Speaker Recognition using the GMM-UBM Framework in Python

João António Fernandes da Costa, *Master in Bioengineering*

***Abstract*—Speech is a key biometric factor with interesting identification capabilities. A quick overview of the main challenges of Machine Learning related to speech analysis is made, with emphasis on speaker recognition systems.**

A brief report on the main speaker features and speaker models is performed, with main focus on the golden standards for speaker recognition tasks, such as MFCC features and GMM-UBM frameworks speaker models.

A GMM-UBM speaker identification system is implemented in Python 3.6, evaluated with a 10 speaker subset of the VoxCeleb dataset. The influence of model parameters such as number of components and iterations is studied. The overall model achieves accuracy scores of 96%. This work presents itself as a good starting point for future development and research on speaker recognition.

***Index Terms*—Speaker Recognition, Machine Learning, GMM-UBM, MFCC.**

I. Introduction

SPEECH is a fundamental tool of communication for mankind, being the main way humans transmit ideas and emotions to the world around them. As such, the ability to distinguish and identify different speakers based on their speech is paramount to live in society.

Humans naturally present different characteristics of voice and speech mannerisms, which allows for ease of identification: we know the voices of our family, our friends and can easily identify famous celebrities just by hearing them speak. This variability emerges from the different physiology of the larynx, vocal cords and other necessary organs for voice production, but also from the manner speech is spoken, since every speaker has a distinct rhythm, intonation, accent and vocabulary used [1]. The natural ability of humans to distinguish different speakers is called *naïve speaker recognition*.

Speaker identification can be performed on a more technical basis by trained professionals that qualitatively and systematically analyze several aspects of speech on different comparative samples of different speakers, allowing for a more informed decision on the identity of a speaker, or the fraudulent nature of an attempt at imitation. This is defined as *forensic speaker identification*, mainly used in criminal investigations to identify fraud attempts or criminal activity through sound records.

With the increase of power of computational systems and of Machine Learning, *automatic speaker recognition* is an ever more advanced topic nowadays, with automated decision systems that can identify speakers from a list of enrolled individuals or compare different samples and determine if they were produced by the same person. This ability to identify a speaker is extremely helpful as a biometric recognition factor, allied with, for instance, fingerprints, iris, and face. These biometric factors present themselves as identification factors as well, due to their presumed uniqueness in humans: it is highly improbable two humans share a similar factor (such as similar voices), and almost impossible they share two or more biometric recognition factors. As such, biometric authentication is becoming more and more prevalent to protect private information or to authorize certain tasks or services, using for instance fingerprint or voice scanners.

With the advent of digital assistants such as Siri (Apple) or Alexa (Amazon), speech processing in a computational setting becomes increasingly more relevant for speaker and speech recognition, allowing for more sophisticated and less error-prone speech-to-text systems. New techniques for these tasks are being studied and implemented, mainly based on Deep Learning [2], [3].

In this work, a speaker identification system is implemented in Python using the traditional GMM-UBM framework, applied on a portion of the VoxCeleb Dataset. In Section II a general overview of speech-related Machine Learning challenges is given. Section III refers to the most used speech features and their mathematical formulation. In Section IV, a brief mathematical foundation of the most used models for speaker modeling is described. Section V describes the used methods for this work and their implementation in Python. On Section VI results of the application of the GMM-UBM model on the VoxCeleb Dataset are shown, and discussed in Section VII.

II. Speech Machine Learning Tasks

The area of Speech Processing is rich and diverse in Machine Learning topics, since many questions can be asked based on the speech samples of a group of individuals: when are people speaking? Who is speaking? Is it an enrolled individual, or is it new to the system? Can it be an imitation of said person? Who is speaking when? What is the person saying?

Many of these problems can be modeled as a supervised or unsupervised learning process, allowing for future predictions based on training data, or to attribute probabilities of generation of an utterance by an user.

A. Voice Activity Detection

Voice Activity Detection (VAD) or Speech Activity Detection (SAD) is the speech processing step that identifies the time intervals in a given audio signal where speech is occurring. This is an essential preprocessing step for every task with speech manipulation, more so with noisy conditions, since it is not desirable to extract features from timeframes where there is no active speech in the audio signal, which would result in biased speaker models and lower overall performance. This is a well developed field, mainly due to the potential of VAD/SAD to reduce necessary bandwidth for the transmission of voice, for instance in Voice over Internet Protocol (VoIP) applications.

VAD/SAD can be performed empirically, setting thresholds for energy density in the audio signal. Statistical-based VAD can be performed, modeling the distribution of speech and noise Discrete Fourier Transform (DFT) coefficients as Gaussian random independent variables [4]. Nowadays, VAD is tackled as an unsupervised classification problem, where time and frequency features are extracted, combined, and used for each timeframe to classify it as containing speech or not [5].

B. Speaker Recognition

Speaker Recognition tasks are a main focus of research on computational speech processing, along with Speech Recognition. The developed algorithms allow a computer system to identify or verify a user based on its voice patterns, much like other biometric factors, permitting the use of voice as a personal, unique and intransmissible key to access private information. With the increase of mobile computational power, more and more equipment such as smartphones use the potential of voice as a security measure.

These systems envelop two distinct modes: the enrollment mode (or training phase), where the labeled speech utterances from a set of selected speakers are used to create world and speaker models, and the recognition mode (or test phase), where speech samples of unknown origin are compared to the previously defined speaker models. A schema of the general architecture of a Speaker Recognition system is presented in Fig.1.

Speaker Recognition can be performed in two settings: *text-dependent* and *text-independent* [6]. Text-dependent systems rely on cooperative users where the recognition and test phrases are known beforehand, usually selected from a predetermined set. Text-independent systems are unconstrained in this aspect: user speech is not bound to a predetermined script, allowing it to be more fluent and natural. This is a significantly more difficult challenge than the first setting, since train and test conditions can be wildly different, with different speech context.

Speaker Recognition is a broader term for distinct tasks, each with an important niche to fulfill [1]:

1) Speaker Identification: In a Speaker Identification system the goal is to identify an unknown speaker from a set of known speakers, i.e. to determine the speaker who sounds closest to the unknown sample characteristics. This is a multi-classification problem, where the goal is to determine a single predictive class for a sample of unknown origin from a set of known possibilities. This approach can have two distinct scenarios based on the origin of the unknown speaker sample:

Closed or in-set: where unknown samples originate from known speakers considered in the system,

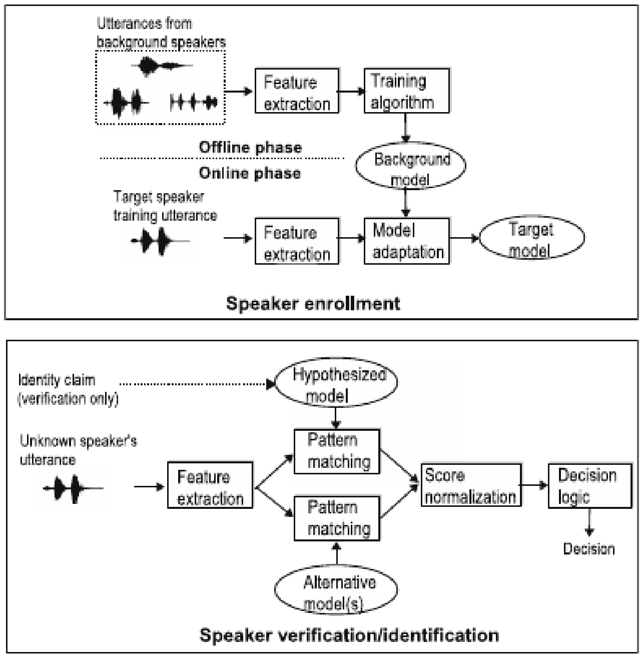

Fig. 1. Schematic of a typical automatic Speaker Recognition system [6]. On top the enrollment phase is represented, which creates the speaker models for the used training data. Below the identification/verification phase is represented, where speech samples of unknown origin are compared to the determined speaker models.

