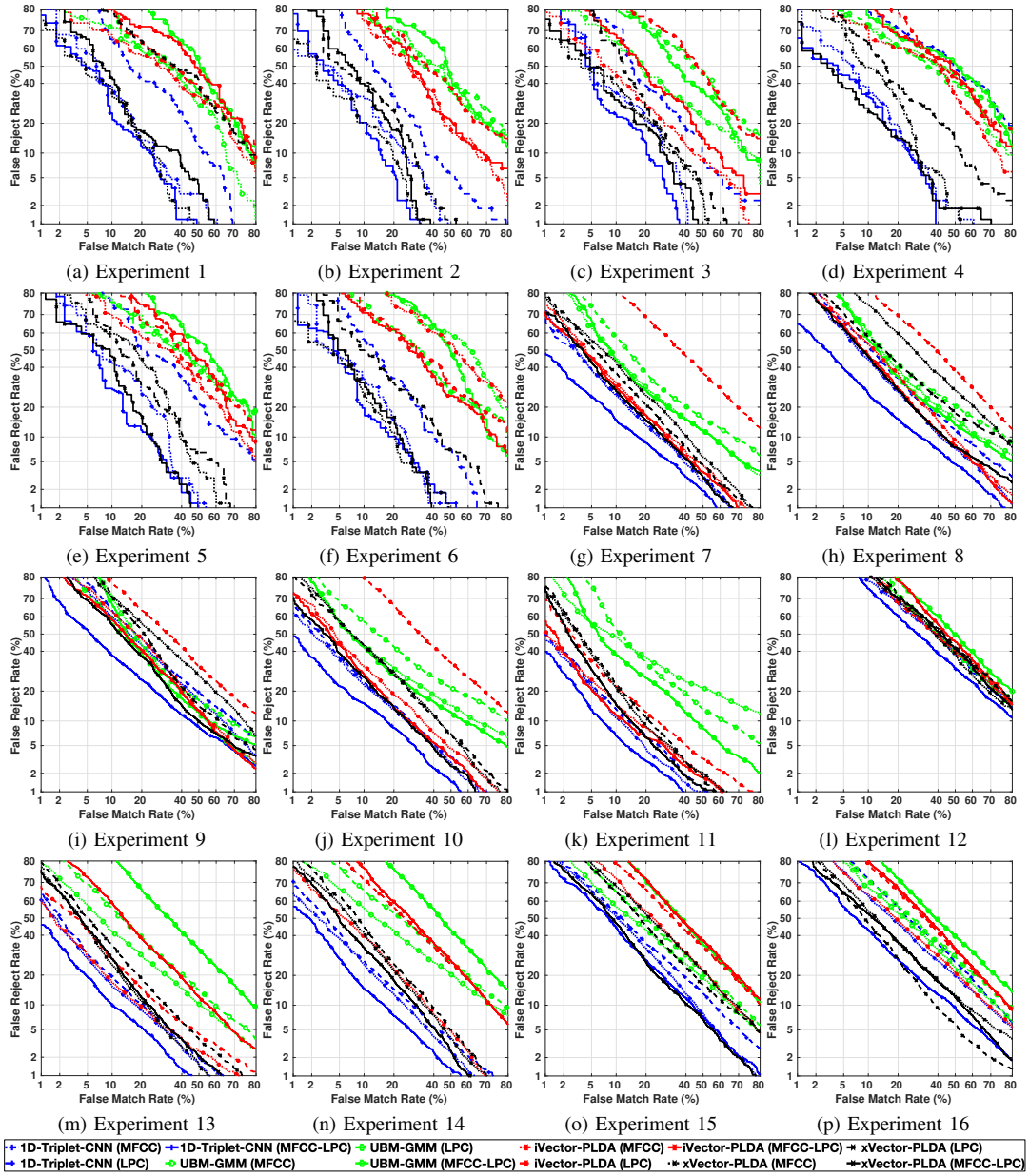


Figure 4. DET curves for the speaker verification experiments on the degraded TIMIT dataset (Exp. 1 to 6), degraded Fisher dataset (Exp. 7 to 10) and, the clean and degraded NIST SRE 2008 and 2010 datasets (Exp. 11 to 16) using UBM-GMM, iVector-PLDA, xVector-PLDA and 1D-Triplet-CNN algorithms on MFCC, LPC and MFCC-LPC feature sets.

