ARASID: Artificial Reverberation-Adjusted Indoor Speaker Identification Dealing with Variable Distances

Zeya Chen
University of Virginia
zeyachen@virginia.edu

Mohsin Y. Ahmed
University of Virginia
mohsin.ahmed@virginia.edu

Asif Salekin
University of Virginia
asifsalekin@virginia.edu

John A. Stankovic
University of Virginia
stankovic@virginia.edu

Abstract

Indoor speaker identification systems have been researched for a long time and are widely used in many human interaction acoustic monitoring systems. Many works have focused on improving accuracy in dealing with different realisms, including noise and varying distances from the microphone. However, these works either require significant extra effort such as measuring room types and dimensions, obtaining many speakers' samples, or requiring expensive hardware such as microphone arrays and complex deployment settings. In this paper, we introduce a complete speaker identification solution using an artificial reverberation generator with different parameters to adjust the original close-distance speech samples so that each speaker has different artificial voice samples. Samples in different environments are not required because these artificial samples are close approximations to different environments. Two kinds of models, GMM-UBM and the i-vector, are evaluated. The models are trained on all samples separately, and testing is done against all in parallel. A score fusing approach with two thresholds, a minimum value and a minimum difference, is applied to the scores in producing the final result. Also, several standard acoustic pre-processing routines, including a voice activity detection algorithm and an overlapped speech remover, are included to make the system fully deployable. Finally, to assess the improvements when applying a reverberation adjustment, we evaluate our system with two literature speech databases, one has 251 people and the other one has four kinds of emotions. Further, we perform an in-lab speaking experiment. The evaluation results show our system has more than 90% accuracy in identifying speakers within 6 meters if the emotion is neutral, and a 10% improvement over no reverberation adjustments when speakers have non-neutral emotions.

Categories and Subject Descriptors

C.3 [**Special-purpose and Application-based Systems**]: Signal processing systems; I.5.4 [**Pattern Recognition**]: Applications—*Signal processing*

General Terms

Machine learning on sensor data

Keywords

speaker identification, reverberation, distance

1 Introduction

Today we are surrounded by millions of digital devices and sensors. These devices and sensors can be anywhere and have various types and purposes, from smartphones in one's pocket to motion sensors at home, from noise detectors on roads to fire alarms in forests. We use them to monitor and improve our daily lives. The key benefit of these systems is their ability to gather and analyze the data from the surrounding environment automatically, robustly, and easily.

An acoustic sensor is one of the most commonly used sensors to deal with human interaction mainly because sound has fewer limitations than other physical signals, such as vision, when being used to monitor people. For example, sound is not affected by light conditions, and it does not require a direct path. For acoustic sensors, which monitor humans, knowing the identity of the subject is crucial.

The idea of Speaker IDentification (SID) has been proposed and developed for more than two decades. The goal of this kind of system is to automatically detect who is speaking within the coverage of a microphone. For example, we can use a SID system to detect who is at home, who is talking in a meeting, or whether there is a malicious person in a security-sensitive place. Recently, virtual assistants in smart homes such as Google Home announced that it can recognize an individual's voice [4] which could be extremely valuable. But, it detects the speaker from fixed sentences.

SID is non-trivial due to different kinds of realities. The accuracy of the system is affected by many factors, such as noise and distances. Noise makes it difficult to obtain the

International Conference on Embedded Wireless Systems and Networks (EWSN) 2019
25–27 February, Beijing, China

ISBN: 978-0-9949886-3-8

actual speaker's voice features, and distance adds reverberation and de-amplification problems which distorts the original signal.

The current state-of-the-art algorithms focus on solving these realisms by measuring room characteristics which can be used to adjust the signal [43], by obtaining significant amount of training data for each speaker in different environments, or by using expensive equipment [24, 36, 37, 42]. All of these methods require huge amounts of additional effort, and they are not practical for rapid or large-scale deployments. Also, a practical and operational SID system is likely to face several environmental and realistic challenges like non-human sound, overlapping speakers, varying distance from microphones, training overhead, and presence of non-trained speakers (both real and from sources like a TV) which need to be solved.

Recently, researchers are deploying health care systems to monitor people's activity of daily living, creating activity diary systems, studying family eating dynamics, and so on. Acoustic monitoring is often a part of these systems. An important issue is that these systems must be deployed with little effort. The motivation for this paper is to solve the SID problem for a system that must be deployed quickly and easily in a home to detect speaker IDs without the room measurements, extensive data samples, fixed sentences, or expensive hardware.

The basic solution approach is that for each speaker, we only need to record a few voice samples (at training time) by a close microphone in a single arbitrary room. The recordings are adjusted with different artificial reverberations to generate a group of artificial samples. The artificial reverberation parameter settings cover various types of rooms. Standard cepstral mean normalization (CMN), feature warping and noise reduction techniques are applied. We treat all these groups of samples as different artificial speakers in the training phase. Conventional GMM-UBM and more state-of-the-art i-vector methods are used to build speakers' model. When testing, scoring is done in parallel for each artificial speaker model. In the end, all the scores are compared, fused, and translated back to the real speaker identity.

The main contribution of this paper is a speaker identification system which does not require huge amount of training data and can handle different environments, far distances from microphones, unknown speakers and emotions. The highlights include:

- **Realisms** The solution takes realisms into consideration and provides a general solution of SID for indoor scenarios. The solution is practical and easy to deploy. It detects and filters out non-speech and overlapped speech samples.
- **Distance-independent**. The solution is distance independent. There is no need to have a particular reverberation measurement as an input before deploying. Speakers can be at different distances to the microphone. It also does not need a localization algorithm nor expensive microphone arrays to locate the speaker's position.
- **Limited training samples required**. Since different reverberation settings are used to adjust the training sample, there is no need to collect many user samples at different distances or in different types of rooms.
- **Non-trained speaker separation**. Non-trained speakers' samples can be separated from trained speakers, which means the system filters out background television speech, or an outside visitor.
- **High confidence in the results**. By applying a difference threshold between the first and second most likely speakers we eliminate almost all false positives.
- **General emotions**. This is the first work to study SID under various moods such as angry, sad, and happy mixed with their voice. The accuracy of our solution at close distances to the microphone is the same as the state-of-the-art results, and the accuracy at distant is improved by 10%.
- **Good overall performance**. By evaluating our solution with two literature datasets and new controlled lab experiments, we show that distant speakers' IDS within 6 meters of the microphones can be detected with more than 90% accuracy.
- **Evaluation**. In the evaluation, we show that our solution outperforms various baselines, i.e., a GMM-UBM solution with MFCC features only by 10.9% at 6 meters, an i-vector PLDA solution with enhanced MFCC features (enhanced by CMN and feature warping) by 11.5% at 6 meters.

Note that ARASID is designed for detecting a small number of speakers in a single deployment. For example, it can be used in a house of 4-6 family members to monitor their daily lives such as conversations during meal time. It can be used in a small conference to keep logs of who is talking. The system can handle arbitrary number of speakers, but we limit the total number of speakers in order to ensure computational efficiency.

2 Related Work

SID has been researched for a very long time. While some research is still being conducted in fundamental algorithms to improve performance, other research is addressing realisms.

One realistic problem is the distance between the speaker and the recording device. McCowan et al. [24] introduced microphone arrays into SID algorithms. They mentioned that a single distant microphone cannot have acceptable performance in very noisy conditions. On the other hand, using a microphone array enhances the signal based on the knowledge of sound direction. So it can improve the robustness of SID systems. Wang et al. [36, 37] used position-dependent Cepstral Mean Normalization to minimize channel distortion caused by distance. The speaker model is trained based on the speaker's position with estimated compensation parameters for position-dependent CMN. To recognize a speaker, the system first estimates the speaker's position, then applies the particular compensation parameters in CMN and performs the SID algorithm. Zieger et al. [42] addressed the problem of user identification and voice control of a TV-system. Reverberation, noise from TV's output and distance affect the system's performance. Techniques includ-

ing source localization, beamforming and echo cancellation were applied by using a microphone array to improve accuracy. As we can see here, all these approaches need a microphone array, which is relatively expensive. The algorithms heavily rely on the localization algorithm. Neither of these assumptions is often practical for a large-scale deployments.

