A Text-Independent Speaker Recognition System

Catie Schwartz
Ph.D. Student, Applied Mathematics and Scientific Computing
Department of Mathematics
University of Maryland, College Park
schwa2cs AT math.umd.edu

Dr. Ramani Duraswaimi
Associate Professor, Department of Computer Science and
Department of the University of Maryland Institute of Advanced Computer Studies (UMIACS)
University of Maryland, College Park
ramani AT umiacs.umd.edu

Abstract

Speaker recognition is the computational task of validating a person's identity based on their voice. The two phases of a speaker recognition system are the enrollment phase where speech samples from the different speakers are turned into models and the verification phase where a sample of speech is tested to determine if it matches a proposed speaker. In a text-independent system, there are no constraints in the words or phrases used during verification. Numerous approaches have been studied to make text-independent speaker recognition systems accurate with very short speech samples and robust against both channel variability (differences due to the medium used to record the speech) and speaker dependent variability (such as health or mood of the speaker). A text-independent speaker recognition system using Gaussian mixture models and factor analysis techniques will be implemented in Matlab and tested against the NIST SRE databases for validation.

1 Project Background/Introduction

Humans have the innate ability to recognize familiar voices within seconds of hearing a person speak. How do we teach a machine to do the same? Research in speaker recognition/verification, the computational task of validating a person's identity based on their voice, began in 1960 with a model based on the analysis of x-rays of individuals making specific phonemic sounds [1]. With the advancements in technology over the past 50 years, robust and highly accurate systems have been developed with applications in automatic password reset capabilities, forensics and home healthcare verification.

There are two phases in a speaker recognition system: an enrollment phase where speech samples from different speakers are turned into models and the verification phase where a sample of speech is tested to determine if it matches a proposed speaker, as displayed in Figure 1. It is assumed that each speech sample pertains to one speaker. A robust system would need to account for differences in the speech signals between the enrollment phase and the verification phase that are due to the channels used to record the speech (landline, mobile phone, handset recorder) and inconsistencies within a speaker (health, mood, effects of aging) which are referred to as channel variability and speaker dependent variability respectively. In text-dependent systems, the words or phrases used for verification are known beforehand and are fixed. In a text-independent system, there are no constraints on the words or phrases used during verification. This project will focus on text-independent speaker verification systems.

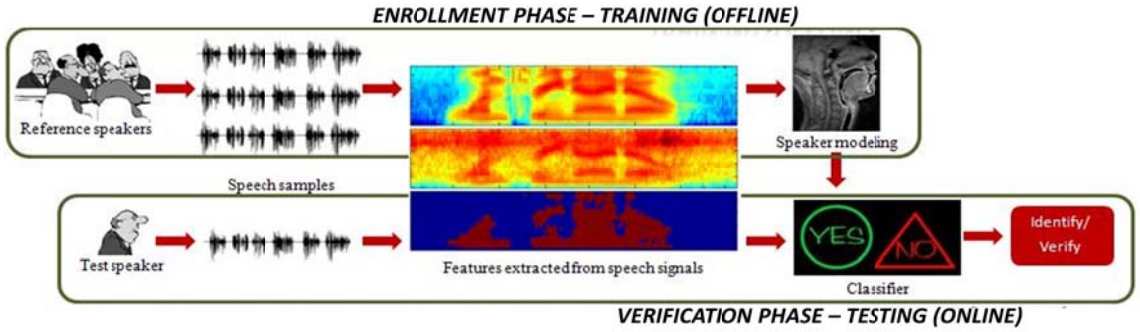

Figure 1: Speaker Recognition System (Courtesy of Balaji Srinivasan)

A variety of different features can be extracted from the speech samples, or utterances. Low level features relate to physiological aspects of a speaker such as the size of the vocal folds or length of vocal tract, prosodic and spectro-temporal features correspond to pitch, energy or rhythm of the speech, and high level features are behavioral characteristics such as accents or pronunciation. When extracting features, voice activity detectors (VADs) can be used to remove segments in an utterance where there is no speech. VADs can be energy based or based on periodicity. Many advanced systems account for multiple features and fusion is used to find the best overall match [2].