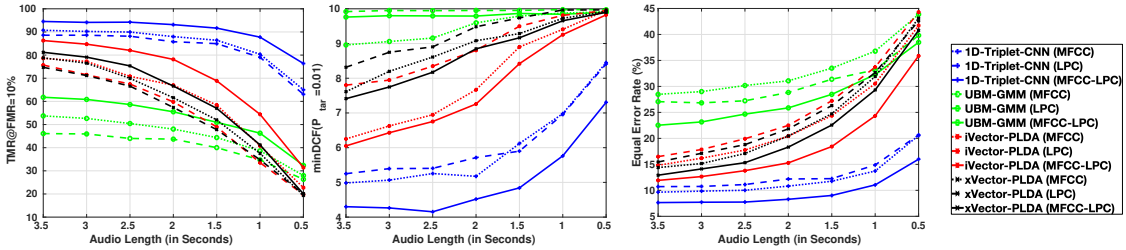
at a resultant SNR of 10dB. The noisy datasets were generated in a simulated room environment of fixed size (4m, side length of cube), referred to as R1 in the protocol. Fixed amount of reverberation was also used to introduce additional noise in the data, referred to as V1 in the protocol.

3) NIST SRE 2008 and 2010 datasets: National Institute of Standards and Technology (NIST) periodically conducts the NIST Speaker Recognition Evaluation (SRE) challenges to evaluate performance of speaker recognition algorithms under various audio characteristics. In our work, we use the NIST SRE 2008 dataset to train our models. For evaluation we use both the NIST SRE 2008 and 2010 datasets. For our experiments on the NIST SRE 2008 [1] dataset, we use multilingual speech data from 'phonecall' and 'interview' speech types, collected across varied audio conditions labeled as '10-sec', 'long' and 'short2'. We choose a random subset of 1136 speakers for training our algorithms and rest 200 speakers are reserved for evaluation purposes. For cross-dataset speaker verification performance evaluation, we use speech data from all the speakers in the NIST SRE 2010 [2] dataset. We have also perturbed the NIST SRE 2008 dataset with synthetic noise from the NOISEX-92 [48] noise dataset. We added F-16 and Babble noise to the NIST SRE 2008 dataset. The Signal to Noise ratio of the resultant *degraded* NIST SRE 2008 dataset is maintained at 0dB.

C. Features

All experiments using the proposed 1D-Triplet-CNN algorithm and the three baselines of UBM-GMM, iVector-PLDA

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2941773, IEEE Transactions on Information Forensics and Security



(a) TMR@FMR=10% (b) minDCF($P_{tar} = 0.01$) (c) EER

Figure 5. (a) TMR@FMR=10%, (b) minDCF($P_{tar} = 0.01$) and (c) EER under varying audio length on the clean NIST SRE 2008 dataset. 1D-Triplet-CNN(MFCC-LPC) performs the best across varying lengths of test audio.

and xVector-PLDA, as detailed in Section V-D, are done using the MFCC and LPC feature sets individually. The same experiments have then been repeated for all the algorithms using the fusion of MFCC and LPC feature sets, referred to as MFCC-LPC. This was done to better understand the ability of the different algorithms at combining information from two seemingly complementary feature sets and leveraging it for performing speaker recognition. For the 1D-Triplet-CNN algorithm, as also discussed in Sections IV-1 and IV-2, the MFCC and LPC features were fused together into a two channel feature matrix yielding an input feature dimensionality of $40 \times 200 \times 2$. However, for the UBM-GMM, iVector-PLDA and xVector-PLDA algorithms, the MFCC and LPC features (in that order) were concatenated end-to-end at frame level, yielding an input feature dimensionality of 80×200. This was done because the VOICEBOX toolkit's implementation of UBM-GMM and iVector-PLDA algorithms does not support multi-channel feature input.

