# A Kernel Between Sets of Vectors

**Risi Kondor** RISI@CS.COLUMBIA.EDU
**Tony Jebara** JEBARA@CS.COLUMBIA.EDU
Computer Science Department, Columbia University M.C. 0401, 1214 Amsterdam Ave., New York, NY10027

## Abstract

In various application domains, including image recognition, it is natural to represent each example as a set of vectors. With a base kernel we can implicitly map these vectors to a Hilbert space and fit a Gaussian distribution to the whole set using Kernel PCA. We define our kernel between examples as Bhattacharyya's measure of affinity between such Gaussians. The resulting kernel is computable in closed form and enjoys many favorable properties, including graceful behavior under transformations, potentially justifying the vector set representation even in cases when more conventional representations also exist.

## 1. Introduction

Kernel methods, such as Support Vector Machines, Gaussian Processes, etc., have proved to be extremely successful at a wide variety of supervised and unsupervised Machine Learning tasks. Whilst the core algorithms in this field are now fairly well crystallized (Schölkopf & Smola, 2001) and their theoretical properties have been thoroughly investigated, finding optimal ways of representing real life data as input to these algorithms is still a largely open issue.

Instead of operating on training and testing examples $\mathcal{x}_1, \mathcal{x}_2, \ldots, \mathcal{x}_m \in \mathcal{X}$ directly (where $\mathcal{X}$ is the input space), kernel based algorithms only make recourse to the value of the kernel function $K(\mathcal{x}, \mathcal{x}')$ evaluated for each pair of examples. The kernel $K$ can be any symmetric similarity measure satisfying positive (semi-) definiteness,

$$\sum_{i,j=1}^{m} c_i c_j K(\mathcal{x}_i, \mathcal{x}_j) \geq 0$$

for any $m \in \mathbb{N}$, any selection of examples $\mathcal{x}_1 \mathcal{x}_2 \ldots \mathcal{x}_m \in \mathcal{X}$, and any set of coefficients $c_1, c_2, \ldots, c_m \in \mathbb{R}$. These conditions ensure the existence of a mapping $\Phi_K : \mathcal{X} \mapsto \mathcal{F}$ to some Hilbert space $\mathcal{F}$ (called feature space), in which $K$ turns into the inner product:

$$K(\mathcal{x}, \mathcal{x}') = \langle \Phi_K(\mathcal{x}), \Phi_K(\mathcal{x}') \rangle .$$

Traditionally, examples have been represented as vectors, $\mathcal{x}_i \in \mathbb{R}^n$, and the kernel was defined as a closed form positive definite function on $\mathbb{R}^n$, such as the Gaussian Radial Basis Function (RBF) kernel

$$K(\mathcal{x}, \mathcal{x}') = e^{-\| \mathcal{x} - \mathcal{x}' \|^2 / (2\sigma^2)}.$$

More recently, it has been realized that one of the strengths of the kernel based learning paradigm is its ability to support much more general representations of the data. Indeed, the input space can be almost anything, as long as we can define a function on it that is both positive definite, and a plausible similarity measure between examples for the task at hand. This idea has given rise to a whole host of novel kernels, such as string kernels (Watkins, 2000)(Lodhi et al., 2002)(Leslie et al., 2003)(Vishwanathan & Smola, 2003), kernels on graphs (Kondor & Lafferty, 2002), kernels on automata (Cortes et al., 2003), kernels on the statistical manifold (Jaakkola & Haussler, 1999)(Lafferty & Lebanon, 2003), and kernels on more general discrete objects (Haussler, 1999)(Collins & Duffy, 2002).

In this paper we focus on representing examples as sets of vectors $\mathcal{x} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_k\}$, $\boldsymbol{x}_i \in \mathbb{R}^n$. Such a "bag of tuples" approach (Figure 1) can suit diverse domains in a natural way. For images, each tuple may correspond to a single pixel, encoding its $(x, y)$ coordinates and the corresponding intensity value. In time series analysis, tuples may encode (value, time) pairs. A video sequence can be seen as a collection of $(x, y, \text{intensity}, \text{time})$ 4-tuples.

In all of the above cases, the emphasis is not on the representation for its own sake, but rather the behavior it confers on the kernel. For instance, our kernel between sets of vectors is automatically invariant under

*Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003)*, Washington DC, 2003.

*Figure 1.* The bag of tuples representation for images. Each tuple encodes the $(x, y)$ coordinates of a pixel and the corresponding intensity value.

permutations of vectors within the set. More to the point, we are interested in kernels that are relatively insensitive to transformations $\boldsymbol{x}_i \mapsto \boldsymbol{x}_i + \boldsymbol{\delta}$, especially when $\boldsymbol{\delta}$ is a smoothly varying function of $\boldsymbol{x}$. Such "soft invariances" match intuitive notions of similarity, and are a key element in the design of high performance kernels. For example, for images, when $\delta$ is a slowly varying function of $(x, y)$, transformations of this kind correspond to translations, rotations and warpings.