For text-independent speaker verification, the most popular modeling approaches are vector quantization (VQ), Gaussian mixture models (GMMs) and support vector machines (SVM). VQ is a technique that divides the features into clusters using a method such as K-means. GMM is an expansion of the VQ model, allowing each feature to have a nonzero probability of originating for each cluster [2]. A universal background model (UBM) representing an average speaker is often used in a GMM-based model. Adaptations of the UBM are used to characterize each of the individual speakers making the models robust even when the full phonemic space is not covered by the training data. SVM take labeled training data and seeks to find an optimized decision boundary between two classes which can be used to discriminate the different speakers.

Various techniques have been researched to assist in compensating for channel variability and speaker dependent variability, including speaker model synthesis (SMS) and feature mapping (FM). Most approaches require the speaker models to be organized into a high- and fixed-dimensional single vector called a supervector so that utterances with varying numbers of features can be represented in a general and compatible form. Popular methods that focus on compensating SVM supervectors include generalized-linear discriminant sequence (GLDS) kernel and maximum likelihood linear regression (MLLR) [2]. Factor analysis (FA) is a common generative modeling technique that is used on supervectors from GMMs to account for variability by learning low-dimensional subspaces. FA methods used in speaker verification include joint factor analysis (JFA) which model channel variability and speaker dependent variability separately, and total variability which model channel variability and speaker dependent variability in the same space. Normalization methods such as nuisance attribute projection (NAP), within-class covariance normalization (WCCN) and linear discriminant analysis (LDA) are also used for intersession variability compensation [2].

2 Approach

In this project, a simple text-independent speaker verification system will be implemented using mel-frequency cepstral coefficients (MFCCs) as the features used to create UBM-adapted GMMs. The mean components in the GMMs will be concatenated into supervectors. FA techniques will also be used on the GMM supervectors to learn the low-dimensional total variability space. i-vectors will be extracted from the total variability space which uniquely represent the same information contained in the GMM supervectors. LDA methods will be applied to the i-vectors corresponding to the total variability space to maximize inter-speaker variability and minimize speaker-dependent variability. Discrete cosine scoring (DCS) will be used for verifying if a test utterance matches a proposed speaker.

2.1 Feature Extraction

Low-level features called mel-frequency cepstral coefficients (MFCCs) will be extracted from the speech samples and used for creating the speaker models. The mel-frequency scale maps lower frequencies linearly and higher frequencies on a logarithmic scale in order to account for the widely-supported result that humans' can differentiate sound best at lower frequencies. Cepstral coefficients are created by taking a discrete cosine transform on the logarithm of the magnitude of the original spectrum. This step removes any relative timing, or phase, information between different frequencies and significantly alters the balance between intense and weak components [3]. MFCCs relate to the physiological aspects of a person such as the size of their vocal folds or length of their vocal tract and were first used starting in the 1980s [2]. They have been found to be fairly successful in speaker discrimination.

Given an utterance, it is first segmented using a 20 ms windowing process at a 10 ms frame rate. Since it is natural for people to pause while speaking, some of the frames will contain no useful information. A simple energy based voice activity detector (VAD) will be applied to the speech signals in order to locate the specific intervals that include speech segments [2]. Once speech segments are detected, MFCCs can be extracted from the signal. If the waveform is sampled at 16kHz, the 20 ms segment will

contain 320 samples. The Fast Fourier Transform (FFT) algorithm is applied to the speech sample. Then a mel-frequency filter bank is used to obtain an M-channel filterbank denoted as $Y(m), m = 1, \dots, M$. The MFCCs are found using the following formula:

$$c_n = \sum_{m=1}^{M} [\log Y(m)] cos\left[\frac{\pi n}{M}(m - \frac{1}{2})\right] \quad (1)$$

where n is the index of the cepstral coefficient. The 19 lowest DCT coefficients will be used for purposes of this project along plus 1 energy value. The complex process of obtaining MFCCs is shown in Figure 2.

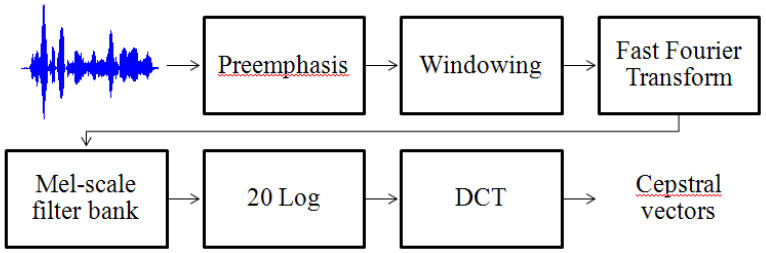

Figure 2: MFCC Feature Extraction Flow Chart (Courtesy of Balaji Srinivasan)