Open or out-of-set: where unknown samples can originate from speakers outside the predefined speaker set. In this case, it is necessary to develop a *Universal Background Model* (*UBM*) to create a world, speaker-independent model of speech.

2) Speaker Verification: Speaker Verification systems have as input an unknown speaker sample and a identity claim, determining if that claim is true or false or predicting the probability of each. This can be translated as a binary class problem, where each claim can be labeled as true or false. Speaker Verification can be performed by comparing samples from the claimed user or by comparing the claimed speaker model to a world model (UBM).

C. Speech Recognition

Speech Recognition focuses on detecting and translating *what* is being said, rather than *who* is saying it. This task, also called Automatic Speech Recognition (ASR) or just "text-to-speech" is a very ample and growing topic of computational linguistics, since it requires complex models of speech and language. Nowadays this task can be done with the use of smartphones and online processing of data, such as the digital assistants present in these devices, which can interpret spoken language and convert to meaningful text.

D. Speech-Related Challenges

Speech, unlike other biometric factors such as fingerprints, presents a very fundamental problem to all computational tasks: speaker variability is large and widely dependent on several internal and external factors, making recognition of speech and speakers difficult and error-prone with the change of these factors [1]. One of the most illustrative examples of this is the way voice changes over time, even on wildly different time scales: our voice changes as we age, but can also drastically change due to disease, accidents or different emotional states of the subject. Furthermore, even with similar external conditions, subjects do not utter the same phrases in the same way. This is called *intraspeaker variability* and is one of the many factors that can have a large influence on speech processing systems.

Variability sources are usually defined as *speaker-based* (stress, emotion, disease, intraspeaker variability,...), *conversation-based* (related to the different scenarios with voice interaction, such as conversations, monologues, spoken text,...), and *technology-based* (quality of audio stream, quality of microphone, background noise,...) [1]. Some of these sources of variability, such as background noise and audio quality, can be mitigated using adequate mathematical modeling, for instance modeling additive noise or transmission channel variability [7].

III. Speech Features

Speech features translate the characteristics of voice and speech to comparable metrics, allowing for speaker model creation and posterior decision making regarding unknown utterances. The chosen features must be good descriptors of speech, with low intra-speaker variability and high inter-speaker variability, robust against mimicry and noise, frequent in natural speech and easy to extract from audio signals [8].

Automatic speaker recognition features can be sub-divided into four groups [6]:

Short-Term Spectral Features: translate the frequency content of speech with time: the audio signal is broken into short ($\approx$ 20 ms) segments and spectral coefficients are extracted, e.g. Mel-frequency Cepstrum Coefficients (MFCCs),

Voice Source Features: characterize glottal activity during speech, such as vocal fold opening or glottal pulses,

Spectro-Temporal Features: modulate variation of the frequency spectrum with time, useful for detection of formant transitions and energy change rate . Derivatives of short-term spectral features can be used, such as ΔMFCCs and $\Delta\Delta$MFCCs,

Prosodic Features: intonation, fundamental frequency, rate of speech, among others. The calculation of the fundamental frequency (F0) is one of the most useful prosodic features, but also a challenging task.

A. Mel-frequency Cepstrum Coefficients

Mel-frequency Cepstrum Coefficients (MFCCs) are one of the most widely used short term spectral features used for speaker and speech recognition [6], capturing well the variability of spectral content between different speakers. These can be used directly to model speaker and background models [9], which are the basis for more advanced systems, as described in Section IV.

The required steps for calculation of the MFCCs are represented in Figure 2. Mathematical formulation of MFCCs is as follows [10]:

Windowing: the input audio signal is windowed with tapered windows (Hamming or Hanning) to short frames of 20 milliseconds (ms). The result is exemplified in Figure 2(a).

Fourier Transform: Discrete Fourier Transform (DFT) is applied to the previously determined frames (zero-padded), as detailed in Equation 1:

$$X_k = \frac{1}{N} \sum_{n=0}^{n=N-1} x_n e^{-j\frac{2k\pi}{N}},\ k \in [0, 1, ..., N] \qquad (1)$$

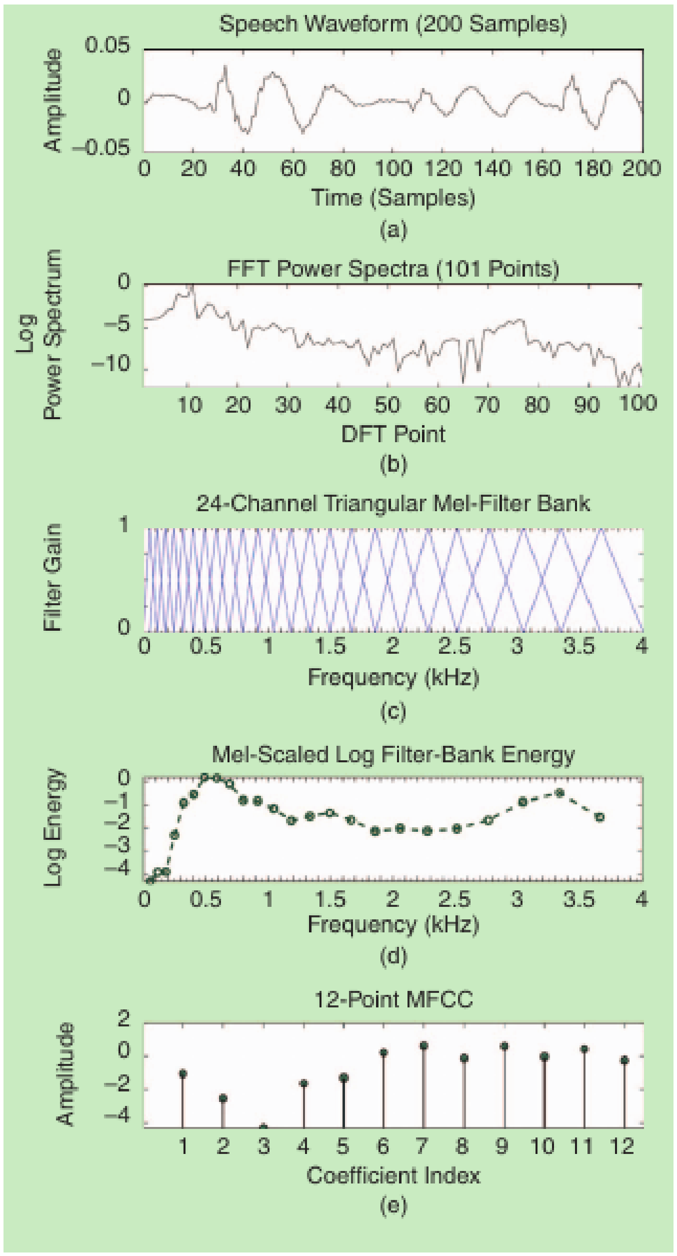

Fig. 2. Steps in MFCC calculation from a considered speech frame [1]. (a) 200 sample speech frame, (b) DFT power obtained through FFT, (c) 24-channel triangular mel-scale filter bank, (d) log filter bank energy, (e) MFCC of size 12 obtained from the first 12 DCT coefficients of log filter bank energy.

