Speaker verification using sequence discriminant support vector machines

Vincent Wan and Steve Renals

***Abstract*— This paper presents a text-independent speaker verification system using support vector machines (SVMs) with score-space kernels. Score-space kernels generalize Fisher kernels and are based on underlying generative models such as Gaussian mixture models (GMMs). This approach provides direct discrimination between whole sequences, in contrast with the frame-level approaches at the heart of most current systems. The resultant SVMs have a very high dimensionality since it is related to the number of parameters in the underlying generative model. To address problems that arise in the resultant optimization we introduce a technique called spherical normalization that preconditions the Hessian matrix. We have performed speaker verification experiments using the PolyVar database. The SVM system presented here reduces the relative error rates by 34% compared to a GMM likelihood ratio system.**

***Index Terms*— Fisher kernel, score-space kernel, speaker verification, support vector machine.**

I. Introduction

Current state-of-the-art speaker verification systems are based on generative speaker models, typically Gaussian mixture models (GMMs) and hidden Markov models (HMMs) [1]. These models usually operate at the frame-level with an overall sequence score obtained by averaging the likelihoods of each frame in the sequence or via the use of an HMM. More accurate verification systems may be constructed by placing these generative models in a discriminative framework, for example taking the likelihood ratio between the model for a particular speaker and a more general world model [2]. A limitation of these approaches arises from the fact that discrimination occurs between frames, whereas speaker verification is concerned with sequence discrimination. Since a discriminative classifier discards information that its objective function considers irrelevant, frame discrimination approaches may inadvertently discard relevant information. In this paper we describe an approach to speaker verification based on the support vector machine (SVM) [3], [4] that enables direct discrimination between sequences.

An SVM has many desirable properties including the ability to classify sparse data without over-training and to make non-linear decisions via kernel functions. However, due to certain practical limitations the SVM has not gained widespread usage in mainstream applications. Initial speaker recognition work using SVMs by Schmidt and Gish [5] highlighted the main problem: SVMs become inefficient when the the number of training frames is large. This can be overcome by using special kernels to classify sequences instead of frames. The family of score-space kernels [6] are such kernels that enable sequence discrimination. Score-space kernels include the Fisher kernel [7] and map a complete sequence onto a single point in a high dimensional space by exploiting generative models. The Fisher kernel has been applied to speaker recognition with limited success [8], [9].

We have applied the score-space kernel SVM approach to text-independent speaker verification, extending some previous work that employed frame discriminant SVMs [10], [11]. We performed experiments on the PolyVar database [12] and report error rates that are better than a GMM likelihood ratio system by 34% and better than the current state-of-the-art system by 25%.

The structure of this paper is as follows: the next section provides an overview of GMM speaker verification systems; section III reviews SVMs for classification; sequence kernels and methods for normalizing them are described in section IV; experimental evaluation and results are presented in section V; section VI concludes the paper.

II. GMM-based speaker verification

An N_g component Gaussian mixture for N_d dimensional input vectors has the following form:

$$P(\mathbf{x}|M) = \sum_{i=1}^{N_g} a_i \frac{1}{(2\pi)^{N_d/2}|\Sigma_i|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mu_i)^T\Sigma_i^{-1}(\mathbf{x}-\mu_i)\right) \quad (1)$$

where $P(\mathbf{x}|M)$ is the likelihood of input vector, $\mathbf{x}$, given the mixture model, M. The mixture model consists of a weighted sum over N_g Gaussian densities each parameterized by a mean vector, μ_i, and a covariance matrix, Σ_i. The coefficients, a_i, are the mixture weights, which are constrained to be non-negative and must sum to one. The parameters of a Gaussian mixture model, a_i, μ_i and Σ_i for $i = 1 \cdots N_g$ may be estimated using the maximum likelihood criterion and the EM (expectation-maximization) algorithm [13], [14].

For reasons of both modelling and estimation, it is usual to employ GMMs consisting of components with diagonal covariance matrices. A detailed discussion on the application of GMMs to speaker modelling can be found in [15]. The basic method is straightforward. A GMM (the *client model*) is trained using maximum likelihood to estimate the probability

This work was supported by a Motorola studentship awarded to Vincent Wan. Vincent Wan is with the Department of Computer Science, University of Sheffield, 211 Portobello St,. Sheffield, S1 4DP, UK (email: v.wan@dcs.shef.ac.uk). Steve Renals is at the Centre for Speech Technology Research, University of Edinburgh, Edinburgh EH8 9LW, UK (email: s.renals@ed.ac.uk).

density function, $P(\mathbf{x}_i|M)$, of the client speaker. The probability, $P(X|M)$, that an utterance, $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_{N_v}\}$, is generated by the model, M, is used as the utterance score, estimated by the mean log likelihood over the sequence:

$$S(X) = \log P(X|M) = \frac{1}{N_v}\sum_{i=1}^{N_v} \log P(\mathbf{x}_i|M). \tag{2}$$

The utterance score is used to make a decision by comparing it against a threshold that has been chosen for a desired trade-off between detection error types.

Speaker verification may be posed as a discriminative problem, with the objective of assigning high scores to those frames, $\mathbf{x}_i$, that are specific to the client and low scores to frames that are common to most speakers. To achieve this a second GMM (the *world model*) may be used to model the properties of speech signals common to all speakers, with parameters estimated from a large number of background speakers. A discriminative score may then be obtained by taking the log likelihood ratio of the client model, M, to the world model, Ω, which is equivalent to the difference between their log likelihood scores. Again, the mean of the frame scores is taken across the sequence:

$$\begin{aligned} S(X) &= \frac{1}{N_v}\sum_{i=1}^{N_v} \log \frac{P(\mathbf{x}_i|M)}{P(\mathbf{x}_i|\Omega)} && (3)\\ &= \frac{1}{N_v}\sum_{i=1}^{N_v} \log P(\mathbf{x}_i|M) - \frac{1}{N_v}\sum_{i=1}^{N_v} \log P(\mathbf{x}_i|\Omega) && (4)\\ &= \log P(X|M) - \log P(X|\Omega). && (5) \end{aligned}$$

We call this approach the GMM likelihood ratio (GMM-LR), and in practice it produces more accurate speaker verification systems. Reynolds [2] used the GMM-LR approach for speaker verification. The GMM-LR method is a simple yet powerful approach and is used here as a baseline.

