Voice Verification System for Mobile Devices based on ALIZE/LIA_RAL

Hussein Sharafeddin[1], Mageda Sharafeddin[2] and Haitham Akkary[2]
[1]*Faculty of Sciences, Lebanese University, Hadat, Beirut, Lebanon*
[2]*Department of Electrical and Computer Engineering, American University of Beirut, Beirut, Lebanon*

Keywords: ALIZE, Biometric Authentication, LIA_RAL, Mobile Security, Speaker Verification, Text-independent Identification, Voice Recognition.

Abstract: The main contribution of this paper is providing an architecture for mobile users to authenticate user identity through short text phrases using robust open source voice recognition library ALIZE and speaker recognition tool LIA_RAL. Our architecture consists of a server connected to a group of subscribed mobile devices. The server is mainly needed for training the world model while user training and verification run on the individual mobile devices. The server uses a number of public random speaker text independent voice files to generate data, including the world model, used in training and calculating scores. The server data are shipped with the initial install of our package and with every subsequent package update to all subscribed mobile devices. For security purposes, training data consisting of raw voice and processed files of each user reside on the user's device only. Verification is based on a short text-independent as well as text-dependent phrases, for ease of use and enhanced performance that gets processed and scored against the user trained model. While we implemented our voice verification in Android, the system will perform as efficiently in iOS. It will in fact be easier to implement since the base libraries are all written in C/C++. We show that the verification success rate of our system is 82%. Our system provides a free robust alternative to replace commercial voice identification and verification tools and extensible to implement more advanced mathematical models available in ALIZE and shown to improve voice recognition.

1 INTRODUCTION

Security is a major barrier to using mobile devices in business according to a study by Boa et al. (Bao et al., 2011). A recent study by Communication Fraud Control Association (Aronoff, 2013) shows that global fraud loss in 2013 is up by 15% compared to 2011 and reports $8.84 Billion USD losses from subscription fraud and identity theft alone.

There has been a lot of interest in the area of biometric information for identification to leverage security. This is mainly due to the fact that authentication with biometric data is based on who the person is and not on the passwords they are supposed to remember. While a lot of work in the literature examines multi modal biometrics where several biometric measures are combined for identification, in this work we focus on one biometric parameter, human voice. In voice or speaker identification the user does not claim an identity, the tool rather determines the user's identity through a trained classifier. Speaker verification on the other hand is when a tool determines whether the user is who he/she claims to be. Speaker verification and identification are generally two types, text dependent and text independent. Current text dependent voice verification tools are awkward to use and perform poorly (Trewin et al. 2012). In some applications the user is expected to remember the phrase used and to pronounce it the way he/she did during training. Hence, it is crucial to improve robustness of voice verification and have the tool automatically identify the user without introducing additional restrictions or costs.

The main contributions of this work are:

1. Introducing the robust open source speaker recognition platform ALIZE/LIA_RAL into mobile development and research. This platform is extensible and has shown potential for improved success rate in Automatic Speaker Recognition (ASR) (Larcher et al. 2013).
2. Our code modifications are contributed to the

Sharafeddin H., Sharafeddin M. and Akkary H..
Voice Verification System for Mobile Devices based on ALIZE/LIA_RAL.
DOI: 10.5220/0005218902480255
In *Proceedings of the International Conference on Pattern Recognition Applications and Methods* (ICPRAM-2015), pages 248-255
ISBN: 978-989-758-077-2

open source community to facilitate enhancements to state of the art voice identification on mobile devices (Sharafeddin et al. 2014).

One voice identification Android application is based on open source code reported in the literature by Chen *et al.* (Chen et al. 2013). They use Modular Audio Recognition Framework (MARF), an open source Java tool. Linear Prediction Coefficients (LPC) model is used in MARF. LPC is based on linear combinations of past values of the voice signal with scaled present values. LPCs greatly simplify the voice recognition problem by ignoring the complete signal modeling of human vocal excitation (Campbel 1997). The Android application in (Chen et al. 2013) reduces errors in voice detection by generating distance information and optimizing results using personal thresholds. Their text-dependent speaker identification tool achieves an 81% success rate. Using thresholds is common in voice verification to adjust two error types: False Acceptance Rate (FAR) when an imposter is authenticated and False Rejection Rate (FRR) when an authentic user is denied access (Faundez-Zanuy 2006).