To achieve this soft invariance property, we fit distributions $p$ and $p'$ to the sets $\mathcal{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_k\}$ and $\mathcal{X}' = \{\boldsymbol{x}'_1, \boldsymbol{x}'_2, \ldots, \boldsymbol{x}'_{k'}\}$ and define the kernel as the Bhattacharyya overlap measure between $p$ and $p'$. The intermediate step of fitting the distributions ensures explicit invariance in permutation and affords a degree of smoothing in $\mathbb{R}^n$. In the following we concern ourselves with finding the right parametric family of distributions to choose $p$ and $p'$ from, being sufficiently general to capture the structure of the objects we wish to represent, afford controllable smoothing, and still allow us to compute the kernel in closed form.

A similar vector set representation for images in conjunction with using the kernel trick has been proposed by Wolf and Shashua (2003). Other approaches to handling sets of vectors, collections of tuples or bags of pixels include applying PCA to the data (i.e. for several images) while maintaining each image's built-in permutational invariance (Jebara, 2003).

## 2. Kernels Between Distributions Estimated from Samples

In this paper we investigate the case when examples are presented as collections $\mathcal{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_k\}$ of $n$ dimensional vectors ($n$-tuples) $\boldsymbol{x}_i \in \mathbb{R}^n$ or $\boldsymbol{x}_i \in \Omega \subset \mathbb{R}^n$.

Monochrome bitmap images can naturally be represented in this form by letting $\boldsymbol{x} = (x, y)^\top$ encode the $x$ and $y$ coordinates of each pixel, and letting $\mathcal{X}$ be the set of all foreground pixels. For gray-scale or color images, representations of the form $\boldsymbol{x} = (x, y, \gamma)^\top$ or $\boldsymbol{x} = (x, y, \gamma_r, \gamma_g, \gamma_b)^\top$ may be used to encode information about brightness or the intensity of the RGB color components. The set $\mathcal{X}$ will then contain one such tuple for every pixel, or a random subset of pixels. If desired, images can be described in terms of more complex features (Gábor wavelets, edge features, etc.), and each tuple can code one occurrence of such a feature.

In this paper, instead of defining a kernel directly between sets of tuples, we regard $\mathcal{X}$ and $\mathcal{X}'$ as i.i.d. samples from unknown distributions $p$ and $p'$ from some parametric family $\mathcal{P}$. We proceed by defining a kernel between members of $\mathcal{P}$, and a statistical procedure for estimating $p$ from $\mathcal{X}$ and $p'$ from $\mathcal{X}'$. The vector set kernel between $\mathcal{X}$ and $\mathcal{X}'$ is then defined as the kernel between the corresponding distributions: $K(\mathcal{X}, \mathcal{X}') = K(p, p')$. To simplify the notation, in the following we omit the use of bold face font for vector quantities, in the understanding that $x$, $x'$, $x_i$, etc. will always denote members of $\mathbb{R}^n$.

### 2.1. Bhattacharyya Kernels

There are several well-known definitions of similarity or distance between distributions, such as the Kullback-Leibler divergence, Fisher kernel, $\chi^2$ distance, and so on. To define our kernel, in this paper we use Bhattacharyya's affinity (Bhattacharyya, 1943)

$$K(\mathcal{X}, \mathcal{X}') = K(p, p') = \int \sqrt{p(x)}\sqrt{p'(x)}\, dx, \qquad (1)$$

trivially related to the better known Hellinger's distance

$$H(p, p') = \left[ \int \left( \sqrt{p(x)} - \sqrt{p'(x)} \right)^2 dx \right]^{1/2}$$

by $H = \sqrt{2 - 2K}$. Expressing $\sqrt{p(x)}$ in any orthogonal basis of functions shows that $K$ is automatically positive definite. In addition, it also satisfies the normalization property

$$K(p, p) = \int p(x)\, dx = 1$$

for any $p \in \mathcal{P}$.

### 2.2. The Multivariate Normal Model

In the following we shall restrict our attention to the case where $\mathcal{P}$ is the family of multivariate normal distributions $\mathcal{N}(\mu, \Sigma)$ with probability density function

$$p(x) = \frac{1}{(2\pi)^{n/2} \left| \Sigma \right|^{1/2}} e^{-(x-\mu)^\top \Sigma^{-1} (x-\mu)/2}$$

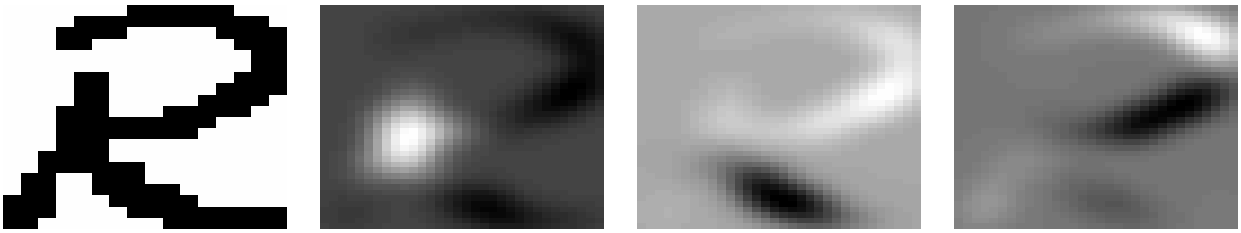

*Figure 2.* Handwritten letter "R" at $16 \times 20$ resolution and the first three kernel principal components for the Gaussian kernel with $\sigma_\kappa = 1$ pixels.

where $|\,\Sigma\,|$ denotes the determinant. For more general applications of kernels of the form (1) see (Jebara & Kondor, 2003).