where X_k are the DFT coefficients, N the total length of the DFT (usually 512), x_n the sequence of audio samples, with $j^2 = -1$.

The power spectrum is then considered, as in $E_k = X_k \cdot X_k^*$, where X_k^* represents the complex conjugate of X_k. Graphical representation of the power spectrum is present in Figure 2(b).

Log Mel-scale Filter Bank: the power spectrum signal is passed through a filterbank of p triangular overlapping filters, with linearly spaced central frequencies in the mel Scale, as seen of Figure 2(c), where $p = 24$. The mel Scale is a psychoacoustic perceptually uniform frequency space [11], usually converted with Equation 2 [12]:

$$m = 2595 \log_{10} \left(1 + \frac{f}{700}\right) \tag{2}$$

with f as the frequency in hertz.

The logarithm of the filter responses is computed, resulting in the log energy output of the filterbank, as seen on Figure 2(d).

Discrete Cosine Transform: the log energy output of the mel filterbank is subjected to the Discrete Cosine Transform (DCT) (similar to (1) but only considering real numbers), decomposing the sequence into a sum of cosines of different frequencies, with different coefficients. For MFCCs, only the lowest 12-15 coefficients are kept. Figure 2(e) shows the MFCC considering the first 12 coefficients of the DCT.

MFCCs are then vector representations with 12 to 15 dimensions of short audio segments. A speaker utterance usually contains hundreds of these MFCC vectors in sequence. To further capture the dynamics of spectral variability and spectro-temporal changes, the derivatives of these MFCC vectors with respect to time can be considered, resulting in the ΔMFCCs for the first derivative and the $\Delta\Delta$MFCCs (or Δ^2MFCCs) for the second derivative. These extra coefficients are usually appended to the base feature vector of each frame, improving general performance.

B. Feature Normalization

To improve robustness in noisy conditions, audio signals can be first preprocessed to perform noise reduction and improve general performance. However, since these tasks can require some computational power and are sometimes based on empirical approaches that better suit the data used, this approach may not be the ideal. The use of feature normalization can overcome and mitigate noise effects and artifacts in the extracted features, allowing for more robust feature extractors [6].

Several feature normalization techniques can be applied to the extracted features (such as MFCCs), reducing noise effects. Some of the most popular alternatives are Cepstral Mean and Variance Normalization (CMVN), Feature Warping (FW), Relative Spectral filtering (RASTA) and Feature Mapping.

Cepstral Mean and Variance Normalization (CMVN) is a simple feature space transformation, which subtracts to all feature vectors the mean of the feature vector set and normalizes variance by dividing by the standard deviation. This results in zero-mean and unit-variance feature vectors, where channel variability effects, such as the presence of environmental noise or the quality of recording, are equalized within the sample set. This process can be performed on a global scale (global-CMVN), but this assumes channel variability to be constant within the same utterance, which may not be the case. To account for this consideration, local-CMVN can be performed, where the same process of mean subtraction and variance normalization is performed across a sliding time window, with sufficient length to perform good normalization, but short enough to capture fluctuations in the time-varying properties of the considered utterance.

Feature Warping (FW) modifies the short term feature distribution to resemble a desired reference distribution, usually Gaussian, by adapting the cumulative distribution function of short term feature windows. This process is more computationally complex than CMVN but generally improves accuracy.

Relative Spectral filtering (RASTA) performs filtering in the time domain of each feature (the evolution with time of a feature) using a bandpass filter. This allows for elimination or suppression of fluctuations not characteristic to speech, such as very high frequency changes or low frequency components due to the presence of noise.

IV. Speaker Modeling

After extracting features from utterances of known speakers, a mathematical model is derived from these, describing the feature properties of a given speaker. This process can be thought of as classifier selection and training with labeled training data. However, due to the natural variability speech features can present over the course of a single utterance, direct application of most predictive classifiers can lead to poor results. Furthermore, as described above, many recognition tasks require the use of a universal background model to evaluate the identity of a subject not present in the training group; many classical classifiers do not allow for such formulation.

As such, it is necessary to create mathematical models that describe in general the feature characteristics of a selected speaker, given its training utterances, which can also compensate for the natural variability present in intra-speaker and intra-utterance data [1]. In this Section, a brief introduction is made to the classical speaker models, as well as the progressive improvement of them and new techniques developed over the years, presented in chronological order.

A. Gaussian Mixture Models

Gaussian Mixture Models (GMM) have shown to be the standard of speaker modeling for speaker recognition [6], serving as a baseline of performance for new methods.

GMMs are probabilistic models that assume that data points are generated from a finite sum of Gaussian distributions. The general formulation for these models is presented in Equation 3, representing the probability density function for a GMM model λ [9]:

$$p\left(\mathbf{x}|\lambda\right) = \sum_{k=1}^{C} \omega_k \cdot \mathcal{N}(\mathbf{x}|\mu_\mathbf{k}, \mathbf{\Sigma_k}) \tag{3}$$

where $\mathbf{x}$ is a F-dimensional feature vector $[F \times 1]$, C the total number of components of the GMM (total number of summed multivariate Gaussian distributions), ω_k the mixing weight (or prior probability) of the k-th Gaussian component $\mathcal{N}(\mathbf{x}|\mu_\mathbf{k}, \mathbf{\Sigma_k})$, with means μ_k $[F \times 1]$ and covariance matrix Σ_k $[F \times F]$. The mixing weights ω_k must satisfy $\sum_{k=1}^{C} \omega_k = 1$.

The parameters of a GMM can be collectively described as $\lambda = \{\omega_k, \mu_\mathbf{k}, \mathbf{\Sigma_k}\}_{k=1}^{C}$. For computational reasons, the calculated covariance matrices are diagonal (variance vectors), having all other entries as zero. This restricts the principal axes of the distributions to the direction of the feature axes, simplifying calculations without harming the overall descriptive and predictive capabilities of the model [9].

