Privacy-Preserving Speaker Verification using Secure Binary Embeddings

José Portêlo*†, Bhiksha Raj‡, Alberto Abad*† and Isabel Trancoso*†
*INESC-ID, Lisboa, Portugal
†Instituto Superior Técnico, Lisboa, Portugal
‡Language Technologies Institute, Carnegie Mellon University, Pittsburgh, PA, USA
Email: {jose.portelo,alberto.abad,isabel.trancoso}@inesc-id.pt, bhiksha@cs.cmu.edu

***Abstract*—Remote speaker verification services typically rely on the system having access to the users recordings, or features derived from them, and/or a model for the users voice. This conventional approach raises several privacy concerns. In this work, we address this privacy problem in the context of a speaker verification system using a factor analysis based front-end extractor, the so-called i-vectors. Preserving privacy in our context means that neither the system observes voice samples or speech models from the user, nor the user observes the universal model owned by the system. This is achieved by transforming speaker i-vectors to bit strings in a way that allows for the computation of approximate distances, instead of exact ones. The key to the transformation uses a hashing scheme known as secure binary embeddings. Then, an SVM classifier with a modified kernel operates on the hashes. Experiments showed that the secure system yielded similar results as its non-private counterpart. The approach may be extended to other types of biometric authentication.**

I. Introduction

The exponential increase of storage capability over the Internet has lead to the dissemination of a variety of online services, such as e-banking, mobile commerce, social networks, image and video organizers, online games, etc. The common aspect among all of them is that users must first register with them and receive a unique user ID and a written password with which they can authenticate themselves to the system. In many situations, an attractive alternative is to authenticate users by their voice instead, since it is a more natural way of communication.

Nevertheless, voice-based authentication systems, also called speaker verification systems, have significant privacy concerns. Current systems require access to recordings of a user's voice (or at least parameterized versions of it). A malicious system, or a hacker who has compromised the system, could edit the recordings to impersonate the user. Even if the user only transmits features extracted from his voice, such that a recording cannot be synthesized from them, other risks remain. An individual's voice also carries information about their gender, nationality, etc., all of which would also be available for the services to use as they please, *e.g.* they could sell this information. Ideally, the service should not have access to such information, although it seems nearly paradoxical to require that a system not have access to many of the key voice characteristics that themselves help establish the user's identity.

In order to protect the user, the system must store both his voice and the parameters representing it in an obfuscated manner, such that neither the system nor a hacker can extract undesired information from them or use them to impersonate the user. This aspect of speaker authentication has been largely ignored until recently, and literature on the topic remains minimal. In [1] homomorphic encryption methods were employed to ensure that the system only sees encrypted data from the user, and only stores encrypted models that it cannot decrypt by itself. To successfully perform speaker authentication, it must engage in secure multiparty computation (SMC) [2] protocols requiring repeated encryption and decryption, with significant computational participation by the user. In [3] an alternate scheme based on locality-sensitive hashing (LSH) [4] was proposed, that converts voice recordings to a password-like string. Authentication is performed based on perfect matches of the password-like strings derived from the voice to stored templates. Both procedures have their drawbacks: on one hand cryptographic methods pose an unacceptably high computational load, while on the other hand the LSH-based method compromises system accuracy to achieve speed, although it is very secure. A more recent approach was presented in [5], where a new technique called secure binary embeddings (SBE) [6] was successfully used for obtaining not only an almost negligible degradation in classification results (when compared to a baseline using supervectors) but also a computational overhead similar to the one required by LSH.

In this paper we further explore the combination of speaker recognition with SBE for privacy-preserving speaker authentication. In particular, we aim to analyze if the previous approach in [5] scales properly across different speaker verification techniques. Thus, we propose a new privacy-preserving scheme using a factor analysis based front-end extractor, the so-called i-vectors, which is the current *de facto* standard for speaker verification.

The remaining of this paper is structured as follows. In Section II we briefly present the speaker verification techniques considered in this paper. Section III describes secure binary embeddings and their properties. In Section IV we present experiments on the speaker verification task. Then we discuss data privacy issues in Section V and finally we present some conclusions.

II. Speaker Verification

In conventional text-independent speaker verification systems, a user wishing to authenticate himself provides the system voice samples during an enrollment phase. The system then employs these samples to build a "model" for the user.