2.2 Gaussian Mixture Models using a Universal Background Model

Gaussian mixture models (GMMs) were first introduced as a method for speaker recognition in the early 1990s and have since become the *de facto* reference method [2, 4]. GMMs represent each speaker, s_i, by a finite mixture of multivariate Gaussians based on the *d*-dimensional feature vector $\boldsymbol{x}$:

$$p(\boldsymbol{x}|s_i) = \sum_{k=1}^{K} \pi_k N(\boldsymbol{x}|\boldsymbol{\mu}_k, \sum_k) \quad (2)$$

where K is the number of components, $\pi_k \geq 0$ represent the mixture weights that are constrained by $\sum_{k=1}^{K} \pi_k = 1$ and

$$N(\boldsymbol{x}|\mu_k, \sum_k) = (2\pi)^{-\frac{d}{2}} |\sum_k|^{-\frac{1}{2}} \exp\{-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_k)^T \sum_k{}^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_k)\} \quad (3)$$

where μ_k of dimension $1 \times d$ represents the mean value of mixture component *k* and $\sum_k$ of dimension $d \times d$ represents the covariance of mixture component *k* [4]. Given the sequence of T training vectors $X = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_T\}$ the GMM likelihood can be rewritten as

$$p(X|s) = \prod_{t=1}^{T} p(\boldsymbol{x}_t | s). \quad (4)$$

The values of $\pi, \mu, \sum$ representing each speaker, s, will be learned using maximum likelihood (ML) estimation techniques, which seek to find model parameters which maximize the likelihood of the GMM given the input training data, $\boldsymbol{x}$. Using full-covariance GMM normally requires a significant amount of training data and is very computationally intensive, therefore diagonal covariance matrices will be used.

A universal background model (UBM) or speaker-independent model is first created using speech samples from a large number, T, of speakers. The parameters of the UBM are found using an

expectation-maximization (EM) algorithm which iteratively refines a random initialization of GMM parameters to monotonically increase the likelihood of the estimated model based on the given feature vectors. In the estimation step, the Bayesian statistics are used to determine the probability of mixture component c:

$$\gamma_t(c) = p(c|\boldsymbol{x}_t, s) = \frac{\pi_c N(\boldsymbol{x}|\boldsymbol{\mu}_c, \sum_c)}{\sum_{k=1}^{K} \pi_k N(\boldsymbol{x}|\boldsymbol{\mu}_k, \sum_k)} \tag{5}$$

In the maximization step, the following are formulas are used which guarantee a monotonic increase in the model's likelihood value [5]:

Mixture weights:

$$\pi_c = \frac{1}{T} \sum_{t=1}^{T} \gamma_t(c) \tag{6}$$

Means:

$$\boldsymbol{\mu}_c = \frac{\sum_{t=1}^{T} \gamma_t(c) \boldsymbol{x}_t}{\sum_{t=1}^{T} \gamma_t(c)} \tag{7}$$

Variances:

$$\sigma_c = \frac{\sum_{t=1}^{T} \gamma_t(c) \boldsymbol{x}_t^2}{\sum_{t=1}^{T} \gamma_t(c)} - \boldsymbol{\mu}_c^2 \tag{8}$$

The expectation step and the maximization step are iterative. To determine the best stopping criteria, a maximum number of iterations will be used and changes to the parameters will be analyzed. Based on the newly found GMM-UBM components, $\pi^{UBM}, \mu^{UBM}, \sum^{UBM}$, a Bayesian adaptation technique is used to determine the components of the GMM for each individual speaker as displayed in Figure 3. This approach utilizes prior knowledge of what speech in general is like and uses the adapted model as the speaker model.