Meanwhile, many works focus on using fundamental algorithms to deal with distance distortion. Jin et al. [16] investigated two approaches in a far-field speaker recognition system. One is reverberation compensation and feature warping, which provides significant improvements under mismatched training-testing conditions. Multiple channel combination strategies including data combination, frame-based score competition, segment-based score fusion, and segment-based decision voting were introduced to deal with multiple channels' data. The second approach uses higher-level linguistic features. In [15], they evaluated minimum-variance distortionless response features, fundamental frequency variation features, factor analysis, and frame-based score competition by using MIXER5 Corpus [3] under mismatched conditions. Similarly, Ye et al. [14] applied spectral subtraction before feature extraction, feature warping, and model compensation. Remus et al. [29] used partial least squares (PLS) to decompose the features and mitigate the degradation in SID performance. Fowler et al. [11] described a standoff multi-microphone speech corpus, and further evaluated the performance of PLS. All these works are trying to adjust the contaminated speech to the clean speech, and then obtain the speech features from the recovered speech. However, they cannot solve all the realistic scenarios because the model is only trained on one scenario, and the recovered speech may not match with the real deployment scenario.

Another focus of current research is reverberation. Reverberation is the most important factor in distant speech. Zieger et al. [43] used artificial reverberation to contaminate clean speech. The real impulse responses were measured at different distances in a typical room. Both clean speech and contaminated speech were used to build the speaker models, and the final decision was generated by a weighted score fusion. Falk et al. [9] also considered room reverberations. A Gammatone filter-bank was used to filter speech signals, and speaker features were extracted from modulation frequency bands. Clean speech, artificially generated reverberant speech and reverberant speech recorded in a meeting room were used to evaluate their models. Also, multichannel score combination and adaptive channel selection techniques were introduced for multi-microphone systems. Zhang et al. [40] trained speaker models using dereverberant speech obtained by suppressing reverberation from arbitrary artificial reverberant speech. Dickerson et al. [8] introduced a system called RESONATE, which compensated for reverberation using simulated room impulse responses to adapt to the real reverberant rooms. These works either need to measure the room reverberation parameters or rely on the dereverberation algorithms. In a real deployment, we cannot obtain each room's parameters because it may take a huge amount of time and effort, and the dereverberation algorithms are still not robust enough to different room types or they require expensive computation overhead such as [41].

Recently, deep neural network approaches were used to achieve better performance. However, the training set and/or the enrollment set for those system is large. [31] used a 100 hours dataset to train the DNN model. In [25], training data was sourced from 800 and 1300 hours of microphone and telephone speech. The English training dataset in [20] has 2,174 speakers, 543,840 utterances, and more than 620 hours of speech. [35] uses 28k recordings from 2.6k speakers from the SWBD dataset and from NIST SREs between 2004 to 2010 which contains about 63k recordings from 4.4k speakers to train their model. To our best knowledge, deep learning methods are generally dependent on availability of large amounts of data. With small training sets, it is more likely to overfit or underfit. The main idea is that our solution does not require a large training dataset.

Some other work [10] mentioned they used an open-source simulator to generate degraded speech recordings from clean speech to train speaker models. They addressed the effect of a wide variety of speech degradation on a SID system. Similar to this work, in [1], they implemented a multiple speaker model approach on a GMM-UBM based speaker identification system by using different reflection coefficients to model realistic levels of reverberation. These works improve the accuracy, but they do not mention the performance of speaker identification at distances.

In this paper, we focus on a speaker identification system which generates speaker models with artificial reverberations. The artificial reverberations mimics various distance effects with a single recording sample, so the system can handle various conditions with a very small training sample set. All the adjustments can be done offline such that no additional human effort is needed to collect more sample data or use special devices.

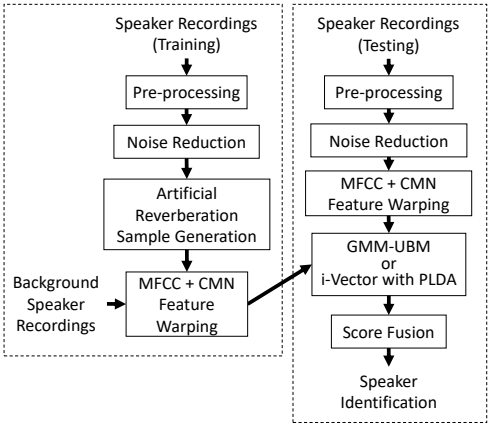

Figure 1. ARASID system overview.

3 ARASID

In this section, we describe the details of our solution, called ARASID. As shown in Figure 1, it consists of several steps including pre-processing, noise reduction, artificial reverberation sample generation, MFCC features, CMN, feature warping, GMM-UBM model training, testing, and score fusion.The major idea here is instead of using one speech sample to generate a speaker model, each speaker has

a group of generated samples and models. We only need to collect a close microphone's recording for each speaker, and all the other samples are generated from the pre-processed and noise-removed close recording by adding different reverberations. When running the system, we try to match a speaker's speech input to all these models in parallel. Because each model covers one kind of room reverberation type, the real environment is more likely to match one of them so that it is expected to have a better performance in realistic scenarios.

3.1 Pre-processing and Noise Reduction

Pre-processing and noise reduction are both applied in the training and testing phases. Both phases share the same procedure. First, the input signal is filtered by a Butterworth bandpass filter in order to get rid of the noise, which is outside the human voice frequency range. A Butterworth filter is widely used in noise filtering. It rolls off slowly around the cutoff frequency, but without ripples. In our system, we use third order bandpass range from 100Hz to 3500Hz. Second, we apply the traditional and standard spectral subtraction method to remove the noise which has some overlapped in the frequency range of human voice. It is a simple and effective method of noise reduction. The first step is to get a spectrum profile from the pure background noise, and form a fingerprint. For each frequency spectrum of each short segment, if the energy in a frequency spectrum is less than the average value in the fingerprint, it gets reduced. After finishing this process in each spectrum, time-smoothing and frequency-smoothing are used to smooth the output signal.

In real time deployments, a simple threshold-based energy filter is used to filter out low energy signals. A voice activity detection algorithm is applied to get rid of non-human speech. Here, we use the LTSD VAD filter from [28]. It measures the long-term spectral divergence (LTSD) between speech and noise, and uses the decision rule to determine speech/non-speech segments. Furthermore, one of the major challenges in realistic conversations is overlapping speech. Overlapping speech could generate many false positives/negatives. In order to eliminate these overlapped speech samples, we apply the overlap speech detector from our recent work [33]. It is a binary neural network classifier with two output classes, overlapped speech and single-person speech. If a segment is output as overlapped speech, we simply ignore it and skip the speaker identification process for this sample.

3.2 Artificial Reverberation Sample Generation

Dereverberation methods have been developed for a long time. But it is still not perfect. The output signal has some level of distortion because of reverberation. Another problem is that it requires measuring room-specific reverberation model parameters as the dereverberation algorithm's input because room types change models significantly. We noticed that, recently, many single-channel and multi-channel dereverberation methods have been proposed, such as [38] and [7]. But there is significant computational overhead.

However, if we address this differently, based on an input signal, we can easily add artificial reverberations. In order to cover as many different types of rooms as possible, we do not need to measure the parameters for each reverberation model in advance, because instead we can use general combinations of reverberation parameter settings.