Another ASR tool for mobile phones that has been recently introduced is SpeakerSense (Lu et al. 2011). The tool focuses on speaker identification, hence it recognizes who is speaking out of a group of many trained speaker voices. As such their work is an $m-$class pattern recognition problem, and since it deals with audio data, it requires the training time to be not less than 3 minutes per trainee. This necessitates studying the power consumption efficiency as a known restriction on small devices. SpeakerSense utilizes similar pre-processing and feature extraction approaches as our tool; however, since the tools differ in the ultimate goals of identification versus verification, the classification models differ. SpeakerSense does utilize a Gaussian Mixture Model (GMM) using only 20 Mel-frequency Cepstral Coefficients (MFCCs) while our tool uses a GMM with a 60-dimensional component vector per user. The extended vector we use includes 19 MFCCs, 40 first and second order derivatives and 1 energy component. Additionally, while the GMM training is done on a remote server running Matlab for SpeakerSense, we run the C++ based user training with the option to perform the Universal Background Model (UBM) training natively onboard the mobile device. The reason why we leave the UBM as an option is that usually we can have one UBM, pre-trained using randomly selected speech segments, work with any individual user GMM. Hence depending on the particular application and deployment policy, we can have the UBM data incorporated within the installation package transparently to the user. This not only simplifies but also lends flexibility to our architecture.

Using a cloud based system for speech recognition using a mobile phone was introduced by Alumae *et al.* (Alumae and Kaljurand 2012). The system uses open source CMU Sphinx system (CMUSphinx 2014) and targets Estonian language recognition where the authors try to provide a user experience comparable to the Google voice search. Unlike our system, their system sends raw voice data from the mobile device to the server for recognition. This is fundamentally different from the role of servers in our system which is mere calculation of the UBM model and processing public speaker data. In this work we use the Mel-Wrapped Cepstrum analysis, which has been shown to work well in speaker recognition (Gish and Schmidt 1994), compared to less efficient techniques such as LPC clustering distances and neural networks.

The rest of this paper is organized as follows. In section 2 we provide background information on main algorithms used in our system. In section 3 we describe our system and in section 4 we discuss our experimental setup. We show performance results and success rates in section 5. We finally conclude in section 6 and discuss future work in section 7.

2 BACKGROUND

Two main components are relevant in speaker verification, features that discriminate among human voices and suitable classification approaches. Among the most common models that examine both components are LPCs and MFCCs. Ideally we were looking for open source packages that implement the full chain of voice pre-processing, feature extraction, classification model creation, user training and classification. Our job would be to evaluate such tools on short speech segments from a limited set of users so that we can port it to a portable device running Android for example.

For speaker verification or identification two open research tools appear to have received research attention in the past decade. The first is MARF and the second is the combination of ALIZE/LIA_RAL packages (D'Avignon Laboratoire Informatique 2011). MARF is an open source (Mokhov et al. 2006) research tool mainly for speaker identification written in Java. Among several functions, it extracts

LPCs and implements several pattern recognition techniques for classification which can be used for speaker identification. Being in Java, and thus platform independent, it is easier to deploy and enhance than other similar tools. We tested MARF on our data set using the LPC features. The best results among the different classifiers it provides did not exceed 80% for speaker identification.

