# Using Discrete Probabilities With Bhattacharyya Measure for SVM-Based Speaker Verification

Kong Aik Lee, *Member, IEEE*, Chang Huai You, *Member, IEEE*, Haizhou Li, *Senior Member, IEEE*, Tomi Kinnunen, and Khe Chai Sim

***Abstract*—Support vector machines (SVMs), and kernel classifiers in general, rely on the kernel functions to measure the pairwise similarity between inputs. This paper advocates the use of discrete representation of speech signals in terms of the probabilities of discrete events as feature for speaker verification and proposes the use of Bhattacharyya coefficient as the similarity measure for this type of inputs to SVM. We analyze the effectiveness of the Bhattacharyya measure from the perspective of feature normalization and distribution warping in the SVM feature space. Experiments conducted on the NIST 2006 speaker verification task indicate that the Bhattacharyya measure outperforms the Fisher kernel, term frequency log-likelihood ratio (TFLLR) scaling, and rank normalization reported earlier in literature. Moreover, the Bhattacharyya measure is computed using a data-independent square-root operation instead of data-driven normalization, which simplifies the implementation. The effectiveness of the Bhattacharyya measure becomes more apparent when channel compensation is applied at the model and score levels. The performance of the proposed method is close to that of the popular GMM supervector with a small margin.**

***Index Terms*—Bhattacharyya coefficient, speaker verification, supervector, support vector machine (SVM).**

## I. Introduction

SPEAKER verification is the task of verifying the identity of a person using his/her voice [1]. The verification process typically consists of extracting a sequence of short-term spectral vectors from the given speech signal, matching the sequence of vectors against the claimed speaker's model, and finally comparing the matched score against a verification threshold. Recent advances reported in [1]–[8] show an emerging trend in using support vector machines (SVMs) for speaker modeling. One reason for the popularity of SVM is its good generalization performance.

Manuscript received December 11, 2009; revised April 09, 2010; accepted July 17, 2010. Date of publication August 09, 2010; date of current version March 30, 2011. The work of H. Li was supported in part by the Nokia Foundation. The work of T. Kinnunen was supported by the Academy of Finland under Project 132129 "Characterizing individual information in speech". The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Nestor Becerra Yoma.

K. A. Lee, C. H. You and H. Li are with the Institute for Infocomm Research, Agency for Science, Technology and Research (A*STAR), Singapore 138632 (e-mail: kalee@i2r.a-star.edu.sg; echyou@i2r.a-star.edu.sg; hli@i2r.a-star.edu.sg).

T. Kinnunen is with the School of Computing, University of Eastern Finland, FI- 80101 Joensuu, Finland (e-mail: tkinnu@cs.joensuu.fi).

K. C. Sim is with the School of Computing, National University of Singapore, Singapore 119275 (e-mail: simkc@comp.nus.edu.sg).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TASL.2010.2064308

The key issue in using SVM for classifying speech signals, which have a varying number of spectral vectors, is how to represent them in a suitable form as SVM can only use input of a fixed dimensionality. A common approach is to map the sequences explicitly into fixed-dimensional vectors known as *supervectors*. Classifying variable-length speech sequences is thereby translated into a simpler task of classifying the supervectors. For instance, in [3] speech vectors are mapped to a high-dimensional space via time-averaged polynomial expansion. In [4], speech vectors are used to train a Gaussian mixture model (GMM) via the adaptation of a so-called *universal background model* (UBM). The supervector is then formed by concatenating the mean vectors of the adapted GMM. In [5], supervectors are formed by stacking the likelihood scores with respect to a cohort of anchor models on a per-utterance basis. In [6], the *maximum likelihood linear regression* (MLLR) transform is used to form the supervectors comprising of the transform coefficients. It should be mentioned that the term "supervector" was originally used in [4] and [9] to refer to the GMM supervector. Here, we use similar term in a broader sense referring to any fixed-dimensional vector that represents a speech sequence as a single point in the vector space, having a much higher dimensionality than the original input space.

This paper advocates the use of discrete events (or symbols) and their probabilities to construct supervectors. Discrete events arise naturally in modeling many types of data, for example, letters, words, and DNA sequences. Speech signals can also be represented as sequences of discrete symbols by using a quantizer. Notably, high-level feature extraction (e.g., idiolect, phonotactic, prosody) usually produces discrete symbols. For instance, in [10] speech signals are converted into sequences of phone symbols and then represented in terms of phone $n$-gram probabilities. Discrete probabilities are also useful in modeling prosodic feature sequences [11]. In [7] and [8], we investigated the use of discrete acoustic events derived using the UBM. In this paper, we show that various discrete representations mentioned above can be summarized under the maximum *a posteriori* (MAP) parameter estimation framework [12]. Since they can be unified within similar framework, an SVM kernel designed for one discrete representation would be useful for the others. Another practical virtue of discrete representation is that the estimation of discrete distribution is simple and any arbitrarily shaped distribution is possible since it is non-parametric.

Another challenge concerning the use of supervectors with SVM is feature normalization—the process where the elements of a feature vector are scaled or warped prior to SVM modeling. Feature normalization in the SVM kernel space is closely related

to the similarity measure of supervectors. In this paper, since the supervectors represent the probability distributions of discrete events, we propose using Bhattacharyya coefficient [13] as the similarity measure. The Bhattacharyya measure is symmetric as opposed to other probabilistic measures such as Kullback–Leibler (KL) divergence [14], which is non-symmetric and has to be simplified and approximated substantially to arrive at a symmetric kernel. While the Bhattacharyya measure is simpler, data-independent and more effective, we will also show how it is related to and different from the Fisher kernel [2], term frequency log-likelihood ratio (TFLLR) [10], and rank normalization [15] proposed earlier for similar form of supervectors.

The remainder of this paper is organized as follows. We introduce the MAP framework for the estimation of discrete probabilities in Section II. Using the UBM as a soft quantizer, we describe the process of constructing supervectors using discrete probabilities and show its relevance to the Fisher kernel in Section III. We analyze the Bhattacharyya measure from feature normalization perspective in Section IV. The performance evaluation is reported in Section V. Finally, Section VI concludes the paper.

## II. Estimation of Discrete Probabilities

Let some discrete events $S = \{e_i, i = 1, 2, \ldots, M\}$ have $M$ possible outcomes, and let $\omega_i$ be the probability of observing the $i$th event $e_i$, i.e., $\omega_i = P(e_i)$. Given a speech segment $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}$, our goal is to estimate the probabilities $\Omega = \{\omega_1, \omega_2, \ldots, \omega_M\}$ of observing individual events in $\mathcal{X}$. Using the maximum *a posteriori* (MAP) estimation criterion [12], the solution is given by the mode of the posterior distribution, as follows:

$$\tilde{\Omega} = \arg\max_{\Omega} \log\{P(\mathcal{X} \mid \Omega) g(\Omega)\} \tag{1}$$

where $g(\Omega)$ is the prior distribution of the parameters $\Omega$. Let

$$n_i(\mathcal{X}) = \sum\nolimits_{t:\mathbf{x}_t \sim e_i} 1 \tag{2}$$

denote the number of occurrences of the event $e_i$ in $\mathcal{X}$, where the notation $\mathbf{x}_t \sim e_i$ denotes that the vector $\mathbf{x}_t$, for $t = 1, 2, \ldots, T$, is encoded as the event $e_i$. The MAP estimate can be expressed as

$$\begin{aligned}\tilde{\Omega} &= \arg\max_{\Omega} \log\left\{ g(\Omega) \prod_{t=1}^{T} P(\mathbf{x}_t \sim e_t \mid \Omega, e_t \in S) \right\} \\ &= \arg\max_{\Omega} \log\left\{ g(\Omega) \prod_{i=1}^{M} \omega_i^{n_i(\mathcal{X})} \right\} \\ &= \arg\max_{\Omega} \left\{ \log g(\Omega) + \sum_{i=1}^{M} n_i(\mathcal{X}) \log \omega_i \right\}. \end{aligned} \tag{3}$$