A reverberation model has several important parameters. In this paper, we refer to the parameters included in the Audio System Toolbox in Matlab 2016a [22]. Wet and dry ratio is the ratio of the reverberated signal to the original signal. The more reverberation the room has, the larger this ratio is. Diffusion is the density rate of the reverberation tail. Higher diffusion rate means the reflection is closer and the sound is thick. Lower diffusion rate has more discrete echoes. Decay factor measures the time duration that a reflection runs out of energy. A larger room has longer reverberation tails and lower decay factors. A smaller room has shorter tails and higher decay factors. There are other parameters such as pre-delay time, which is the time between the direct sound and the first reflection, and high-frequency cutoff / damping, which is the attenuation of higher frequency in reverberated sound output.

In ARASID, we use different combinations of wet / dry ratio, diffusion, and decay factors to model different rooms (see Section 4.1). All the other parameters are set to the default. To each speaker's recording is added reverberation with different reverberation parameters.

3.3 MFCC and CMN

MFCC is the most widely used feature sets in SID algorithms. We use it as basic model features in ARASID.

In order to minimize the energy differences between different distant speech, the signal is scaled to (-1, 1) based on Equation (1) before extracting MFCC features.

$$s_n(t) = \frac{s_o(t)}{\max\left(|s_o(1)|, |s_o(2)|, ..., |s_o(n)|\right)} \tag{1}$$

where $s_n(t)$ is the scaled signal value at time t and $s_o(t)$ is the original signal value at time t. The whole time period has n values. The scaled signal value is the original value divided by the maximum absolute signal value. This ensures the original zero value stays at zero and the range is scaled and fit into (-1, 1).

Cepstral Mean Normalization (CMN) subtracts the overall mean value from each cepstral value. It can compensate for distortion such as those caused by different microphone channels. However, CMN does not work when the impulse response lasts longer than the short time analysis window [16]. For those input signals which include noise, the CMN results have biases.

Let us assume the original signal is $x(t)$, noise is $n(t)$, $h(t)$ is the impulse response of reverberation, and $y(t)$ is the recorded signal.

$$y(t) = x(t) * h(t) + n(t) \tag{2}$$

After taking Fourier Transform, the time-domain relation in (2) becomes the frequency-domain relation (3).

$$Y(f) = X(f) * H(f) + N(f) \tag{3}$$

We use the logarithm when calculating the cepstral. Then

we get (4).

$$\begin{aligned}\log Y(f) &= \log[X(f) * (H(f) + \frac{N(f)}{X(f)})] \\ &= \log X(f) + \log[H(f) + \frac{N(f)}{X(f)}]\end{aligned} \tag{4}$$

If we subtract the mean value from each cepstral, the last part in (4), $\frac{N(f)}{x(f)}$, cannot be subtracted and distorts the estimation. The model trained on this estimation has an unpredictable performance.

Instead of relying on the noise reduction algorithm to remove the unhandled part of Equation (4) or separating impulse responses like [16] did, our solution idea is straightforward. It applies CMN to all the generated recordings and the original recordings. The assumption is that one of the models (including generated and original ones) matches closely enough to the actual speaker's environment. It has similar $h(t)$ and $n(t)$ both in training and testing, which minimizes the side effects of CMN.

3.4 Feature Warping

Feature Warping [27] is widely used in many robust SID systems. It was intended to solve channel mismatch and additive noise problems. In our experiments, we find feature warping also improves the accuracy of emotion mismatch between training and testing data.

The feature warping warps the short-time MFCC values into a standard distribution. The process we used is taken from [16]. Each dimension of MFCC is considered independent and runs the warping algorithm separately. First, we define a short time window, and only the middle value of that short time window is warped. Then we shift the window by one and do another round of warping. Zeros are appended at the beginning and end of the original MFCC stream.

The warping has three steps. The first step is to find the rank r of the middle value among all the N values in that short time window. The second is to calculate the matching CDF value by using $((r-\frac{1}{2})/N)$. The last step is to find the inverse CDF value by searching the standard normal CDF table. The inverse CDF value is the warped value of the original middle value.

After doing the above processes on all the dimensions of the MFCC streams, we obtain our model features.

3.5 GMM-UBM Model or i-Vector Model with Multiple Reverberation-adjusted Samples

In this paper, in order to evaluate the performance improvements when using reverberation adjustment, we choose two models, the traditional GMM-UBM model and the more recent i-vector with PLDA model, to model speakers. These two models are replaceable, and in the final deployment, the system uses only one model. During the evaluation, the system learns each model and tests on each model separately. We compare the results to see which model works better under the reverberation adjustments.

For each speaker, we only record one speech sample by using a close microphone. We use this recording to generate multiple (e.g., 80) reverberation-adjusted samples (see Section 5.1), which represent each speaker in different reverberation environments. We treat each sample as a different speaker in training. The GMM-UBM model training algorithm we used has no difference from the standard algorithm. A huge amount of human voice is used to train the universal background model (See details in the Evaluation Section). MapAdapt is used to adapt the universal background model to each speaker's model [30].

The i-vector extraction algorithm is a well-known algorithm, which is taken from [23], [17] and [6], and implemented in [32]. We did not change the algorithm, and the basic steps of this algorithm are as follows. After UBM is trained from the background speaker samples, the zero-th and first order Baum-Welch statistics are computed from the UBM. Then the total variability subspace is trained from the statistics. The dimension of total variability subspace is chosen as 400. In the next step, the i-vector of each speaker is extracted by using the statistics, the UBM model, and the total variability subspace. As recommended in [32], we do LDA on the extracted i-vector. The dimension of LDA is chosen as 200. Later a Gaussian probabilistic LDA (PLDA) model is learned by using the EM algorithm.

3.6 Testing

For each segment of a sample, we perform the same pre-processing, noise reduction, MFCC feature extraction, CMN, and feature warping.

If the GMM-UBM model is used, the log-likelihood ratio for each test segment is calculated in Equation (5) [30].

$$\begin{aligned}score(X) = &\log P(X|Speaker's\ GMM\ model) \\ &- \log P(X|UBM\ model)\end{aligned} \tag{5}$$

where $score(X)$ is the log-likelihood of speech segment X. It is the difference between the log probabilities of the incoming speech belonging to one specific speaker and the log probability of the incoming speech belonging to the universal background speakers.

If i-vector with PLDA is used, the verification score is calculated as Equation (6) [12].

$$\begin{aligned}score(X) = &\log P(x_1, x_2|H_1) \\ &- \log P(x_1|H_0) - \log P(x_2|H_0)\end{aligned} \tag{6}$$

where x_1 and x_2 are two i-vectors, H_1 means both i-vectors share the same identity latent variable, H_0 means different identity latent variables.

If the score is larger than some threshold (for example, a non-negative value in GMM-UBM model, such as 0), it means, to some degree, it is more likely to be a trained speaker than the background speakers in GMM-UBM model, or it is more likely to be from the same speaker than from a different speaker. If we compare all the scores above the threshold, the largest one indicates the detected speaker. If all the scores are below the threshold, it means the algorithm cannot identify the speaker. It is possible when the algorithm tries to identify a non-trained speaker. This point is valuable under certain circumstances. For example, if the system is used to monitor and keep a log of family's daily conversations, background speakers from a television

or visitors to the home should be filtered out by the threshold. Because those speakers are not trained speakers in the model, the scores are below the threshold.

3.7 Score Fusion

In ARASID, each speaker has multiple models. When testing, each model outputs a score indicating the log-likelihood difference between the speaker and the background speakers. To get the real speaker's identity, we need to integrate these scores. This integration process in ARASID is called score fusion.