Considering a speech utterance consisting of T features vectors $X = \{\mathbf{x_1}, ..., \mathbf{x_T}\}$, the probability of observing this sequence given the GMM λ is described in Equation 4. Due to added simplicity in calculation, the log-probability (or log-likelihood) of an observation is more frequently used.

$$\begin{aligned} p\left(X|\lambda\right) &= \prod_{t=1}^{T} p(\mathbf{x_t}|\lambda) \\ \Leftrightarrow \log p\left(X|\lambda\right) &= \sum_{t=1}^{T} \log p(\mathbf{x_t}|\lambda) \end{aligned} \tag{4}$$

GMMs are usually trained using the Expectation-Maximization (EM) algorithm: GMM parameters are randomly initiated (or learned using k-means algorithm) and the probabilities of each feature vector being generated by each Gaussian component calculated (expectation or E step). The GMM parameters are then adjusted to maximize the global sum of log probabilities (maximization or M step), and the process is repeated until convergence or until a predetermined umber of iterations.

1) UBM and Model Adaptation: For speech applications, it is frequently required to create a model that translates the speaker-independent variability, called a world model or a universal background model. The UBM allows for calculation of likelihoods of a certain utterance being produced by a considered speaker in the training set of the model: if an utterance is from someone outside the considered set, it will present higher score for the UBM than for any other speaker model.

In addition, the creation of speaker-specific models from the adaptation of UBMs can be a beneficial factor, since it can help to reduce variability due to intra-speaker or inter-channel variability [1], compared with speaker-specific modeling.

Allying this to the Gaussian Mixture Model framework, one can construct a GMM-UBM framework: training of the UBM is performed using the complete set of training data from all speakers, for a limited number of iterations (usually 10 to 20). UBMs are usually modeled with 2048 Gaussian components, although some authors subdivide into two UBM with 1024 components, one for each speaker gender [9].

The trained UBM is then adapted during speaker enrollment by the Maximum A Posteriori (MAP) algorithm (or Bayesian Learning), using statistics obtained with speaker-specific data. This process is illustrated in Figure 3.

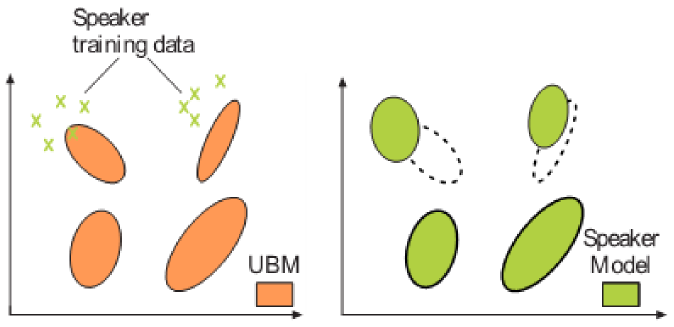

Fig. 3. MAP adaptation of a UBM to create a speaker model [9]. On the left, a 4 component, 2 dimensional UBM is illustrated, together with speaker-specific training data. On the right the resulting adapted speaker dependent model. Note that only components that can relevantly model the speaker data are tweaked.

MAP speaker adaptation is performed as follows [9]: given a trained UBM and a set of speaker training data $X = \{\mathbf{x_1}, ..., \mathbf{x_T}\}$, the probabilistic alignment of each feature vector with respect to the UBM is calculated, according to Equation 5:

$$Pr\left(i|\mathbf{x_t}\right) = \frac{\omega_i p_i(\mathbf{x_t})}{\sum_{k=1}^{C} \omega_k p_k(\mathbf{x_t})} \tag{5}$$

From (5), sufficient statistics for MAP adaptation for each Gaussian component i are calculated according to Equations 6 to 8:

$$N_i = \sum_{t=1}^{T} Pr(i|\mathbf{x_t}) \tag{6}$$

$$F_i(\mathbf{x}) = \frac{1}{N_i} \sum_{t=1}^{T} Pr(i|\mathbf{x_t})\mathbf{x_t} \tag{7}$$

$$S_i(\mathbf{x}) = \frac{1}{N_i} \sum_{t=1}^{T} Pr(i|\mathbf{x_t}) \cdot \mathrm{diag}(\mathbf{x_t} \cdot \mathbf{x_t}') \tag{8}$$

These statistics are then applied to the weight, mean and covariance of each component of the UBM to create a speaker adapted model with adjusted weights $\hat{\omega}$, adjusted means $\hat{\mu}$ and adjusted variances $\hat{\sigma^2}$, following Equations 9 to 11:

$$\hat{\omega}_i = \gamma \left[\frac{\alpha_i^\omega n_i}{T} + (1 - \alpha_i^\omega)\omega_i \right] \tag{9}$$

$$\hat{\mu_\mathbf{i}} = \alpha_i^\mu F_i(\mathbf{x}) + (1 - \alpha_i^\mu)\mu_\mathbf{i} \tag{10}$$

$$\hat{\sigma_\mathbf{i}^\mathbf{2}} = \alpha_i^\sigma S_i(\mathbf{x}) + (1 - \alpha_i^\sigma)(\sigma_\mathbf{i}^\mathbf{2} + \mu_\mathbf{i}^\mathbf{2}) - \hat{\mu_\mathbf{i}}^\mathbf{2} \tag{11}$$

γ is a scale factor, ensuring that $\sum_{k=1}^{C} \hat{\omega_i} = 1$ remains true. $\alpha_i^\omega, \alpha_i^\mu, \alpha_i^\sigma$ are parameter-specific, data-dependent terms that modulate the strength of the MAP adaptation for the weights, means, and covariances, respectively, as defined in Equation 12:

$$\alpha_i^\rho = \frac{N_i}{N_i + r^\rho}, \rho \in \{\omega, \mu, \sigma\} \tag{12}$$

r^ρ is a fixed relevance factor for each parameter. All r are set to 16 in normal conditions, with the overall result being independent of these relevance factors, within the range 8-20. By being data-dependent, the relevance factors allow the adaptation of only the most meaningful Gaussian components to model the speaker data, since low probability components will have $\alpha_i \to 0$. Furthermore, quicker computation times can be achieved by setting some α^ρ to 0, effectively ignoring some parameters in the adaptation step. For instance, one could set $\alpha^\sigma = \alpha^\mu = 0$, thus only adapting the means of the Gaussian components of the UBM: this adaptation reveals itself as the best performing one from all possible combinations of parameters to adapt [9].

2) Likelihood Ratio: Considering single-speaker detection as a simple hypothesis test where an utterance X is hypothesized to be from speaker S, a decision system can be formulated using Equation 13, with H_0 being "X is from speaker S" and H_1 being "X is not from speaker S":

$$\frac{p(X|H_0)}{p(X|H_1)} \begin{cases} \geq \theta, \text{accept } H_0 \\ < \theta, \text{reject } H_0 \end{cases} \tag{13}$$

with threshold θ. Considering (4) and $p(X|H_1)$ as being modeled by the UBM with $p(X|\lambda_{UBM})$, a log-likelihood ratio score can be established as Equation 14:

$$\Lambda(X|S) = \log p(X|\lambda_S) - \log P(X|\lambda_{UBM}) \tag{14}$$

The use of (14) allows for comparable measure of log-likelihood scores across all considered speakers, matching score ranges.