Later, in the verification phase, new incoming speech signals are compared to this model to verify the user.

In terms of speech parameterization, speaker verification systems are typically built upon mel-frequency cepstral coefficient (MFCC) vectors. Like in other speech applications, feature vectors are usually augmented with their derivatives. As a first step, a large collection of recordings of non-target speakers is used to train a universal background model (UBM). The UBM is a Gaussian mixture model (GMM) representing the distribution of speech from all potential imposters. Next, the Gaussian means of the UBM are adjusted through maximum a posteriori (MAP) adaptation [7] to the user's enrollment data to learn a GMM for that user. In addition to reducing enrollment data requirements, MAP adaptation also guarantees a one-to-one correspondence between the Gaussians in the UBM and those in the user's model. Given a new recording purported to be from a target user, the log likelihood assigned to it by the GMM for the target user is compared to that obtained from the UBM to determine if the user should be accepted or not [7].

An alternate equally-successful approach to likelihood ratio tests obtains a separate GMM for *each* of multiple enrollment recordings by the user, through MAP adaptation of the UBM to the recording. The parameters (usually the means) of the resulting GMMs are mapped to a high-dimensional vectors called "supervector" [8], one for each recording. Supervectors are similarly obtained for recordings by putative imposters. A support vector machine (SVM) is then trained to distinguish between the two sets [9]. To verify that a given test recording was indeed spoken by the user, the supervector derived from the recording is classified by the SVM.

More recently, additional variations to this GMM-UBM scheme have resulted in the proliferation of new successful vector-based methods, such as the joint factor analysis (JFA) [10] or total variability (TV) [11] based compensation methods. TV modeling has rapidly emerged as one of the most powerful approaches for speaker verification and has become the current *de facto* standard. In this approach, closely related to the JFA, the speaker and the channel variabilities of the high-dimensional GMM supervectors are jointly modeled as a single low-rank total-variability space. The low-dimensionality total variability factors extracted from a given speech segment form a vector, named i-vector, which represents the speech segment in a very compact and efficient way. Since the i-vector comprises both speaker and channel variabilities, in the i-vector framework for speaker verification some sort of channel compensation or channel modeling technique usually follows the i-vector extraction process. Regarding channel compensation, linear discriminant analysis (LDA) or within-class covariance normalization (WCCN) are typically applied to compensate for channel nuisance in the i-vector space [12]. Then, the verification score can be obtained either based on a simple cosine similarity between the target user i-vector and the the test utterance i-vector, or by evaluating the test utterance i-vector with a previously trained SVM. In the latter, cosine kernel is usually preferred. Recently, new channel modeling techniques for i-vectors, such as probabilistic linear discriminant analysis (PLDA) [13], have been reported to overcome classical cosine-distance scoring of i-vectors with channel compensation.

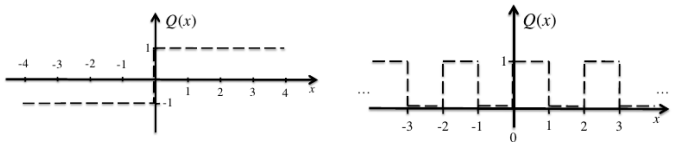

Fig. 1. 1-bit quantization functions.

In our approach we consider the i-vector technique together with SVMs (trained on target and impostor i-vectors) for speaker modeling and scoring. We exploit the total-variability modeling as a sort of factor analysis based front-end extractor able to provide compact speaker representations of fixed length. Moreover, we do not consider any of the previously mentioned channel compensation methods, since this issue is out of the scope of the present study. Nevertheless, it would be straightforward incorporating (at least) LDA and WCCN compensation without significant alterations of the proposed scheme.

III. Secure Binary Embeddings

A secure binary embedding (SBE) [6] is a scheme for converting vectors to bit sequences using band-quantized random projections. These bit sequences, which we will refer to as *hashes*, possess an interesting property: if the Euclidean distance between two vectors is lower than a certain threshold, then the Hamming distance between their hashes is directly proportional to the Euclidean distance between the vectors; if it is higher, then the hashes provide no information regarding the true distance between the two vectors. This scheme is based on the concept of universal quantization (UQ) [14], which redefines scalar quantization by forcing the quantization function to have non-contiguous quantization regions.