D. Experimental Protocols

In all the experiments, across all the datasets, we ensure disjoint speakers in training and testing sets. The split of noise characteristics, however, in the training and testing sets are experimented with both disjoint noise and same noise scenarios, as given in Tables I, II, III. For example, in experiment 1, the training set consists of audio samples that are simulated to be recorded in a room of size R1 and reverberation coefficient V1, with additive background noise of type "Babble" and "F16".

1) UBM-GMM [40] based Speaker Verification Experiments: To obtain baseline performance on the experiments laid out in Tables I, II, III and IV, we train a Universal Background Model (UBM) [40] using data from the speakers in the training set. We evaluate the trained model on verification audio pairs from speakers in the testing set. For evaluation, we adapt the trained UBM to each of the audio samples in a verification pair to obtain two separate speaker-adapted GMM models. Which are then scored against each other to render a match score.

2) iVector-PLDA [20] based Speaker Verification Experiments: To obtain a second baseline performance on the experiments laid out in Tables I, II, III and IV, we perform iVector-PLDA based speaker recognition experiments using the implementation in the MSR identity toolkit [45]. Similar to the protocol for the UBM-GMM experiment, we first train an UBM on audio data from speakers in the training set. The trained UBM is then used to learn a total variability subspace of 400 dimensions, from background statistics. Development i-vectors are then extracted from speech features using the trained total variability subspace and UBM. Finally a Gaussian PLDA model with development i-vectors is learnt. For evaluation, i-vectors are generated for both the audio samples in a verification pair and then they are compared using the trained PLDA model to render a match score.

3) xVector-PLDA [47] based Speaker Verification Experiments: To obtain a neural network based baseline performance for the experiments reported in Tables I, II, III and IV, we use xVector-PLDA. Since the xVector implementation in Kaldi [38] toolkit only supports 24-dimensional MFCC features, we re-implemented the xVector algorithm in PyTorch for enabling support for 40-dimensional MFCC and LPC features and the 80-dimensional MFCC-LPC features. The PyTorch implementation of xVector algorithm was used together with the gaussian PLDA implementation given in the MSR identity toolkit [45] for performing the xVector-PLDA based speaker recognition experiments.

4) 1D-Triplet-CNN based Speaker Verification Experiments: The experiments, given in Tables I, II, III and IV, were conducted using the proposed 1D-Triplet-CNN based Speaker Verification algorithm. For training the 1D-Triplet-CNN, we generate data triplets using audio data from speakers in the training set. For evaluation we generate genuine-impostor verification pairs from speakers in the testing set and match them as explained in Section IV.

5) Speaker verification experiments on audio samples of varying length: We also perform speaker verification experiments using the UBM-GMM, iVector-PLDA, xVector-PLDA and 1D-Triplet-CNN algorithms on speech data of varying lengths from the NIST SRE 2008 dataset. This experiment is aimed at evaluating the effect of variation in length of test audio on the performance of speaker verification algorithms. In practical scenarios, the probe audio sample is often of limited length in which the amount of usable speech audio is further reduced greatly by audio perturbations. Therefore it is important for speaker verification algorithms to be robust across different lengths of audio samples.

We compare the True Match Rate at a False Match Rate of 10% (TMR@FMR=10%), minimum Detection Cost Function at a priori probability of the specified target speaker, P_{tar}, of 0.01 (minDCF($P_{tar} = 0.01$)) and Equal Error Rate (EER, in %) for both baseline and proposed algorithms on audio samples of varying number of frames. We vary the audio length from 3.5 to 0.5 seconds in steps of 0.5 second.