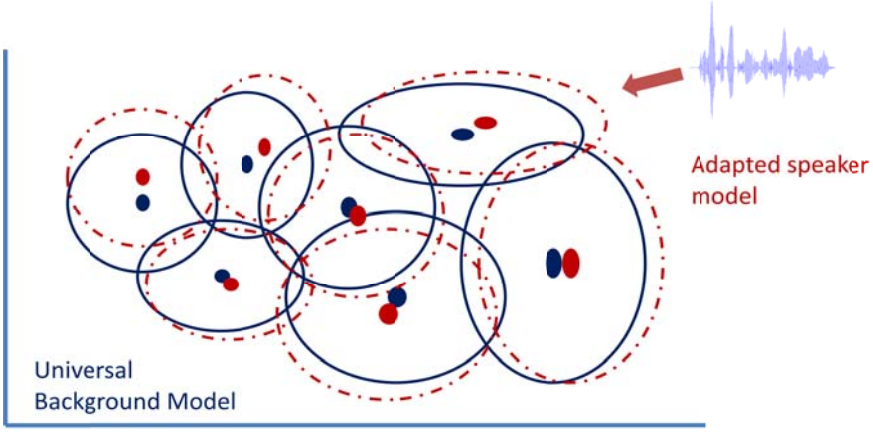

Figure 3: Maximum a posteriori (MAP) algorithm used to adapt the UBM (Courtesy of Balaji Srinivasan)

The first steps of the MAP algorithm are the same as the EM algorithm, that is, the values of π_c, μ_c are found using Bayesian statistics and ML estimations as in (6) and (7). Covariance weights will not be modified because of limited data to enable complete adaptation. Once these parameters are found, they are used to update the old UBM parameters π_c^{UBM}, $\boldsymbol{\mu}_c^{UBM}$ for mixture component c to create the adapted parameters for mixture component c with the equations:

$$\hat{\pi}_c = [\alpha_c^w \pi_c + (1 - \alpha_c^w)\pi_c^{UBM}]\gamma \tag{9}$$

and

$$\hat{\boldsymbol{\mu}}_c = \alpha_c^m \boldsymbol{\mu}_c + (1 - \alpha_c^m)\boldsymbol{\mu}_c^{UBM} \tag{10}$$

where $\alpha_c^\rho, \rho\epsilon\{w, m\}$ represent the adaptation coefficients controlling the balance between the old and the new estimates for the weights and means, respectively and the scale factor, γ, is computed to ensure that weights sum to unity [4]. The values of α_i^ρ are defined as

$$\alpha_c^\rho = \frac{\sum_{t=1}^{T} \gamma_t(c)}{\sum_{t=1}^{T} \gamma_t(c) + r^\rho} \tag{11}$$

where r^ρ is a fixed relevance factor for parameter ρ. For this project, $r^\rho = 16$ will be used for both relevance factors since experimental results have found performance to be rather insensitive to values in the range of 8-20 [4]. If it is decided to only adjust the mean coefficients, r^w will be set to 0. Using data-dependent adaptation coefficients enables de-emphasis on new parameters when a mixture component has a low probabilistic count with more emphasis on the old parameters. If a mixture component has high probabilistic counts, more emphasis can be placed on the new parameters. Having the ability to adjust the adaptation coefficients based on the data leads to robustness against limited training data.

The mean components of the GMMs for each speaker can be concatenated into a high- and fixed-dimensional single vector of dimension $Kd \times 1$ where K is the number of Gaussian centers and d is the number of features. The vector is called a supervector and is illustrated in Figure 4. Supervectors are useful because utterances with varying numbers of features can be represented in a general and compatible form [2]. FA techniques will use the GMM supervectors as described in the next section.

Figure 4: Creation of supervectors from GMMs (Courtesy of Balaji Srinivasan)

2.3 Factor Analysis

Factor analysis is a statistical method used to describe variability among observed variables in terms of potentially lower number of unobserved variables called factors [6]. This method can be used to separate variability due to differences in channels or other nuisances from variability inherently within speakers as illustrated in Figure 5.

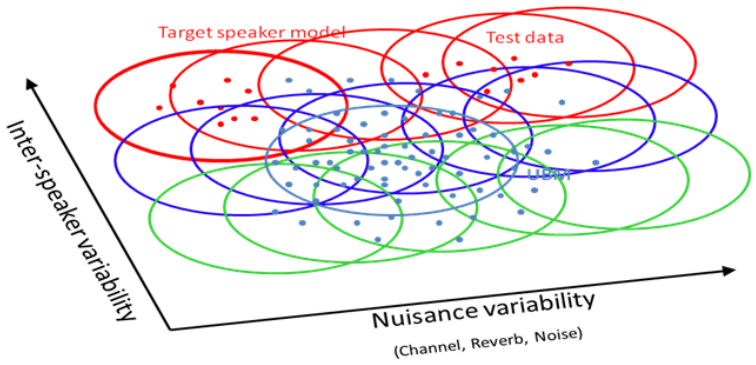

Figure 5: Inter-speaker variability versus nuiance variability. (Courtesy of Balaji Srinivasan)