One method is to find the index of the largest score among each speaker's scores, and compare these largest scores. The largest of the largest is the detected speaker. We notate this method as the top-1 score fusion approach. It only compares the models with the highest score from each speaker.

Another method is to use the sum of each speaker's top n scores as the speaker's score. This considers not only the highest-score model, but also other related or similar models to get an overall evaluation of the speaker.

The third method is the sum of each speaker's top n scores, which are above a pre-defined threshold. This excludes those models which output the result showing universal background speakers.

The fourth method is to vote. In this way, the number of scores, which are above the threshold, is calculated for each speaker. Comparing these numbers, we label the speaker who has the most votes as the detected speaker.

Here is an example that explains all the above methods. To keep it simple, in this example, there are three speakers, and each speaker has four artificial reverberation models and one original model. Table 1 shows the results of each speaker model's score.

Table 1. A sample result

	Speaker 1	Speaker 2	Speaker 3
Model 1	1.66	0.69	2.78
Model 2	3.41	2.78	2.77
Model 3	3.37	2.86	2.79
Model 4	3.39	2.87	2.33
Model 5	3.40	2.92	2.28

In the top-1 approach, the final score is 3.41, which belongs to Speaker 1, and reverberation model 2 is closest to the real environment. In the top-3 approach, the final score is 10.2 (3.41+3.39+3.40), which also belongs to Speaker 1. Speaker 2 has a score of 8.65 (2.86+2.87 +2.92), and Speaker 3 has a score of 8.34 (2.78+2.77+2.79). In the top-3 with threshold approach, let us assume the threshold is 2.8. The final decision is the same, but Speaker 3's score is 0 because none of the scores is above the threshold. In the voting approach, we use the same threshold. Speaker 1 has four votes, Speaker 2 has three votes and Speaker 3 has zero votes. So the final decision is Speaker 1.

As we can see here, there are some differences between each score fusion strategy. But most of the time the final decision is the same. We perform more evaluations on these different score fusion approaches in the evaluation section, and find out the top-1 approach shows the best performance.

In addition to the minimum value threshold, we apply another threshold on the difference between the largest and the second largest speaker's value. When we use the top-1 approach, each speaker has one value. If the largest and the second largest value among the speakers is not greater than the threshold, we output 'cannot decide' as the decision instead of outputting the identity of the largest value. The motivation is that when we deploy the system, we care more about the confidence of the results. A wrong identity is more severe than a missed identity. In the evaluation section, we show that we can achieve a high confidence in the results with a low compromise on missed detections.

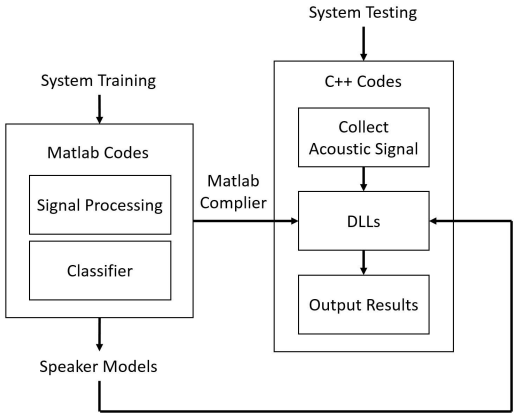

Figure 2. Implementation of ARASID.

4 Implementation

In this section, we describe the implementation our system. Matlab is used to implement the signal processing and the classifier. C++ is used to build a program to collect acoustic signals and output the results. The link between the Matlab code and the C++ program is made by the Matlab Compiler. It compiles Matlab functions into Dynamic-Link Libraries (DLLs) and the C++ program invokes these functions from the DLLs. The training is done offline by Matlab, which generates the trained speaker models. The testing can be either online or offline by the combination of C++ and DLLs generated from Matlab complier. Figure 2 indicates the structure of the system.

4.1 Training

A speaker is required to speak in front of a close microphone to obtain the training samples. Then each sample is passed to a Butterworth filter (details in Section 4.1) which is included in Matlab R2016a. Then we apply frequency subtraction on the filtered signal. The frequency subtraction algorithm is implemented by [2].

The reverberation generator is selected from the Audio System Toolbox of Matlab 2016a [22] which used the idea from [5]. The idea of adding reverberation is to match and simulate as many real scenarios as possible. In practice, we found that a typical indoor environment has the wet/dry ratio of $\{0.2, 0.3, 0.4, 0.5$ and $0.6\}$, the decay factor of $\{0.2\ 0.4\ 0.6\ 0.8\}$ and the diffusion of $\{0.2\ 0.4\ 0.6\ 0.8\}$. More fine-grained combinations have closer matches, but requires more computational complexity. The computation complexity is linear to the number of combinations. 80 $(5*4*4)$ different

combinations of these parameters are chosen because of the trade-off between performance and complexity.

All these speaker samples including the actual and artificial speakers are treated as different speakers to train GMM-UBM models. If the total number of speakers is n, we have n actual speaker models and $80 * n$ artificial speaker models with one universal background model. The GMM-UBM training algorithm and the i-vector with a PLDA training algorithm are implemented by [32].

Each speech recording is split into small segments and is scaled as described in Section 4.2. The MFCC feature extraction algorithm is implemented by the HTK MFCC MATLAB Library [39]. The CMN is simply implemented by subtracting the mean of cepstral value from each cepstral values. 13 MFCC coefficients are used, as well as the deltas and delta-deltas. Feature Warping is implemented by window shifting and using the *icdf* function in Matlab.

We implemented these on a Windows machine with Intel Core i7 CPU having four physical cores. It takes an average of 2 seconds to add one type of reverberation to a wave file of 2.5 minutes by using one core. If we applied the default parallelism in Matlab, four cores can parallelize the work separately. When training the model, it is obvious that the time complexity is related to the total number of actual and artificial speakers. We found adding 80 artificial reverberations per speaker is reasonable for a small group of speakers. The time complexity is almost linear to the total number of reverberation parameters.

4.2 Testing

In the offline version, each recording is passed to the same filter and frequency subtraction function. Then, MFCC features are extracted. CMN and feature warping are applied. Later, the GMM-UBM model or the i-vector with PLDA trials are performed, generating the log-likelihood ratios for each speaker. Using the score fusion techniques, we get the detected speaker's identity.

The online (real-time) version is implemented by the cooperation of Matlab and C++. The C++ program reads the acoustic signal from the connected hardware devices and puts the signal into a buffer. In our solution, the buffer is set to 5 seconds for each microphone channel. Once the buffer is filled, it copies and sends all the buffered signals to the detection thread. Then it clears all the contents and stores the next incoming signal. The detection thread gets the data and calls the functions in the DLLs which are generated from the Matlab complier. These functions have all the required processes as the offline version, including filter, frequency subtraction, MFCC feature extraction, CMN, feature warping, GMM-UBM model scoring, and score fusion. The final returned value of these functions is the detected speaker's identity. The testing takes about 1 second to run the classifier built from 80 artificial reverberations on a 5-second speech sample.

5 Evaluation

In order to show the performance improvement from our artificial reverberation adjustment, we did a comprehensive evaluation on different combinations of feature processing methods. In this section, we present three evaluations on two literature speech datasets and one in-lab experiment. We choose two different literature datasets because the first dataset is used to evaluate the system on groups of people from a large population, and the second one is to evaluate the performance improvement on emotional speech. We also perform an in-lab experiment because for the two literature datasets, we need to play and re-record them in order to generate the distance issue. We want to know the difference between re-recording and actual collected voices in terms of our system's performance. The baseline is the MFCC only approach. Further, we evaluated our two thresholds. The results show that our solution improves the accuracy dealing with different distances.