6) Speaker Verification Experiments for comparing the performance benefits of dilated 1D convolutions over traditional

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2941773, IEEE Transactions on Information Forensics and Security

1D and 2D CNN architectures: We also perform additional speaker recognition experiments, as given in Table V for experimentally validating the design choice of using dilated 1D convolutions along the feature dimension. We use data from degraded TIMIT dataset for these experiments. Audio data degraded with Babble and F16 noise is used to train the models, while data in the testing set is perturbed with Car and Factory noise. All the CNN designs are kept mutually identical in all other aspects (e.g. number of layers, number of input and output channels etc.), except the shape of the convolution filters used and the usage of dilation versus pooling operation. Following CNN designs are explored in these set of experiments:

- 1D-CNN (with dilation, along feature dimension): This is the design introduced in our proposed method where dilated 1D convolution filters are learnt along the feature dimension for extracting speaker dependent information at frame-level.
- 1D-CNN (with dilation, along time dimension): This design uses 1D dilated convolution filters learnt along the time dimension for extracting speaker dependent information at multiple temporal scales.
- 1D-CNN (with pooling, along feature dimension): This design replaces the use of dilated 1D convolution layers as done in proposed 1D-Triplet-CNN architecture with regular 1D convolution layers paired with average pooling layers.
- 2D-CNN (with pooling): This design replaces the use of dilated 1D convolution layers as done in proposed 1D-Triplet-CNN architecture with regular 2D convolution layers paired with average pooling layers.

We report performance of the different CNN designs on both the training and testing sets along with their number of learnable model parameters.

7) Score-level fusion experiments for combining speaker recognition models trained on MFCC and LPC features separately: We also performed score-level fusion of the speaker recognition models trained individually on MFCC and LPC features, using the sum and product fusion rules. We performed these experiments on the six subsets (S1 - S6) of the degraded TIMIT dataset, defined in Table I, to compare the performance benefits of the score-level and feature-level fusion approaches. For all the baseline and proposed algorithms, both sum and product rule based score-level fusion approaches outperformed the models trained individually on the LPC features. However, they failed to outperform the models trained on the MFCC features alone. When compared to the performance of feature-level fusion approach, the score-level fusion approaches lag behind by $\sim 22\%$ for the 1D-Triplet-CNN algorithm and by $\sim 2\%$ for the xVector-PLDA algorithm. However, for the UBM-GMM and iVector-PLDA algorithms the score-level fusion approaches outperform the feature-level fusion approach by $\sim 10\%$ and $\sim 2\%$ respectively. Since, the score level fusion strategies did not appear to benefit the overall speaker verification performance for the 1D-Triplet-CNN and xVector-PLDA algorithms, therefore they have been excluded from the scope of this work.

VI. Results and Analysis

The results of the verification experiments are presented in Tables I, II, III and IV. We have chosen to report True Match Rate at False Match Rate of 10% (TMR@FMR=10%), minimum Detection Cost Function at a priori probability of the specified target speaker, P_{tar}, of 0.01 (minDCF($P_{tar} = 0.01$)) and Equal Error Rate (EER, in $\%$) as our performance metric for comparison of the baseline methods and the proposed method. The Detection Error Tradeoff (DET) curves are given in Figures 4 and 5. Additionally, we also determined and reported the subsets of test data pairs that were correctly matched using the proposed and the baseline algorithms at False Match Rate of 10%. This was used to determine the proportion of the test data pairs where the proposed algorithm performed better or worse than the baseline algorithms.

- The proposed algorithm vastly outperforms the baseline algorithms, in majority of the experiments given in Tables I, II, III and IV, when trained/tested on MFCC and LPC features separately and also when fused together. Also, it is interesting to note that, unlike the baseline algorithms, the proposed algorithm successfully fuses the MFCC and LPC features to gain consistent performance benefits over the individual features, across all the experiments. The main reason for the performance improvement can be attributed to the design of the 1D-Triplet-CNN architecture, which: (a) successfully extracts speaker dependent features from MFCC and LPC features drawn from degraded audio signals and (b) successfully combines the extracted speaker dependent features to learn a highly discriminative joint-embedding for improving speaker recognition performance.
- On average, across the six experiments in Table I, UBM-GMM, iVector-PLDA, xVector-PLDA, and 1D-Triplet-CNN correctly verifies the same 34.97% of the test samples. 1D-Triplet-CNN correctly verifies an additional 14.08% of the test samples over xVector-PLDA, 28.72% over iVector-PLDA, and 33.48% over UBM-GMM based algorithms. However, the 1D-Triplet-CNN based algorithm fails to correctly verify 4.4% of the test samples that were correctly verified by all the baseline algorithms.
- On average, across the six experiments in Table I, the best baseline performance (TMR at FMR=10%) is achieved by xVector-PLDA(MFCC-LPC) algorithm. The proposed, 1D-Triplet-CNN(MFCC-LPC), algorithm further improves upon the best average baseline performance, by 12%. It also improved the average EER from 18% to 15% and minDCF($P_{tar} = 0.01$) from 8.46 to 7.89.
- On average, across the four experiments in Table II, UBM-GMM, iVector-PLDA, xVector-PLDA, and 1D-Triplet-CNN correctly verified the same 47.5% of the test samples. 1D-Triplet-CNN correctly verifies an additional 17.18% of the test samples over xVector-PLDA, 13.55% over iVector-PLDA, and 16.06% over UBM-GMM based algorithms. However, the 1D-Triplet-CNN based algorithm fails to correctly verify 4.3% of the test samples that were correctly verified by all the baseline algorithms.
- On average, across the four experiments in Table II, the best baseline performance (TMR at FMR=10%) is achieved