By Bayes' decision rule, equation (5) is optimal so long as the client and impostors are well modelled. Bengio and Mariéthoz [16] proposed that the probability estimates are not perfect and that a more accurate version would be:

$$S(X) = a \log P(X|M) - b \log P(X|\Omega) + c, \tag{6}$$

where a, b and c are adjustable parameters estimated using an SVM for which the input is the two dimensional vector composed of the client and world models' log likelihoods. This results in a small improvement in accuracy as discussed in section V.

III. Support vector classification

In its basic form, the SVM (support vector machine) is a binary linear classifier. It is described in detail by Vapnik [3] and in Burges' tutorial [4]. Given a set of linearly separable two-class training data, there are many possible solutions for a discriminative classifier. In the case of the SVM, a separating hyperplane is chosen so as to maximize the *margin* between the two classes. Essentially this involves orienting the separating hyperplane to be perpendicular to the shortest line separating the convex hulls of the training data for each class, and locating it midway along this line.

Let the separating hyperplane be defined by $\mathbf{x}\cdot\mathbf{w} + b = 0$ where $\mathbf{w}$ is its normal. For linearly separable data labelled $\{\mathbf{x}_i, y_i\}$, $\mathbf{x}_i \in \Re^{N_d}$, $y_i \in \{-1, 1\}$, $i = 1 \ldots N$, the optimum boundary chosen according to the maximum margin criterion is found by minimizing the objective function:

$$\begin{aligned} E &= ||\mathbf{w}||_2^2 && (7)\\ \text{subject to } &(\mathbf{x}_i \cdot \mathbf{w} + b) y_i \geq 1 \quad \text{for all } i. \end{aligned}$$

The solution for the optimum boundary, $\mathbf{w}_0$, is a linear combination of a subset of the training data, $\mathbf{x}_s$, $s \in \{1 \ldots N\}$: the *support vectors*. These support vectors define the margin edges and satisfy the equality $(\mathbf{x}_s \cdot \mathbf{w}_0 + b) y_s = 1$. Data may be classified by computing the sign of $\mathbf{x}\cdot\mathbf{w}_0 + b$.

Generally, the data are not separable and the above inequalities cannot be satisfied. In this case we may introduce "slack" variables ξ_i which represent the amount by which each point is misclassified. In this case the objective function is reformulated as:

$$\begin{aligned} E &= \frac{1}{2}||\mathbf{w}||_2^2 + C\sum_i L(\xi_i) && (8)\\ \text{subject to } &(\mathbf{x}_i \cdot \mathbf{w} + b) y_i \geq 1 - \xi_i \quad \text{for all } i. \end{aligned}$$

The second term on the right hand side of (8) is the *empirical risk* associated with those points that are misclassified or lie within the margin. L is a cost function and C is a hyper-parameter that trades off the effects of minimizing the empirical risk against maximizing the margin. The first term can be thought of as a regularization term, derived from maximizing the margin, which gives the SVM its ability to generalize well on sparse training data. This property will be seen to be important when classifying sequences.

The linear-error cost function is most commonly used since it is robust to outliers. The dual formulation (which is more conveniently solved) of equation (8) with $L(\xi_i) = \xi_i$ is

$$\alpha^* = \max_{\alpha} \left(\sum_i \alpha_i + \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j \right) \tag{9}$$

$$\text{subject to} \quad \begin{array}{l} 0 \leq \alpha_i \leq C \\ \sum_i \alpha_i y_i = 0 \end{array} \tag{10}$$

in which $\alpha = \{\alpha_1, \ldots, \alpha_N\}$ is the set of Lagrange multipliers of the constraints in the primal optimization problem [4]. The dual can be solved using standard quadratic programming techniques. The optimum decision boundary, $\mathbf{w}_0$, is given by

$$\mathbf{w}_0 = \sum_i \alpha_i y_i \mathbf{x}_i \tag{11}$$

and is a linear combination of all vectors that have $\alpha_i \neq 0$.

The extension to non-linear boundaries is achieved through the use of kernel functions that satisfy Mercer's condition [17]. In essence, a non-linear mapping is defined from the *input space*, in which the data are observed, to a manifold in higher dimensional *feature space*, which is defined implicitly by the kernel functions. The hyperplane is constructed in the feature space and intersects with the manifold creating a non-linear boundary in the input space. In practice, the mapping is achieved by replacing the value of dot products between two vectors in input space with the value that results when

the same dot product is carried out in the feature space. The dot product in the feature space is expressed conveniently by the kernel as some function of the two vectors in input space. The polynomial and RBF (radial basis function) kernels are commonly used, and take the form:

$$K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i \cdot \mathbf{x}_j + 1)^n \tag{12}$$

and

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left[-\frac{1}{2}\left(\frac{||\mathbf{x}_i - \mathbf{x}_j||}{\sigma}\right)^2\right] \tag{13}$$

respectively, where n is the order of the polynomial and σ is the width of the radial basis function. The dual for the non-linear case is thus

$$\alpha^* = \max_{\alpha}\left(\sum_i \alpha_i + \sum_{i,j}\alpha_i\alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)\right) \tag{14}$$

$$\text{subject to} \quad \begin{array}{l} 0 \le \alpha_i \le C \\ \sum_i \alpha_i y_i = 0. \end{array} \tag{15}$$

The use of kernels means that an explicit transformation of the data to the feature space is not required.