5.1 Literature Dataset I – Librispeech

We choose Librispeech [26], a corpus of read English speech, as one literature dataset to evaluate our solution. It is chosen because it includes a large number of speakers. The Librispeech was originally collected as a dataset for evaluating speech recognition systems. It was derived from audiobooks, containing 1000 hours of speech sampled at 16 kHz, and the speech was recorded on the mono channel with 32-bit resolution. 251 people were involved in one of the subsets named 'train-clean-100'. It has a total of 100 hours of speech, and each person speaks for about 20-25 minutes. We use all the recordings to train the UBM model and a small portion of speakers to form groups as training speakers.

Table 2. Librispeech dataset results for close distance

	# of Mixtures		
Group #	64	128	256
1 (3 people)	98.8%	98.8%	98.5%
2 (3 people)	97.7%	98.2%	97.9%
3 (4 people)	95.4%	96.2%	97.5%

In this dataset, we only apply the GMM-UBM algorithm. Later in Section 5.4, we compare the GMM-UBM with the i-Vector algorithm. We took approximately 2 minutes of speech recordings from each person in the original recording as the training set for the UBM model. Three groups of speakers were randomly chosen to train the GMM speaker models, which were two 3-person groups and one 4-person group. The models had different amounts of mixtures, which were 64, 128, and 256. Table 2 shows the results when training and testing were both based on the original recordings. The decision window was set to 2 seconds.

When the training and testing samples were clean and matched, as shown in the Table 2, the MFCC-only based GMM-UBM model worked well as it should. We only used 13 MFCC features along with the deltas and delta-deltas. No reverberation model, CMN, or feature warping were introduced. The group of three people had almost 98% accuracy. The group of four had less accuracy, but still remained about 96%. The number of mixtures slightly changed the accuracy.

To evaluate the accuracy when the sound source has distance from the microphone, we developed a script which can play the original sound through a laptop speaker and record it at the same time by a microphone system. We used 6 microphones. Microphone #1 was close (0 meters) to the laptop speaker. Microphone #2 and Microphone #3 were about 1

meter away. Microphone #4 was about 3 meters away. Microphone #5 was about 4 meters away and Microphone #6 was about 5 meters away. The microphones were directional and were faced towards the laptop speaker. It sampled at 16kHz and had 32-bit resolution. All the recordings were done in a quiet room except the air-conditioner was on and made small noises. Due to the limited volume of the laptop speaker, the Microphone #6 could barely record clear voices because it was buried under the background noise. So we dropped results from Microphone #6.

We tested different approaches to solving the distance issue. The UBMs were trained on the same 2 minutes speech recordings of each person in the original dataset. The speaker models were trained only on the closest microphone data with different approaches, only-MFCC, MFCC with CMN, MFCC with CMN and Reverberation, MFCC with Reverberation, and MFCC with CMN, feature warping and Reverberation. 81 artificial reverberation models were generated in those models with reverberation. The highest testing score among models was selected as the output speaker. Table 3 shows the average accuracy in different distances. The decision window was set to 5 seconds.

Table 3. Results on the distant Librispeech dataset. Applying all methods has the best performance

		Distance (meters)			
Only MFCC (Baseline)	**Mixtures**	0	1	3	4
	64	100%	82.6%	83.2%	82.5%
	128	100%	83.8%	83.9%	84.8%
	256	100%	91.5%	90.9%	86.6%
		Distance (meters)			
MFCC, CMN	**Mixtures**	0	1	3	4
	64	99.5%	78.5%	68.2%	68.0%
	128	99.8%	80.2%	69.3%	69.8%
	256	99.8%	82.7%	70.5%	74.3%
		Distance (meters)			
MFCC, Reverb	**Mixtures**	0	1	3	4
	64	99.6%	81.5%	77.7%	77.1%
	128	99.6%	81.2%	78.3%	77.4%
	256	100.0%	80.0%	78.0%	75.4%
		Distance (meters)			
MFCC, CMN, Reverb	**Mixtures**	0	1	3	4
	64	99.3%	97.8%	95.9%	95.9%
	128	99.3%	98.1%	95.7%	96.8%
	256	99.5%	98.6%	96.1%	96.1%
		Distance (meters)			
MFCC, CMN, Warp, Reverb	**Mixtures**	0	1	3	4
	64	99.3%	95.0%	95.7%	95.9%
	128	99.3%	96.3%	96.1%	96.8%
	256	99.5%	97.0%	96.4%	97.5%

As we can see in Table 3, if we only use the MFCC approach to detect a distant speaker, the accuracy drops to 80%, especially when the number of mixtures in the model is low. For 64 and 128 mixtures, the accuracy is around 83% which means there is a 17% accuracy gap between the close microphone and the distant microphone. The model with 256 mixtures has better results, but it is still low (86% at 4 meters). The further the distance is, the lower accuracy it has.

For the MFCC-CMN approach, the results indicate that when we perform CMN on distant recordings, some characteristics of the speaker are removed. CMN was supposed to minimize the distortion of recording. But as some other related works [16, 14] shows, it does not work perfectly in a reverberation environment. Its functionality is highly related to the type of reverberation. When the length of the channel impulse response is longer than the short-time spectral analysis window, it actually does not help, but adds some uncertainty. In this evaluation, it lowers the accuracy.

The MFCC-Reverb has the better results because the artificial reverberation we added matches the real scenario better. The MFCC-CMN-Reverb approach's result shows even better performance compared to the first three approaches. The overall accuracy within 4 meters is over 95.9%. The MFCC-CMN-Warp-Reverb approach is similar to the MFCC-CMN-Reverb approach, which also shows great performance. Notice that, the results of 4 meters are a little better than 1 meter's and 3 meters' in the last approach. It can happen because the artificial models are generated to match the real scenario regardless of distances, and the distance effect is further minimized by feature warping. So it is possible that the 4 meters' speech has a slightly better match.

The above results showed that when training and detecting are both based on the closest microphone, the standard algorithm works well. When the voice distance and training/testing is mismatched, the accuracy drops rapidly from 100% to around 85%, and only applying CMN is not enough because the result is even worse (below 80%). However, the accuracy can be improved back to over 95% when we apply both CMN and the artificial reverberation approach, and applying the feature warping as in our solution, ARASID, has almost the same results.

Table 4. Accuracy of other score fusion methods is lower than the Top-1 Method

Method	**Distance (meters)**			
	0	1	3	4
Top 1 Positive	99.3%	95.0%	95.7%	95.9%
Top 3 Positive	97.1%	94.8%	90.1%	93.0 %
Top 5 Positive	96.5%	94.8%	90.1%	93.0 %
Top 10 Positive	96.5%	94.8%	90.7%	93.2 %
Top 3	97.1%	94.8%	90.1%	93.0 %
Top 5	96.5%	94.8%	90.1%	93.0 %
Top 10	96.5%	94.2%	90.7%	93.0 %
Top 15	96.5%	93.0%	90.1%	93.0 %
Vote	91.9%	84.9%	57.6%	62.8 %

We further analyze our score fusion methods based on the approach in our solution. In the above evaluation, only the Top-1 method is used. Now we test all the score fusion methods. The results are shown in Table 4. Each speaker has 81 different scores from the original model and each reverberation-adjusted models. The top n positive is the method that sums the highest n positive scores of each speaker as the fused score. The top n is the method that sums up the highest scores no matter they are positive or not for each speaker. The vote is to count the number of positive scores for each speaker. As we can see in the table, when we consider more artificial models, the accuracy drops, and the voting method has the lowest results. This is mainly because

only one of the artificial reverberation models will actually match reality. If we combine more than one model's score, it actually degrades the performance. So in the later evaluations, we only use the highest score among all the scores as the speaker's score.