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2019.2941773, IEEE Transactions on Information Forensics and Security

by xVector-PLDA(MFCC-LPC) algorithm. The proposed 1D-Triplet-CNN(MFCC-LPC) algorithm further improves upon the best average baseline performance, by almost 16%. It also improved the average EER from 20% to 16% and minDCF($P_{tar} = 0.01$) from 8.68 to 6.81.

- On average, across the six experiments in Table III, UBM-GMM, iVector-PLDA, xVector-PLDA, and 1D-Triplet-CNN correctly verified the same 36.35% of the test samples. 1D-Triplet-CNN correctly verifies an additional 17.76% of the test samples over xVector-PLDA, 20.62% over iVector-PLDA, and 25.33% over UBM-GMM based algorithms. However, the 1D-Triplet-CNN based algorithm fails to correctly verify 6.73% of the test samples that were correctly verified by all the baseline algorithms.
- On average, across the six experiments in Table III, the best baseline performance (TMR at FMR=10%) is achieved by xVector-PLDA(MFCC) algorithm. The proposed, 1D-Triplet-CNN(MFCC-LPC), algorithm further improves upon the best average baseline performance, by almost 11%. It also improved the average EER from 23% to 19% and minDCF($P_{tar} = 0.01$) from 8.79 to 7.44.
- In the experimental results given in Table IV and illustrated in Figure 5, we notice a decreasing trend in verification performance, i.e. decrease in TMR at FMR=10% and increase in minDCF($P_{tar} = 0.01$) and EER, with decrease in length of audio samples in the testing data, across all the algorithms. However, it is interesting to note the vastly different rates of decrease in performance across all the algorithms. The iVector-PLDA and xVector-PLDA baseline algorithms exhibit a comparatively sharper decrease in performance when compared to others. On average, the iVector-PLDA and xVector-PLDA algorithms, on MFCC-LPC feature set, lose 55% and 60% performance (TMR@FMR=10%) respectively, when the audio length decreases from 3.5 to 0.5 seconds. The UBM-GMM and 1D-Triplet-CNN, however, lose about 30% and 18% performance (TMR@FMR=10%) respectively in the same experimental setting.
- The aggregate TMR@FMR=10% for 1D-Triplet-CNN, on MFCC-LPC feature set given in Table IV, is vastly superior at $90 \pm 6\%$ as compared to $52 \pm 10\%$ for UBM-GMM, $69 \pm 20\%$ for iVector-PLDA, and $60 \pm 22\%$ for xVector-PLDA. The 1D-Triplet-CNN, on MFCC-LPC feature set, maintains an aggregate minDCF($P_{tar} = 0.01$) of $5.01 \pm 1.14\%$ as compared to $9.82 \pm 0.04\%$ for UBM-GMM, $7.70 \pm 1.46\%$ for iVector-PLDA, and $8.69 \pm 0.95\%$ for xVector-PLDA. The 1D-Triplet-CNN, on MFCC-LPC feature set, also maintains an EER of $9.62 \pm 3.04\%$ as compared to $27.91 \pm 5.73\%$ for UBM-GMM, $18.89 \pm 8.61\%$ for iVector-PLDA, and $21.90 \pm 10.08\%$ for xVector-PLDA.
- Thus, we can establish that the performance of the 1D-Triplet-CNN is relatively robust to the variation in length of audio samples in the testing data. This can be attributed to the architecture of 1D-Triplet-CNN that performs dilated 1D convolutions only along individual frames and is independent of the length of context. However, the iVector-PLDA and xVector-PLDA algorithms, use statistic pooling across the frames in an audio sample for characterizing it. Reliability of such statistic pooling operations are heavily dependent on the number of available audio frames. Thus, reducing the length of audio has a greater detrimental effect on iVector-PLDA and xVector-PLDA algorithms.
- Across the four experiments, given in Table V, the best performance is achieved by our proposed architecture design of using dilated 1D convolutions along the feature dimension. We compare the effect of performing dilated 1D convolutions along the feature and time axes individually and found the former to perform vastly superior. This validates the assumption of speaker-dependent voice characteristics to be stable only within individual frames of short time scales (25ms). We further compare the effect of using dilation against pooling operation with 1D convolutions (along the frames) for increasing the receptive field of 1D-convolution filters deeper in the network. As evidenced in the results, pooling operation results in inferior performance as compared to dilation operation, thereby, confirming the detrimental effect of data loss incurred by the pooling operation. This supports the design choice of using dilation over pooling operation. Finally, we also train a network with 2D convolution filters paired with average pooling operation, popularly used in image classification networks. It is interesting to note that the second-best training performance attained by this design while falling behind considerably on testing performance when compared to architectures using 1D convolution filters (along frames). This indicates signs of overfitting in case of 2D-CNN. It is also important to note that the proposed 1D-CNN architecture has only 89K trainable model parameters compared to the 768K model parameters on 2D-CNN and 4.2M parameters in the xVector network design. The 1D-Triplet-CNN model is therefore much easier to train and converge using limited data and computational resources.