Since the likelihood function $P(\mathcal{X} \mid \Omega)$ follows a multinomial distribution, the prior density can be assumed as a Dirichlet distribution (i.e., a conjugate prior for the parameters of the multinomial distribution) [12], as follows:

$$g(\Omega) = K_c \prod_{i=1}^{M} \omega_i^{\nu_i - 1} \tag{4}$$

where $\nu_i$ are the set of positive parameters for the Dirichlet distribution and $K_c$ is a normalization factor. Using (4) in (3), the MAP estimate can then be solved, subject to the constraints $\sum_{i=1}^{M} \omega_i = 1$ and $\omega_i \geq 0$, using the method of Lagrange multipliers, to give

$$\tilde{\omega}_i = \frac{n_i(\mathcal{X}) + \nu_i - 1}{\sum_{j=1}^{M} [n_j(\mathcal{X}) + \nu_j - 1]},\ i = 1, 2, \ldots, M. \tag{5}$$

The MAP estimate of discrete probabilities in (5) is given by the sum of the observed statistics $n_i$ and the parameters $\nu_i$ of the prior distribution. For a flat prior, whereby $\nu_i - 1 = 0$, (5) reduces to the popular maximum-likelihood (ML) estimate

$$\tilde{\omega}_i = \frac{n_i(\mathcal{X})}{T},\ i = 1, 2, \ldots, M \tag{6}$$

since $\sum_{i=1}^{M} n_i(\mathcal{X}) = T$. ML estimate is used when we have higher belief in the observed statistics than the prior information.

We assume in (5) that a rule of correspondence has been defined between the speech feature vectors and the set of events such that $n_i(\mathcal{X})$ for $i = 1, 2, \ldots, M$ represent the counts of observing those events in $\mathcal{X}$. In (2), speech feature vectors are quantized as discrete events or symbols on a frame-by-frame basis. These events may correspond to the codewords of a vector quantization (VQ) codebook [14] or the Gaussian densities in a UBM as shown in Section III. The discrete events may also correspond to abstract linguistic units such as phonemes, syllables, words, or subsequences of $n$ symbols (i.e., $n$-grams). For instance, in spoken language recognition [16] and speaker recognition utilizing high-level features [10], [11], the events represent $n$-grams of phones, words or some prosodic features. In these methods, phone recognizers or prosodic feature extractors are used to discover the events set from the speech signals.

## III. Constructing Supervector Using Discrete Probabilities

### A. UBM as Soft Quantizer

A universal background model (UBM) is a GMM trained to represent a speaker-independent distribution [17]. In this regard, the UBM is usually trained, using the expectation maximization (EM) algorithm [14], from tens or hundreds of hours of speech data gathered from a large number of speakers.

A UBM, denoted by $\boldsymbol{\Theta}$, with $M$ mixture components is characterized by the following probability density function:

$$p(\mathbf{x} \mid \boldsymbol{\Theta}) = \sum_{i=1}^{M} \lambda_i \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \tag{7}$$

where $\lambda_i$ is the mixture weight, $\boldsymbol{\mu}_i$ is the mean vector, and $\boldsymbol{\Sigma}_i$ is the covariance matrix of the $i$th Gaussian component. The mixture weights satisfy the constraint $\sum_{i=1}^{M} \lambda_i = 1$ and the covariance matrices are assumed to be diagonal in this paper. Let each of the Gaussian densities represent a discrete event $e_i$.

Given a speech segment $\mathcal{X}$, the number of occurrences of event $e_i$ is computed by accumulating the posterior probabilities

$$P(i \mid \mathbf{x}_t, \boldsymbol{\Theta}) = \frac{\lambda_i \mathcal{N}(\mathbf{x}_t \mid \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)}{\sum_{j=1}^{M} \lambda_j \mathcal{N}(\mathbf{x}_t \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \tag{8}$$

evaluated for the $i$th Gaussian component for the whole utterance, as follows:

$$n_i(\mathcal{X}) = \sum_{t=1}^{T} P(i \mid \mathbf{x}_t, \boldsymbol{\Theta}). \tag{9}$$

The UBM quantizes the input vectors into discrete symbols, much similar to the VQ codebook except that the codewords are now modeled as Gaussian densities. Since the Gaussian densities can be overlapped, rather than partitioned, soft membership can be computed based on the Bayes rule as given in (8). The UBM $\boldsymbol{\Theta}$ together with (9) thereby define the set of discrete events, $S$, and the rule of correspondence between the feature vectors and the events.

Finally, to obtain the MAP estimate, the parameters $\nu_i$ in (5) are set to

$$\nu_i - 1 = \tau \cdot M \cdot \lambda_i \tag{10}$$

where $\lambda_i$ are the weights of the UBM and the controlled parameter $\tau$ has to be greater or equal to 0. This is known as the $\tau$-initialization method in [12]. Feasible values for $\tau$ range from 0 to 1, which we have found effective for this application. Equation (10) controls the broadness of the prior density $g(\Omega)$ in (4) with the parameter $\tau$. When $\tau$ is large, the prior density is sharply peaked around the UBM weights $\lambda_i$, in which case the resulting MAP estimate approaches $\lambda_i$. Conversely, if $\tau$ is small the MAP estimate approaches the ML estimate. In particular, the MAP estimate in (5) reduces to the ML estimate in (6) for $\tau = 0$.

### *B. Constructing Supervector*

The discrete probabilities $\tilde{\Omega} = \{\tilde{\omega}_1, \tilde{\omega}_2, \ldots, \tilde{\omega}_M\}$ can be conveniently represented in functional form as $P(h)$, where the variable $h$ represents any event in $S$ such that $P(h = e_i) = \tilde{\omega}_i$ for $h \in S$. That is, the function $P(h)$ is the probability mass function (PMF) [14]. We can express the PMF in vector form as

$$\mathbf{p} = [P(e_1), P(e_2), P(e_M)]^{\mathrm{T}} = [\tilde{\omega}_1, \tilde{\omega}_2, \ldots, \tilde{\omega}_M]^{\mathrm{T}} \tag{11}$$

where the superscript $^{\mathrm{T}}$ denotes transposition. The vector $\mathbf{p}$ has a fixed dimensionality $M$ equivalent to the cardinality of the event set $S$. It represents the speech segment $\mathcal{X}$ in terms of the distribution of discrete events observed in $\mathcal{X}$. These attributes fulfill our requirement of supervector representation. For the case of UBM, the relation between $\tilde{\Omega}$ and $\mathcal{X}$ is given by (5) and (9), and the dimensionality of the supervector is determined by the number of Gaussian densities in the UBM.

### *C. Fisher Information*

The concept of mapping sequences into supervectors is commonly interpreted as a sequence (or dynamic) kernel. The earliest example of sequence kernel can be traced back to [18] in which the *Fisher kernel* was proposed. The Fisher kernel maps a sequence into a supervector by taking the derivatives of the log-likelihood function with respect to the parameters of the model. Let the model be the UBM as defined in (7). Taking the derivative of the log-likelihood function with respect to the weights $\lambda_i$, for $i = 1, 2, \ldots, M$, and normalizing by the duration $T$, we obtain

$$F_i = \frac{\nabla_{\lambda_i} \log p(\mathcal{X} \mid \boldsymbol{\Theta})}{T} = \frac{\sum_{t=1}^{T} P(i \mid \mathbf{x}_t, \boldsymbol{\Theta})}{\lambda_i \cdot T}. \tag{12}$$

We deliberately write (12) in terms of $P(i \mid \mathbf{x}_t, \boldsymbol{\Theta})$, as defined in (8), to establish the connection to our earlier discussion. Using (9) in (12) and normalizing by the respective Fisher information [18] we arrive at

$$F_i' = \frac{n_i(\mathcal{X})/T}{\sqrt{I_i}} \tag{13}$$

where the constant $\lambda_i$ has been absorbed as part of the Fisher information definition as $I_i = E\{(\lambda_i F_i)^2\}$, where $F_i$ is as defined in (12).