IV. Discriminative sequence classification

The approaches to speaker verification outlined in section II are discriminative between frames rather than between complete utterances. Discriminative classification of sequences is difficult since sequences have different lengths. However, if a mapping from a variable length sequence to a fixed length vector can be achieved, then standard classification procedures may be applied. Such a mapping was first developed by Jaakkola and Haussler [7] and is known as the *Fisher kernel*. This approach was generalized by Smith and Gales [18], [19] as a technique referred to as *score-spaces*. Using these approaches for whole sequence classification results in a sparse data problem, for which SVMs are well suited: a set of sequences are mapped to a comparatively high dimensional feature space. Furthermore, the concept of mapping each sequence to a feature space may be interpreted as an SVM kernel. We shall now describe the score-space transformations that we have used.

A. Score-spaces

Score-space kernels [6], [18], [19], which generalize Fisher kernels [7], enable SVMs to classify whole sequences by exploiting a set of parametric generative models. In this approach a variable length sequence is mapped explicitly onto a single point in a fixed-dimension space, the *score-space*. Such a mapping is achieved by applying some operator to the likelihood score of a generative model. Hence the fixed-dimension score-space allows a dot product to be computed between two sequences even if they were originally different lengths. This section first describes a generic formulation to achieve such mappings followed by a detailed explaination of some special cases of the score-space method: the Fisher kernel [7] and the likelihood ratio kernel.

The score-space is defined by and derived from the likelihood score of a set of k generative models, $\{p_k(X|M_k, \theta_k)\}$.

TABLE I
Some examples of score operators

Operator	expression
first derivative	$\hat{F} = \nabla_\theta$
first derivative and argument	$\hat{F} = [\nabla_\theta, 1]^T$
first and second derivative	$\hat{F} = [\nabla_\theta, \text{vec}(\nabla^2\theta)^T]^T$

The generic formulation for mapping a sequence, $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_{N_v}\}$, to the score-space is given by

$$\psi_{\hat{F},f}(X) = \hat{F} f(\{p_k(X|M_k, \theta_k)\}), \tag{16}$$

where $\psi_{\hat{F},f}(X)$ is called the *score-vector*, $f(\{p_k(X|M_k, \theta_k)\})$, a function of the scores of the set of generative models, is called the *score-argument* and $\hat{F}$ is the *score-operator* that maps the scalar score-argument to the score-space. The properties of the resulting score-space depend upon the choice of operator and argument that is used. Several options for score-operators were proposed by Smith *et. al.* [6] and are summarized in table I.

Almost any function may be used as a score-argument. We shall show two specific cases that lead to the likelihood score-space kernel (more commonly known as the Fisher kernel [7]) and the likelihood ratio score-space kernel.

1) The likelihood score-space: The likelihood score-space is obtained by setting the score-argument to be the log likelihood of a generative model, M, parameterized by θ, and choosing the first derivative score-operator from table I:

$$f(\{p_k(X|M_k, \theta_k)\}) = \log P(X|M, \theta) \tag{17}$$

$$\psi_{\text{Fisher}}(X) = \nabla_\theta \log P(X|M, \theta). \tag{18}$$

This mapping, known as the *Fisher* mapping, was first developed and applied to biological sequence analysis by Jaakkola and Haussler [7].

Each component of the score-space, $\psi(X)$, corresponds to the derivative of the log likelihood score with respect to one of the parameters of the model. In some ways it is a measure of how well the sequence, X, matches the model. Consider a generative model trained using the maximum likelihood criterion and gradient descent. In order to maximize the likelihood of a given sequence, the same set of derivatives to equation (18) must be computed so that the parameters may be updated. When the derivatives are small then the likelihood may be close to a local maximum; when the derivatives are large then the likelihood has yet to reach a maximum. Whether the derivatives will provide additional information that is not already encoded in the likelihood score may be examined by augmenting the score-vector with the score-argument (the log likelihood score in this case):

$$\psi_{\text{Fisher+}}(X) = \begin{bmatrix} \nabla_\theta \log P(X|M, \theta) \\ \log P(X|M, \theta) \end{bmatrix}. \tag{19}$$

2) The likelihood ratio score-space: An alternative score-argument is the ratio of two generative models, M_1 and M_2,

$$f(\{p_k(X|M_k, \theta_k)\}) = \log \frac{P(X|M_1, \theta_1)}{P(X|M_2, \theta_2)}, \tag{20}$$

where $\theta = \{\theta_1, \theta_2\}$. The corresponding mapping using the first derivative score-operator is

$$\psi_{\text{LR}}(X) = \nabla_\theta \log \frac{P(X|M_1,\theta_1)}{P(X|M_2,\theta_2)} \tag{21}$$

and again the score-argument may be added to the score-space:

$$\psi_{\text{LR+}}(X) = \begin{bmatrix} \nabla_\theta \log \frac{P(X|M_1,\theta_1)}{P(X|M_2,\theta_2)} \\ \log \frac{P(X|M_1,\theta_1)}{P(X|M_2,\theta_2)} \end{bmatrix}. \tag{22}$$

The likelihood ratio score-space is motivated by the GMM likelihood ratio (GMM-LR) classifier described in section II. In the same way that the GMM-LR is a more discriminative classifier than a single GMM, so should the likelihood ratio score-space kernel be. A GMM likelihood ratio forces the classifier to model the class boundaries more accurately. The discrimination information encoded in the likelihood ratio score will also be in its derivatives.

B. Computing the score-vectors

In this section we derive the formulae for computing derivatives of the log likelihoods when the generative model is a diagonal covariance GMM. The formulae for the derivatives when the generative model is an HMM may be found in [6].