ALIZE (D'Avignon Laboratoire Informatique 2011) is a core toolkit that encapsulates several base-level functionalities for managing speech files and the subsequent data extracted thereof. Utilizing ALIZE functionalities is LIA_RAL, a set of modules that implement different speech processing and recognition algorithms which can be cascaded into a customized tool chain for testing different mathematical models. Both ALIZE and LIA_RAL, developed at the Laboratoire d'Informatique d'Avignon (LIA) –France, are open source platforms for research in speaker verification and speech recognition. Several methods have been investigated for the classification model including GMM, Joint Factor Analysis (JFA), and Support Vector Machines (SVM). In this work we focus on the GMM implementation leaving the others for future work. It is worth mentioning here that LIA_RAL uses speech features extracted by other tools. One such tool is the open source toolkit for Signal Processing (SPro) (Spro 2004), a speech processing and feature extraction library that implements commonly used pre-processing steps and feature extraction algorithms. In this work we utilize it for pre-emphasis, windowing and MFCC extraction. These are extracted directly from the raw voice files and stored into corresponding binary files.

Reynolds *et al.* (Reynolds and Rose 1995) detailed the GMM model and justified its veracity for speaker identification. Since the main interest here is to validate a speaker by processing a short speech segment, the model is desired to utilize features that effectively represent the speaker's vocal tract physiological and acoustic properties. Several feature types have been studied in the literature including LPC that model the vocal tract as a linear system. Although those features offer good representation of a person's vocal tract properties, they are sensitive to additive noise such as microphone background.

Spectral analysis based features offer frequency range selectivity that might help in applications where noise is inevitable (Reynolds and Rose 1995). In particular, MFCCs have been widely used in speech processing and recognition applications. The extraction process can be summarized by the following steps: voice signal pre-processing which includes pre-emphasis and windowing; Fourier transform and coefficient modulus calculation; filtering through a Mel-frequency filterbank for smoothing and envelope extraction and therefore vector size reduction.

$$f_{Mel} = 1000 \log_2 \left(1 + {}^{f}/_{1000}\right) \quad (1)$$

Finally, performing a discrete cosine transform (DCT) on the log scaled values (Rabiner and Schafer 2010) (Petrovska-Delacrétaz, et al. 2009):

$$c_n = \sum_{k=1}^{K} S_k \cos\left(\frac{n\left(k-\frac{1}{2}\right)\pi}{K}\right); \quad n = 1,2,\dots,L, \quad (2)$$

where $S_k (k = 1, \dots, K)$ is the *log*-absolute value of the k^{th} Fast Fourier Transform (FFT) coefficient; K is the number of FFT coefficients; L $(L \leq K)$ is the number of cepstral coefficients c_n to keep. Each window will then be represented by its own cepstral vectors. The logarithm calculation in the final stage makes the model closer to the human hearing system. The DCT decorrelates the log filterbank energies because of their overlap. This simplifies downstream model calculations by using diagonal covariance matrices.

The MFCC features calculated using Spro (Spro 2004) are further normalized to have a zero-mean and unit variances by the feature normalization step shown in figure 1a-c for noise reduction. Energy extraction (Spro 2004) also shown in figure 1a-c is applied next to the voice frames in order to minimize error contribution from unvoiced speech, silence or background noise; their corresponding feature vectors, linked by the frame time labels, are then easily eliminated.

The remaining features are well suited for use in speaker verification, and hence, make up the feature vector components of the GMM model that we also use in this work. Moreover, the smoothing property of the Gaussian model yields robustness to the stochastic parameter estimation of a speaker's voice underlying component distributions. The GMM represents a multi-modal distribution that provides a better and smoother fit when compared with the simpler uni-modal Gaussian or the vector quantization codebook model.

For a single speaker D-dimensional feature vector by $\vec{x}$, a weighted sum of M Gaussian densities $p_i(\vec{x}; \mu_i, \Sigma_i)$ with weights w_i constitutes a GMM for $\vec{x}$ to be utilized for the likelihood ratio test:

$$p(\vec{x}|\lambda) \triangleq p(\vec{x}|w_i, \mu_i, \Sigma_i) = \sum_{i=1}^{M} w_i p_i(\vec{x}); \quad (3)$$

$$where \quad i = 1, \dots, M \text{ and } \sum_{i=1}^{M} w_i = 1;$$

$$p_i(\vec{x}) = \frac{1}{(2\pi)^{\frac{D}{2}} |\Sigma_i|^{\frac{1}{2}}} e^{\frac{-1}{2}(\vec{x_i}-\vec{\mu_i})' \Sigma_i^{-1} (\vec{x_i}-\vec{\mu_i})} \quad (4)$$

The current approach in the field for estimating the individual speaker GMMs is to first estimate a UBM, also called GMM world model, from random features taken from numerous speakers. In our work we demonstrated this approach by recording several short speech segments (< 10 seconds) from 7 randomly selected speakers. We subsequently pass their features to the training step shown in figure 1a. This UBM represents speaker-independent feature probability density function (pdf) mixture and is used in adapting the individual speaker GMM parameters to better represent the acoustic classes present in the speech segments.