5.2 Literature Dataset II – EMA

Another literature dataset, Electromagnetic Articulography (EMA) database from [19], is used to evaluate our system. This dataset is chosen because it not only has clean speech like the previous one, but also it includes some moods while speaking. Three talkers in this database produced acted emotional speech on a set of 10 sentences. Each sentence was recorded 5 times in four different moods (angry, happy, neutral and sad). In each mood, we took 40 recordings to train the speaker model and took the remaining 10 recordings to test.

In order to evaluate our system on the distant recording, we performed the same procedures as we did on the Librispeech dataset. The original recordings were played through a speaker and recorded by microphones at different distances, which were at 1, 2 and 3 meters.

The first evaluation only focuses on neutral speech. Table 5 shows the accuracy results when we applied MFCC only, MFCC with CMN, feature warping, and MFCC with CMN, feature warping and artificial reverberation.

Table 5. Results on the EMA dataset shows applying all these methods has a better performance

	# of Mixtures	Only MFCC (Baseline)	MFCC CMN Warp	MFCC CMN Warp Reverb
0 Meter	64	100%	100%	100%
	128	100%	97.8%	97.8%
	256	100%	97.8%	97.8%
1 Meter	64	88.5%	88.5%	94.2%
	128	90.4%	88.5%	96.2%
	256	90.4%	92.3%	96.2%
2 Meters	64	84.6%	84.3%	88.5%
	128	88.5%	80.4%	90.4%
	256	88.5%	86.3%	94.2%
3 Meters	64	80.8%	80.8%	86.6%
	128	86.5%	82.7%	88.5%
	256	86.5%	86.5%	92.3%

When comparing the results of MFCC+CMN+Warp+ Reverberation to the results of only MFCC or MFCC+CMN +Warp, we can see improvements. The accuracy within 3 meters are over 90%.

Although this paper is not focusing on identifying speakers under different moods, it is still valuable to see the performance after adding artificial reverberations when training on different moods speech data. Table 6 shows the speaker id results on speech segments with three different moods (Angry, Happy, and Neutral) when using the close neutral speech as the training set. The number of mixtures is 256.

We only use neutral speech as the training set because in reality, it is difficult for speakers to pretend to be happy, angry or sad when training the models. This is due to several reasons. First, not all people are good at acting. Second, each person has his own understanding of moods. Different people change speech differently. Third, when training models, it is usually done once in a short time which cannot cover the whole scope of moods in the real life. If we do emotion-dependent modeling, the model may be trained on happy. When testing, it may not work when the speaker is extremely happy or slightly happy. In order to address these realisms, we can only use the neutral speech to train speaker models.

From the results in Table 6, we can see the overall performance on emotional speaker identification is low when training set and testing sets have different moods. Lots of works such as [34, 13, 18, 21] have shown that moods have a huge influence on the performance of SID systems. The dataset we use is collected from the professional actors. They were trained to change their voices and express moods explicitly, so the voice is more distinguishable and different.

Table 6. Results on the distant emotional dataset show applying all these methods has a better performance

	Only MFCC (Baseline)			MFCC CMN Warp Reverb		
1 Meter	**Emotion**			**Emotion**		
	Angry	Happy	Sad	Angry	Happy	Sad
	51.7%	50.9%	60.3%	61.7%	61.4%	71.4%
2 Meters	**Emotion**			**Emotion**		
	Angry	Happy	Sad	Angry	Happy	Sad
	48.3%	50.9%	50.8%	61.7%	59.7%	60.3%
3 Meters	**Emotion**			**Emotion**		
	Angry	Happy	Sad	Angry	Happy	Sad
	50.0%	50.9%	49.2%	58.3%	52.6%	65.1%

The results also show, as expected, that when the distance is further, the accuracy drops. Our artificial reverberation method has less improvement on the emotion recordings, but it still mitigates the distance influence. When we add reverberation adjustments, the performance on different distances improves by about 10%.

5.3 Controlled Lab Experiment

We recruited 12 people to read scripts in our controlled lab experiment. We used a VocoPro UHF-8800 Wireless Microphone System and a transmitter, M-Audio Fast Track Ultra 8R USB 2.0, to record and transmit sound. The microphone setting is shown in Figure 3. It was a rectangular room and 7 microphones were placed facing the speaker. One was 0.5 meters away, three were about 1.5 meters away, two were about 3 meters away and the last one were about 6 meters away. Each microphone recorded speaker's voice separately, but simultaneously, into 44.1kHz, 32-bit wave format files. Each speaker read two one-minute long scripts and four 6-second sentences in his/her natural volume. All the microphones could record the speech.

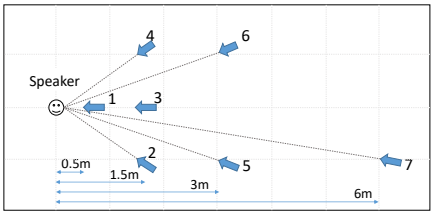

Figure 3. Microphone positions

We evaluated five approaches with the GMM-UBM model, which are MFCC only, MFCC with CMN, MFCC with artificial reverberation, MFCC with CMN and artificial reverberation, and MFCC with CMN, feature warping and

Table 7. Results on the distant controlled lab experiment show applying all these methods has the best performance

		Distance (meters)			
Only MFCC (Baseline)	**Mixtures**	0.5	1.5	3	6
	64	100%	82.7%	77.7%	75.0%
	128	100%	85.0%	77.8%	76.7%
	256	100%	85.0%	78.5%	78.3%
		Distance (meters)			
MFCC, CMN	**Mixtures**	0.5	1.5	3	6
	64	100%	92.9%	87.4%	86.7%
	128	100%	91.3%	88.2%	88.3%
	256	100%	92.1%	88.2%	90.0%
		Distance (meters)			
MFCC, Reverb	**Mixtures**	0.5	1.5	3	6
	64	100%	83.4%	88.5%	85.0%
	128	100%	85.8%	89.3%	86.7%
	256	100%	90.6%	90.1%	88.3%
		Distance (meters)			
MFCC, CMN, Reverb	**Mixtures**	0.5	1.5	3	6
	64	100%	93.7%	91.0%	88.3%
	128	100%	94.5%	91.7%	90.0%
	256	100%	95.3%	93.4%	91.7%
		Distance (meters)			
MFCC, CMN, Warp, Reverb	**Mixtures**	0.5	1.5	3	6
	64	100%	95.3%	90.0%	88.3%
	128	100%	96.9%	90.9%	90.0%
	256	100%	95.3%	90.2%	90.0%

artificial reverberation. The UBM was the same. The two one-minute long speech clips were used to train the speaker model, and the four 6-second sentences were used to test. Only the closest (0.5 meters) microphone recordings were used as the training samples. 80 artificial reverberation models were applied to each raw recording. So each speaker had 81 different GMM models (80 reverberation models and 1 original).

12 people were split into four groups evenly. The accuracy was calculated as the ratio of a total number of correctly identified segments to the total number of segments. The decision window was set to 5 seconds. Table 7 shows the results of the averaged accuracy.

The MFCC-Only approach shows the lowest results when detecting the distant speech. Obviously, it is because the closest recording has quite different features compared with the distant speech. MFCC-CMN and MFCC- Reverberation approaches show similar improvement. In this experiment, CMN did minimize some distortion caused by distances. Each result was improved by about 10%. But when considering the negative effect in the previous experiment, we know only applying CMN to the reverberation scenario has uncertain effects. MFCC-Reverberation improves the accuracy of 3-meter and 6-meter speech by 10%, while the accuracy of 1.5-meter stays almost the same.

MFCC-CMN-Reverberation shows acceptable results, which are about 94% at 1.5 meters, about 92% at 3 meters, and about 90% at 6 meters. MFCC-CMN-Warp- Reverberation approach, as we do in ARASID, has similar results. It has a slight improvement at 1.5 meters while the overall performances are almost the same.