Let

$$R(i,j) = \prod_{\ell=1}^{N_d} \frac{1}{\sigma_j^\ell \sqrt{2\pi}} \exp\left\{ -\frac{1}{2} \left(\frac{x_i^\ell - \mu_j^\ell}{\sigma_j^\ell} \right)^2 \right\} \tag{23}$$

so that the diagonal covariance GMM likelihood is

$$P(\mathbf{x}_i|M,\theta) = \sum_{j=1}^{N_g} a_j R(i,j) \tag{24}$$

where $\theta = \{a_j, \mu_j^\ell, \sigma_j^\ell\}$ is the set of parameters in the GMM, M. In particular, a_j is the prior of the j^{th} Gaussian component of the GMM, μ_j is the mean vector of the j^{th} component and σ_j is the corresponding diagonal covariance vector. The superscript on the mean and covariance enumerate the components of the vectors. N_g is the number of Gaussians that make up the mixture model and N_d is the dimensionality of the input vectors with components $\mathbf{x}_i = [x_i^1, x_i^2, \cdots, x_i^{N_d}]^T$.

The global log likelihood of a sequence $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_{N_v}\}$ is

$$\log P(X|M,\theta) = \sum_{i=1}^{N_v} \log P(\mathbf{x}_i|M,\theta) \tag{25}$$

where N_v is the number of frames in the sequence. From (18) the score-vector is the vector of the derivatives with respect to each parameter of the log of (25). The derivatives are with respect to the covariances, means and priors of the Gaussian mixture model. The derivative with respect to the j^{th} prior is

$$\frac{d}{da_{j^*}} \log P(X|M,\theta) = \sum_{i=1}^{N_v} \frac{R(i,j^*)}{\sum_{j=1}^{N_g} a_j R(i,j)}. \tag{26}$$

The derivative with respect to the ℓ^{th} component of the j^{th} mean is

$$\frac{d}{d\mu_{j^*}^{\ell^*}} \log P(X|M,\theta) = \sum_{i=1}^{N_v} \frac{a_{j^*} R(i,j^*)}{\sum_{j=1}^{N_g} a_j R(i,j)} \cdot \frac{1}{\sigma_{j^*}^{\ell^*}} \left(\frac{x_i^{\ell^*} - \mu_{j^*}^{\ell^*}}{\sigma_{j^*}^{\ell^*}} \right). \tag{27}$$

The derivative with respect to the ℓ^{th} component of the j^{th} covariance is

$$\frac{d}{d\sigma_{j^*}^{\ell^*}} \log P(X|M,\theta) = \sum_{i=1}^{N_v} \frac{a_{j^*} R(i,j^*)}{\sum_{j=1}^{N_g} a_j R(i,j)} \cdot \left(\frac{(x_i^{\ell^*} - \mu_{j^*}^{\ell^*})^2}{(\sigma_{j^*}^{\ell^*})^3} - \frac{1}{\sigma_{j^*}^{\ell^*}} \right). \tag{28}$$

The likelihood score-vector can then be expressed as:

$$\psi_{\text{Fisher}}(X) = \left[\frac{d}{da_{j^*}}, \cdots, \frac{d}{d\mu_{j^*}^{\ell^*}}, \cdots, \frac{d}{d\sigma_{j^*}^{\ell^*}} \right]^T \log P(X|M,\theta) \tag{29}$$

for $j^* = 1, \ldots, N_g$ and $\ell^* = 1, \ldots, N_d$.

The likelihood ratio kernel is also computed using equations (26) to (29). The score-vectors of the likelihood ratio kernel (21) can be expressed as the difference of two terms,

$$\psi_{\text{LR}}(X) = \nabla_\theta \log P(X|M_1,\theta_1) - \nabla_\theta \log P(X|M_2,\theta_2). \tag{30}$$

Let $\theta = \{\theta_1, \theta_2\}$ be the vector of all parameters that exist in both models, M_1 and M_2. The derivatives of $\log P(X|M_1,\theta_1)$ with respect to the parameters θ_2 in M_2 are zero and vice-versa. Thus $\psi_{\text{LR}}(X)$ can be split so that the derivatives are computed with respect to one model at a time. When the differentiated parameter belongs to model M_1 then

$$\psi_{\theta_1}(X) = \nabla_{\theta_1} \log P(X|M_1,\theta_1) \tag{31}$$

is computed. Likewise, when the parameter belongs to model M_2 then

$$\psi_{\theta_2}(X) = \nabla_{\theta_2} \log P(X|M_2,\theta_2) \tag{32}$$

is computed. These derivatives are identical to the derivatives computed by the Fisher kernel. The likelihood ratio score-vector is

$$\psi_{\text{LR}}(X) = \begin{bmatrix} \psi_{\theta_1}(X) \\ -\psi_{\theta_2}(X) \end{bmatrix}. \tag{33}$$

From equations (29) and (33) it can be seen that the dimensionality of the score-space is equal to the total number of parameters in the generative models. Having mapped the sequence to the score-space, any discriminative classifier may be used to classify vectors and hence obtain a classification for the complete sequence. However, it is not unusual for generative models to have several thousand parameters. This means that the discriminative classifier must be able to classify vectors of that size. Classifiers such as multilayer perceptrons cannot be easily trained on such data due to problems in parameterization. The SVM, fortunately, is well suited to classify high dimensional data.

C. Score-space normalization

SVMs are not invariant to linear transformations in feature space, so normalization of the feature vectors is desirable. We used two stages of normalization: whitening the data in the score-space by normalizing the components of the vectors, $\psi(X)$, to zero mean and unit variance; then applying *spherical normalization*, which may be interpreted as a Hessian preconditioning step and involves making a further nonlinear transformation to a higher dimension space.