Verifying whether a given speech vector $\vec{x}$ would belong to speaker A is done by a hypothesis test. The pdf for the null hypothesis H_0 is $p(\vec{x}|\lambda_A)$; the pdf for the alternative hypothesis H_1 is $p(\vec{x}|\lambda_{UBM})$ where λ_{UBM} represents all models not $\lambda_A(\overline{\lambda_A})$. In fact this is the dominant approach in the literature as done by LIA_RAL. The likelihood ratio test becomes:

$$LR(\vec{x}) = \frac{p(\vec{x}|\lambda_A)}{p(\vec{x}|\overline{\lambda_{UBM}})} \begin{cases} \geq \theta, \text{authenticate as } A \\ < \theta, \text{reject as impostor} \end{cases} \quad (5)$$

The decision threshold is set empirically. In our implementation, we introduced a tool sensitivity selector to be set by the user. Fauve et al. (Fauve et al. 2007) studied limitations of the basic GMM model and of a modified model that uses support vector machines; they also demonstrated improvements on a modified eigenvoice model.

Bousquet et al. (Bousquet et al. 2011) proposed an improvement to the i-vector approach by performing simple linear and nonlinear transformations to remove the session effects. They also proposed an improvement on the scoring technique by utilizing the Mahalanobis distance in the classifier. In this work we do not include Fauve et al.'s (Fauve et al. 2007) and Bousquet et al.'s (Bousquet et al. 2011) enhancement and leave this for future work.

3 PROPOSED SYSTEM

In this work we advocate using high quality open source speech recognition techniques to build a speaker verification application that can be used stand alone or integrated with other authentication methods. Most of the available code suitable for this task is written in C and C++. While this can be integrated into iOS applications easily, it is not straight forward to implement in Android. The Java Native Interface (JNI) environment is needed for building dynamic libraries to be loaded at run-time. Specialized Makefiles and coding rules for function names and parameters need to be followed in order for the library to properly interface with the Java Dalvik Virtual Machine inside Android. This multi-layered hybrid system makes it challenging to debug the application.

3.1 System Overview

Figure 1 shows a general overview of our proposed system. The figure shows the three separate steps in our system. Figure 1a shows the server training step, a process that is performed offline and involves public speaker voice signals or system speaker voice signals who allow the use of these signals. Spro (Spro 2004) is an open source ANSI C library with various speech analysis techniques. In our work we use Spro version 4 which is released under the GPL license. The default settings released with the ALIZE/LIA_RAL code assume the audio file format is SPHERE (.sph) which is an older format used initially by the National Institute of Standards and Technology (NIST) (NIST 2014). In this work we implement an application running on current mobile devices and hence we use the WAVE format (.wav) which is also supported by the Spro file reader. Our main use for Spro is to perform the first preprocessing steps which are pre-emphasis and windowing and to extract the MFCC features. The main problem we faced with Spro was a portability to Android. Some memory management system calls ended up using uninitialized memory blocks and causing core dumps. We wrote macros to circumvent the problem in a way not to violate the license. We contacted the author of Spro with this fix which will be added to the new release. The system was developed on an Intel machine running Ubuntu linux using eclipse IDE; the native code was cross-compiled to ARM-based dlls using JNI.