Notice that $n_i(\mathcal{X})/T$ gives the ML estimate as shown in (6). Hence, Fisher mapping essentially boils down to the ML estimate of discrete distribution, with additional normalization factors depending on the Fisher information. The supervector [cf. Equation (11)] is now given by

$$\mathbf{p}'_{\mathrm{FISHER}} = \left[\frac{\tilde{\omega}_1}{\sqrt{I_1}}, \frac{\tilde{\omega}_2}{\sqrt{I_2}}, \ldots, \frac{\tilde{\omega}_M}{\sqrt{I_M}}\right]^{\mathrm{T}}. \tag{14}$$

In practice, the Fisher information $I_i$, $i = 1, 2, \ldots, M$, are estimated by replacing the expectation with sample average computed from a large background corpus. The Fisher information normalizes individual dimensions of the supervector to the same scale corresponding to the mean-square value of the discrete probabilities estimated from the background samples. By so doing, all dimensions are treated equally in the mean-square sense when used as inputs to SVM.

Recall that the Fisher mapping in (12) was obtained by taking the derivative of the log-likelihood function with respect to the weights of the UBM. In addition to the weights, taking the derivative with respect to the mean vectors and covariance matrices, as originally proposed in [18], increases the dimensionality of the supervector. These additional dimensions are not considered in this paper as they do not correspond to any discrete probability interpretation, which is the focus of the paper. A full account of using this form of supervector for speaker verification can be found in [2].

## IV. Bhattacharyya Measure

The Bhattacharyya coefficient [13] is commonly used in statistics to measure the similarity of two probability distributions. It is computed by integrating the square root of the product of the two distributions. For discrete distributions, the Bhattacharyya coefficient is given by

$$\rho = \sum_{h \in S} \sqrt{P_a(h) P_b(h)} \tag{15}$$

where $S$ is the set of discrete events. The coefficient $\rho$ lies between zero and unity, where $\rho = 1$ indicates that the two distri-

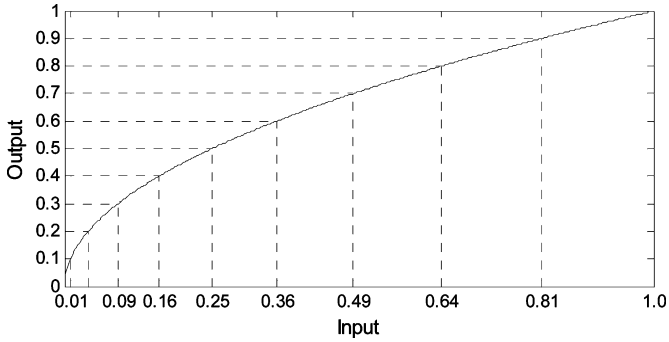



Fig. 1. Horizontal axis is warped according to a square-root function.

butions are fully overlapped, while $\rho = 0$ occurs for the case of non-overlapping distributions.

Using $P_a(h = e_i) = \tilde{\omega}_{i,a}$ and $P_b(h = e_i) = \tilde{\omega}_{i,b}$ in (15), the Bhattacharyya measure can be written in vector form as

$$\rho = \sum_{i=1}^{M} \sqrt{\tilde{\omega}_{i,a}\tilde{\omega}_{i,b}} = \mathbf{p'}_a^{\mathrm{T}}\mathbf{p'}_b \tag{16}$$

where the supervector is now given by

$$\mathbf{p'}_{\mathrm{BHAT}} = \left[\sqrt{\tilde{\omega}_1}, \sqrt{\tilde{\omega}_2}, \ldots, \sqrt{\tilde{\omega}_M}\right]^{\mathrm{T}}. \tag{17}$$

Clearly, the Bhattacharyya measure is symmetric and it represents an inner product in the supervector space. Hence, it can be used as a legitimate kernel function [19] in SVM.

From feature normalization perspective, the square-root operator has an effect in normalizing the contribution of individual dimensions to the inner product. As shown by the solid curve in Fig. 1, higher gain is applied to rare events, i.e., those events with lower probabilities. The gain reduces gradually (so as the slope of the curve) when the input approaches unity. By so doing, the situation where rare events are outweighed by those with higher probabilities is avoided. The square-root operator could also be interpreted as a warping function, where the horizontal axis is shrunk for inputs close the zero and stretched for inputs close to unity, as depicted Fig. 1. This is different from the normalization scheme in the Fisher kernel, where constant scaling is applied to individual dimension based on the Fisher information estimated from a background corpus.

### A. Term Frequency Log-Likelihood Ratio (TFLLR)

Term frequency log-likelihood ratio (TFLLR) was introduced in [10] for the scaling of $n$-gram probabilities. Since each $n$-gram can be regarded as a discrete event $e_i$, the $n$-gram probabilities can be expressed as PMF and supervector as given in (11). In this regard, the event set $S$ consists of all unique $n$-grams. The TFLLR scales individual dimensions of the supervector (i.e., the $n$-gram probabilities) in proportion to the square root of the inverse $n$-gram probabilities computed from a large background corpus. Denoting the background probabilities as $\lambda_i$, the supervector is now given by

$$\mathbf{p'}_{\mathrm{TFLLR}} = \left[\frac{\tilde{\omega}_1}{\sqrt{\lambda_1}}, \frac{\tilde{\omega}_2}{\sqrt{\lambda_2}}, \ldots, \frac{\tilde{\omega}_M}{\sqrt{\lambda_M}}\right]^{\mathrm{T}}. \tag{18}$$

For discrete probabilities derived from the UBM quantizer, the background probabilities $\lambda_i$ correspond directly to the weights of the UBM since the weights are estimated from a large background corpus.

The TFLLR de-emphasizes frequent events and emphasizes rare events. This is similar in spirit with the Bhattacharyya measure, except that individual dimension is subjected to constant scaling instead of warping. Hence, TFLLR scaling falls into the same category as the Fisher kernel from the perspective of feature normalization in the supervector space.

### B. Rank Normalization

In [15], rank normalization was proposed for normalizing the supervectors of $n$-gram probabilities. Elements of the supervector are processed separately where a warping function is used for mapping each dimension to a uniform distribution over the interval from zero to unity. The warping function is non-parametric and is derived from the concept of cumulative density function (cdf) matching, similar to that used in *histogram equalization* (HEQ) [20] and *feature warping* [21]. Since the cdf of the targeted uniform distribution is linear (and monotonically increasing) in the interval $[0, 1]$, cdf matching amounts to a procedure where each element of the supervector is replaced by its rank in a background corpus. Denoting the rank of $\tilde{\omega}_i$ as $r_i$, the supervector is now given by

$$\mathbf{p'}_{\mathrm{rank}} = \left[\frac{r_1}{R}, \frac{r_2}{R}, \ldots, \frac{r_M}{R}\right]^{\mathrm{T}} \tag{19}$$

where $R$ is the number of reference samples in the background data. Let $B_i$ be the set of $R$ values for the $i$th dimension. The rank $r_i$ of $\tilde{\omega}_i$ is given by the number of elements in the background set $B_i$ whose values are smaller than $\tilde{\omega}_i$

$$r_i = \left|\{b \in B_i : b < \tilde{\omega}_i\}\right| \tag{20}$$

where $|\cdot|$ denotes the cardinality of a set and $|B_i| = R$.

In Section II, we assume that the probability estimates $\tilde{\Omega}$ follow a Dirichlet distribution. This assumption implies that each element $\tilde{\omega}_i$, when treated independently, follows a beta distribution [22], which is warped to a uniform distribution via rank normalization. The normalization stretches the high-density areas of the feature space and shrinks it in areas of low density. The warping functions are shown in Fig. 2 for individual dimensions and their ensemble average.