1) Score-space whitening: For a given score-space, the metric of the space is determined by the generative model(s) and is generally non-Euclidean. The dot product in a non-Euclidean space is defined as $\mathbf{x}^T G^T G\mathbf{y}$ where G is a matrix that maps the vectors to a Euclidean space. A kernel constructed from any of the above mappings is

$$K(X_p, X_q) = \psi(X_p)^T G^T G \psi(X_q) \tag{34}$$

where $G^T G$ is the metric of the space and the subscript on X enumerates the sequences. In Euclidean space, G is the identity matrix. In the case of the log likelihood score-space mapping, $G^T G$ is the inverse Fisher information matrix (the inverse of the covariance matrix of the score-vectors):

$$G^T G = \left(E\{U(X)U(X)^T\}\right)^{-1} \tag{35}$$

where $U(X) = \psi(X) - E\{\psi(X)\}$ and E is the expectation operator.

This can be interpreted as a whitening step where the score-vector components are normalized to zero mean and unit variance (*i.e.* the basis vectors of the score-space are mapped to an orthonormal set). Whitening is important since SVMs are not invariant to linear transformations in the feature space. Consider a two dimensional space where the variance in one dimension is significantly higher than in the other. A dot product in this case will be dominated by the high variance component, effectively reducing the dimensionality of the space to one.

Computing dot products in score-space relies on the ability to estimate a full covariance matrix, which will normalize the scaling in each dimension and make the principal component axes orthogonal. However, the score-space space dimensionality, which is equal to the number of parameters in the generative model, may be large thus making estimation impractical. The required computation may be reduced by normalizing with a diagonal covariance matrix (so the scale of each dimension will be the same), or a block diagonal covariance matrix (making some of the principal component axes orthogonal).

2) Spherical normalization: Spherical normalization, developed in the context of SVMs using high order polynomial kernels [11], is a preconditioning step employing a transformation that maps each feature vector onto the surface of a unit hypersphere embedded in a space that has one dimension more than the feature vector itself.

Dot products between high dimensional vectors may lead to an ill-conditioned Hessian since the dynamic range of the result is large. This occurs even when each individual component of the vectors has been normalised to zero mean and unit variance. In particular, score-space kernels based on generative models that have many tens of thousands (or more) of parameters are likely to suffer from ill-conditioning. It is possible to compress the dynamic range of a dot product by exploiting its cosine interpretation, *i.e.* $\mathbf{x}\cdot\mathbf{y} = \|\mathbf{x}\|\|\mathbf{y}\|\cos\theta$. If the vectors have unit length then the dot product is just the cosine of the angle in between and the result must be in the range -1 to $+1$.

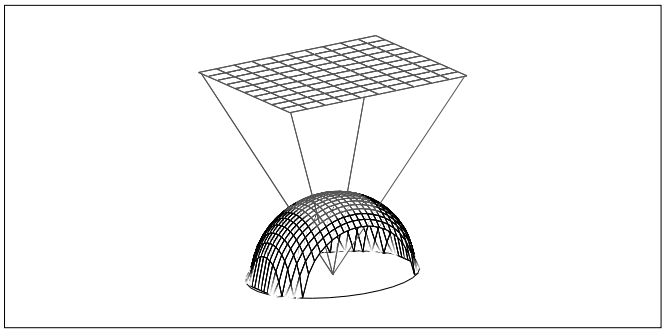
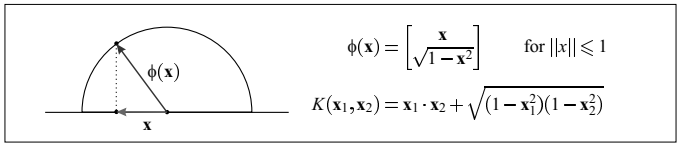


Fig. 1. Spherical vector-length normalization: mapping onto a sphere

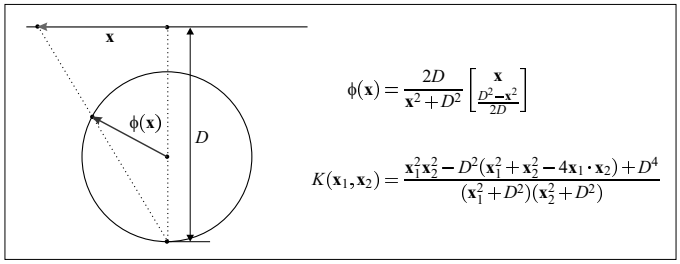
A vector can be normalized easily by dividing by its Euclidean length. But this results in information loss causing greater classification uncertainty. For example, two points in the input space represented by $\mathbf{x}$ and $2\mathbf{x}$ will both be normalized to $\hat{\mathbf{x}}$. Alternatively, normalization without information loss may be done by projecting to a higher dimensional space. Consider the mapping from a 2D plane to a 3D unit sphere, as in figure 1. Any point in 2D space may be mapped onto the surface of a unit sphere in 3D space. The new vectors representing the data are the unit vectors from the centre of the sphere to its surface. The mapping is reversible so no information is lost. We call this spherical vector-length normalization, or spherical normalization for short.

Mapping a plane onto a sphere's surface may be achieved by many different projections, all of which may be generalized to arbitrary dimensions: this is the inverse of the problem faced by cartographers when mapping the Earth but extended to much higher dimensions. Standard projections used by cartographers are the azimuthal, conical and cylindrical projections. We consider three different azimuthal projections (illustrated in figure 2 along with the explicit transformations and the corresponding kernel functions): the orthographic projection, the stereographic projection and a modified stereographic projection.

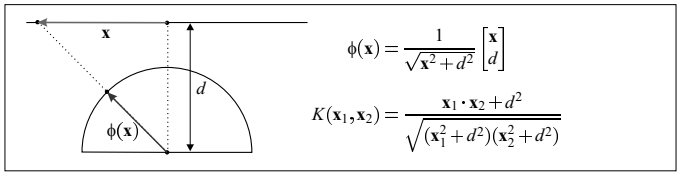
The orthographic projection (figure 2a) is limited since the input data are restricted to lie within a small finite region directly beneath the hemisphere. The stereographic projection (figure 2b) does not suffer from this restriction but the space represented by the sphere wraps around. With this projection, points located at $+\infty$ and $-\infty$ in the input space project to the same point on the hypersphere. We used the modified stereographic projection (figure 2c) because it does not suffer from these issues. The projection is made by augmenting a vector with a constant, d, and normalizing the new vector by its Euclidean length.



a: The orthographic projection



b: The stereographic projection



c: The modified stereographic projection

Fig. 2. Spherical vector-length normalization: various ways of projecting data onto unit hyperspheres.