Given an L-dimensional vector $\mathbf{x} \in \mathbb{R}^L$, the UQ process converts it to an M-bit binary sequence, where the m^{th} bit is given by

$$q_m(\mathbf{x}) = Q\left(\frac{\langle \mathbf{x}, \mathbf{a}_m \rangle + w_m}{\Delta}\right), \tag{1}$$

where $\langle \cdot, \cdot \rangle$ represents a dot product, $\mathbf{a}_m \in \mathbb{R}^L$ is a random projection vector comprising L i.i.d samples drawn from $\mathcal{N}(\mu = 0, \sigma^2)$, Δ is a precision parameter, w_m is a random dither drawn from a uniform distribution over $[0, \Delta]$ and $Q(\cdot)$ is a quantization function given by $Q(x) = \lfloor x \mod 2 \rfloor$. We can represent the complete quantization into M bits compactly in vector form:

$$\mathbf{q}(\mathbf{x}) = Q\left(\mathbf{\Delta}^{-1}(\mathbf{A}\mathbf{x} + \mathbf{w})\right), \tag{2}$$

where $\mathbf{q}(\mathbf{x})$ is an M-bit binary vector which we will refer to as the *hash* of $\mathbf{x}$, $\mathbf{A} \in \mathbb{R}^{M \times L}$ is a matrix composed of the row vectors $\mathbf{a}_m$, $\mathbf{\Delta}$ is a diagonal matrix with entries Δ and $\mathbf{w} \in \mathbb{R}^M$ is a vector containing the dither values w_m.

The universal 1-bit quantizer of Equation 1 maps the real line onto $1/0$ in a banded manner, where each band is Δ wide. Figure 1 compares conventional scalar 1-bit quantization (left had side panel) with the equivalent universal 1-bit quantization (right hand side panel).

The binary hash generated by the universal quantizer of Equation 2 has the following properties [6]: the probability

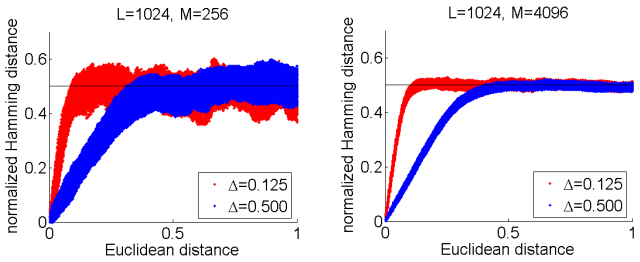

Fig. 2. SBE behavior as a function of Δ for two values of M.

that the i^{th} bits $q_i(\mathbf{x})$ and $q_i(\mathbf{x}')$ of the hashes of two vectors $\mathbf{x}$ and $\mathbf{x}'$, respectively, are identical depends only on the Euclidean distance $d = \|\mathbf{x} - \mathbf{x}'\|$ between the two vectors and not on their actual values. As a consequence, the following relationship can be shown [6]: given any two vectors $\mathbf{x}$ and $\mathbf{x}'$ with a Euclidean distance d, with probability at most e^{-2t^2M} the normalized (per-bit) Hamming distance $d_H(\mathbf{q}(\mathbf{x}), \mathbf{q}(\mathbf{x}'))$ between the hashes of $\mathbf{x}$ and $\mathbf{x}'$ is bounded by:

$$\frac{1}{2} - \frac{1}{2}e^{-\left(\frac{\pi\sigma d}{\sqrt{2}\Delta}\right)^2} - t \leq d_H(\mathbf{q}(\mathbf{x}), \mathbf{q}(\mathbf{x}')) \leq \frac{1}{2} - \frac{4}{\pi^2}e^{-\left(\frac{\pi\sigma d}{\sqrt{2}\Delta}\right)^2} + t,$$