B. Supervectors

One of the challenges in speaker recognition is the variability in length of feature vector sets corresponding to spoken utterances, since these can be of varying time spans. Adequate representation of an utterance with a single vector is necessary to overcome this difficulty. A possible solution would be to time-average all features, but this leads to poor accuracy results.

In [13] a novel approach to this issue was demonstrated: using GMM *supervectors*. Supervectors are a representation method that translate full speaker utterances into a single, highly dimensional vector, that can be compared to similar UBM supervectors through the use of classifiers such as SVM.

[13] suggests the creation of GMM supervectors by concatenation on the Gaussian component means: considering a GMM of C Gaussian components and F-dimensional feature vectors, the GMM supervector is the $CF \times 1$ resulting from the concatenation of the means vectors for each component C. The proposed verification system creates a UBM from all training speech utterances and generates a utterance-based GMM by MAP adaptation. The created models are then supervectorized and compared using SVM with an adequate comparison kernel.

Supervectors allied with SVM show a relevant improvement over the previously described GMM-UBM framework, with an achieved equal error-rate (EER) of 3.77%, compared with 5.68% with the latter, evaluated on the 2005 NIST speaker recognition corpus [14].

C. Joint Factor Analysis

The supervector approach described above can translate speaker-dependent models through MAP adaptation of the UBM followed by supervectorization. However, for a given speaker, utterance supervectors may not be similar, especially if these are collected with different environmental and handset conditions. As such, channel compensation is necessary to compare data obtained from different channels, which implies the development of an explicit model of channel variability.

[7] introduces the concept of Joint Factor Analysis (JFA), where channel and speaker-dependent supervectors M can be decomposed into two independent supervectors: $M = s + c$, with s as the speaker independent supervector and c the channel independent supervector.

c can be decomposed further into $c = Ux$, with U being a rectangular matrix whose columns are the *eigenchannels* of the dataset and x the *channel factors* of a given speech utterance.

The speaker supervector s is further decomposed into $s = m + Vy + Dz$, with m as the UBM supervector, V a low rank rectangular matrix whose columns are referred as *eigenvoices*, and y as the *speaker factors*. This can be seen as an extension of the classical MAP adaptation, which corresponds to $s = m + Dz$.

The general model for JFA is described in Equation 15:

$$M = m + Vy + Ux + Dz \tag{15}$$

U, V and D are the JFA model hyperparameters, estimated beforehand on large training datasets. For a given training utterance, estimation of the latent factors x, y and z is performed, considering s as the speaker model, accomplishing channel compensation. These supervectors can be applied as a probabilistic model similar to those previously discussed here, or can be used in conjunction with SVM, as in [15].

JFA models achieve EER of 3.01% on all trials of the 2006 NIST SRE, where JFA allied with SVM obtain comparable results, with EER of 4.92% in the same conditions [15].

D. i-Vectors

JFA systems have shown themselves to be powerful basis for speaker modeling in speaker recognition applications, allowing the removal of channel related variations. These systems can also be thought of feature extractors for SVM, as seen on [15], with the speakers factors used as features for SVM classifiers.

However, as described in [16], although supposed to only model channel effects, channel factors were shown to possess speaker-dependent information. Classical JFA defines two variability spaces: the speaker space, defined by the eigenvoice matrix V and the channel space, defined by the eigenchannel matrix U, as described in (15). In [16], both the speaker and channel factors were mapped to a single space defined as the *total variability space* T, as described in Equation 16:

$$M = m + T\omega \tag{16}$$

with T as a rectangular, low rank matrix and ω as the total factor vector, also called the identity vector, or simply *i-vector*.

The training of T is similar to the training of the V matrix in JFA (described in [7]), but assuming each utterance was obtained from different speakers. The process of transformation of GMM supervectors to i-vectors can be seen as a specific model of Principal Component Analysis (PCA), where large, highly dimensional GMM supervectors are dimensionally reduced to low dimensional i-vectors. Further processing of i-vectors can be performed, reducing the dimensionality of these, using for instance PCA or Linear Discriminant Analysis (LDA).

Processed i-vectors can then be used as utterance feature vectors for SVM classification. Kernel adaptation can be performed, using for instance cosine distance transform kernels form i-vector comparison. This approach achieves the best results using the NIST 2006 and 2008 SRE datasets, compared with JFA, with EERs of 2.72% and 6.09%, respectively [16].

E. Deep Learning Approaches

Recent research in speaker recognition systems has mainly implemented Deep Learning methods improving over the previous i-vector approaches. The use of Deep Neural Networks (DNN) has shown to be an incredibly powerful tool for Machine Learning, with very interesting applications in speaker and speech recognition [1], [2].

The architecture of a DNN can be designed to allow for the extraction of low rank features that best describe the speaker utterances, with high discriminative power, that can then be applied to other classification methods.

[3] applies DNNs as direct score estimators for speaker verification, using DNNs with temporal pooling layers, allowing for time-varying utterances to be considered. These networks are trained with extracted MFCC features, with an objective function that maximizes probability in a same-speaker scenario and minimizes for different speakers. The system combination of both i-vector and DNN methods show large improvements of EER (5.3%) compared with the standalone i-vector system (8.5%), for a dataset of English phone calls of 102 thousand different speakers.

Recently [17] show promising results for the more complex task of speaker diarization (identifying segments of audio where a certain speaker is active, in a multi-speaker context) using *d-vectors*, which are speaker embeddings derived from neural networks such as Long Short-Term Memory networks (LSTM). Speaker diarization tasks with d-vectors show an improved Diarization Error Rate (DER) of 12.30%, compared with similar i-vector approaches, with DER of 21.12% for the same conditions, on the 2003 NIST Rich Transcription dataset.

The authors of [18] apply a deep Convolutional Neural Network (CNN) classifier for speaker recognition and verification on a large speaker dataset collected from Youtube videos on unconstrained settings. MFCCs are not used as the basis feature vectors; instead, short-term spectrograms are used as image inputs for the network. For the considered database, this approach achieved EER of 7.1%, compared with 8.8% obtained with an i-vector approach and 15.0% with the GMM-UBM framework. These results show the flexibility of the use of Deep Learning, where simple raw and unprocessed inputs of speaker utterances can achieve high levels of recognition with the use of well trained deep networks.

Deep Learning approaches show a very promising outcome for the future for speaker and speech recognition. For the latter task, most speech recognition algorithms present in current day smartphones utilize the power of DNNs and online processing to perform speech-to-text activities [19].

However, Deep Learning may not always be the most adequate tool for certain speaker recognition tasks, since a very large dataset with hundreds of thousands of speakers and thousands of hours of speech training data are required to achieve good results, and are very computationally expensive. Simpler models such as GMM-UBM can perform better on more sparse datasets, and do not require very long training time to achieve good results. The extensive hours of training data required for Deep Learning systems training were previously a major deterring factor, since not many sufficiently large datasets were available. However, more and more open-source and free large datasets are available, such as [18], [20], [21].

V. Methods

On this work, a speaker identification system is implemented in Python 3.6. This section describes the overall pipeline of the system, which is illustrated in Figure 4.