For the score-space kernels presented here, the mapping applied explicitly to the score-vectors is

$$\Psi(X) :\rightarrow \phi(\Psi(X)) = \frac{1}{\sqrt{\Psi(X)\cdot\Psi(X)+d^2}}\begin{bmatrix}\Psi(X)\\ d\end{bmatrix} \quad (36)$$

where $\Psi(X) = G\psi(X)$ is the whitened score-vector of the sequence, X. The spherically normalized sequence kernel becomes

$$K(X_p,X_q) = \phi(\Psi(X_p)).\phi(\Psi(X_q)) \quad (37)$$

Spherical normalization is discussed in greater depth, and in the context of polynomial and RBF kernels, in [20].

V. Experiments

We carried out a number of development experiments [10] using the YOHO database [21]. Following these, we evaluated these approaches using text-independent speaker verification on the PolyVar database [12]. The PolyVar database consists of 38 client speakers, 24 male and 14 female, recorded over a telephone network. 85 utterances were recorded from each speaker in 5 sessions, with 17 utterances per session. There are also 952 impostor utterances from 56 speakers, each contributing 17 utterances in a single session. The evaluation followed the protocol for speaker model training and testing used on the European Telematics PICASSO project [22]. Approximately one thousand test utterances, including both client and impostors, were presented for each client speaker.

The speech was parameterized as 12^{th} order perceptual linear prediction (PLP) cepstral coefficients, computed using a 32ms window and a 10ms frame shift. The 12 cepstral coefficients were augmented with an energy term and first and second derivatives were estimated, resulting in frames of 39 dimensions. Cepstral mean subtraction was applied to remove the effects of the communication channel. Silence frames within each utterance were segmented out using a multilayer perceptron pre-trained on a different dataset [23].

Our baseline systems for these experiments were based on GMMs. The simplest baseline uses the client model only (2). State-of-the-art results for this database have been obtained by a GMM-LR system (5) and a modified GMM-LR/SVM system (6) in which the likelihood ratio is parameterized using an SVM to estimate the parameters [16]. To enable a direct comparison, the GMMs used in the experiments here were trained using identical conditions to those used in [16] in which cross-validation was used to estimate the optimal model complexity. This resulted in a world GMM containing 1000 Gaussian components, and client GMMs containing 200 Gaussian components.

Using the GMM-LR system on PolyVar, a text independent speaker verification result of 5.55% minimum half total error rate (HTER)[1] was reported on the PolyVar test data using 19 speakers.[2] A 4.73% minimum HTER was reported for the GMM-LR/SVM [16]. We replicated these results: the results for 38 speakers are shown in table II, and the corresponding DET curves are shown in figure 3.

We applied the score-space kernel approaches to PolyVar, based on the GMMs that made up the baseline systems. These GMMs were used to generate the score-spaces used by the kernels discussed in section IV. Using one of these kernels, each complete utterance was mapped onto a single score-vector. An SVM was trained for each client speaker using a total of 1,037 utterances (85 client and 952 background utterances), each mapped to the score-space. The SVM optimization problem for training sets of this size is straightforward and does not require any of the special techniques that have been developed to train SVMs on large quantities of data. We trained SVMs using the following kernels:

- Fisher kernel (18);
- Fisher kernel with argument (19);
- LR kernel (21);
- LR kernel with argument (22).

The Fisher kernel uses the derivatives of the client GMMs from the baseline systems to achieve the mapping to score-space, whereas the LR kernel uses both the client and world models. The number of parameters in the GMM and GMM-LR baselines are 15,800 and 94,800 respectively. The score-space dimensionalities of the Fisher and LR kernels are thus 15,800 and 94,800 respectively, with an additional dimension for each if the argument is included.

The high score-space dimensionalities, particularly that of the LR kernel, causes computational problems for SVM optimization. We addressed this problem by whitening the score-vectors to zero-mean and unit diagonal variance (section IV-C.1), and spherically normalizing them using (36). Since the

[1]The HTER is the arithmetic mean of the false acceptance rate and the false rejection rate at a given threshold. The threshold can be adjusted to minimise the HTER.

[2]Bengio and Mariéthoz [16] used 19 speakers for development and the remaining 19 speakers for evaluation. In our work, development was done using YOHO which meant that all 38 speakers could be used for evaluation.

TABLE II
RESULTS OF THE POLYVAR EXPERIMENTS. THE GMM BASELINE HAS 200 DIAGONAL COVARIANCE GAUSSIAN COMPONENTS FOR MODELLING CLIENTS. THE GMM-LR CONSISTS OF THE ABOVE PLUS A WORLD GMM WITH 1,000 DIAGONAL COVARIANCE GAUSSIANS. THE GMM-LR/SVM USES THE SAME CLIENT AND WORLD GMMS BUT COMPUTES A WEIGHTED LOG LIKELIHOOD RATIO. THE FISHER KERNEL EXPLOITS THE CLIENT GMMS WHILE THE LR KERNEL EXPLOITS BOTH CLIENT AND WORLD GMMS.