We set $\mu$ and $\Sigma$ to their Maximum Likelihood estimates, given by the sample mean

$$\hat{\mu} = \frac{1}{k}\sum_{i=1}^{k} x_i \tag{2}$$

and empirical covariance matrix

$$\hat{\Sigma} = \frac{1}{k}\sum_{i=1}^{k} (x_i - \hat{\mu})\,(x_i - \hat{\mu})^\top. \tag{3}$$

A short computation shows that the Bhattacharyya kernel (1) between $p = \mathcal{N}(\mu, \Sigma)$ and $p' = \mathcal{N}(\mu', \Sigma')$ is

$$K(p, p') = |\,\Sigma\,|^{-1/4}\;|\,\Sigma'\,|^{-1/4}\;\left|\,\Sigma^\dagger\,\right|^{1/2} \exp\left(-\frac{1}{4}\mu^\top\Sigma^{-1}\mu - \frac{1}{4}{\mu'}^\top{\Sigma'}^{-1}\mu' + \frac{1}{2}{\mu^\dagger}^\top\Sigma^\dagger\mu^\dagger\right) \tag{4}$$

where $\Sigma^\dagger = \left(\frac{1}{2}\Sigma^{-1} + \frac{1}{2}{\Sigma'}^{-1}\right)^{-1}$ and $\mu^\dagger = \frac{1}{2}\Sigma^{-1}\mu + \frac{1}{2}{\Sigma'}^{-1}\mu'$. Hence, by plugging (2) and (3) into (4), the kernel $K(\mathcal{x}, \mathcal{x}')$ can be computed in closed form.

## 3. Distributions on Hilbert Space

At this point, the representational power of our kernels might seem rather limited. Certainly, for images we cannot truly hope that two dimensional Gaussians will capture sufficient detail for successful learning. To overcome this difficulty, we introduce a second positive definite kernel, $\kappa : \mathbb{R}^n \times \mathbb{R}^n \mapsto \mathbb{R}$, this time defined between the elementary vectors $x$.

Recall that for any such kernel, we can construct a Hilbert space $\mathcal{H}$ and a mapping $\Phi\colon \mathbb{R}^n \mapsto \mathcal{H}$ such that

$$\kappa(x, x') = \langle \Phi(x), \Phi(x') \rangle\,.$$

We can now repeat the construction of Section 2, this time letting $\mathcal{P}$ be a family of distributions over $\mathcal{H}$, fitting $p$ to the Hilbert space points $\Phi(x_1), \ldots, \Phi(x_k)$, and defining the kernel as

$$K(\mathcal{x}, \mathcal{x}') = K(p, p') = \int_{\mathcal{H}} \sqrt{p(z)}\sqrt{p'(z)}\,dz. \tag{5}$$

For typical kernels, the $\Phi(x_i)$ will span a subspace in $\mathcal{H}$ of dimensionality much greater than $n$, allowing simple parametric families of distributions over $\mathcal{H}$ to capture complex structures in the original sample.

Our choice of $\kappa$ in our experiments on images in Section 5 will be the familiar Gaussian RBF kernel

$$\kappa(x, x') = e^{-\|\,x - x'\,\|^2/(2\sigma_\kappa^2)}$$

but our method is not in any way limited to this particular kernel.

Independent of our work, the same idea of defining a kernel between a swarm of Hilbert space vectors induced by another kernel has recently been proposed by Wolf and Shashua (2003), also in the context of representing images as sets of vectors. In contrast to our distribution-based approach, Wolf and Shashua concentrate on the subspaces spanned by the $\Phi(x_i)$ and define their kernel via principal angles between such subspaces.

We now discuss how, in the special case of the multivariate normal model, (5) can be computed in closed form without the need to explicitly construct the images $\Phi(x_i)$.

### 3.1. Normal Distributions on $\mathcal{H}$

To facilitate the following discussion, we adopt Dirac's bra-ket notation (Dirac, 1930) for Hilbert space objects. The "ket" $|x\rangle$ will denote $\Phi(x)$ and the "bra" $\langle x|$ will denote its dual, the analog of $\Phi(x)^\top$ for finite dimensional vector spaces. Bras and kets labeled with letters other than $x$ will denote general elements of $\mathcal{H}$, which might or might not be the images of some $x \in \mathbb{R}^n$ under $\Phi$.