VII. Implementation and Reproducibility

The 1D-Triplet-CNN model was implemented using PyTorch [37] toolkit and trained using the Adam optimizer [28] with a starting learning rate of 0.001 for 150 epochs. The α_{margin} hyper-parameter at the value of 0.25, in our cosine triplet embedding loss, was found to provide the best trade-off between time-to-convergence and generalizability. A higher value of α_{margin} increased the time-to-convergence considerably while only improving the performance marginally. On the other hand, reducing the value of α_{margin} below 0.25 led to a loss of generalizability as the network failed to separate harder negative samples from positive samples.

VIII. Conclusion

Noise in audio data often distorts the speaker dependent characteristics present in it, thereby confounding speaker verification methods. As discussed here and also in other literature ([13], [22]), MFCC as a speech representation technique is not very robust to audio degradations; therefore, speaker recognition performance of methods that solely rely on MFCC features will suffer in the presence of audio degradations. In contrast, the proposed 1D-Triplet-CNN algorithm is observed to be robust to a wide range of audio degradations as evidenced in the experimental results. When compared to xVector-PLDA, the proposed algorithm using MFCC-LPC features, improves

the average TMR by 12% on the degraded-TIMIT dataset, 16% on degraded-Fisher and 11% on degraded NIST SRE 2008 and 2010 datasets at FMR=10%.

While our proposed method outperforms all the baseline methods by a substantial margin, it still fails to correctly verify almost 14% of the samples in the degraded-TIMIT dataset, 16% of the samples in the degraded-Fisher dataset and almost 21% of the samples in the clean and degraded NIST SRE 2008 and 2010 datasets. Therefore, we plan to extend our work by extracting features directly from raw audio data, thus removing our reliance on MFCC and LPC based speech features.

ACKNOWLEDGMENT

We would like to thank Dr. Shantanu Chakrabartty and Dr. Kenji Aono from Washington University in St. Louis for providing us access to their audio degradation software.