Comparing Fig. 2 to Fig. 1, it can be seen that the warping function closely resembles the square-root curve in the sense that the input axis is shrunk for values closer to the origin and stretched at the other end. However, there is an important difference regarding computational complexity. Since rank normalization is non-parametric, the background sets $B_i$ have to be stored and therefore computation of (20) is far more expensive than the square-root operation. Recall that the Bhattacharyya coefficient has a dynamic range bounded between zero and unity. This is unachievable with rank normalization, where the inner product of the supervectors in (19) generally results in unpredictable dynamic range. Similar problem happens for the Fisher kernel and TFLLR scaling. This has profound impact on the performance of the SVM, as shown in Section V.

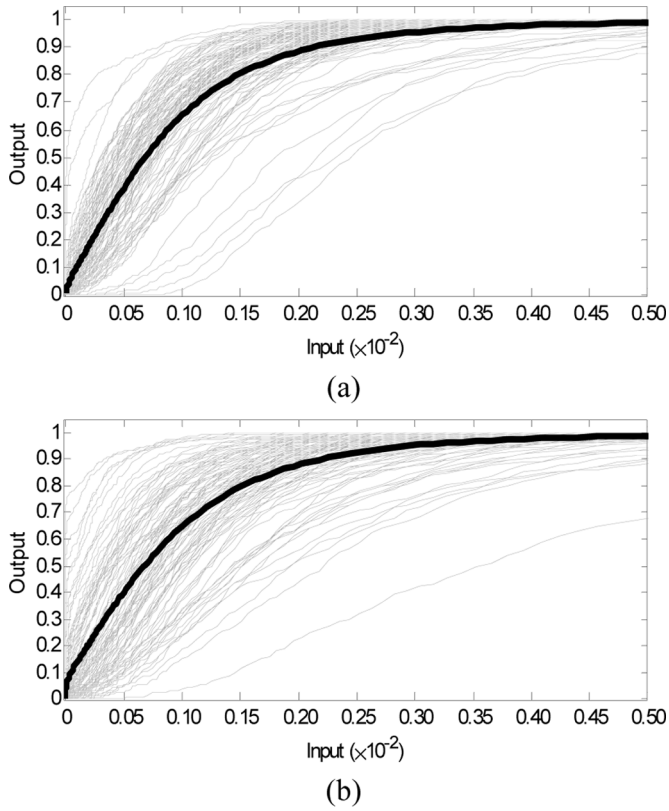



Fig. 2. Warping functions produced by rank normalizing individual dimension of the supervector for (a) male and (b) female populations. The solid curves were obtained by ensemble averaging the warping functions across all dimensions, where the size of UBM was set to $M = 1000$ in the experiment.

## V. Experiments and Results

### A. Experimental Setup

The experiments were carried out on the NIST 2006 speaker recognition evaluation (SRE) task [23]. The core task consists of 810 target speakers, each enrolled with one side of a five-minute conversation, which roughly contains two minutes of speech. There are 3616 genuine and 47 452 imposter trials, where test utterances are scored against target speakers of the same gender. All speech utterances were first preprocessed to remove silence and converted into sequences of 36-dimensional feature vectors, each consisting of 12 Mel frequency cepstral coefficients (MFCCs) appended with deltas and double deltas. Relative spectral (RASTA) filtering [24] and utterance-level mean and variance normalization were performed. We use two well-known metrics in evaluating the performance of the speaker verification systems—equal error rate (EER) and the minimum detection cost function (MinDCF) [23]. The EER corresponds to the decision that gives equal false acceptance rate (FAR) and false rejection rate (FRR). The MinDCF is defined as the minimum value of the function $0.1 \times \text{FRR} + 0.99 \times \text{FAR}$.

The speaker verification system was designed to be gender-dependent.[1] Two gender-dependent UBMs were trained using data drawn from the NIST 2004 dataset. The same dataset was used to form the background data for SVM training. The commonly available libSVM toolkit [25] was used for this purpose. HTK toolkit [26] was used for training the UBMs.

[1]Gender information is provided and there is no cross-gender trial in NIST SREs. Gender-dependent systems have shown better result than gender-independent systems in past evaluations.

### B. ML Versus MAP Estimation

This section investigates the difference between ML and MAP estimation and the influence of the parameter $\tau$ on the system performance. We increased the value of $\tau$ in (10) from 0 to 1.0 with a step size of 0.1. Recall that setting $\tau = 0$ leads to the ML estimate in (6). Similar procedure was repeated for various sizes of UBM from 128 to 4096. In all the experiments Bhattacharyya measure was used for normalizing the supervectors and t-norm [27] was performed at the score level (see Section V-D for more details about t-norm). The results are presented in Fig. 3(a). It can be seen that, for $\tau = 0.1, 0.2, \ldots, 1.0$, MAP estimation gives lower EER as compared to the ML estimation of discrete probabilities. This observation is consistent across different UBM sizes. The empirical results in Fig. 3(a) also show that the optimum value of $\tau$ varies for different UBM size $M$. For instance, the optimum $\tau$ for $M = 2048$ is 0.8, but the value changes to 0.7 for larger UBM with $M = 4096$. To further investigate the influence of $\tau$ on MAP estimation, we gradually increased the value from 0.1 to 100. Fig. 3(b) shows the EER as a function of $\tau$. Notice that we took the ensemble average over different UBM sizes in order to smooth out the empirical noise from individual curves. It can be observed that EER increases drastically for $\tau$ greater than 10. Larger value of $\tau$ pushes the MAP estimate toward the prior weights. This weakens the effect of the observed statistics, which contain speaker characteristics. In particular, using a very large $\tau$ in (10) and (5) would cause the MAP estimation to give the prior weights as the probabilities estimate, and thus losing all speaker-related information. In general, the parameter $\tau$ has to be optimized empirically for a given set of data conditions (duration, signal-to-noise ratio, etc.).

The size of the UBM (i.e., the cardinality of the discrete event set) has a great impact on the performance as shown in Fig. 3(a). It can be seen that the EER reduces as the size of the UBM increases. Similar trend can be observed for the ML and MAP with different values of $\tau$. These results motivate us to use larger UBM, and therefore larger event set, for discrete probabilities modeling in the next and subsequent sections.

### C. Computational Speed Up: Gaussian Selection

For a large UBM (large as compared to the dimensionality of the feature vectors) an input vector will be close only to a few Gaussian components. We can therefore compute the probabilities of a small subset of components located in the vicinity of the input vector; the remaining components are assumed to have zero probability. *Gaussian selection* technique [28] as described below can then be used to speed up the probability computation in (8).