The inner product between $|\xi\rangle$ and $|\xi'\rangle$ is simply written $\langle \xi | \xi' \rangle$, while expressions of the form $|\xi\rangle\,\langle\xi'|$, or

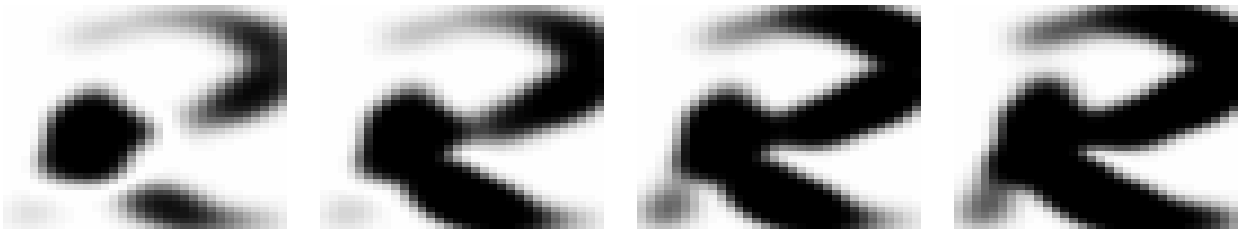

*Figure 3.* Reconstruction of the letter "R" from the first $r = 1, 2, 3$ and 4 principal components in $\mathcal{H}$. For each pixel $x$, the shading reflects the p.d.f. at $\Phi(x)$ of a Gaussian fitted to the first $r$ principal components of the images under $\Phi$ of elementary $(x, y)$ vectors of the black pixels in the original figure (in all orthogonal directions, the p.d.f. is uniform).

weighted sums of such expressions, $\Sigma = \sum_i |\xi_i\rangle a_i \langle\xi_i|$, are symmetric bilinear forms on $\mathcal{H}$, corresponding to symmetric matrices in the finite dimensional case. The power of Dirac's notation begins to show when considering the corresponding linear mapping $\Sigma : \mathcal{H} \mapsto \mathcal{H}$:

$$|\zeta\rangle \mapsto \left( \sum_i |\xi_i\rangle a_i \langle\xi_i| \right) |\zeta\rangle = \sum_i |\xi_i\rangle a_i \langle\xi_i|\zeta\rangle \,,$$

where, of course, each $\langle\xi_i|\zeta\rangle$ is just a number. Let $V_\Sigma$ denote the orthogonal complement to the nullspace of $\Sigma$, $V_\Sigma = \{\, |z\rangle \in \mathcal{H} : \langle z|z'\rangle = 0 \quad \forall\, |z'\rangle \in \mathcal{H}$ such that $\Sigma\,|z'\rangle = |0\rangle \,\}$. Note that for invertible $\Sigma$ ($V_\Sigma = \mathcal{H}$), provided the $|\xi_i\rangle$ form an orthonormal set ($\langle\xi_i|\xi_j\rangle = \delta_{ij}$), the inverse of the $\Sigma$ will simply be $\Sigma^{-1} = \sum_i |\xi_i\rangle a_i^{-1} \langle\xi_i|$.

A finite dimensional Normal distribution $\mathcal{N}(|\mu\rangle\,, \Sigma)$ on $\mathcal{H}$ is of the form

$$p(|z\rangle) = \frac{1}{(2\pi)^{d/2}\,|\,\Sigma\,|} e^{-(\langle z|-\langle\mu|)\,\Sigma^{-1}(|z\rangle-|\mu\rangle))/2} \qquad (6)$$

where $\Sigma$ is a symmetric, positive definite bilinear form of rank $d$. Note that this is a proper distribution only on $V_\Sigma$, not on the whole of $\mathcal{H}$, since $p(|z\rangle)$ is uniform in all directions orthogonal to $V_\Sigma$. Plugging in the empirical mean and covariance

$$|\hat{\mu}\rangle = \frac{1}{k} \sum_{i=1}^{k} |x_i\rangle \qquad (7)$$

$$\hat{\Sigma} = \frac{1}{k} \sum_{i=1}^{k} (|x_i\rangle - |\hat{\mu}\rangle)\,(\langle x_i| - \langle\hat{\mu}|)$$

as before is unlikely to lead to good results, since in the Bhattacharyya kernel this will not penalize for the lack of alignment between the spaces $V_{\hat{\Sigma}}$ and $V_{\hat{\Sigma}'}$.

A related problem is that of overfitting. In particular, for the Gaussian RBF kernel it can be shown that the $|x_i\rangle$ will span a subspace of dimension exactly $k$ in $\mathcal{H}$. Fitting a $k$ dimensional Normal distribution to $k$ data points is not robust in the directions of low covariance, nor are these directions informative. Generally, the first few eigenvectors of the covariance matrix give a good description of the data: carrying around all the eigenvectors is wasteful and potentially misleading.