where t is a control factor. The importance of this bound is that the Hamming distance $d_H(\mathbf{q}(\mathbf{x}), \mathbf{q}(\mathbf{x}'))$ is correlated to the Euclidean distance $\|\mathbf{x} - \mathbf{x}'\|$ between the two vectors if and only if the Euclidean distance between the two vectors is below a predetermined threshold (depends on Δ). Specifically, for small d, the expected Hamming distance $E[d_H(\mathbf{q}(\mathbf{x}), \mathbf{q}(\mathbf{x}'))]$ can be shown to be bounded from above by $\sqrt{2\pi^{-1}}\sigma\Delta^{-1}d$, which is linear in d. However, if the distance between $\mathbf{x}$ and $\mathbf{x}'$ is greater than this threshold, $d_H(\mathbf{q}(\mathbf{x}), \mathbf{q}(\mathbf{x}'))$ is bounded by $0.5 - 4\pi^{-2}\exp\left(-0.5\pi^2\sigma^2\Delta^{-2}d^2\right)$, which rapidly converges to 0.5 and effectively gives us no information whatsoever about the true distance between $\mathbf{x}$ and $\mathbf{x}'$.

In order to better illustrate how this scheme works, we randomly generated samples in a high-dimensional space ($L = 1024$) and plotted the normalized Hamming distance between their hashes against the Euclidean distance between the respective samples. This is presented in Figure 2. The number of bits in the hash is also shown in the figures. We note that in all cases, once the normalized distance exceeds Δ, the Hamming distance between the hashes of two vectors ceases to provide any information about the true distance between the vectors. We will find this property useful in developing our privacy-preserving speaker verification system. We also note that changing the value of the precision parameter Δ allows us to adjust the distance threshold until which the Hamming distance is informative. Moreover, increasing the number of bits M leads to a reduction of the variance of the Hamming distance.

A converse property of the embeddings is that for all $\mathbf{x}'$ except the ones that lie within a small radius of any $\mathbf{x}$, $d_H(\mathbf{q}(\mathbf{x}), \mathbf{q}(\mathbf{x}'))$ provides little information about how close $\mathbf{x}'$ and $\mathbf{x}$ are. It can, in fact, be shown that the embedding provides information theoretic security beyond this radius, if the embedding parameters $\mathbf{A}$ and $\mathbf{w}$ remain unknown to the potential eavesdropper. Any algorithm attempting to recover a signal $\mathbf{x}$ from its embedding $\mathbf{q}(\mathbf{x})$ or to infer anything about the relationship between two signals sufficiently far apart using only their embeddings will fail to do so. Furthermore, even in the case where $\mathbf{A}$ and $\mathbf{w}$ are known, it seems computationally intractable to derive $\mathbf{x}$ from $\mathbf{q}(\mathbf{x})$, unless one can guess a starting point very close to $\mathbf{x}$. In effect, it is infeasible to invert the SBE without strong a priori assumptions about $\mathbf{x}$.

IV. Speaker Verification using SBE

The application of the SBE to speaker verification systems is quite straightforward: if the classifier could be made to operate on SBE hashes of supervectors/i-vectors rather than on the supervectors/i-vectors themselves, speaker verification may be performed without exposing the user's data. In this work, we consider non-private reference speaker verification systems that use SVMs for speaker modeling. Then, in order to achieve a privacy-preserving scheme, SVM kernels must be modified to work with Hamming distances between SBE hashes: $k(\mathbf{x}, \mathbf{x}') = e^{-\gamma \cdot d_H^2(\mathbf{q}(\mathbf{x}), \mathbf{q}(\mathbf{x}'))}$. Note that for a given $\mathbf{A}$ and $\mathbf{w}$, the modified kernel closely approximates the conventional RBF for small $d(\mathbf{x}, \mathbf{x}')$, but shows significant differences for larger $d(\mathbf{x}, \mathbf{x}')$. While it does not satisfy Mercer's conditions and cannot be considered a true kernel, in practice it is effective as we shall see in the experiments below. For comparison sake, these experiments include previous work on using supervectors together with SBE hashes for performing privacy-preserving speaker verification, fully described in [5], as well as our current approach using i-vectors together with SBE hashes for addressing the same task.

The implementation of a privacy-preserving speaker verification system is as follows: the user communicates with the server through a smartphone or computation-capable device. In the enrollment phase, the supervectors/i-vectors for both the enrollment recordings and imposter recordings are computed by the user. Imposter recordings may be obtained from any public resource. The user computes SBE hashes from the supervectors/i-vectors and transmits them to the server. He retains the parameters $\mathbf{A}$ and $\mathbf{w}$ employed by the SBE as his private keys. The system trains an SVM with the obtained SBE hashes. During verification, the user computes the SBE hash for the supervector/i-vector obtained from the test recording and transmits it to the system, which classifies it.