A smaller GMM, referred to as the *hash model* [28], is trained with the same training data as the UBM. Mahalanobis distance is then computed for each pair of Gaussian components of the UBM and the hash model. Since the Gaussian components may have different covariance matrices, their average is used in computing the Mahalanobis distance. A shortlist is then generated

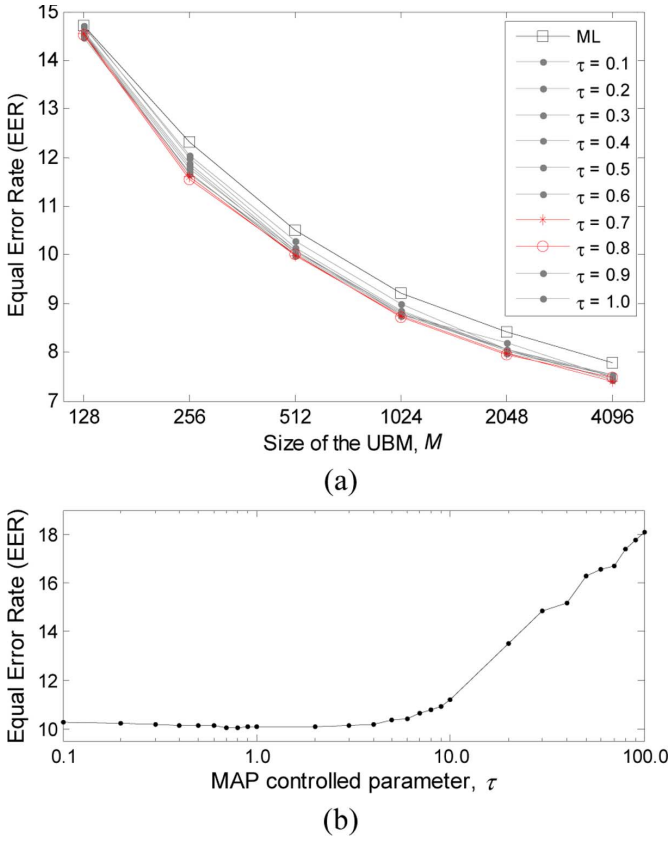



Fig. 3. Performance comparison between ML and MAP in terms of EER. (a) The value of $\tau$ was increased from 0 to 1.0 with a step size of 0.1 for UBM with various sizes. Letting $\tau = 0$ leads to the ML estimation. The curves for $\tau = 0.7$ and $\tau = 0.8$, which are farther apart from the ML curve, are highlighted in red. (b) The EER was evaluated as a function of $\tau$ with value increases from 0.1 to 100. Significant degradation in EER can be observed for $\tau$ greater than 10.

for each component of the hash model. The shortlist contains indices of those components of the UBM having the closest distance to a particular component of the hash model. For a given input vector, we first determine the top scoring component in the hash model. The probabilities of the components in the shortlist of the top-scoring component are then computed.

Let $H$ be the size of the hash model and $Q$ be the length of the shortlists. Gaussian selection results in $M/(H+Q)$ times faster computation, where $M$ is the order of the UBM. Table I shows the average accuracy of the Gaussian selection technique for different sizes of UBM. We evaluate the accuracy by comparing the probability estimates obtained with and without Gaussian selection in terms of the Bhattacharyya measure (16) averaged over 100 random samples, which were selected from our development data. Formally,

$$\text{Average accuracy} = \frac{1}{K}\sum_{k=1}^{K}\left(\sum_{i=1}^{M}\sqrt{\tilde{\omega}_{i,k}\hat{\omega}_{i,k}}\right) \tag{21}$$

where $\{\tilde{\omega}_{1,k}, \tilde{\omega}_{2,k}, \ldots, \tilde{\omega}_{M,k}\}$ and $\{\hat{\omega}_{1,k}, \hat{\omega}_{2,k}, \ldots, \hat{\omega}_{M,k}\}$ are the probability estimates obtained with and without Gaussian selection for the $k$th utterance, respectively, and $K = 100$ is the number of utterances. We fixed the ratio $M/H$ to be 32, and $Q = 4H$, which results in 6.4 times faster computation. Also,

TABLE I
ACCURACY $\rho \times 100\%$ OF GAUSSIAN SELECTION TECHNIQUE AVERAGED OVER 100 SAMPLES FOR DIFFERENT SIZES OF UBM AND HASH MODEL. THE COMPUTATION SPEED-UP IS FIXED AT $\times$ 6.4 FOR ALL COMBINATIONS

| Size of the UBM, $M$ | Size of the hash model, $H$ | Average accuracy (×100%) | |
|---|---|---|---|
| | | Female | Male |
| 1024 | 32 | 98.59 | 98.19 |
| 2048 | 64 | 99.19 | 99.13 |
| 4096 | 128 | 99.48 | 99.46 |
| 8192 | 256 | 99.62 | 99.58 |
| 16384 | 512 | 99.68 | 99.68 |

we used $\nu - 1 = 0$ in the experiment to mitigate the influence of prior on the result. Considerably high accuracy ($\approx 99\%$) is achieved with Gaussian selection technique for all cases listed in Table I.

### D. Nuisance Attribute Projection and SVM Training

Since a supervector represents a speech utterance as a single point in the vector space, it becomes possible to remove unwanted variability, due to different handsets, channels, and phonetic content, from the supervector by linear projection. Let $\mathbf{E}$ be an $M$-by-$N$ matrix representing the unwanted subspace that causes the variability. Nuisance attribute projection (NAP) [29] removes the unwanted variability from a supervector via a projection to the subspace complementary to $\mathbf{E}$, as follows:

$$\mathbf{p}'' = (\mathbf{I} - \mathbf{E}\mathbf{E}^{\mathrm{T}})\mathbf{p}'. \tag{22}$$

NAP assumes that the variability is confined in a relatively low-dimensional subspace such that $N \ll M$. The columns of $\mathbf{E}$ are the eigenvectors of the within-speaker covariance matrix estimated from a development dataset with a large number of speakers, each having several training sessions.

In (22), $\mathbf{p}'$ denotes the supervectors that have been normalized with any of the methods mentioned listed in Sections III and IV. SVM modeling is then performed in the supervector space that has been properly scaled or warped and compensated for session variability. The discriminant function of an SVM [19] can be expressed in terms of the supervector as follows:

$$f(\mathbf{p}'') = \sum_{l=1}^{L}\alpha_l y_l (\mathbf{p}_l'')^{\mathrm{T}}\mathbf{p}'' + \beta. \tag{23}$$

where $L$ is the number of support vectors, $\alpha_l$ are the weights assigned to the $l$th support vector with its label given by $y_l \in \{-1, +1\}$ and $\beta$ is the bias parameter.

Table II shows the results with and without feature normalization, channel compensation and score normalization. Notably, 15.40% relative improvement in EER and 12.31% relative improvement in MinDCF are obtained by applying the Bhattacharyya measure on the raw discrete probabilities. Further improvement (24.55% in EER and 21.90% in MinDCF) is obtained with NAP and t-norm which compensate for session variability at the model and score levels, respectively. Feature normalization is essential for effective SVM modeling. The reason is that SVMs are not invariant to linear transformations,

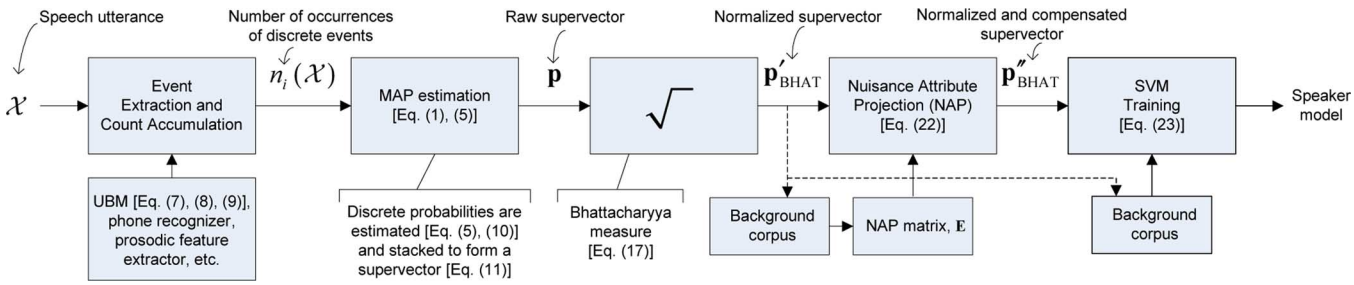



Fig. 4. Speech utterance $\mathcal{X}$ is mapped to a supervector of discrete probabilities $\mathbf{p}$, normalized in accordance with the Bhattacharya measure, and channel compensated prior to SVM modeling. Similar mapping operation is performed on the training utterance of the target speaker, all the utterances in the background corpus (as indicated by the dotted lines), and test utterances (not shown in the figure).