To address both problems at the same time, instead of $\hat{\Sigma}$, we take $\Sigma$ to be the regularized covariance form

$$\Sigma_{\text{reg}} = \sum_{l=1}^{r} |v_l\rangle\, \lambda_l\, \langle v_l| + \eta \sum_i |\zeta_i\rangle\, \langle\zeta_i|\,, \qquad (8)$$

where $|v_1\rangle\,, \ldots, |v_r\rangle$ are the $r$ largest eigenvectors of $\hat{\Sigma}$, $\lambda_1, \ldots, \lambda_r$ are the corresponding eigenvalues, $\eta$ is a regularization constant, and the $|\zeta_i\rangle$ form an orthonormal basis for $\mathcal{H}$.

Note that in the case that $\mathcal{H}$ is infinite dimensional, the denominator in (6) becomes divergent. Strictly speaking, $p$ is not a normal distribution anymore but a Gaussian Process, as we shall discuss in Section 4. However, in the formula for the Bhattacharyya kernel (4) these diverging normalization factors will cancel, conforming to our intuition that all action is limited to the finite dimensional subset of $\mathcal{H}$ spanned by the data.

The technique of computing eigenvectors in feature space is known as Kernel Principal Component Analysis (Kernel PCA), and was developed in (Schölkopf et al., 1998) in the context of unsupervised learning. We now review this technique and show how to construct the eigenvectors $|v_j\rangle$ without any explicit calculations in $\mathcal{H}$.

### 3.2. Kernel PCA

The key observation in Kernel PCA is that the eigenvectors $|v_l\rangle$ lie in the span of the images $|x_i\rangle$, or, equivalently, the centered images $|x_i^*\rangle = |x_i\rangle - |\hat{\mu}\rangle$:

$$|v\rangle = \sum_{i=1}^{k} \alpha_i\, |x_i^*\rangle \qquad \alpha_i \in \mathbb{R}. \qquad (9)$$

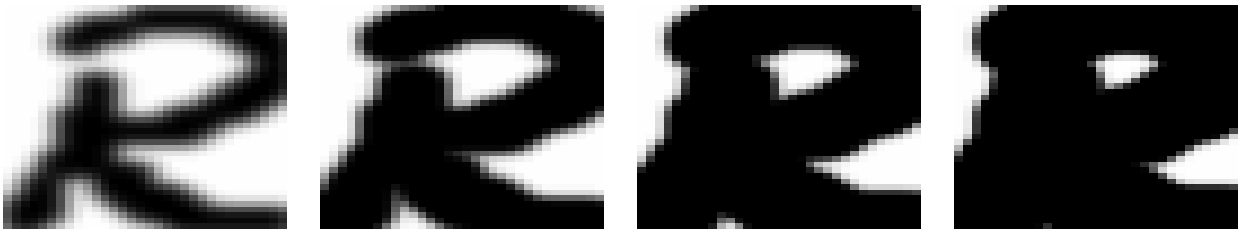

*Figure 4.* Reconstruction of the letter "R" from a Gaussian based on the first 3 KPCA components augmented by a diagonal term with $\eta = 1,\ 0.1,\ 0.01$ and $0.001$.

Plugging (9) in the eigenvector equation $\hat{\Sigma}\,|v\rangle = \lambda\,|v\rangle$,

$$\frac{1}{k}\sum_{j=1}^{k} |x_j^*\rangle\,\langle x_j^*|\sum_{i=1}^{k}\alpha_i\,|x_i^*\rangle = \lambda\sum_{i=1}^{k}\alpha_i\,|x_i^*\rangle,$$

and multiplying on the left by any $\langle x_l^*|$ gives

$$\frac{1}{k}\sum_{j=1}^{k}\sum_{i=1}^{k}\langle x_l^*|x_j^*\rangle\,\langle x_j^*|x_i^*\rangle\,\alpha_i = \lambda\sum_{i=1}^{k}\langle x_l^*|x_i^*\rangle\,\alpha_i.$$

Observe that these sums are but regular matrix multiplications in disguise, so equivalently,

$$K^{*2}\alpha = \lambda\,k\,K^*\alpha,$$

where $K^*$ is the centered Gram matrix, $K^*_{i,j} = \langle x_i^*|x_j^*\rangle = \kappa(x_i^*, x_j^*)$ and $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_k)^\top$. Hence, finding the principal components of the typically very high, possibly infinite, dimensional vectors $|x_i\rangle$ reduces to the $k$-dimensional eigenvector problem

$$K^*\alpha^{(l)} = k\,\lambda_l\,\alpha^{(l)}. \tag{10}$$

Figure 2 shows the first three kernel principal components of $\hat{\Sigma}$ for a handwritten letter $R$, mapped back to the original image plane by $v_l^{\text{induced}}(x) = \langle x|v_l\rangle$. The principal components capture visually recognizable features of the figure.

Figure 3 shows the reconstruction of the same letter from $1, 2, 3$ and $4$ dimensional Normal distributions in $\mathcal{H}$ with no regularization term in $\Sigma$. Note that thanks to the nonlinearity of $\Phi$, four components can already capture the appearance of the original letter quite well.

Finally, Figure 4 shows the reconstruction based on regularized Gaussian model with three principal components. Note that the recovered images are closer to the original than in Figure 3 and that the effect of tuning $\eta$ is similar to smoothing in the image plane.