The system never observes the actual speech from the user. Its model for the user can only be used with the SBE hashes computed using the private hash key from that specific user, and is not usable without the user's participation. Moreover, an attacker wishing to pose as a specific user must not only manage to steal the embedding parameters from the user's client device, but also gain access to voice recordings from the same user, as either of these alone are insufficient to gain unauthorized access to the server. Therefore, our privacy requirements are satisfied.

A. Experimental Setup

The corpus considered for all experiments is the YOHO Speaker Verification corpus [15], consisting of short utterances by 138 speakers. Each utterance contains a set of three two-digit numbers. The corpus is divided into two sets: enrollment and verification. The enrollment set (used for training) contains 96 utterances from each speaker, totaling 14.54 hours of audio. The verification set (used for testing) contains 40 utterances from each speaker, totaling 6.24 hours of audio. We did

TABLE I. SPEAKER VERIFICATION EER (%AGE), SUPERVECTORS.

#Gauss	4	8	16	32	64	128	256	512	1024
EER	3.55	1.52	0.60	0.25	0.22	0.21	–	–	–

TABLE II. SPEAKER VERIFICATION EER (%AGE), SBE HASHES OF SUPERVECTORS, 32 GAUSSIANS.

leakage	$\sim 5\%$	$\sim 25\%$	$\sim 50\%$
bpc=4	2.27	1.40	1.25
bpc=8	1.32	0.89	0.80
bpc=16	0.76	0.60	0.51

not explicitly record imposters. Instead, for each of the 138 speakers in the corpus, the remaining 137 were used as imposters. Both the supervectors and the i-vectors were based on MFCC features extracted in frames of 25ms, at the rate of 100 frames per second. For each frame we extracted 12 MFCC coefficients and the log-energy, augmenting them with the temporal differences and double-differences to result in a total of 39 features. A UBM was trained from the enrollment data for all the speakers and it was adapted to each recording on the verification set to obtain a single supervector/i-vector.

The choice of the YOHO corpus for a proof of concept of the i-vector based approach was mainly justified by the need to compare our results with the ones obtained in [5]. This was the reason for not consider more challenging corpora such as the NIST Speaker Recognition Evaluation corpus [16] and the MIT Mobile Device Speaker Verification corpus [17].

B. Experiments using feature supervectors

In [5] the authors presented a supervector-based approach for performing privacy preserving speaker verification using SBE hashes. The results they obtained are replicated in Tables I and II, and they will serve as a baseline for our approach. The secure binary embeddings have two parameters that can be customized: the quantization step size Δ and the number of bits M. The value of M by itself is not a useful number, as different values of L (dimensionality of the supervector/i-vector) require different values of M. Therefore, the results are reported in terms of *bits per coefficient* (*bpc*), computed as M/L. Speaker *leakage* in this context refers to the fraction of recordings from any speakers whose SBE hashes have a normalized Hamming distance below the threshold at which Hamming distance d_H is proportional to the Euclidean distance d with respect to any recording from another speaker. This threshold was empirically set at 0.475. The amount of leakage is exclusively controlled by Δ, and its implications will be further discussed in Section V.

C. Experiments using i-vectors

The first step for obtaining the i-vectors was to compute a total variability factor matrix $\mathbf{T}$ on the enrollment set, using the technique described in [18]. The dimension of the total variability sub-space was fixed to 400. Then, 10 EM iterations were applied consisting of an initial ML estimation followed by minimum divergence update. The covariance matrix was not updated in any of the EM iterations. The estimated $\mathbf{T}$ matrix was used for extraction of the total variability factors of the processing speech segments as described in [18]. Finally, SVM speaker models trained with the i-vectors were obtained using the LIBSVM toolkit [19]. The results obtained for the speaker verification task using i-vectors and its privacy-preserving counterpart using SBE hashes are presented in Tables III and IV.

TABLE III. SPEAKER VERIFICATION EER (%AGE), I-VECTORS.