TABLE II
COMPARISON OF EER AND MINDCF FOR SUPERVECTORS WITH AND WITHOUT CHANNEL COMPENSATION AND SCORE NORMALIZATION

| Supervector | % EER | MinDCF (×100) |
|---|---|---|
| Raw, $\mathbf{p}$ | 8.57 | 3.90 |
| + NAP | 7.05 | 3.31 |
| + NAP + t-norm | 6.60 | 3.15 |
| Bhattacharyya, $\mathbf{p}'_{\text{BHAT}}$ | 7.25 | 3.42 |
| + NAP | 5.53 | 2.73 |
| + NAP + t-norm | 4.98 | 2.46 |

i.e., any form of scaling would cause some of the dimensions to dominate the overall decision.

For the experiments in Table II, the UBM has a model size of $M = 16384$, while the hash model for Gaussian selection has model size of 512. For the MAP estimation, the parameter $\tau$ in (10) is set to 0.1. For the NAP, the projection matrix has a rank of 60 and was derived from NIST 2004 and 2005 SRE datasets. For the score normalization [27], t-norm cohorts were selected from NIST 2005 SRE dataset. We use the same configuration for subsequent experiments. The overall process from supervector construction to SVM training is summarized and illustrated in Fig. 4. Also included in the figure are references to equations used at each stage.

### *E. Comparison of Normalization Methods*

We compare the performance of the Bhattacharyya measure, Fisher kernel, TFLLR scaling and rank normalization using the same configuration as mentioned above in Table III (see the upper panel). Fig. 5 shows the detection error trade-off (DET) curves. It can be seen that the Fisher kernel and TFLLR scaling perform better than just using the raw discrete probabilities, which indicates that kernel normalization is important. Comparing these results to the Bhattacharyya measure, on the other hand, shows that the square-root operator is more appropriate than constant scaling in the Fisher kernel and TFLLR scaling. The Bhattacharyya measure performs consistently better than the rank normalization in terms of EER and MinDCF. The effectiveness of the rank normalization depends on the extent the supervectors matches the background distribution.

It is also possible to use bigram (i.e., subsequences of two Gaussian indexes) probabilities to construct supervectors and to compare the performance of various normalization methods. For a UBM of size $M'$, bigram probability modeling leads to a set of $M = M' \times M'$ discrete events. Let $M' = 128$ be the size of the UBM, the supervector of bigram probabilities will have

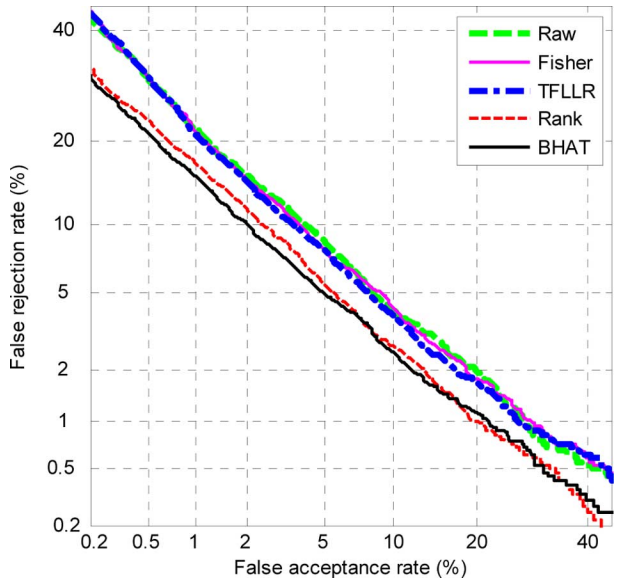



Fig. 5. DET curves showing a comparison of various normalization methods (or kernels) on the supervector of discrete probabilities.

TABLE III
COMPARISON OF EER AND MINDCF FOR DIFFERENT NORMALIZATION METHODS AND SUPERVECTORS. NAP AND t-NORM WERE APPLIED USING EXACTLY THE SAME DATASET

| *Unigram supervector* | % EER | MinDCF (×100) |
|---|---|---|
| Raw | 6.60 | 3.15 |
| Fisher kernel | 6.47 | 3.11 |
| TFLLR scaling | 6.25 | 3.07 |
| Rank normalization | 5.23 | 2.63 |
| Bhattacharyya measure | 4.98 | 2.46 |
| *Bigram supervector* | % EER | MinDCF (×100) |
| Raw | 15.49 | 5.96 |
| Fisher kernel | - | - |
| TFLLR scaling | 13.66 | 5.85 |
| Rank normalization | 13.77 | 5.68 |
| Bhattacharyya measure | 11.99 | 4.80 |

a dimensionality of $M = 16384$. The lower panel of Table III shows the performance of bigram supervector using different normalization methods. We used exactly the same training data and parameter settings for all the experiments in Table III. It can be seen that the bigram supervector gives poorer accuracy compared with the unigram supervector. This is likely due to the fact that we have significantly reduced the size of the UBM to

$M' = 128$ so that the resulting bigram supervector has the same dimensionality as the unigram supervector. For most low-level acoustic quantizers, e.g., UBM and VQ codebook, the event set that contains only the unigrams can be made sufficiently large. This is different from high-level events, e.g., phones [10] and prosodic features [11], which usually rely on bigram or trigram to form a larger event set.

It is worth noting that bigram supervector gives better result when the same UBM is used for deriving the unigram supervector. This can be seen from Fig. 3(a), where the EER is around 14.5% for unigram supervector with $M = 128$, compared to 11.99% of EER (in the last row of Table III) for bigram supervector with $M = 128 \times 128$. Clearly, bigram probabilities are useful but not as effective as simply increasing the UBM size to obtain unigram supervector having the same dimensionality.

Regarding the normalization of the bigram supervectors, our conclusion is clear—Bhattacharyya measure performs consistently better than other normalization methods for both unigram and bigram supervectors. A McNemar's statistical test [14], [30] was conducted to see if the EER and MinDCF of the Bhattacharyya measure are significantly better than other normalization methods. The $p$-values obtained were all less than 0.05, which means that the improvements are significant with a confidence level of 95%. Notice that we do not provide results for Fisher kernel in the lower panel of Table III as the kernel is not readily applicable to bigram probabilities.

### F. Comparison and Fusion of Supervectors

Finally, we evaluate the performance of the supervector of discrete probabilities (with the Bhattacharyya measure) in comparison with the GLDS kernel [3] and GMM supervector [4]. For the GLDS kernel, we used all monomials up to the third order. The resulting supervectors have a dimensionality of 9139. For the GMM supervector, the UBM consists of 512 mixtures leading to supervectors of dimensionality 18 432. Recall that the supervector of discrete probabilities has a comparable dimensionality of $M = 16384$. The datasets used for UBM training, SVM background data, NAP, and t-norm are the same for all systems.

Table IV shows the EER and MinDCF. Fig. 6 shows the DET curves. The Bhattacharyya system exhibits competitive performance compared to the other two systems, with the GMM supervector being the best. We fused the Bhattacharyya system with the other two at the score level using equal weights summation. The fusion with the GLDS gives relative improvement of 15.06% in EER and 8.13% in MinDCF over the best single system. Some improvement can also be observed for the fusion with the GMM supervector, which amounts to 5.35% and 0.95% relative reduction in EER and MinDCF, respectively, over the best single system. A McNemar statistical test [14], [30] was conducted to see if the differences in EER and MinDCF are significant. The $p$-values are shown in Table V. Clearly, the improvement in EER is significant at a confidence level of 95% for both fusions since the $p$-values are less than 0.05. The improvement in MinDCF for the first fusion is also significant with 95% confidence, which, however does not hold for the second fusion.