The initial paradigm that incorporated factor analysis techniques looked into modeling channel-dependent variability explicitly different than speaker-dependent variability. This technique is called Joint Factor Analysis (JFA) and it separated the supervector of a speaker model μ_i into a speaker supervector s and a channel supervector c:

$$\mu_i = s + c \tag{12}$$

where s and c are normally distributed. The idea is to get both s and c in low-dimensional spaces, which is completed for the speaker supervector s by decomposing the supervector into speaker factors and residual factors:

$$s = m + Vy_i + Dz_i \tag{13}$$

In this equation, m is the speaker- and channel-independent supervector (UBM), V is a rectangular matrix of low rank, D is a diagonal matrix, and y and z are independent random vectors with standard normal distributions. The channel supervector c can be rewritten as

$$c = Ux_i \tag{14}$$

where U is a rectangular matrix of low rank and x has a standard normal distribution. Combining the two equations shows that the speaker model μ_i can be decomposed into low dimension spaces.

$$\mu_i = m + Vy_i + Ux_i + Dz_i \tag{15}$$

Dehak et al. [7] found that the subspaces U and V are not completely independent; therefore proposed a combined "total variability" space that will be used in this project. The speaker model supervector, μ_i will be decomposed as shown in the following equation

$$\mu_i = m + Tw_i \tag{16}$$

where T is rectangular matrix of low-rank representing the total variability space and w_i has a standard normal distribution. w_i represents the total variability factors and are often called intermediate/identity vectors or i-vectors. Equation (16) implies that μ_i is normally distributed with mean vector m and covariance matrix TT^*.

The rank of T is set prior to training. The value 400 is normally used but a smaller number can be used given limited data. To train T, an algorithm using concepts of estimation-maximization (EM) is used. The method is very similar to a Probabilistic Principal Component Analysis (PPCA) approach [8] and is the same algorithm used to train the V matrix in JFA. The only difference between training the V matrix in JFA is that in JFA, all recordings of a given speaker are considered to belong to the same person. In the total variability space, all utterances produced by a given speaker are regarded as having been produced by different speakers [7].

First, the Baum-Welch statistics [8] are calculated for a given speaker s and acoustic features $x_1, x_2, \dots, x_T$ for each mixture component c using equation (5):

$$N_c(s) = \sum_{t=1}^{T} \gamma_t(c) \tag{17}$$

$$F_c(s) = \sum_{t=1}^{T} \gamma_t(c) x_t \tag{18}$$

$$S_c(s) = diag(\sum_{t=1}^{T} \gamma_t(c)\, x_t x_t^*) \tag{19}$$

where $N_c(s)$, $F_c(s)$ and $S_c(s)$ represent the 0th, 1st and 2nd order statistics respectively. The 1st and 2nd order Baum-Welsh statistics are then centralized:

$$\tilde{F}_c(s) = F_c(s) - N_c(s) m_c \tag{20}$$

$$\tilde{S}_c(s) = S_c(s)(diag(F_c(s) m_c^* + m_c F_c(s)^* - N_c(s) m_c m_c^*) \tag{21}$$

Several matrices and vectors are defined based on the Baum-Welsh statistics. Let $\mathrm{NN(s)}$ be the CFxCF diagonal matrix whose diagonal block are $N_c(s)I$ $(c = 1, \dots, C)$. Let $\mathrm{FF(s)}$ be the $\mathrm{CFx1}$ supervector obtained by concatenating $\tilde{F}_c(s)$ $(c = 1, \dots, C)$. Let $\mathrm{SS(s)}$ be the CFxCF diagonal matrix whose diagonal blocks are $\tilde{S}_c(s)$ $(c = 1, \dots, C)$.