A. The VoxCeleb Dataset

The VoxCeleb Dataset [18] is a freely available speaker recognition dataset, with utterances extracted from Youtube videos of 1251 celebrities/Person of Interest (POI). Each POI has an average of 116 utterances, each being on average of 8 seconds of duration. Utterance times are variable, with a maximum of 145 seconds and a minimum of 4 seconds. Several different utterances can be extracted from the same origin video.

The considered videos contain unconstrained environments, unlike other previous speaker recognition datasets, where the conditions are strictly controlled (e.g. the TIMIT dataset [22]). Videos include multiple noise environments, with background noise variability, such as quiet individual interviews, to public speeches and red carpet interviews. All are degraded with real life noise, such as laughter, overlapping speech, among other effects, and the range of recording equipment quality is varied and not constant.

The complete VoxCeleb dataset is downloaded, consisting of 1251 folders, one for each speaker, each containing the corresponding utterances in *.wav* format. The file filenames are the Youtube video ID of the original utterance video, with an appended counter for distinguishing different utterances from the same video.

B. Feature Extraction and Normalization

For each *.wav* utterance file, MFCC feature vectors are extracted, with the use of the SpeechPy module for Python [23]. This module implements a MFCC extractor, following the steps described in Section III-A.

The audio file is read in Python and converted to an audio vector, with Fs sampling frequency. The audio vector is first pre-emphasized with a high-pass filter, which removes the DC component of the signals and increases the magnitude of higher frequencies, compared with lower ones. This is

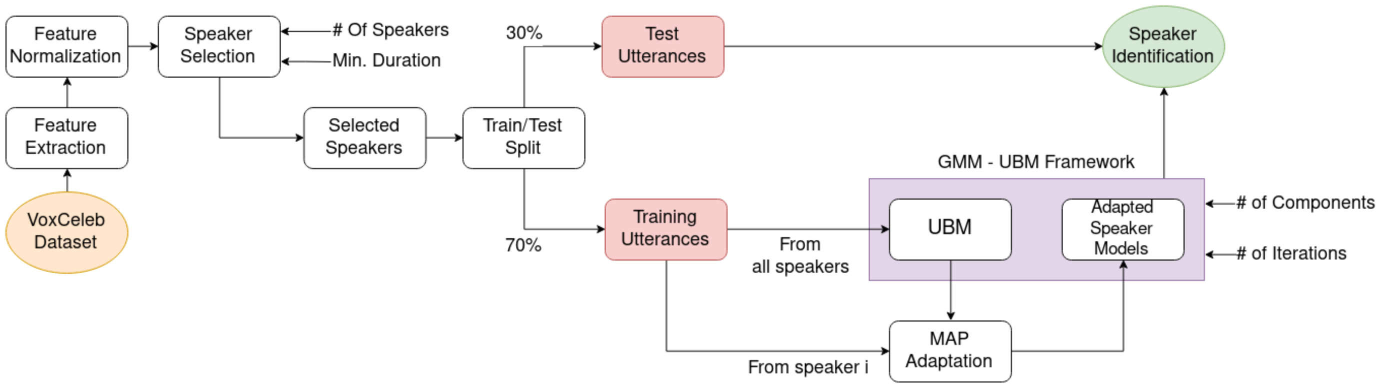

Fig. 4. Implemented Speaker Identification System Pipeline. Note that train/test splitting is done on a speaker level, i.e., for each speaker, approximately 30% of utterances are kept for testing. The UBM is trained with training utterances from all speakers, whereas MAP adaptation is performed for each speaker individually, using all training utterances from speaker i to generate the adapted speaker i model from UBM parameter adaptation.

desirable since a considerable amount of speech information resides on high band frequencies.

MFCC features are extracted for each utterance, with non-overlapping window frames of 0.02s, FFT of length 512, 40-channel mel filter banks and considering the first 13 coefficients. ΔMFCC and Δ^2MFCC are also extracted for each window frame (considering two frames for derivative calculation) and appended to the feature vector, resulting in 39 features per frame.

Global Cepstral Mean and Variance Normalization is performed on an utterance level, as discussed in Section III-B.

Features are extracted for all utterances of all speakers. The extraction and normalization steps are performed beforehand, with features saved to *.pickle* files for each speaker. Each compressed file has a list of extracted feature vectors for individual utterances, i.e., it is possible to separate utterances on future steps.

C. Speaker Selection and Train/Test Split

Do to the enormous size of the VoxCeleb Dataset and to available computational limitations, a subset of speakers was considered for the speaker verification system.

The total number of minutes of speech for each speaker on the dataset is calculated. To avoid random selection of speakers with wildly different total speech minutes, a minimum threshold was established for subset creation; in this case, 5 minutes. N speakers are then selected starting on the minimum duration threshold and introducing the next speaker with the lowest total duration above the duration of the last speaker used, i.e., the N speakers with the closest total durations above 5 minutes are selected. In this case, $N = 10$, which allows for a sufficiently large variety of speakers to exemplify the capabilities of the identification system, without too much class imbalance. Furthermore, the number of speakers of each gender is approximately balanced, with 6 female and 4 male speakers.

After speaker selection, the data was split into a training set of utterances (to model the GMM-UBM framework) and a test set (to evaluate performance of the trained model). Data was split using a 70/30 proportion: for each speaker, approximately 70% of utterances are kept for training, while the remaining ones are stored for testing.

Statistics related to the selected speakers are presented in Table I.

TABLE I
STATISTICS FOR TRAIN AND TEST DATA FOR SPEAKER IDENTIFICATION TASK

	Avg. # of utt.	Avg utt. length (s)	Min. utt. length (s)	Max. utt. length (s)
Train	37	12.4	7.9	35.3
Test	13	12.9	7.9	40.2

D. UBM Modeling

After splitting the utterances for train and test, the train set is used to create a UBM, which will be the foundation of the GMM-UBM framework. The UBM is modeled as a Gaussian Mixture Model, with C components and trained for I iterations. To evaluate the influence of these parameters, different models were generated for each possible combination and compared against the test utterances: $C = \{1, 4, 16, 64, 256\}$ and $I = \{5, 10, 20\}$, giving a total of 12 possible combinations.

The GMM implementation is performed using *GaussianMixtureModel* [24] from *scikit-learn* [25]. Covariance matrices are considered diagonal for computational efficiency.

E. Speaker-GMM Adaptation

After training of the UBM, speaker adapted GMM are constructed via MAP adaptation, as described previously in Section IV-A.

For each speaker, the probabilistic alignment of each Gaussian component of the UBM is calculated as in (5). From this, sufficient statistics are calculated with (6-8) and model parameters adjusted according to (9-11). All parameters are subjected to MAP adaptation, with $r = 16$ in (12) for all. For each speaker, a new instance of *GaussianMixtureModel* is initialized and its parameters are replaced by the adapted parameters calculated using the respective speaker training utterances.