### 3.3. Computing the Bhattacharyya Kernel

It remains to put all the pieces together and compute the Bhattacharyya kernel between $p = \mathcal{N}(|\mu\rangle, \Sigma)$ and $p' = \mathcal{N}(|\mu'\rangle, \Sigma')$, where now $\Sigma = \hat{\Sigma}$ and $\Sigma' = \hat{\Sigma}'$.

Recall that $V_\Sigma$ is the orthogonal complement of the nullspace of $\Sigma$. It is easy to see that in (5) dimensions orthogonal to $W = V_\Sigma \oplus V_{\Sigma'}$ integrate out to 1, relieving us of the need to take determinants, etc., of infinite dimensional forms: as in (4), the kernel is given by

$$K(p, p') = |\,\Sigma_W\,|^{-1/4}\,|\,\Sigma'_W\,|^{-1/4}\,|\Sigma^\dagger_W|^{1/2} \\ e^{-\langle\mu|\Sigma^{-1}|\mu\rangle/4}\,e^{-\langle\mu'|\Sigma'^{-1}|\mu'\rangle/4}\,e^{-\langle\mu^\dagger|\Sigma^{\dagger\,-1}|\mu^\dagger\rangle/2}$$

where $|\Sigma^\dagger\rangle = \left(\frac{1}{2}|\Sigma^{-1}\rangle + \frac{1}{2}|\Sigma'^{-1}\rangle\right)^{-1}$ and $|\mu^\dagger\rangle = \frac{1}{2}\Sigma^{-1}|\mu\rangle + \frac{1}{2}\Sigma'^{-1}|\mu\rangle$. The subscript $W$ denotes the matrix corresponding to the restriction of the given form to the subspace $W$.

The term $\langle\mu|\Sigma^{-1}|\mu\rangle$ and its dashed counterpart can be evaluated by expansion into linear combinations of centered and then uncentered bras and kets, ultimately reducing it to a weighted sum of kernel evaluations $\langle x_i|x_j\rangle = \kappa(x_i, x_j)$. The determinant $|\,\Sigma\,|$ is easily computed via $|\,\Sigma\,| = \eta^{\dim(W)-r}\prod_{i=1}^{r}(\lambda_i + \eta)$ and similarly for $|\,\Sigma'\,|$.

The mixed determinant and the mixed term in the exponent require explicit construction of the matrices $\Sigma_W = [\langle\xi_i|\,\Sigma\,|\xi_j\rangle]_{i,j}$ and $\Sigma'_W$, where $\{|\xi_l\rangle\}_l$ is an orthonormal basis for $W$. It is easiest to construct this basis by starting with the basis of $V_\Sigma$ given by the eigenvectors $|v_l\rangle$ and extending it one vector at a time by adding the eigenvectors of $\Sigma'$ and performing Gram-Schmidt orthogonalization.

## 4. Relationship to Gaussian Processes

So far, we have not said anything about what the elements of $\mathcal{H}$ actually are. We now show that the natural interpretation is that they are functions over our original space, $\mathbb{R}^n$.

Let us identify $|x\rangle = \Phi(x)$ with the function $f_x = \kappa(x, \cdot)$ and extend this linearly, $|\xi\rangle = \sum_i c_i\,|x_i\rangle$ for any $x_1, x_2, \ldots \in \mathbb{R}^n$ and $c_1, c_2, \ldots \in \mathbb{R}$ corresponding to $f_\xi = \sum_i c_i f_{x_i}$. In the continuous limit $|\xi\rangle = \int c(x)\,|x\rangle\,dx$ is identified with $f_\xi = \int c(x) f_x\,dx$. In the following,

$|\xi\rangle$ and $|f_\xi\rangle$ will be used interchangeably. The curious-looking property

$$\langle f_\xi | f_x \rangle = \left( \int c\,(x') \, \langle f_{x'}| \; dx' \right) |f_x\rangle = \int c\,(x')\, \kappa(x,x')\, dx' = f_\xi(x),$$

in particular, $\langle f_{x'}|f_x\rangle = f_{x'}(x) = \kappa(x,x')$, lends this construction the name of Reproducing Kernel Hilbert Space, commonly abbreviated RKHS.

For images, the interpretation of the above is particularly clear. Suppose that we are dealing with monochrome images over the unit square, i.e. $x \in \Omega = [0,1]^2$. Each $|\xi\rangle$ is now a function $f_\xi : [0,1]^2 \mapsto \mathbb{R}$ with $f_\xi(x) = \langle \xi | f_x \rangle$ i.e., it is itself an image.

The analog of the normal distribution for function spaces is the Gaussian Process. More precisely, a set of real valued random variables $\{y_z \; : \; z \in Z\}$ for some index set $Z$ is said to form a Gaussian Process $\mathcal{G}$ if for any $z_1, z_2, \ldots, z_k \in Z$, the marginal distribution $p\,(y_{z_1}, y_{z_2}, \ldots, y_{z_k})$ is multivariate normal.