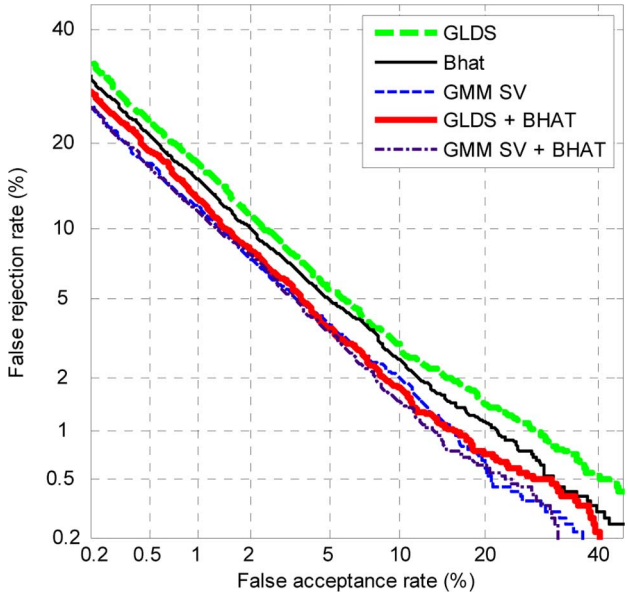



Fig. 6. DET curves showing a comparison of the performance of the GLDS kernel, GMM supervector, and the fusion with the proposed Bhattacharyya system.

TABLE IV
COMPARISON OF EER AND MinDCF FOR DIFFERENT SUPERVECTORS. NAP AND t-NORM WERE APPLIED USING EXACTLY THE SAME DATASET. THE FUSION RESULTS WERE OBTAINED VIA LINEAR COMBINATION WITH EQUAL WEIGHTS AT THE SCORE LEVEL

| Supervector | % EER | MinDCF (×100) |
|---|---|---|
| Bhattacharyya (proposed) | 4.98 | 2.46 |
| GLDS | 5.39 | 2.67 |
| GMM supervector | 4.30 | 2.10 |
| Bhattacharyya + GLDS | 4.23 | 2.26 |
| Bhattacharyya + GMM supervector | 4.07 | 2.08 |

TABLE V
$p$-VALUES OF MCNEMAR'S TESTS ON THE DIFFERENCES IN THE PERFORMANCE OF THE FUSION COMPARED TO THE SINGLE BEST SYSTEM FOR OPERATING POINTS AT EER AND MinDCF. A $p$-VALUE LESS THAN 0.05 MEANS THAT WE ARE OBSERVING A SIGNIFICANT DIFFERENCE IN THE PERFORMANCE AT A CONFIDENCE LEVEL OF 95%

| | The best single system | |
|---|---|---|
| Operating point: | EER | MinDCF |
| Bhattacharyya + GLDS | $2.2\times10^{-16}$ | $1.5\times10^{-2}$ |
| Bhattacharyya + GMM supervector | $1.8\times10^{-4}$ | $6.8\times10^{-1}$ |

The fusion of Bhattacharyya and GMM supervector is less successful because the same datasets were used for UBM training, SVM background data, NAP, and t-norm. This was purposely done so as to have controlled comparisons. We would like to emphasize that, even though we use the UBM as the quantizer in this paper, the occurrence counts of the discrete events could come from a phone recognizer, a prosodic feature extractor or a speech recognition system as shown in Fig. 4. We anticipate that, had we chosen such a completely different

front-end, we would likely observe higher fusion gain. This is a point for future research.

## VI. Conclusion

Speech signals can be represented in terms of the probability distribution of acoustic, idiolect, phonotactic, or some high-level discrete events. Formulated under the maximum *a posteriori* (MAP) estimation framework, we have demonstrated the usefulness of modeling speech signals as discrete distributions for SVM-based speaker verification. We further proposed and analyzed the use of Bhattacharyya coefficient as the similarity measure between supervectors constructed from the discrete probabilities. From the perspective of feature normalization in the supervector space, the Bhattacharyya measure warps the distribution of each dimension with a square-root function, a much simpler and data-independent operation, yet leading to higher accuracy compared to the Fisher kernel, TFLLR scaling, and rank normalization. Experiments conducted on the NIST 2006 SRE showed that relative reduction in EER was 15.40% with the Bhattacharyya measure and 24.55% when used in conjunction with NAP and t-norm. These results suggest that the Bhattacharyya measure is a strong candidate for measuring the similarity between discrete distributions with SVM classifier. The proposed method gives comparable performance to the state-of-the-art GMM supervector approach. Their fusion gave 5.35% relative improvement in EER, even though the improvement in MinDCF was marginal.

It is worth emphasizing that, even though the current work uses a UBM quantizer to construct supervectors, this is not necessarily the case; the proposed method can be used with other types of front-end quantizer. In future, it would be interesting to compare how much we would benefit by using the proposed method with a different front-end quantizer such as a phone recognizer or a prosodic feature extractor. We also expect the method to be readily applicable for spoken language recognition, and applications beyond speech technology that operate on discrete symbols, such as natural language processing (NLP) and bioinformatics.

## References


[1] T. Kinnunen and H. Li, "An overview of text-independent speaker recognition: From features to supervectors," *Speech Commun.*, vol. 52, no. 1, pp. 12–40, Jan. 2010.

[2] V. Wan and S. Renals, "Speaker verification using sequence discriminant support vector machines," *IEEE Trans. Speech Audio Process.*, vol. 13, no. 2, pp. 203–210, Mar. 2005.

[3] W. M. Campbell, J. P. Campbell, D. A. Reynolds, E. Singer, and P. A. Torres-Carrasquillo, "Support vector machines for speaker and language recognition," *Comput. Speech Lang.*, vol. 20, no. 2–3, pp. 210–229, 2006.

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

[5] Y. Mami and D. Charlet, "Speaker recognition by location in the space of reference speakers," *Speech Commun.*, vol. 48, no. 2, pp. 127–141, 2006.

[6] A. Stolcke, S. S. Kajarekar, L. Ferrer, and E. Shriberg, "Speaker recognition with session variability normalization based on MLLR adaptation transforms," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 15, no. 7, pp. 1987–1998, Sep. 2007.

[7] K. A. Lee, C. You, H. Li, and T. Kinnunen, "A GMM-based probabilistic sequence kernel for speaker recognition," in *Proc. Interspeech*, 2007, pp. 294–297.

[8] K. A. Lee, C. You, H. Li, T. Kinnunen, and D. Zhu, "Characterizing speech utterances for speaker verification with sequence kernel SVM," in *Proc. Interspeech*, Sep. 2008, pp. 1397–1400.

[9] P. Kenny, M. Mihoubi, and P. Dumouchel, "New MAP estimates for speaker recognition," in *Proc. Eurospeech*, 2003, pp. 2964–2967.

[10] W. M. Campbell, J. P. Campbell, T. P. Gleason, D. A. Reynolds, and W. Shen, "Speaker verification using support vector machines and high-level features," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 15, no. 7, pp. 2085–2094, Sep. 2007.

[11] E. Shriberg, L. Ferrer, S. Kajarekar, A. Venkataraman, and A. Stolcke, "Modeling prosodic feature sequences for speaker recognition," *Speech Commun.*, vol. 46, no. 3–4, pp. 455–472, Jul. 2005.

[12] C. H. Lee and Q. Huo, "On adaptive decision rules and decision parameter adaptation for automatic speech recognition," *Proc. IEEE*, vol. 88, no. 8, pp. 1241–1269, Aug. 2000.

[13] T. Kailath, "The divergence and Bhattacharyya distance measures in signal detection," *IEEE Trans. Commun. Technol.*, vol. COM-15, no. 1, pp. 52–60, Feb. 1967.

[14] X. Huang, A. Acero, and H. W. Hon, *Spoken Language Processing: A Guide to Theory, Algorithm, and System Development*. Upper Saddle River, NJ: Prentice-Hall, 2001.

[15] A. Stolcke, S. Kajarekar, and L. Ferrer, "Nonparametric feature normalization for SVM-based speaker verification," in *Proc. ICASSP*, 2008, pp. 1577–1580.