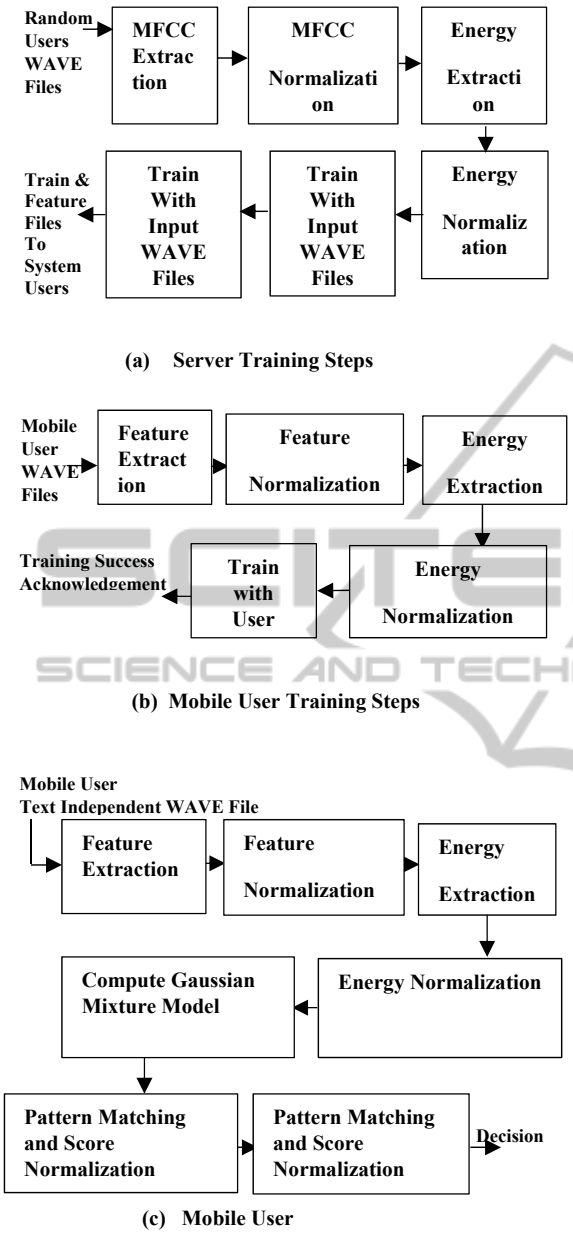

Figure 1: Our System's Main Activities- (a) server training, (b) user training, and (c) user identification.

3.2 Server Training

The voice signals of the random users undergo four Spro steps, MFCC extraction and normalization, energy extraction and normalization shown in figure 1a. This will generate PRM files which include normalized feature vectors and LBL files denoting durations of speech periods in a given speaker voice file. The last two steps involve server training the input WAV files to create the UBM model. All of these files are needed in the mobile device user training and verification and are hence packaged and shipped with the application to the various users. Periodically, the server can resend updated data in the form of application upgrade to all subscribed users. Note that the design of server training carefully separates user data from public data for two main reasons, first to remove any security concerns by not sending user voice files over the network and second to reduce the mobile user training time.

3.3 Mobile User Training

Figure 1b shows the mobile user training steps. Note that the same first four Spro steps in server training are applied here. Various PRM files are generated as a result. Additionally, the user is asked to train his/her device. This is a timed training session which lasts 8s to 30s. During this time, the user is supposed to utter a sentence of their choice. The generated user PRM files with the LBL files denoting periods of utterance, as opposed to silence, are saved in the package file space to avoid exposing the files when the device is compromised temporarily. The imposter will need to have the root id to be able to extract these files.

3.4 Mobile User Verification

Finally, figure 1c shows the user verification step. In this step, the user is asked to utter sentences of his/her choice. This is also a timed session that lasts 8s to 15s. After feature and energy extraction and normalization, the new data is compared against the world UBM plus owner model to create verification scores. The scores are finally normalized. Score normalization is based on the following three normalization techniques:

1. Zero normalization technique to compensate for inter-speaker score variations.
2. Test normalization technique to compensate for inter-session score variation.
3. Confused Zero and Text normalization to do both and can perform better that 1 & 2 (Srikanth and Hegde 2010).