When $Z \subset \mathbb{R}^n$, it is natural to regard $\mathcal{G}$ as a distribution over functions $g : Z \mapsto \mathbb{R}$, $g(z) = y_z$. An important property is that by defining the mean $\mathrm{E}[g(z)]$ and covariance function $\mathrm{Cov}(g(z), g(z'))$, all the marginals, and hence $\mathcal{G}$ itself, is uniquely defined.

The Gaussian Process concept meshes in naturally with the above RHKS point of view. Setting $Z = \Omega$, replacing the $z$'s with $x$'s and letting $g(x) = \langle \xi | x \rangle = f_\xi(x)$ makes $\mathcal{G}$ into a distribution on $\mathcal{H}$. We see that the previously laboriously fitted "generalized normal distribution" $p$ over $\mathcal{H}$ is nothing but a Gaussian Process with mean $\mathrm{E}[f(x)] = \frac{1}{k} \sum_i \kappa(x_i, x)$ and covariance $\mathrm{Cov}(f(x), f(x')) = \langle f_x | \, \Sigma_{\mathrm{reg}} \, | f_{x'} \rangle$.

Our kernel PCA-based procedure can then be interpreted as fitting a Gaussian Process to the sample of functions $\{ \, f_{x_i} = \kappa(x_i, \cdot) \; : \; x_i \in \mathcal{X} \, \}$. The resulting distribution over functions, $p(f)$ really encodes our beliefs of how similar each $f$ is to the image whose pixels are $\mathcal{X}$. The Bhattacharyya kernel (5) then defines similarity between $\mathcal{X}$ and $\mathcal{X}'$ as the integral over all $f$ of (the square root of) how similar $\mathcal{X}$ is to $f$ and how similar $f$ is to $\mathcal{X}'$.

In the Machine Learning literature, there is a long history of using Gaussian Processes as a compact Bayesian function learning tool by itself, without the need to invoke any outside estimation procedure (Zhu et al., 1997)(Mackay, 1997). This method is based on the fact that a Gaussian Process prior updated with observations $t(x_i) = f(x_i) + \epsilon$ (where $\epsilon$ is additive Gaussian noise of known variance) gives rise to a posterior that is also a Gaussian Process.

The question arises as to why we do not estimate $p$ using this Bayesian approach. The answer is that although both methods yield GP estimators, they are fundamentally different: whereas the "classical" GP procedure is a regression tool, our Kernel PCA-based procedure is a density estimator.

The justification for our estimation procedure is essentially the same as that for the traditional MLE estimator for Normal distributions. Given a sample of functions $\{ \, f_{x_i}(x) = \kappa(x_i, x) \}_{i=1}^k$, the maximum likelihood Gaussian Process to generate these functions is that with mean $\mathrm{E}[f(x)] = \frac{1}{k} \sum_{i=1}^k \kappa(x_i, x)$ and covariance $\mathrm{Cov}[f(x), f(x')] = \frac{1}{k} \sum_{i=1}^k \kappa(x, x_i) \kappa(x_i, x')$. Our estimator $p$ is a regularized approximation to this GP, using only the first $r$ components of the covariance form. A potentially more satisfying Bayesian approach to estimating $\mu$ and $\Sigma$ that would also involve estimating $r$ and $\eta$ along the lines of (Zhu et al., 1997) remains the subject of further research.

## 5. Experiments

### 5.1. Crosses and Squares

To explore the robustness of the vector sets kernel to spatial variation, in a preliminary classification experiment we generated 100 monochromatic images of crosses and squares at various positions and scales in a $40 \times 40$ pixel field (Figure 5).

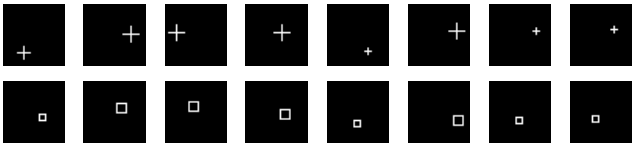

*Figure 5.* Synthetic Images Data Set. Crosses and squares were generated under random translation and rescaling.

We trained a support vector machine to separate the cross images from the square images using half the dataset for training and the other half for testing. As a baseline, we compare against the standard method of treating each image as a single vector in $\mathbb{R}^{1600}$ to which we apply a conventional Gaussian RBF kernel. Figure 6 depicts the classification accuracy as a function of the SVM regularization parameter $C$. Multiple curves are shown for the various settings of $\sigma$ for the conventional RBF and for various settings of the analogous $\sigma_\kappa$ parameter in the Gaussian base kernel of our novel point set kernel. For regularization we keep the first $r = 4$ principal components and use $\eta = 0.01$, which

were empirically found to be reasonable values.

Clearly, provided that $\sigma_\kappa$ is appropriate, the point set kernel can easily outperform the traditional RBF. The latter is severely handicapped by the crosses and squares appearing in different parts of the figure because it is only sensitive to coincidence of pixels and is unaware of the relative position of pixels. In contrast, the point set kernel can abstract shape from position to some degree.