[16] M. A. Zissman, "Comparison of four approaches to automatic language identification of telephone speech," *IEEE Trans. Speech Audio Process.*, vol. 4, no. 1, pp. 31–44, Jan. 1996.

[17] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted Gaussian mixture models," *Digital Signal Process.*, vol. 10, no. 1–3, pp. 19–41, 2000.

[18] T. S. Jaakkola and D. Haussler, "Exploiting generative models in discriminative classifisers," in *Advances in Nueral Information Processing Systems 11*, M. S. Kearns, S. A. Solla, and D. A. Cohn, Eds. Cambridge, MA: MIT Press, 1998.

[19] V. Kecman, *Learning and Soft Computing: Support Vector Machines, Neural Networks, and Fuzzy Logic Models*. Cambridge, MA: MIT Press, 2001.

[20] Á de la Torre, A. M. Peinado, J. C. Segura, J. L. Pérez-Córdoba, M. C. Benítez, and A. J. Rubio, "Histogram equalization of speech representation for robust speech recognition," *IEEE Trans. Speech Audio Process.*, vol. 13, no. 3, pp. 355–366, May 2005.

[21] J. Pelecanos and S. Sridharan, "Feature warping for robust speaker verification," in *Proc. Odyssey*, 2001.

[22] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. New York: Wiley, 2001.

[23] *The NIST Year 2006 Speaker Recognition Evaluation Plan*. : National Inst. Standards Technol., 2006.

[24] H. Hermansky and N. Morgan, "RASTA processing of speech," *IEEE Trans. Speech Audio Process.*, vol. 2, no. 4, pp. 578–589, Oct. 1994.

[25] C.-C. Chang and C.-J. Lin, LIBSVM: A library for support vector machines 2001 [Online]. Available: http://www.csie.ntu.edu.tw/~cjlin/libsvm

[26] S. Young, G. Evermann, M. J. F. Gales, T. Hain, D. Kershaw, X. Liu, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. Woodland, *The HTK Book*. Cambridge, U.K.: Cambridge Univ. Eng. Dept., 2006.

[27] R. Auckenthaler, M. Carey, and H. Lloyd-Thomas, "Score-normalization for text-independent speaker verification," *Digital Signal Process.*, vol. 10, pp. 42–54, Jan. 2000.

[28] R. Auckenthaler and J. S. Mason, "Gaussian selection applied to text-independent speaker verification," in *Proc. Odyssey*, 2001, pp. 83–88.

[29] A. Solomonoff, W. M. Campbell, and C. Quillen, "Channel compensation for SVM speaker recognition," in *Proc. Odyssey*, 2004, pp. 57–62.

[30] L. Gillick and S. J. Cox, "Some statistical issues in the comparison of speech recognition algorithms," in *Proc. ICASSP*, 1989, pp. 532–535.

**Kong Aik Lee** (M'05) received the B.Eng. (first class honors) degree from University Technology Malaysia, Skudai, in 1999, and the Ph.D. degree from Nanyang Technological University, Singapore, in 2006.

He is currently a Senior Research Fellow with the Human Language Technology Department, Institute for Infocomm Research (I2R), Singapore. His research focuses on statistical methods for speaker and spoken language recognition, adaptive echo and noise control, and subband adaptive filtering. He is the leading author of the book *Subband Adaptive Filtering: Theory and Implementation* (Wiley, 2009).

**Chang Huai You** (M'08) received the B.Sc. degree in physics and wireless from Xiamen University, Xiamen, China, in 1986, the M.Eng. degree in communication and electronics engineering from Shanghai University of Science and Technology, Shanghai, China, in 1989, and the Ph.D. degree in electrical and electronic engineering from Nanyang Technological University (NTU), Singapore, in 2006.

From 1989 to 1992, he was an Engineer with Fujian Hitachi Television Corporation, Ltd., Fuzhou City, China. From 1992 to 1998, he was an Engineering Specialist with Seagate International, Singapore. He joined the Centre for Signal Processing, NTU, as a Research Engineer in 1998 and became a Senior Research Engineer in 2001. In 2002, he joined the Agency for Science, Technology, and Research, Singapore, and was appointed as a Member of Associate Research Staff. In 2003, he was appointed as a Scientist with the Institute for Infocomm Research (I2R), Singapore. In 2006, he was appointed as a Senior Research Fellow with (I2R). Since 2007, he has been a Research Scientist with Human Language Technology department of (I2R). His research interests include speaker recognition, language recognition, speech enhancement, speech recognition, acoustic noise reduction, array signal processing, audio signal processing, and image processing. He is a reviewer of many international conferences and journals.

Dr. You was the recipient of Silver Prize of EEE Technology Exhibition at NTU for his "Intelligent Karaoke Project" as a major designer and project leader in 2001.

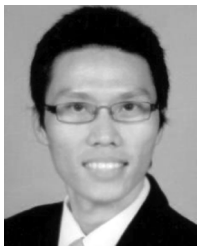
**Haizhou Li** (M'91–SM'01) is currently the Principal Scientist and Department Head of Human Language Technology at the Institute for Infocomm Research, Signapore. He is also the Program Manager of Social Robotics at the Science and Engineering Research Council of A*STAR, Singapore.

Dr Li has worked on speech and language technology in academia and industry since 1988. He taught in the University of Hong Kong (1988–1990), South China University of Technology (1990–1994), and Nanyang Technological University (2006-present). He was a Visiting Professor at CRIN in France (1994–1995), and at the University of New South Wales in Australia (2008). As a technologist, he was appointed as Research Manager at the Apple-ISS Research Centre (1996–1998), Research Director in Lernout & Hauspie Asia Pacific (1999–2001), and Vice President in InfoTalk Corp., Ltd., (2001–2003). His current research interests include automatic speech recognition, speaker and language recognition, and natural language processing. He has published over 150 technical papers in international journals and conferences. He holds five international patents.

Dr Li now serves as an Associate Editor of the IEEE TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING and the *Springer International Journal of Social Robotics*. He is an elected Board Member of the International Speech Communication Association (ISCA, 2009–2013), a Vice President of the Chinese and Oriental Language Information Processing Society (COLIPS, 2009–2011), an Executive Board Member of the Asian Federation of Natural Language Processing (AFNLP, 2006–2010), and a Member of the ACL. He was a recipient of the National Infocomm Award of Singapore in 2001. He was named one the two Nokia Visiting Professors 2009 by the Nokia Foundation in recognition of his contribution to speaker and language recognition technologies.

**Tomi Kinnunen** received the M.Sc. and Ph.D. degrees in computer science from the University of Joensuu, Joensuu, Finland, in 1999 and 2005, respectively.

He worked as an Associate Scientist at the Speech and Dialogue Processing Lab, Institute for Infocomm Research (I2R), Singapore, and as a Senior Assistant in the Department of Computer Science and Statistics, University of Joensuu. He works currently as a Post-Doctoral Researcher in the University of Eastern Finland, Joensuu, and his research is funded by the Academy of Finland. His research areas cover speaker recognition and speech signal processing.



**Khe Chai Sim** received the B.A. degree in electrical and information sciences and the M.Eng. degree in computer speech, text, and Internet technology, both from the University of Cambridge, Cambridge, U.K., in 2001 and 2002, respectively, and the M.Phil. degree from the Machine Intelligence Laboratory, Cambridge University Engineering Department in 2006.

After the Ph.D. degree, he joined the Institute for Infocomm Research, Singapore, as a Research Engineer. Since 2008, he has been an Assistant Professor at the School of Computing, National University of Singapore, Singapore. His research interests include statistical pattern classification, automatic speech recognition, speaker recognition, and spoken language recognition. He has also worked on the DARPA-funded EARS project from 2002 to 2005 and the GALE project from 2005 to 2006.