Classifier	% min HTER	% EER
Baseline		
GMM	11.22	12.07
GMM-LR	5.53	6.12
GMM-LR/SVM	5.37	5.94
Fisher kernel		
whitening	6.54	6.98
whitening + sph. norm	6.50	6.87
whitening + argument	6.50	6.92
whitening + argument + sph. norm	6.47	6.87
LR kernel		
whitening	5.13	5.55
whitening + sph. norm	3.72	4.03
whitening + argument	5.03	5.55
whitening + argument + sph. norm	3.71	4.03

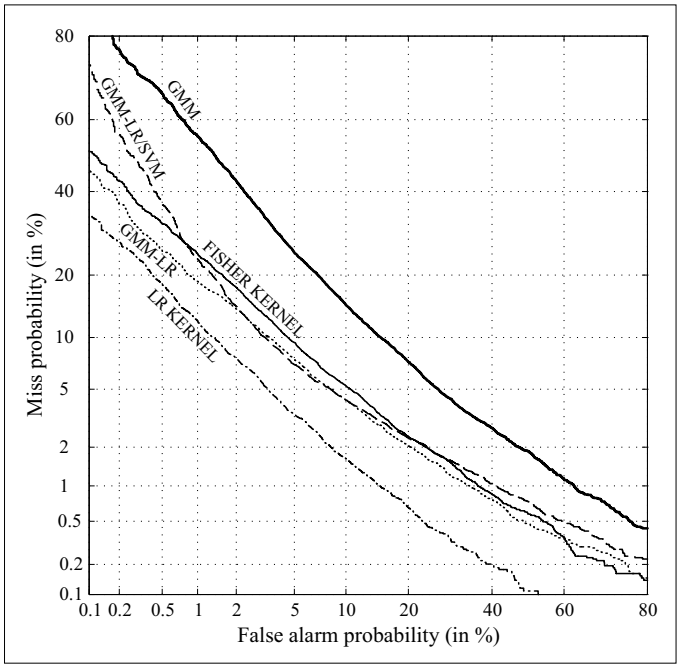


Fig. 3. DET curves of the GMM, GMM-LR, GMM-LR/SVM, Fisher kernel SVM and spherically normalised LR kernel SVM systems for text-independent speaker verification on PolyVar.

vectors are whitened, setting the spherical normalization parameter, d, to one will spread the data over a reasonably large portion of the hypersphere. Each of the four kernels itemized above were trained with and without spherical normalization.

Classification in the score-space was carried out using linear SVMs. A static RBF or polynomial kernel could be used to make non-linear decision boundaries in score-space. However, since the dimensionality of the score-space is significantly higher than the number of training vectors, the classification problem is linearly separable and non-linear boundaries are unnecessary. Also, since the problem is known to be linearly separable, the regularization parameter (C in equation 10 or 15) was set to infinity (*i.e.* the formulation for an SVM that maximizes the margin when the data is linearly separable was used). To give an indication of the value of the regularization parameter that should be used if more regularization were needed, the Lagrange multipliers in the spherically normalized likelihood ratio score-space kernel SVMs had a mean about 0.25 and an average maximum value of about 4.

We evaluated the results of our experiments using equal error rate (EER) and minimum HTER, and the results of the various systems are summarized in table II. Scores such as EER and minimum HTER reflect performance at a single operating point on the detection error trade-off (DET) curve; figure 3 uses DET curves to show the performance of the baseline systems and the SVM approaches at all operating points on PolyVar.

The GMM baseline from which the Fisher kernel is derived yielded 12.07% average EER. The Fisher kernel with whitening, but without spherical normalization or augmentation with the score-argument, reduced the average EER to 6.98%, a relative reduction of 42%. The application of spherical normalization and augmentation of the score-operator with the argument both reduced the EER but insignificantly (less than 2% relative). However, despite the improvement that the SVM imparted to the GMM system, the EER of the GMM-LR system was a further 11% lower relative to the spherically normalized Fisher kernel.

The basic LR kernel, without spherical normalization, reduces the EER to 5.55%, relative reductions of 9% compared with the GMM-LR system and 7% compared with the GMM-LR/SVM system of [16]. Spherical normalization reduced the EER to 4.03%, a further relative reduction of 27% and an overall relative reduction of 34% compared with the GMM-LR system. Spherically normalizing the LR kernel resulted in a greater error reduction compared with applying the same normalization to the Fisher kernel. The dimension of the likelihood ratio score-space is six times larger than that of the likelihood score-space due to the inclusion of the world model, hence the Hessian computed from the LR kernel is more likely to be ill-conditioned. As was observed with the Fisher kernel, augmenting the kernel with the score-argument had a negligible effect.

Figure 3 shows the DET curves of the GMM-LR, GMM-LR/SVM and spherically normalized LR kernel systems. It can be seen that the LR kernel results in a lower miss probability at all false alarm probabilities — in contrast to the GMM-LR/SVM system which has evidently optimized the parameters for a particular set of operating points (corresponding to EER and minimum HTER) at the expense of other points. At low false alarm probabilities, the LR kernel reduces the miss probability by over 20% compared to the GMM-LR system: when the probability of misclassifying an impostor is 0.1%, the GMM-LR baseline has a probability of 45.5% of rejecting a client but the LR kernel SVM has a lower probability of 35.5%.

The SVM solutions found in these experiments included nearly a thousand support vectors (from a training set of 1,037). But since a linear SVM is used, all the support vectors may be represented by a single resultant vector $\mathbf{w}_0$, defined by (11). Thus, the number of parameters required per speaker is about four times that of the underlying client and world generative models: the lengths of $\mathbf{w}_0$ and the mean and diagonal covariance vectors, for whitening the score-space, plus the total number of client and world GMM parameters, plus one for the SVM bias.

VI. Conclusion

This paper has presented and evaluated a text-independent speaker verification system based on SVMs. The SVMs use the score-space kernels approach, which subsumes the Fisher kernel, to provide direct classification of whole sequences. Two score-space kernels were examined: the Fisher (likelihood score-space) kernel and the likelihood ratio score-space kernel. The Fisher kernel exploits one generative model (that of the client speaker) to map variable length sequences onto a single vector of fixed length, while the likelihood ratio kernel exploits two models (the client model and a world model).