We found that using scores from either 1 or 2 are sufficient in addition to scores from 3. The verification rates reported in section 5 are based on scores from 1 & 3. Normalization techniques allow setting a global threshold independent of speakers using our system.

4 EXPERIMENTAL SETUP

We carried several build tests and application experiments in order to achieve the following goals: 1. *Concept:* A proof of concept assurance that all the required code gets ported and works as expected. This was achieved through a set of C/C++ macro definitions and some recoding needed for memory stability and exception handling. We ran several tests on a 64-bit Intel-based linux machine as well as on an ARM-based handheld Android device using the same NIST04 data and configuration sets released with the ALIZE_GMM (D'Avignon Laboratoire Informatique 2011) tutorial which utilizes the baseline GMM MAP approach. Figure 2 below demonstrates that the final normalization steps on either machine were verified to be very similar, the minor differences being due to floating point precision. 2. *Threshold:* Since making the authentication decision requires that the normalization scores be thresholded, finding a global working threshold is a known challenge in the field especially that this depends on training and on testing conditions. Being a first study in implementing the ALIZE/LIA_RAL system on mobile devices, the main focus here is not methodical especially that there are further published improvements to the basic GMM MAP model that we did not include yet. That said, we automated anonymous voice, score, and performance data collection on the device for further analysis. 3. *Data source variance:* Since this application is intended for everyday mobile device users, practical considerations in terms of noise, device variability and text similarity should be reflected in fresh recorded speech data. Device variability in terms of performance is due to the long chain of processing and calculations involved and microphone operating characteristics.

We conducted simultaneous tests on two different ARM-based Android devices: a single core LG Optimus L5 phone running Android 4.0.3 on an ARM Cortex-A5 800MHz processor and a quad-core Samsung SIII phone running Android 4.3 on an ARM Cortex-A9 1.4GHz processor. The audio recorders were configured at 8 KHz sampling rate with16 bits per sample using a single audio channel. Eight volunteers enrolled their voices for UBM training with short duration speech data (8-15 seconds) recorded. Another set of eight volunteers enrolled as *targets* in a quiet environment for 8 seconds in one experiment and for 30 seconds in another. Each target then performed three true positive tests against their own trained model, and similarly three true negative tests against other targets. The processing times and normalized scores were recorded for each test. All true positive tests were repeated using the same trained text as well as using different text with variable recording times. The test sets were conducted first in a quiet setting then repeated with a noisy background.

5 RESULTS

Goal 1: Initial tests on public NIST04 data proved the concept and produced very close results with minor differences due to floating point precision. Using our recorded data also confirmed feasibility; however, the results were suboptimal due to the limited number of UBM training and recording durations. Figure 3 shows the overall Equal Error Rate (EER) achieved with this limited data set.

Goal 2: Since our tests were repeated in different conditions, we separated the results first based on background noise then based on text similarity. Since our goal is text-independence, no discrete time warping (DTW, done for voice segment alignment) was implemented; yet and as expected, the text-similar tests yielded 24% better results than random speech test cases. The superiority in the former is mainly attributed to more energy similarities in the same frequency bands uttered. Noisy backgrounds did contribute to degraded performance compared to quiet test setting by 21% EER increase. The 30s enrollment duration also yielded better separation distance as opposed to the 8s enrollment.

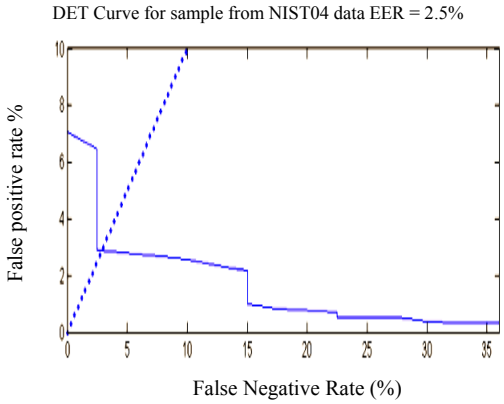

Figure 2: Sample result on mobile phone using NIST04 sample confirming previous results.