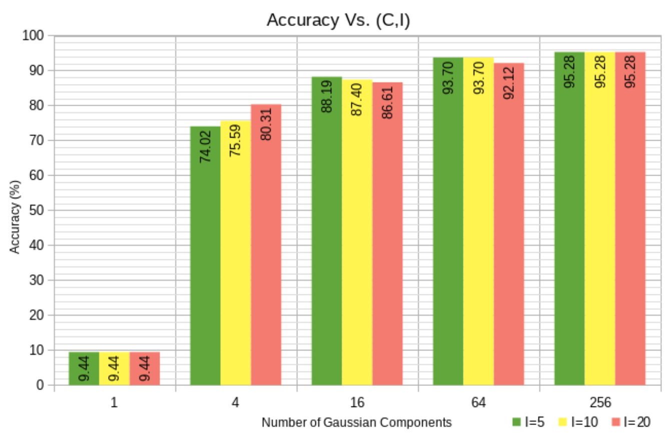

Fig. 5. Accuracy results obtained for multi-class classification of test utterances.

VI. Results

After UBM modeling and speaker model adaptation, test utterances scores for each speaker in the model are calculated according to (14). For a given test utterance T, a score vector $\Lambda(T) = \{\Lambda(T|S_1), ..., \Lambda(T|S_N)\}$ is obtained for N speakers. The predicted speaker S_p will have $p = \max_i \{\Lambda(T|S_i)\}_{i=1}^{N}$, i.e., S_p will be the speaker corresponding to the highest log-likelihood score out of the 10 considered speakers in the model.

Predicted labels and ground truth labels for all test utterances are compared and relevant metrics are calculated. Results are shown in Figure 5, for each pair of number of components and number of iterations (C, I) of the GMM-UBM framework. Detailed results are presented in Table II.

To evaluate the performance of the GMM-UBM framework across a wide range of operating points, Detection Error Trade-off (DET) curves [26] plotted for each pair (C, I) considered. DET curves are similar to Receiver Operating Characteristic (ROC) curves, but are shown on log-log axis for better discrimination of systems with good results, since these tend to have curves that bundle together in the ROC plot. DET curves are plotted with the x-axis as the False Alarm Probability in % (also known as False Positive Rate, or FPR) and the y-a xis as the Miss Probability in % (also known as False Negative Rate, or FNR): for a range of considered thresholds, each operating threshold for the system will correspond to a point (FPR,FNR) on the DET curve. As the performance of a system improves, the DET curve moves closer to the origin.

Since the considered system performs multi-class classification and DET curves are designed for binary output systems, DET curves are obtained by averaging all speaker-specific DET curves for a given pair (C, I).

An important point on DET curves is the Equal Error Rate (EER) point, where $FPR = FNR$. This value is frequently used as a metric for measuring speaker verification system, along with the NIST C_{det}^{min} metric [1]. EER values for considered (C, I) pairs are reported on Table II.

The obtained DET curves are shown on Figure 6.

VII. Discussion Of Results

The obtained accuracy results for a 10 speaker identification system show the powerful capabilities of the GMM-UBM framework for this task. Comparing with the obtained results of [18] with a CNN framework, similar accuracy scores are obtained. However, the CNN system is trained for the full 1251 speaker set of VoxCeleb, so direct comparison of accuracy scores is unfair. Nonetheless, the achieved score of 96% for correct test utterance speaker identification is an exciting result.

Observing the results of Figure 5, it is evident to see that increase in the number of considered Gaussian components for modeling leads to an increase in performance of the system. Small numbers of C are not enough to model the variability of the training data. However, the increase in performance reaches a plateau for large numbers of C, which do not lead to an automatic increase in overall accuracy. This can be due to the number of components being sufficient to model speaker variability, with further increase in C resulting in the addition of redundant components that do not contribute to the modeling capabilities of the system, with low probabilistic alignment for all speakers.

Similar GMM-UBM identification/verification systems [9], [18] usually develop frameworks with $C = 1024$, or two separate UBM for each gender, with 1024 components each. This number of components is not tested on these experimental conditions due to computational restrictions. However, due to the much smaller subset of considered speakers, the use of such a large number of Gaussian components would probably be fruitless, since a small number of components is sufficient to adequately model the variability in the speaker subset: added components would probably be redundant or non-informative, not necessary to model the small-scale variability of the 10-speaker subset.

Increasing the number of iterations used for UBM training only caused relevant increase in accuracy for the 4-component model. For 1 Gaussian, only 1 iteration was performed, since

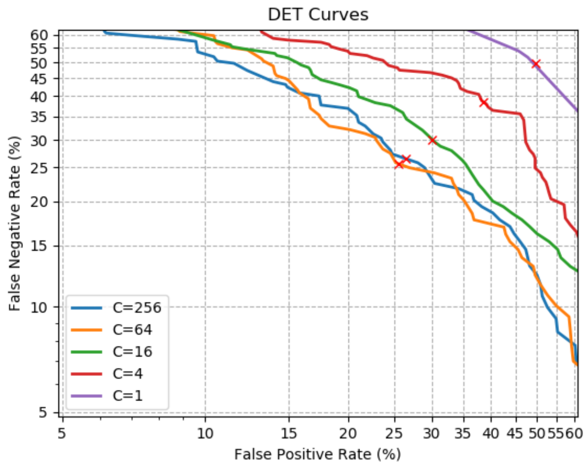

Fig. 6. DET curves for GMM-UBM framework parameter pairs $(C, I = 10)$. The EER point is indicated by a red cross on each curve.

the model converged after 1 iteration. For models with more components, the number of iterations does not bring significant changes, probably because the addition of more components is enough to model speaker variability, instead of utilizing fully-converged models. [18] performs 10 iterations for UBM training, which shows to be efficient in this case as well.

The analysis of Figure 6 shows increasing the number of components for the GMM-UBM framework increases the performance shown by the system's DET curve, bringing the curve closer to the origin (bottom left corner). However, comparing with similar systems in literature, as in [9], DET curves are obtained with EER of approximately 10%, comparing with the best obtained EER in the developed system of 26.49%, with other more sophisticated systems with even lower EER (some scores are referred in Section IV). This shows the developed system still needs improvement for better discernment of different speakers with higher confidence.

VIII. Conclusions and Future Work

The described GMM-UBM framework was applied to a 10-speaker subset of the large VoxCeleb dataset. Parameters of the model such as the number of Gaussian components and the number of iterations for training achieving high values of accuracy of speaker identification from test utterances (not used for model training), the highest being 96%. DET curves are plotted for varying thresholds, with a minimum EER of 26.49% achieved, which is underwhelming compared to literature.

The application of the GMM-UBM was shown to be a success when applied to the VoxCeleb 10-speaker subset. To obtain more comparable results to literature, GMM-UBM training must be performed on the full VoxCeleb dataset. This would also make more evident the modeling capabilities and detrimental factors of the use of this type of model.

Future work could be the application of more recent speaker modeling techniques, such as supervectors with SVM, and i-vectors, since these could bring a major improvement in the operating capabilities of the system, and would illustrate the evolution of speaker systems and their increasing performance. This has not been done on this work due to computational limitations, but is an excellent start for future development.

References

[1] J. H. Hansen and T. Hasan, "Speaker recognition by machines and humans: A tutorial review," *IEEE Signal Processing Magazine*, vol. 32, no. 6, pp. 74–99, 2015.