Mapping to a fixed length representation allows sequences of different durations to be compared and classified directly using traditional machine learning approaches. However, the score-space representation exists in a high dimension space such that most classification strategies will suffer parameterization difficulties. Fortunately, SVMs are well suited to this task and have the advantage of permitting discriminant analysis between whole sequences, unlike, for example, HMMs which only allow discriminant analysis between frames.

In order for the SVM to classify the score-space representation effectively, two normalization steps were necessary: a whitening step, which normalizes the components of the score-space to zero mean and unit variance, and spherical normalization, which tackles the variability in the dynamic range of elements in the Hessian associated with SVM optimization.

The PolyVar database was used in our evaluation. Compared to the GMM likelihood ratio baseline, the SVM approach without the use of spherical normalization reduced the average equal error rate by a relative amount of 9%. Spherical normalization enabled a much greater 34% relative reduction in the average equal error rate.

References

[1] G. Doddington, M. Przybocki, A. Martin, and D. Reynolds, "The NIST speaker recognition evaluation – overview, methodology, systems, results, perspective," *Speech Communication*, vol. 31, no. 2-3, pp. 225–254, 2000.

[2] D. A. Reynolds, "Speaker identification and verification using Gaussian mixture speaker models," *Speech Communication*, vol. 17, pp. 91–108, 1995.

[3] V. N. Vapnik, *Statistical Learning Theory*, Wiley, 1998.

[4] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 1–47, 1998.

[5] M. Schmidt and H. Gish, "Speaker identification via support vector classifiers," in *Proc. ICASSP*, 1996, vol. 1, pp. 105–108.

[6] N. Smith, M. Gales, and M. Niranjan, "Data-dependent kernels in SVM classification of speech patterns," Tech. Rep. CUED/F-INFENG/TR.387, Cambridge University Engineering Dept., 2001, http://svr-www.eng.cam.ac.uk/∼nds1002.

[7] T. S. Jaakkola and D. Haussler, "Exploiting generative models in discriminative classifiers," in *Advances in Neural Information Processing Systems 11*, M. S. Kearns, S. A. Solla, and D. A. Cohn, Eds. MIT Press, 1998.

[8] S. Fine, J. Navrátil, and R. A. Gopinath, "A hybrid GMM/SVM approach to speaker identification," in *Proc. ICASSP*, 2001, vol. 1, pp. 417–420.

[9] L. Quan and S. Bengio, "Hybrid generative-discriminative models for speech and speaker recognition," Tech. Rep. IDIAP-RR 02-06, IDIAP, March 2002.

[10] V. Wan and S. Renals, "Evaluation of kernel methods for speaker verification and identification," in *Proc. ICASSP*, 2002, vol. 1, pp. 669–672.

[11] V. Wan and W. M. Campbell, "Support vector machines for speaker verification and identification," in *Proc. Neural Networks for Signal Processing X*, 2000, pp. 775–784.

[12] G. Chollet, J. L. Cochard, A. Constantinescu, C. Jaboulet, and P. Langlais, "Swiss French PolyPhone and PolyVar: Telephone speech databases to model inter- and intra-speaker variability," in *Linguistic Databases*, John Nerbonne, Ed., pp. 117–135. 1997.

[13] A. P. Dempster, M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society*, pp. 1–39, 1977.

[14] R. A. Redner and H. F. Walker, "Mixture densities, maximum likelihood and the EM algorithm," *SIAM Review 26*, pp. 195–202, 1984.

[15] D. A. Reynolds, *A Gaussian Mixture Modelling Approach to Text-independent Speaker Identification*, Ph.D. thesis, Georgia Institute of Technology, September 1992.

[16] S. Bengio and J. Mariéthoz, "Learning the decision function for speaker verification," in *Proc. ICASSP*, 2001, vol. 1, pp. 425–428.

[17] R. Courant and D. Hilbert, *Methods of Mathematical Physics*, Interscience, 1953.

[18] N. Smith and M. J. F. Gales, "Using SVMs and discriminative models for speech recognition," in *Proc. ICASSP*, 2002, vol. 1, pp. 77–80.

[19] N. Smith and M. Gales, "Speech recognition using SVMs," in *Advances in Neural Information Processing Systems 14*. 2002, MIT Press.

[20] V. Wan, *Speaker Verification using Support Vector Machines*, Ph.D. thesis, University of Sheffield, June 2003.

[21] J. P. Campbell Jr., "Testing with the YOHO CD-ROM voice verification corpus," in *Proc. ICASSP*, 1995, vol. 1, pp. 341–344.

[22] F. Bimbot, M. Blomberg, and L. Boves, "An overview of the PICASSO project research activities in speaker verification for telephone applications," in *Proc. Eurospeech*, 1999, vol. 5, pp. 1963–1966.

[23] K. Koumpis, S. Renals, and M. Niranjan, "Extractive summarization of voicemail using lexical and prosodic feature subset selection," in *Proc. Eurospeech*, 2001, vol. 4, pp. 2377–2380.

Vincent Wan received a BA in Physics from the University of Oxford in 1997. In 1998 and 1999 he worked on hybrid speech recognition at the University of Sheffield then spent 2000 working at the Motorola Human Interface Labs at Palo Alto, California, on speech and handwriting recognition. In 2003 he recieved his Ph.D. in speaker verification and support vector machines from the University of Sheffield, where he is presently holding a post-doctoral position at the Department of Computer Science. His interests include machine learning, biometrics and speech processing.

Steve Renals received a BSc in Chemistry from the University of Sheffield, followed by an MSc in Artificial Intelligence and a PhD in Speech Recognition and Neural Networks from the University of Edinburgh. He held postdoctoral fellowships at the International Computer Science Institute, Berkeley and the University of Cambridge. He was a lecturer, then reader, in computer science at the University of Sheffield for nine years. In 2003 he was appointed professor of speech technology in the School of Informatics at the University of Edinburgh, where he is also director of the Centre for Speech Technology Research. He has published over 80 papers in the area of spoken language processing.