REFERENCES

[1] 2008 NIST speaker recognition evaluation training set part 2 ldc2011s07. https://catalog.ldc.upenn.edu/LDC2011S05. Accessed: 2018-03-06.
[2] 2010 NIST speaker recognition evaluation test set ldc2017s06. https://catalog.ldc.upenn.edu/LDC2017S06. Accessed: 2018-03-06.
[3] Amazon Alexa voice recognition. https://www.theverge.com/circuitbreaker/2017/10/11/16460120/amazon-echo-multi-user-voice-new-feature. Accessed: 2017-12-29.
[4] Apple Siri voice recognition. http://www.zdnet.com/article/apple-adds-individual-voice-recognition-to-hey-siri-in-ios-9. Accessed: 2017-12-29.
[5] HSBC voice id making telephone banking safer than ever. https://www.hsbc.co.uk/1/2/voice-id. Accessed: 2017-12-29.
[6] Morpho and agnitio partner, bring voice biometrics to criminal id. https://findbiometrics.com/morpho-and-agnitio-partner-bring-voice-biometrics-to-criminal-id-21261/. Accessed: 2018-06-13.
[7] Siri background noise. https://support.apple.com/en-us/HT204389. Accessed: 2017-12-29.
[8] Voicevault biometrics to protect payments. https://findbiometrics.com/voicevault-biometrics-to-protect-payments-25131/. Accessed: 2018-06-13.
[9] J. Allen and D. Berkley. Image method for efficiently simulating small-room acoustics. *The Journal of the Acoustical Society of America*, 1979.
[10] S. Boll. Suppression of acoustic noise in speech using spectral subtraction. *IEEE TASSP*, 1979.
[11] M. Brookes et al. Voicebox: Speech processing toolbox for MATLAB. *Software, available [Mar. 2011] from www. ee. ic. ac. uk/hp/staff/dmb/voicebox/voicebox. html*, 1997.
[12] W. Campbell, D. Sturim, and D. Reynolds. Support vector machines using GMM supervectors for speaker verification. *IEEE SPL*, 2006.
[13] A. Chowdhury and A. Ross. Extracting sub-glottal and supra-glottal features from MFCC using convolutional neural networks for speaker identification in degraded audio signals. In *IEEE IJCB*, 2017.
[14] C. Cieri, D. Miller, and K. Walker. Fisher English training speech parts 1 and 2. *Philadelphia: LDC*, 2004.
[15] N. Dehak. *Discriminative and generative approaches for long-and short-term speaker characteristics modeling: application to speaker verification*. PhD thesis, École de technologie supérieure, 2009.
[16] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet. Front-end factor analysis for speaker verification. *IEEE TASLP*, 2011.
[17] P. B. Denes and E. Pinson. *The speech chain*. Macmillan, 1993.
[18] J. S. Erkelens. *Autoregressive modelling for speech coding: estimation, interpolation and quantisation*.
[19] W. Fisher, G. Doddington, and K. Goudie-Marshall. The DARPA speech recognition research database: specifications and status. In *Proc. DARPA Workshop on speech recognition*, 1986.
[20] D. Garcia-Romero and C. Espy-Wilson. Analysis of i-vector length normalization in speaker recognition systems. In *Interspeech*, 2011.
[21] A. Gibiansky, S. Arik, G. Diamos, J. Miller, K. Peng, W. Ping, J. Raiman, and Y. Zhou. Deep voice 2: Multi-speaker neural text-to-speech. In *NIPS*, 2017.
[22] J. Guo, R. Yang, H. Arsikere, and A. Alwan. Robust speaker identification via fusion of subglottal resonances and cepstral features. *The Journal of the Acoustical Society of America*, 2017.
[23] M. Hamad and M. Hussain. *Mazin's Thesis (064055)*. 08 2016.
[24] J. Hansen and T. Hasan. Speaker recognition by machines and humans: A tutorial review. *IEEE Signal processing magazine*, 2015.
[25] J. Hansen and V. Varadarajan. Analysis and compensation of Lombard speech across noise type and levels with application to in-set/out-of-set speaker recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 2009.
[26] H. Hirsch and D. Pearce. The aurora experimental framework for the performance evaluation of speech recognition systems under noisy conditions. In *ISCA Tutorial and Research Workshop*, 2000.