Goal 3: Our tests did not show significant impact of microphone variability between the two devices; however, performance was noticeably faster on the

multicore device, as expected. Data processing itself was just 35% faster since our application runs on a single thread on either core, the improvement being mostly due to the clock speed; however, the overall application user experience was much better due to different threads handling context switching on the multicore device.

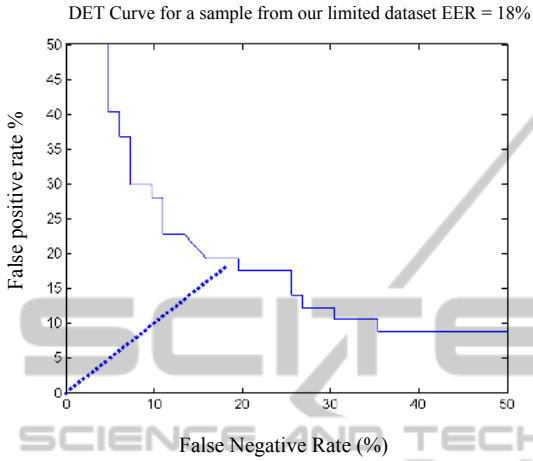

Figure 3: Sample result from our limited data set recorded on mobile device.

6 CONCLUSIONS

In this work we introduce a system for voice verification consisting of a server and subscribed mobile users. We use open source code SPro and ALIZE library as well as LIA_RAL speaker tool for user training and verification. Our system preliminary results show an 82% identification success rate.

The full android application with all dependencies is available online (Sharafeddin et al. 2014). This includes a README file with all modifications made to both Spro and LIARAL. It also has a sample UBM model and user wave files. The various modules used in our system, shown in figure 1, are documented in the respective tools.

7 FUTURE WORK

Results can be significantly improved first by collecting enrollment segments from many users anonymously using different devices in regular daily life settings (noisy or quiet) in order to build a more reliable UBM to be distributed with the application. Moreover, the enrollment time would be set to a minimum of 30s. Further improvements to the baseline GMM MAP system including Joint Factor Analysis (Kenney et al. 2007), i-vectors (Dehak et al. 2009) and SVM (Campbell et al. 2006) based classification as included in ALIZE_3.0 (Larcher et al. 2013) will be incorporated and compared. We intend to evaluate the work of Chan et al. (Chan et al. 2007) where wavelets for features are compared with the MFCC features on a GMM model. Finally, we would also like to understand if implementing dynamic time warping to create distance information for determining personal thresholds as reported by (Chen et al. 2013) improves verification rates.

Finally, previous baseline GMM studies showed suboptimal performance of the GMM MAP model especially with short duration training and testing (Fauve et al. 2008). We intend to use this work as a start for evaluating ALIZE/LIA_RAL on different mobile phones. Our application works in production mode as well as in research mode where it keeps speech data and collects score and performance information. This will be leveraged as new volunteers enroll to further enhance the UBM and the methods tested.

REFERENCES

Alumae, T., Kaljurand, K., 2012. Open and extendable speech recognition application architecture for mobile environments. *In SLTU'12, The 2nd International Workshop on Spoken Language Technologies for Under-resourced Languages.*

Aronoff, R., 2013. *Global fraud loss survey*. Communications Fraud Conrol Association. Roseland, NJ.

Bao, P., Pierce, J., Wittkaer, S. , Zhai, S., 2011. Smart Phone Use by Non-Mobile Business Users. *In MobileHCI.*

Bousquet, P. M., Matrouf,, D., Bonastre, J. F., 2011. Intersession Compensation and Scoring Methods in the i-vectors Space for Speaker Recognition. *In 12th Annual Conference of the International Speech Communication Association.*

Campbel, J. Jr., 1997. Speaker recognition: a tutorial. *In Proceedings of the IEEE. Vol. 85, no. 9, Sept., pp. 1437-1462.*

Campbell, W. M., J. P. Campbell, J. P., Reynolds, D. A., Singer, E., Torres-Carrasquillo, P. A., 2006. Support Vector Machines for Speaker and Language Recognition. *In Computer Speech & Language, Elsevier, Vol. 20.*