#Gauss	4	8	16	32	64	128	256	512	1024
EER	4.12	1.65	0.74	0.35	0.18	0.15	0.11	0.09	0.07

TABLE IV. SPEAKER VERIFICATION EER (%AGE), SBE HASHES OF I-VECTORS, 256 GAUSSIANS.

leakage	$\sim 5\%$	$\sim 25\%$	$\sim 50\%$
bpc=4	21.21	5.62	3.54
bpc=8	5.55	1.98	1.05
bpc=16	1.27	0.64	0.47

TABLE V. AVERAGE NORMALIZED SPEAKER LEAKAGE ENTROPY.

	supervectors+SBE	i-vectors+SBE
bpc=4	0.884	0.962
bpc=8	0.862	0.941
bpc=16	0.851	0.931

Regarding the non-private approach, we observe that using either supervectors or i-vectors for performing speaker verification on the YOHO corpus produces similar results. The only relevant difference is that only when 64 Gaussians or more are considered, it is better to consider i-vectors instead of supervectors. As for the SBE hashes, in order to compare our results with the ones presented in [5], we ran experiments for the same values of *bpc* and speaker leakage. We performed experiments considering 256 Gaussians instead of 32 Gaussians because, for the i-vectors, it is only when this amount of Gaussians is reached that no significant improvements are obtained in terms of EER on our baseline experiment. We can see that changing either the value of *bpc* or the amount of leakage has much more impact on the classification results when i-vectors are considered instead of supervectors. A possible reason for this might be that the size of the i-vectors does not depend on the number of Gaussians, unlike what happens with the supervectors. This means that each individual feature of the i-vectors contains much more discriminative information than each feature of the supervectors for larger amounts of Gaussians, which leads to the noise introduced by mapping the i-vectors into SBE hashes having much more visible effects. Overall, as expected, both the supervector and the i-vectors approaches obtained good results in the speaker verification task.

V. DATA PRIVACY AND SBE

SBE provides a basic but strong form of security: a vector $\mathbf{x}$ cannot be recovered, even in part, from its hash $\mathbf{q}(\mathbf{x})$, if the projection matrix $\mathbf{A}$ and dither vector $\mathbf{w}$ are unknown. The primary benefit of using SBE is that it now becomes possible for the system to perform classification using the hashes $\mathbf{q}(\mathbf{x})$, without being able to recover the actual data $\mathbf{x}$ from them. Nevertheless, alternative factors that may provide information about the speaker must be considered. One of them is speaker *leakage*, already explained in Section IV-B. Another one is the *normalized entropy* of the distribution over imposters of the leaked vectors for each of the speakers in the verification set. The results obtained in terms of normalized entropy are presented in Table V.

Ideally, we would like to obtain high values for the nor-

malized entropy, as 1 represents completely random behavior and 0 indicates neighborhood to a single speaker. According to the obtained results, for the interesting values of *bpc* and for 50% speaker leakage, the identifiable bias of any speaker towards any other speaker is very low. In particular, when the i-vectors are considered, almost no information regarding any possible speaker clustering is revealed. Therefore, even if the system has registration data from a user, in either case it must retrieve a very large number of putative recordings from a target user to make any inferences about other users in its database. However, this does not provide a strong guarantee of privacy against the motivated adversary.

VI. Conclusion

In this work we presented a comparison between two techniques for privacy-preserving speaker verification using SBE hashes: previous work using feature supervectors and our approach considering i-vectors. We verified that the two approaches performed in a similar fashion both on the baseline experiments and when the SBE hashes were considered. Although for the i-vector approach a more computationally demanding parameter configuration is required for obtaining the desired results, they provide a stronger security regarding the clustering of different users given the information leaked through the Hamming distances between the hashes. Although our technique was presented in the scope of a voice-based speaker verification system, it can be easily extended to other types of biometric authentication.

Acknowledgment

José Portêlo, Alberto Abad and Isabel Trancoso were supported by FCT grants SFRH/BD/71349/2010, PTDC/EIA-CCO/122542/2010 and PEst-OE/EEI/LA0021/2013. Bhiksha Raj was supported by NSF grant number 1017256.

References

[1] M. Pathak and B. Raj, "Privacy Preserving Speaker Verification using adapted GMMs", in *Proc. Interspeech*, Florence, Italy, 2011, pp. 2405–2408.