[27] C. Jankowski, A. Kalyanswamy, S. Basson, and J. Spitz. Ntimit: A phonetically balanced, continuous speech, telephone bandwidth speech database. In *IEEE ICASSP*, 1990.
[28] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
[29] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter. Self-normalizing neural networks. *arXiv preprint arXiv:1706.02515*, 2017.
[30] N. Krishnamurthy and J. Hansen. Babble noise: modeling, analysis, and applications. *IEEE TASLP*, 2009.
[31] P. Matějka, O. Glembek, O. Novotnỳ, O. Plchot, F. Grézl, L. Burget, and J. Cernockỳ. Analysis of dnn approaches to speaker identification. In *IEEE ICASSP*, 2016.
[32] T. May, S. Van De Par, and A. Kohlrausch. Noise-robust speaker recognition combining missing data techniques and universal background modeling. *IEEE Transactions on Audio, Speech, and Language Processing*, 2012.
[33] B. Milner and X. Shao. Speech reconstruction from mel-frequency cepstral coefficients using a source-filter model. In *Interspeech*, 2002.
[34] J. Ming, T. Hazen, J. Glass, and D. Reynolds. Robust speaker recognition in noisy conditions. *IEEE Transactions on Audio, Speech, and Language Processing*, 2007.
[35] L. Muda, M. Begam, and I. Elamvazuthi. Voice recognition algorithms using mel frequency cepstral coefficient (MFCC) and dynamic time warping (DTW) techniques. *CoRR*, 2010.
[36] A. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.
[37] A. Paszke, S. Gross, S. Chintala, and G. Chanan. Automatic differentiation in PyTorch. In *NIPS*, 2017.
[38] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, et al. The kaldi speech recognition toolkit. Technical report, IEEE Signal Processing Society, 2011.
[39] D. Reynolds. Speaker identification and verification using Gaussian mixture speaker models. *Speech Communication*, 1995.
[40] D. Reynolds, T. Quatieri, and R. Dunn. Speaker verification using adapted Gaussian mixture models. *Digital Signal Processing*, 2000.
[41] D. Reynolds and R. Rose. Robust text-independent speaker identification using Gaussian mixture speaker models. *IEEE TSAP*, 1995.
[42] F. Richardson, M. Brandstein, J. Melot, and D. Reynolds. Speaker recognition using real vs synthetic parallel data for dnn channel compensation. *Interspeech*, 2016.
[43] F. Richardson, D. Reynolds, and N. Dehak. Deep neural network approaches to speaker and language recognition. *IEEE SPL*, 2015.
[44] S. Sadjadi and J. Hansen. Robust front-end processing for speaker identification over extremely degraded communication channels. In *IEEE ICASSP*, 2013.
[45] S. Sadjadi, M. Slaney, and L. Heck. MSR identity toolbox v1.0: A MATLAB toolbox for speaker-recognition research. *Speech and Language Processing Technical Committee Newsletter*, 2013.
[46] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *IEEE CVPR*, 2015.
[47] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur. X-vectors: Robust DNN embeddings for speaker recognition. In *IEEE ICASSP*, 2018.
[48] A. Varga and H. Steeneken. Assessment for automatic speech recognition: II. NOISEX-92: a database and an experiment to study the effect of additive noise on speech recognition systems. *Speech communication*, 1993.
[49] N. Wang, P. Ching, N. Zheng, and T. Lee. Robust speaker recognition using both vocal source and vocal tract features estimated from noisy input utterances. In *IEEE International Symposium on Signal Processing and Information Technology*, 2007.
[50] Z. Zhang, L. Wang, A. Kai, T. Yamada, W. Li, and M. Iwahashi. Deep neural network-based bottleneck feature and denoising autoencoder-based dereverberation for distant-talking speaker identification. *EURASIP Journal on Audio, Speech, and Music Processing*, 2015.
[51] X. Zhao, Y. Wang, and D. Wang. Robust speaker identification in noisy and reverberant conditions. *IEEE/ACM TASLP*, 2014.