[2] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/6296526/

[3] D. Snyder, D. Garcia-Romero, D. Povey, and S. Khudanpur, "Deep Neural Network Embeddings for Text-Independent Speaker Verification," *Interspeech*, pp. 999–1003, 2017. [Online]. Available: https://pdfs.semanticscholar.org/3697/28d7576683a25de8890e4bc02fae6132fccb.pdf

[4] Jongseo Sohn, Nam Soo Kim, and Wonyong Sung, "A statistical model-based voice activity detection," *IEEE Signal Processing Letters*, vol. 6, no. 1, pp. 1–3, 1 1999. [Online]. Available: http://ieeexplore.ieee.org/document/736233/

[5] S. O. Sadjadi and J. H. L. Hansen, "Unsupervised Speech Activity Detection Using Voicing Measures and Perceptual Spectral Flux," *IEEE Signal Processing Letters*, vol. 20, no. 3, pp. 197–200, 3 2013. [Online]. Available: http://ieeexplore.ieee.org/document/6403507/

[6] T. Kinnunen and H. Li, "An overview of text-independent speaker recognition: From features to supervectors," *Speech Communication*, vol. 52, no. 1, pp. 12–40, 2010. [Online]. Available: http://dx.doi.org/10.1016/j.specom.2009.08.009

[7] P. Kenny, P. Ouellet, N. Dehak, V. Gupta, and P. Dumouchel, "A study of interspeaker variability in speaker verification," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 16, no. 5, pp. 980–988, 2008.

[8] F. J. D. Nolan, "The phonetic bases of speaker recognition," 1980. [Online]. Available: http://ethos.bl.uk/OrderDetails.do?uin=uk.bl.ethos.259225

[9] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted Gaussian mixture models," *Digital Signal Processing: A Review Journal*, vol. 10, no. 1, pp. 19–41, 2000.

[10] M. Sahidullah and G. Saha, "Design, analysis and experimental evaluation of block based transformation in MFCC computation for speaker recognition," *Speech Communication*, vol. 54, no. 4, pp. 543–565, 5 2012. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0167639311001622

[11] S. S. Stevens, J. Volkmann, and E. B. Newman, "A Scale for the Measurement of the Psychological Magnitude Pitch," *The Journal of the Acoustical Society of America*, vol. 8, no. 3, pp. 185–190, 1 1937. [Online]. Available: http://asa.scitation.org/doi/10.1121/1.1915893

[12] D. O'Shaughnessy, *Speech communication : human and machine*. Addison-Wesley Pub. Co, 1987. [Online]. Available: https://books.google.pt/books?ei=rhFfSpa-BJOCNsTT4IUG&id=mHFQAAAAMAAJ&dq=Speech+Communications:+Human+and+Machine&q=2595&redir_esc=y#search_anchor

[13] W. M. Campbell, D. E. Sturim, and D. A. Reynolds, "Support vector machines using GMM supervectors for speaker verification," *IEEE Signal Processing Letters*, vol. 13, no. 5, pp. 308–311, 2006.

[14] Linguistic Data Consortium. and National Institute of Standards and Technology (U.S.), *2005 NIST speaker recognition evaluation test data*. Linguistic Data Consortium, 2011. [Online]. Available: https://catalog.ldc.upenn.edu/LDC2011S04

[15] N. Dehak, P. Kenny, R. Dehak, O. Glembek, P. Dumouchel, L. Burget, V. Hubeika, and F. Castaldo, "Support vector machines and Joint Factor Analysis for speaker verification," in *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 4 2009, pp. 4237–4240. [Online]. Available: http://ieeexplore.ieee.org/document/4960564/

[16] P. Kenny, N. Dehak, P. Kenny, P. Dumouchel, and P. Ouellet, "Front-End Factor Analysis for Speaker Verification Front-End Factor Analysis For Speaker Verification," no. May 2017, pp. 1–12, 2011.

[17] Q. Wang, C. Downey, L. Wan, P. A. Mansfield, and I. L. Moreno, "Speaker Diarization with LSTM," 2017. [Online]. Available: http://arxiv.org/abs/1710.10468

[18] A. Nagraniy, J. S. Chungy, and A. Zisserman, "VoxCeleb: A large-scale speaker identification dataset," *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, vol. 2017-Augus, pp. 2616–2620, 2017.

[19] I. McGraw, R. Prabhavalkar, R. Alvarez, M. G. Arenas, K. Rao, D. Rybach, O. Alsharif, H. Sak, A. Gruenstein, F. Beaufays, and C. Parada, "Personalized Speech Recognition On Mobile Devices," 2016. [Online]. Available: https://ai.google/research/pubs/pub44631

[20] M. McLaren, L. Ferrer, D. Castan, and A. Lawson, "The speakers in the wild (SITW) speaker recognition database," *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, vol. 08-12-Sept, pp. 818–822, 2016.

[21] Mozilla, "Common Voice." [Online]. Available: https://voice.mozilla.org/en

[22] Linguistic Data Consortium, "TIMIT Acoustic-Phonetic Continuous Speech Corpus - Linguistic Data Consortium." [Online]. Available: https://catalog.ldc.upenn.edu/ldc93s1

[23] A. Torfi, "SpeechPy: Speech recognition and feature extraction," 8 2017. [Online]. Available: https://doi.org/10.5281/zenodo.840395

[24] "Gaussian mixture models — scikit-learn documentation." [Online]. Available: http://scikit-learn.org/stable/modules/mixture.html

[25] "scikit-learn: machine learning in Python." [Online]. Available: http://scikit-learn.org/stable/index.html

[26] A. Martin, G. Doddington#, T. Kamm+, M. Ordowski+, and M. Przybocki, "The DET Curve in Assessment of Detection Task Performance."

APPENDIX

TABLE II
FULL RESULTS TABLE FOR SPEAKER IDENTIFICATION SYSTEM WITH THE GMM-UBM FRAMEWORK

C	I	Accuracy (%)	Mean Precision (%)	Mean Recall (%)	Mean F1 (%)	EER (%)
1	5	**9.44**	**0.89**	**9.45**	**1.63**	49.73
	10	9.44	0.89	9.45	1.63	49.61
	20	9.44	0.89	9.45	1.63	**49.55**
4	5	73.23	77.28	73.23	73.44	39.77
	10	74.80	79.22	74.80	75.12	38.54
	20	**80.31**	**82.10**	**80.31**	**80.50**	**36.03**
16	5	85.83	86.39	85.83	85.90	30.22
	10	**87.40**	**88.55**	**87.40**	**87.41**	**30.08**
	20	87.40	87.91	87.40	87.39	30.43
64	5	93.70	94.01	93.70	93.72	26.75
	10	93.70	94.04	93.70	93.71	**25.61**
	20	**94.45**	**94.66**	**94.49**	**94.48**	27.41
256	5	**96.06**	**96.23**	**96.06**	**96.06**	26.69
	10	94.49	94.70	94.49	94.48	**26.49**
	20	95.28	94.44	95.28	95.27	27.15