An iterative method is now used to determine the matrix T as described in [8, 9]. The first step is to determine the posterior distribution of the variables $w(s)$ given T. For the first iteration, a random initialization can be used for T. For each speaker, the following equation is defined

$$l_T(s) = I + T^* \sum^{-1} NN(s) T. \tag{22}$$

This will result in the posterior distribution of $w(s)$ conditioned on the acoustic observations of the speaker to be Gaussian distributed with mean $l_T^{-1}(s)T^*\sum^{-1}\tilde{F}(s)$ and covariance matrix $l_T^{-1}(s)$ [10]. The maximum-likelihood re-estimation step requires accumulating statistics over all the training speaker:

$$N_c = \sum_s N_c(s) \quad (c = 1, \dots, C) \tag{23}$$

$$A_c = \sum_s N_c(s) l_T^{-1}(s) \quad (c = 1, \dots, C) \tag{24}$$

$$\mathbb{C} = \sum_s FF(s)(l_T^{-1}(s)T^*\sum^{-1}FF(s))^* \quad (c = 1, \dots, C) \tag{25}$$

Given these values, a new estimate of the total variability space can be computed

$$T = \begin{bmatrix} T_1 \\ \vdots \\ T_C \end{bmatrix} = \begin{bmatrix} A_1^{-1}\mathbb{C}_{\mathbf{1}} \\ \vdots \\ A_1^{-1}\mathbb{C}_{\boldsymbol{c}} \end{bmatrix} \tag{26}$$

where

$$\mathbb{C} = \begin{bmatrix} \mathbb{C}_{\mathbf{1}} \\ \vdots \\ \mathbb{C}_{\boldsymbol{c}} \end{bmatrix}. \tag{27}$$

Several iterations (approximately 20) will be competed to obtain the trained total variability space. Once the space is defined, i-vectors are extracted are extracted using the knowledge that from (22), the expected value of an acoustic feature $w(s)$ is $l_T^{-1}(s)T^*\sum^{-1}\tilde{F}(s)$.

2.3 Linear Discriminant Analysis

Another dimensionality reduction technique called linear discriminant analysis (LDA) will be used. Once the total variability space, T, and the i-vectors, w from equation (16) are learned, LDA can be used to project the i-vectors into a lower-dimensional space, ω using the following equation:

$$\omega = Aw \tag{28}$$

The matrix A is chosen such that within-speaker, or speaker-dependent, variability is minimized and inter-speaker variability is maximized within the space. The matrix can be found by solving the eigenvalue problem

$$J(A) = Tr\{s_w^{-1}s_B\} \tag{29}$$

where s_w represents the within-class covariance matrix and s_B represents the between class covariance matrix.

2.4 Classifiers

Two classifiers will be used for the accept/reject decision. A log-likelihood ratio test will be used based on the GMMs models and cosine distance scoring will be used on both the i-vectors and the intersession-compensated LDA vectors.

2.4.1 Log-likelihood ratio test

Given a GMM speaker model s_{hyp} and the GMM-UBM s_{UBM}, a log-likelihood ratio test can be applied on the extracted features of a test utterance, X using the following formula [4]:

$$\Lambda(X) = \log p\big(X|s_{hyp}\big) - \log p(X|s_{UBM}) \tag{30}$$

where $\Lambda(X) \geq \theta$ will lead to verification of the hypothesized speaker, s_{hyp} , and $\Lambda(X) < \theta$ will lead to rejection.

2.4.2 Discrete cosine score

The discrete cosine score (DCS) can be applied to both the i-vectors, w, and the intersession-compensated i-vectors using LDA, ω using the following equation [9]:

$$score(\omega_1, \omega_2) = \frac{\omega_1^* \omega_2}{\|\omega_1\|\|\omega_2\|} = \cos(\theta_{\omega_1,\omega_2}) \tag{31}$$

where $score(\omega_1, \omega_2) \geq \varphi$ will lead to verification of the hypothesized speaker and $score(\omega_1, \omega_2) < \varphi$ will lead to rejection.

3 Implementation

In Phases II-IV (described in Section 7), a simple yet complete speaker recognition system will be implemented in Matlab on a modern Dell desktop computer. A software package that extracts the MFCCs from the utterances will be used [12], but all other code will be developed. Two classifier tests will be included to validate code at different the phases.

The implemented code will not be able to processes large amounts of data which is typically necessary for a robust speaker recognition system, especially in obtaining the GMM-UBM and the total variability matrix, T. Therefore, lower dimensional features and modest sized training sets will be used for initial test and validation. To test on larger data sets, numerical complexities and high memory requirements are expected and techniques will have to be implemented make the code work satisfactorily.

The results of Phase II-IV will impact the decision of what to implement in Phase V. If reasonable results are obtained using the implemented code from Phase II-IV, more features may be added to the system. If the code written to obtain the GMM-UBM or the total variability matrix is found to be inefficient, Phase V may be to parallelize/optimize the inefficient code. If this is the case, the task will be too computationally intensive to complete the task on a single computer and will therefore be completed on

a cluster. The code will most likely be implemented in Matlab, using C and MPI if necessary. Another option of Phase V is to complete more extensive testing using different inputs into the vetted speaker recognition system.