[2] Y. Lindell and B. Pinkas, "Secure Multiparty Computation for Privacy-Preserving Data Mining", in *Journal of Privacy and Condentiality*, Vol. 1, No. 1, pp. 59–98, 2009.

[3] M. Pathak and B. Raj, "Privacy-Preserving Speaker Verification as Password Matching", in *Proc. ICASSP*, Kyoto, Japan, 2012, pp. 1849–1852.

[4] P. Indyk and R. Motwani, "Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality", in *ACM Symposium on Theory of Computing (STOC)*, Dallas, TX, USA, 1998, pp. 604–613.

[5] J. Portêlo, B. Raj, P. Boufounos, I. Trancoso and A. Abad, "Speaker Verification using Secure Binary Embeddings", in *Proc. Eusipco*, Marrakech, Morocco, 2013.

[6] P. Boufounos and S. Rane, "Secure Binary Embeddings for Privacy Preserving Nearest Neighbors", in *Proc. Workshop on Information Forensics and Security (WIFS)*, Foz do Iguaçu, Brazil, 2011, pp. 1–6.

[7] D. A. Reynolds, T. F. Quatieri and R. B. Dunn, "Speaker Verification using Adapted Gaussian Mixture Models", in *Digital Signal Processing*, Vol. 10, Issues 1–3, pp. 19–41, 2000.

[8] W. M. Campbell, D. E. Sturim and D. A. Reynolds, "Support Vector Machines using GMM Supervectors for Speaker Verification", in *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 13, No. 5, pp. 308–311, 2006.

[9] W. M. Campbell, J. R. Campbell, D. A. Reynolds, E. Singer and P. A. Torres-Carrasquillo, "Support Vector Machines for Speaker and Language Recognition", in *Computer Speech and Language*, Vol. 20, Issues 2–3, pp. 210–229, 2006.

[10] P. Kenny, G. Boulianne, P. Ouellet and P. Dumouchel, "Joint Factor Analysis Versus Eigenchannels in Speaker Recognition", in *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 15, No. 4, pp. 1435–1447, 2007.

[11] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel and P. Ouellet, "Front-End Factor Analysis for Speaker Verification", in *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 19, No. 4, pp. 788–798, 2011.

[12] N. Dehak, R. Dehak, P. Kenny, N. Brümmer, P. Ouellet and P. Dumouchel, "Support Vector Machines Versus Fast Scoring in the Low-Dimensional Total Variability Space for Speaker Verification", in *Proc. Interspeech*, Brighton, United Kingdom, 2009, pp. 1559–1562.

[13] L. Burget, O. Plchot, S. Cumani, O. Glembek, P. Matejka and N. Brümmer, "Discriminatively Trained Probabilistic Linear Discriminant Analysis for Speaker Verification", in *Proc. ICASSP*, Prague, Czech Republic, 2011, pp. 4832–4835.

[14] P. Boufounos, "Universal Rate-Efficient Scalar Quantization", *IEEE Transactions on Information Theory*, Vol. 58, No. 3, pp. 1861–1872, 2012.

[15] J. P. Campbell, "Testing with the YOHO CD-ROM Voice Verification Corpus", in *Proc. ICASSP*, Detroit, MI, USA, 1995, pp. 341–344.

[16] A. Martin and C. Greenberg, "The NIST 2010 Speaker Recognition Evaluation", in *Proc. Interspeech*, Makuhari, Japan, 2010, pp. 2726–2729.

[17] R. Woo, A. Park and T. Hazen, "The MIT Mobile Device Speaker Verification Corpus: Data Collection and Preliminary Experiments", in *Proc. IEEE Odyssey: The Speaker and Language Recognition Workshop*, San Juan, Puerto Rico, 2006, pp. 1–6.

[18] P. Kenny, P. Ouellet, N. Dehak, V. Gupta and P. Dumouchel, "A Study of Inter-Speaker Variability in Speaker Verification", in *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 16, No. 5, pp. 980–988, 2008.

[19] C.-C. Chang and C.-J. Lin, "LIBSVM : A Library for Support Vector Machines", in *ACM Transactions on Intelligent Systems and Technology*, Vol. 2, Issue 3, No. 27, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.