Chan, W. N., Zheng, N., Lee, T., 2007. Discrimination Power of Vocal Source and Vocal Tract Related Features for Speaker Segmentation. *In IEEE TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING.*

Chen, Y., Heimark, E., Gligorski, D., 2013. Personal threshold in a small scale text-dependent speaker recognition. *In International Symposium on Biometrics and Security Technologies.*

CMUSphinx, 2014. Carnegie Melon University. *http://cmusphinx.sourceforge.net.*

D'Avignon, Laboratoire Informatique, 2011. *http://mistral.univ-avignon.fr/index_en.html*.

Dehak, N., Dehak, R., Kenny, P., Brummer, N., Ouellet, P., Dumouchel, P., 2009. Support Vector Machines versus Fast Scoring in the Low-Dimensional Total Variability Space for Speaker Verification. *In InterSpeech 10th Annual Conference of the International Speech Communication Association.*

Faundez-Zanuy, M., 2006. Biometric security technology. *In IEEE Aerospace and Electronic Systems Magazine No. 21, pp. 15-26.*

Fauve, B., Evans, N., Mason, J., 2008. *Improving the performance of text-independent short duration.* Odyssey.

Fauve, B.G.B., Matrouf, D., Scheffer, N., Bonastre, J. F., Masin, J. S. D., 2007. State-of-the-Art Performance in Text-Independent Speaker Verification Through Open-Source Software. *In IEEE Transactions on Audio Speech and Language Processing*.

Gish, H., Schmidt, M., 1994. Text-independent speaker identification. *In IEEE Signal Processing Magazine, Vol. 1, pp. 18-32*.

Kenney, P., Boulianne, G., Ouellet, P., Dumouchel, P., 2007. Joint Factor Analysis versus Eigenchannels in Speaker Recognition. *In IEEE Transactions on Audio, Speech, and Language Processing, Vol. 15.*

Larcher, A., et al., 2013. ALIZE 3.0 - Open source toolkit for state-of-the-art speaker recognition. *In Interspeech*.

Lu, H., Brush, A., Priyantha, B., Karlson, A., Liu, J., 2011. SpeakerSense: Energy Efficient Unobtrusive Speaker Identification on Mobile Phones. *In The Ninth International Conference on Pervasive Computing.*

Mokhov, S., Clement, I., Sinclair, S., Nicolacopoulos, D., 2002. *Modular Audio Recognition Framework. Department of Computer Science and Software Engineering, Concordia University. http://marf.sourceforge.net.*

NIST, 2014, *www.nist.gov*.

Petrovska-Delacrétaz, D., Chollet, G., Dorizzi, B., Jain, A., 2009. *Guide to Biometric Reference Systems and Performance Evaluation. Springer.*

Rabiner, L., Schafer, R., 2010. Theory and Applications of Digital Speech Processing 1st (first) Edition . *Prentice Hall.*

Reynolds, D. A., Rose, R., 1995. Robust Text-Independent Speaker Identification Using Gaussian Mixture Speaker Models. *In IEEE Transactions on Speech and Audio Processing, Vol. 3, No. 1.*

Sharafeddin, M., Sharafeddin, H, Akkary, H., 2014. Android-Voice-IDentification-App-using-SPRO-ALIZE-LIARAL. *github.com/umbatoul/Android-Voice-IDentification-App-using-SPRO-ALIZE-LIARAL.*

Srikanth, N, Hegde, R. M., 2010. On line client-wise cohort set selection for speaker verification using iterative normalization of confusion matrices. *In EUSIPCO European Signal Processing Conference, pp. 576-580.*

Spro, 2004. *http://www.irisa.fr/metiss/guig/spro/.*

Trewin, S., Swart, C., Koved, L., Matino, J., Singh, K., Ben-Davic, J., 2012. Biometric Authentication on a Mobile Device: A Study of User Effort, Error, and Task Disruption. *In ACSAC'12 Proceedings of the 28th Annual Computer Security Applications Conference pp. 159-168.*