4 Databases

The National Institute of Standards and Technology (NIST) has coordinated Speaker Recognition Evaluations (SRE) approximately every two years since 1996. In support of the SRE, datasets consisting of *.wav and *.sph formatted files with a sampling rate of 8kHz are provided for use of the participants. The databases that will be used for this project is the NIST 2008 SRE database and the NIST 2010 SRE database, both of which contain speech data in several different conditions including data from interviews, microphones, telephone conversations. The NIST 2008 SRE database will only be used if the amount of data from the NIST 2010 SRE is too much to process for phases II-IV. The NIST 2010 SRE database contains utterances from approximately 12000 different speakers.

5 Validation

Three commonly used metrics will be used for validation in this project. Equal error rate (ERR) is a measure that gives the accuracy at decision threshold for which the probabilities of false rejection (miss) and false acceptance (false alarm) are equal. This measure is good at obtaining a first quick understanding of whether there are any bugs because large values are not expected. Detection error trade-off (DET) curves will also be used for visual inspection. Lastly, the MinDCF algorithm used by NIST in the evaluation of the SRE will be examined.

Validation will ensure that the code is working properly in order to complete Phases II, III and V. Phase II marks the completion of using the EM and the MAP algorithm to generate the speaker model supervectors. A likelihood ratio test will be used as the classifier to validate results at this phase. Phase III uses FA techniques and LDA to create a low-dimensional space in which interspeaker variability is maximized and with-speaker variability is minimized. Discrete cosine scoring (DCS) can be used as the classifier and results can be tested after the FA step on the i-vectors and after the LDA step. Results from the DCS of the i-vectors should be an improvement over the likelihood ratio test in Phase II. The results from LDA should be an improvement over both of the previous score.

6 Testing

Several different tests will be conducted during Phase IV. Selection of tests will be chosen from the variety of different conditions made available by the NIST 2010 SRE databases. Smaller scaled testing will be completed in order to minimize the probability of running into difficulties processing the data on a modern desktop computer. If needed, the NIST 2008 SRE database can be used which contains smaller datasets. Larger tests will be used to determine the capabilities of the Matlab tool.

All tests completed on the Matlab code will also be tested on an already vetted speaker recognition system created by researchers at UMD and JHU. Side by side results can be compared and used as a higher level means validation.

7 Project Schedule

Fall 2011

Phase I: ~(5 weeks)

Aug. 29 – Sept. 28 ~(4 weeks)
- Read a variety of Text-Independent Speaker Identification papers to obtain an understanding of the proposed project

Sept. 28 – Oct. 4 ~(1 week)
- Write proposal and prepare for class presentation

Phase II: ~(4 weeks)

Oct. 5 – Oct. 21 ~(2 weeks)
- Be able to extract MFCCs from speech data and apply simple VAD algorithm
- Understand SRE databases

Oct. 22 – Nov. 4 ~(2 weeks)
- Develop EM algorithm to trained UBM
- Add MAP algorithm to create speaker models
- Add likelihood ratio test as a classifier
- Validate results using likelihood ratio test as classifier with EER and DET curves, bug fix when necessary

Phase III: ~(5 weeks)

Nov. 5 – Dec. 2 ~(3 weeks + Thanksgiving Break)
- Create supervectors from GMMs
- Write code to train total variability space
- Add ability to extract i-vectors from the total variability space
- Add cosine distance scoring (CDS) as a classifier
- Validate results using the CDS classifier with EER and DET curves, bug fix when necessary

Dec. 3 – Dec. 9 ~(1 week) *overlap*
- Prepare Project Progress Report

Dec. 3 – Dec. 19 ~(2 week) *overlap*
- Implement LDA on the i-vectors
- Validate results using the CDS classifier with EER and DET curves, bug fix when necessary

Spring 2012

Phase IV: ~(4 weeks)

Jan. 25 – Feb. 24 ~(4 weeks)
- Obtain familiarity with vetted a speaker recognition system
- Test algorithms of Phase II and Phase III on several different conditions and compare against results of vetted system
- Bug fix when necessary

Phase V ~(7 weeks)