Notice that the overall accuracy in this controlled lab experiment is less than the Librispeech experiment. It is mainly because of the following reasons. First, the part in Librispeech dataset we used was noise-free, and we trained on the microphone recordings next to the laptop speaker (at 0 meters). We can assume the recordings were relatively clean as well. However, the controlled lab experiment used the microphone recordings at 0.5 meter distance which contained some level of noise. Second, the sound quality in the original Librispeech experiment was higher because the speakers were selected professionals. But in our controlled lab experiment, we have never trained our speakers. Third, the relative position and distance between the laptop speaker and the microphone stayed fixed in the Librispeech experiment, and the volumes of all speech were stable. But it was not true in our controlled lab experiment, for example, a speaker may change his volume randomly.

5.4 Comparison between GMM-UBM and i-Vector with PLDA on Reverberation Adjustment

In the previous two evaluations, we only use GMM-UBM as the model to detect a speaker. In this subsection, we compare the performance between the GMM-UBM model and the i-vector with PLDA model.

In this evaluation, the same dataset we collected in Section 5.3 is used, and the GMM-UBM model is built almost the same as the previous setting. The only difference is more mixtures (512, 1024, and 2048 mixtures) are chosen. We compare the two models results of three different feature processing approaches, which are MFCC-only, MFCC with CMN and Warping, and MFCC with CMN, Warping and Reverberation adjustment. The results are shown in Table 8.

As we can see from the results, both GMM-UBM and the i-Vector with PLDA have high accuracy at the close distance and low accuracy at far distances when only MFCC was used. Especially, when distances become further, the i-Vector with PLDA solution almost does not work. This means that the i-Vector solution is more sensitive than GMM-UBM when the training and testing channels are mismatched.

When applying CMN and feature warping, both methods' accuracy are increased. The results show that GMM-UBM and i-Vector have similar results, and differences are within about 3%. It is important because when applying reverberation adjustment, the accuracy gets further improved. GMM-UBM shows slightly better results than the i-Vector with PLDA at different distances, although both of them achieve more than 90% accuracy. In the final deployment version, we only used the GMM-UBM algorithm to model and detect speakers.

5.5 Result Confidence vs Detection Miss

In the previous section, we discussed that a wrong identity is more severe than a missed identity, and by applying a minimum difference threshold, we can achieve a high confidence with a low compromise on the missing detection. Here we

Table 8. Comparison between GMM-UBM and i-Vector with PLDA

		GMM-UBM				i-Vector with PLDA			
Only MFCC (Baseline)		**Distance (meters)**				**Distance (meters)**			
	Mixtures	0.5	1.5	3	6	0.5	1.5	3	6
	512	100%	80.9%	86.6%	85.2%	100%	66.5%	70.0%	63.5%
	1024	100%	85.9%	87.0%	86.2%	99.8%	59.1%	59.1%	51.5%
	2048	100%	89.3%	87.3%	85.3%	99.8%	53.5%	52.5%	48.1%
MFCC, CMN, Warp		**Distance (meters)**				**Distance (meters)**			
	Mixtures	0.5	1.5	3	6	0.5	1.5	3	6
	512	100%	84.3%	82.7%	81.0%	99.3%	84.0%	83.6%	83.4%
	1024	100%	86.5%	83.0%	81.9%	100%	85.1%	85.5%	83.7%
	2048	100%	87.5%	85.2%	82.3%	99.6%	82.2%	82.0%	81.2%
MFCC, CMN, Warp, Reverb		**Distance (meters)**				**Distance (meters)**			
	Mixtures	0.5	1.5	3	6	0.5	1.5	3	6
	512	99.8%	98.6%	93.0%	92.3%	98.2%	93.8%	91.4%	90.7%
	1024	100%	98.3%	93.4%	94.1%	99.5%	93.5%	90.0%	91.2%
	2048	100%	98.5%	95.0%	94.6%	98.2%	92.8%	90.0%	90.5%

perform an evaluation on all the outputs of the GMM-UBM model in all the datasets.

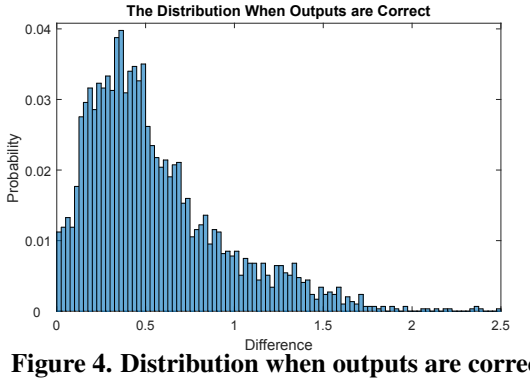

Figure 4. Distribution when outputs are correct.

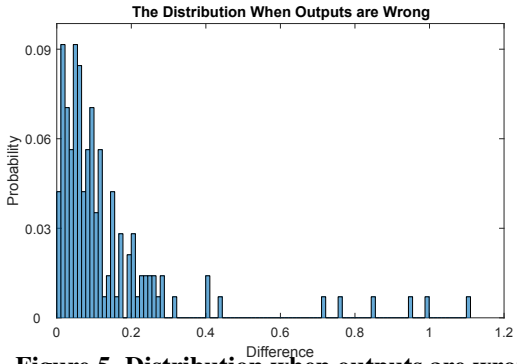

Figure 5. Distribution when outputs are wrong.

Figure 4 shows the distribution of the differences between the largest and the second largest value when the system outputs correct identities. Figure 5 shows the distribution when outputs are wrong. As we can see in the figures, when the results are correct, the majority of the differences is larger than 0.2 while the majority of the differences is less than 0.2 when the results are wrong.

Figure 6 further illustrates that a threshold (e.g. 0.2) guarantees a high confidence with a low compromise on the missing detection. If the threshold is 0.2, a 'cannot decide' result is generated. We may miss about 10% of the times when we could have output correct identities, but we remove more than 80% of times when a wrong identity is generated.

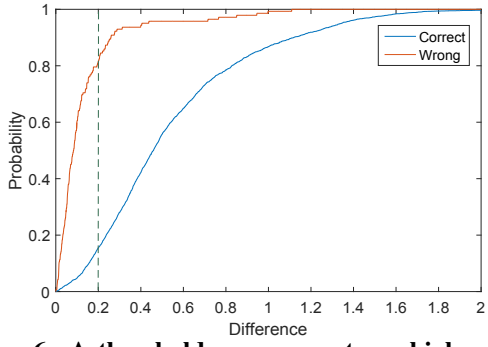

Figure 6. A threshold can guarantee a high confidence with a low compromise on the missing detection.

6 Conclusion

In this paper, we discuss the problem of distance in speaker identification systems. A realistic and easy-and-ready-to-deploy indoor speaker identification system, ARASID, is presented. In our solution, there is no need to measure room sizes or types, to obtain many speech samples, or to use expensive equipment. By adjusting the original speaker's recording samples with different artificial reverberations, a group of artificial speaker models covering various room types and speaker-microphone distances is generated. Combined with cepstral mean normalization and feature warping, the standard GMM-UBM or i-vector with PLDA modeled SID system's performance is improved. In the evaluation, results show that distant speakers in the neutral mood within 6 meters can be detected with more than 90% accuracy. Furthermore, our solution includes standard acoustic pipelines, e.g. filters, voice activity detector, and an overlap speech remover for real deployments. Two kinds of thresholds, minimum value and minimum difference threshold, are applied to improve the confidence of the results with a low compromise on the missing detection.

Acknowledgement

This work was supported, in part, by NSF IIS-1521722 and CNS-1646470. We thank the reviewers and our shepherd for the insightful reviews and suggestions.

7 References

[1] A. Akula, V. R. Apsingekar, and P. L. De Leon. Speaker identification in room reverberation using gmm-ubm. In *Digital Signal Processing Workshop and 5th IEEE Signal Processing Education Workshop, 2009. DSP/SPE 2009. IEEE 13th*, pages 37–41. IEEE, 2009.

[2] S. Boll. Suppression of acoustic noise in speech using spectral subtraction. *IEEE Transactions on acoustics, speech, and signal processing*, 27(2):113–120, 1979.

[3] L. Brandschain, C. Cieri, D. Graff, A. Neely, and K. Walker. Speaker recognition: Building the mixer 4 and 5 corpora. In *LREC*, 2008.

[4] CNN. Google home now recognizes your individual voice. `http://money.cnn.com/2017/04/20/technology/google-home-voice-recognition`, 2017. Accessed: 2017-05-03.

[5] J. Dattorro. Effect design, part 1: Reverberator and other filters. *Journal of the Audio Engineering Society*, 45(9):660–684, 1997.

[6] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet. Front-end factor analysis for speaker verification. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(4):788–798, 2011.

[7] M. Delcroix, T. Hikichi, and M. Miyoshi. Precise dereverberation using multichannel linear prediction. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(2):430–440, 2007.

[8] R. F. Dickerson, E. Hoque, P. Asare, S. Nirjon, and J. A. Stankovic. Resonate: reverberation environment simulation for improved classification of speech models. In *Proceedings of the 13th international symposium on Information processing in sensor networks*, pages 107–118. IEEE Press, 2014.

[9] T. H. Falk and W.-Y. Chan. Modulation spectral features for robust far-field speaker identification. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(1):90–100, 2010.

[10] M. Ferras, S. Madikeri, P. Motlicek, S. Dey, and H. Bourlard. A large-scale open-source acoustic simulator for speaker recognition. *IEEE Signal Processing Letters*, 23(4):527–531, 2016.

[11] M. Fowler, M. McCurry, J. Bramsen, K. Dunsin, and J. Remus. Stand-off speaker recognition: effects of recording distance mismatch on speaker recognition system performance. In *INTERSPEECH*, pages 3713–3716, 2013.

[12] D. Garcia-Romero and C. Y. Espy-Wilson. Analysis of i-vector length normalization in speaker recognition systems. In *Interspeech*, volume 2011, pages 249–252, 2011.

[13] M. V. Ghiurcau, C. Rusu, and J. Astola. A study of the effect of emotional state upon text-independent speaker identification. In *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4944–4947. IEEE, 2011.

[14] Y. Jiang, Z. Tang, and L. Wang. Identification of a distant speaker and its robustness. *Chinese Journal of Electronics*, 20(2), 2011.

[15] Q. Jin, R. Li, Q. Yang, K. Laskowski, and T. Schultz. Speaker identification with distant microphone speech. In *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 4518–4521. IEEE, 2010.

[16] Q. Jin, T. Schultz, and A. Waibel. Far-field speaker recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(7):2023–2032, 2007.

[17] P. Kenny. A small footprint i-vector extractor. In *Odyssey*, volume 2012, pages 1–6, 2012.

[18] S. G. Koolagudi, K. Sharma, and K. S. Rao. Speaker recognition in emotional environment. In *Eco-friendly Computing and Communication Systems*, pages 117–124. Springer, 2012.

[19] S. Lee, S. Yildirim, A. Kazemzadeh, and S. Narayanan. An articulatory study of emotional speech production. In *INTERSPEECH*, pages 497–500, 2005.

[20] C. Li, X. Ma, B. Jiang, X. Li, X. Zhang, X. Liu, Y. Cao, A. Kannan, and Z. Zhu. Deep speaker: an end-to-end neural speaker embedding system. *arXiv preprint arXiv:1705.02304*, 2017.

[21] A. Mansour and Z. Lachiri. Emotional speaker recognition in simulated and spontaneous context. In *2016 2nd International Conference on Advanced Technologies for Signal and Image Processing (ATSIP)*, pages 776–781. IEEE, 2016.

[22] Matlab. Matlab audio toolbox: Design and test audio processing systems. `https://www.mathworks.com/products/audio-system.html`, 2017. Accessed: 2017-05-03.

[23] D. Matrouf, N. Scheffer, B. G. Fauve, and J.-F. Bonastre. A straightforward and efficient implementation of the factor analysis model for speaker verification. In *Interspeech*, pages 1242–1245, 2007.

[24] I. A. McCowan, J. Pelecanos, and S. Sridharan. Robust speaker recognition using microphone arrays. In *2001: A Speaker Odyssey-The Speaker Recognition Workshop*, 2001.

[25] M. McLaren, Y. Lei, and L. Ferrer. Advances in deep neural network approaches to speaker recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pages 4814–4818. IEEE, 2015.

[26] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210. IEEE, 2015.

[27] J. Pelecanos and S. Sridharan. Feature warping for robust speaker verification. 2001.

[28] J. Ramırez, J. C. Segura, C. Benıtez, A. De La Torre, and A. Rubio. Efficient voice activity detection algorithms using long-term speech information. *Speech communication*, 42(3):271–287, 2004.

[29] J. Remus, J. Estrada, and S. A. Schuckers. Mitigating effects of recording condition mismatch in speaker recognition using partial least squares. In *INTERSPEECH*, pages 2674–2677, 2012.

[30] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn. Speaker verification using adapted gaussian mixture models. *Digital signal processing*, 10(1):19–41, 2000.

[31] F. Richardson, D. Reynolds, and N. Dehak. A unified deep neural network for speaker and language recognition. *arXiv preprint arXiv:1504.00923*, 2015.

[32] S. O. 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*, 1(4), 2013.

[33] A. Salekin, Z. Chen, M. Y. Ahmed, J. Lach, D. Metz, K. De La Haye, B. Bell, and J. A. Stankovic. Distant emotion recognition. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 1(3):96, 2017.

[34] I. Shahin. Speaker identification in emotional environments. *Iranian Journal of Electrical and Computer Engineering*, 8(1):41–46, 2009.

[35] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur. X-vectors: Robust dnn embeddings for speaker recognition. pages 5329–5333, 04 2018.

[36] L. Wang, N. Kitaoka, and S. Nakagawa. Robust distant speaker recognition based on position dependent cepstral mean normalization. In *INTERSPEECH*, pages 1977–1980, 2005.

[37] L. Wang, N. Kitaoka, and S. Nakagawa. Robust distant speaker recognition based on position-dependent cmn by combining speaker-specific gmm with speaker-adapted hmm. *Speech communication*, 49(6):501–513, 2007.

[38] M. Wu and D. Wang. A two-stage algorithm for one-microphone reverberant speech enhancement. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(3):774–784, 2006.

[39] S. Young, G. Evermann, D. Kershaw, G. Moore, J. Odell, D. Ollason, V. Valtchev, and P. Woodland. The htk book (revised for htk version 3.4.1). *Cambridge University*, 2009.

[40] Z. Zhang, L. Wang, and A. Kai. Distant-talking speaker identification by generalized spectral subtraction-based dereverberation and its efficient computation. *EURASIP Journal on Audio, Speech, and Music Processing*, 2014(1):1–12, 2014.

[41] 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(1):12, 2015.

[42] C. Zieger, M. Matassoni, and M. Omologo. Experiments on distant-talking speaker verification in tv scenario. In *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 4538–4541. IEEE, 2010.

[43] C. Zieger and M. Omologo. Combination of clean and contaminated gmm/svm for far-field text-independent speaker verification. In *INTERSPEECH*, pages 1949–1952, 2008.