Feb. 25 – Mar. 2 ~(1 week) *overlap*
- Make Decision to either: (1) parallelize/optimize inefficient code, (2) Add more features, or (3) test in various conditions
- Read appropriate background material to make decision

Feb. 25 – Mar. 2 ~(1 week) *overlap*
- Work on Project Status Presentation

Mar. 3 – Apr. 20 ~(6 weeks + Spring Break)
- Update Schedule to reflect decision made in Phase IV
- Finish (1) or (2) in a 6 week time period including time for validation and test

Phase VI: ~(3 weeks)

Apr. 21 – May 10 ~(3 weeks)
- Create final report and prepare for final presentation

8 Milestones

Fall 2011

October 4
- Have a good general understanding on the full project and have proposal completed. Present proposal in class by this date.
  Marks completion of Phase I

November 4
- Validation of system based on supervectors generated by the EM and MAP algorithms
  Marks completion of Phase II

December 19
- Validation of system based on extracted i-vectors
- Validation of system based on nuisance-compensated i-vectors from LDA
- Mid-Year Project Progress Report completed. Present in class by this date.
  Marks completion of Phase III

Spring 2012

Feb. 25
- Testing algorithms from Phase II and Phase III will be completed and compared against results of vetted system. Will be familiar with vetted Speaker Recognition System by this time.
  Marks completion of Phase IV

March 18
- Decision made on next step in project. Schedule updated and present status update in class by this date.

April 20
- Completion of all tasks for project.
  Marks completion of Phase V

May 10
- Final Report completed. Present in class by this date.
  Marks completion of Phase VI

9 Deliverables

A fully validated and complete Matlab implementation of a speaker recognition system will be delivered with at least two classification algorithms.

Both a mid-year progress report and a final report will be delivered which will include validation and test results.

10 Bibliography

[1]*Biometrics.gov - Home*. Web. 02 Oct. 2011. <http://www.biometrics.gov/>.

[2] Kinnunen, Tomi, and Haizhou Li. "An Overview of Text-independent Speaker Recognition: From Features to Supervectors." *Speech Communication* 52.1 (2010): 12-40. Print.

[3] Ellis, Daniel. "An introduction to signal processing for speech." *The Handbook of Phonetic Science,* ed. Hardcastle and Laver, 2nd ed., 2009.

[4] Reynolds, D. "Speaker Verification Using Adapted Gaussian Mixture Models." *Digital Signal Processing* 10.1-3 (2000): 19-41. Print.

[5] Reynolds, Douglas A., and Richard C. Rose. "Robust Text-independent Speaker Identification Using Gaussian Mixture Speaker Models." *IEEE Transations on Speech and Audio Processing* IEEE 3.1 (1995): 72-83. Print.

[6] "Factor Analysis." *Wikipedia, the Free Encyclopedia*. Web. 03 Oct. 2011. <http://en.wikipedia.org/wiki/Factor_analysis>.

[7] Dehak, Najim, and Dehak, Reda. "Support Vector Machines versus Fast Scoring in the Low-Dimensional Total Variability Space for Speaker Verification." *Interspeech 2009 Brighton.* 1559-1562.

[8] Kenny, Patrick, Pierre Ouellet, Najim Dehak, Vishwa Gupta, and Pierre Dumouchel. "A Study of Interspeaker Variability in Speaker Verification." *IEEE Transactions on Audio, Speech, and Language Processing* 16.5 (2008): 980-88. Print.

[9] Lei, Howard. "Joint Factor Analysis (JFA) and i-vector Tutorial." *ICSI. Web. 02 Oct. 2011.* http://www.icsi.berkeley.edu/Speech/presentations/AFRL_ICSI_visit2_JFA_tutorial_icsitalk.pdf

[10] Kenny, P., G. Boulianne, and P. Dumouchel. "Eigenvoice Modeling with Sparse Training Data." *IEEE Transactions on Speech and Audio Processing* 13.3 (2005): 345-54. Print.

[11] Bishop, Christopher M. "4.1.6 Fisher's Discriminant for Multiple Classes." *Pattern Recognition and Machine Learning*. New York: Springer, 2006. Print.

[12] Ellis, Daniel P. W. *PLP and RASTA (and MFCC, and Inversion) in Matlab*. *PLP and RASTA (and MFCC, and Inversion) in Matlab*. Vers. Ellis05-rastamat. 2005. Web. 1 Oct. 2011. <http://www.ee.columbia.edu/~dpwe/resources/matlab/rastamat/>.