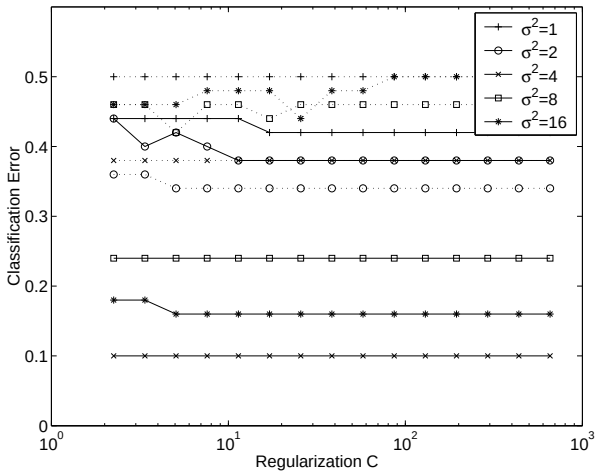


*Figure 6.* The Bhattacharyya point set kernel (solid lines) can achieve much lower testing error on the synthetic image dataset than the best conventional RBF (dotted lines).

### 5.2. Handwritten Digits

Towards comparing our kernel with common benchmarks in a familiar setting, we conducted experiments on an intentionally small dataset of handwritten digits, consisting of just the first twenty examples of each of the digits $0, 1, \ldots, 9$ from the NIST dataset. To test how well we can learn visual patterns from sparse, noisy examples, instead of the original images, we sampled 30 pixels from the foreground region of each image (intensity greater than 191 on a 0 to 255 scale) and only presented the coordinates of these pixels to the algorithm.

Experiments were performed by training on 120 images and testing on the remaining 80, averaging the performance over 10 such random splits, thereby on average giving 12 positive training examples for each class. Pursuing a simple one-versus-all strategy, separate Support Vector Machines were built for each class, and in testing the predicted class was chosen to be the one with highest $\sum_i \alpha_i K(\mathcal{X}_i, \mathcal{X}) + b$, where $\alpha_i$ are the usual support vector coefficients and $b$ is the bias term.

Results are compared to the baseline of using a conventional RBF or a dot product kernel on the sparsified images (Figure 7). Clearly, the performance of the point set kernel is very sensitive to the choice of $\sigma_\kappa$, but has the potential to far outperform the baseline. As in the previous experiment, no attempt has been made to optimize performance over $r$ and $\eta$ (10 and 0.1, respectively): a more systematic study would set these parameters by cross-validation or by identifying a drop-off point (eigen-gap) in the spectrum of $\Sigma$.

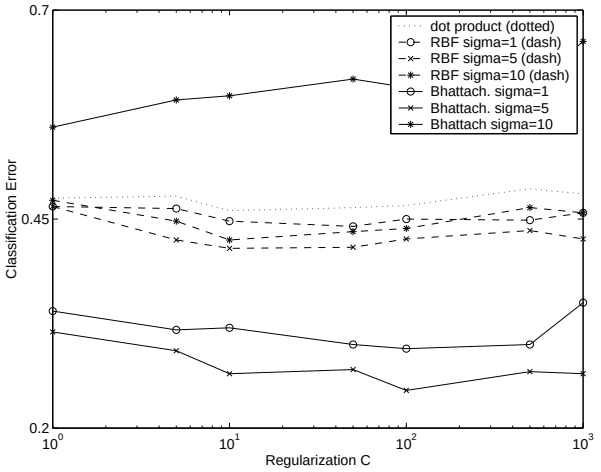


*Figure 7.* The Bhattacharyya point set kernel (solid lines) is very sensitive to $\sigma_\kappa$, but can far outperform the conventional Gaussian RBF (dashed) and dot product (dotted) kernels on the sparsified NIST images task.

## 6. Conclusions

We have proposed a novel kernel that applies to a wide class of learning problems where instances can be represented as sets of vectors. The kernel is defined as Bhattacharyya's affinity between Gaussian models fitted to the set. The resulting kernel becomes powerful when the whole procedure is "kernelized" by the introduction of a second kernel $\kappa$, defined between elementary vectors.

The "bag of tuples" representation of instances is itself worthy of further exploration. In addition to explicit invariance to permutation, by treating all variables on the same footing, the base kernel can extend its favorable smoothness properties to all variables. Contrast this with a conventional Gaussian RBF kernel between images, where $(x, y)$ pixel coordinates are treated as indices and the kernel only behaves gracefully in intensity. Such a traditional kernel has no concept of the metric structure of $(x, y)$ and consequently behaves poorly under translation, rotation, etc..

The choice of parametric model is essentially constrained to Gaussians by the dual requirements of ker-

nelizability and the existence of a closed form formula for Bahattacharyya's affinity. On the other hand, the base kernel $\kappa$ can be chosen freely, making our method quite flexible. Indeed, the restriction to sets of vectors in the title is unnecessary: the $x_i$ could come from any continuous or discrete set $X$ on which a meaningful kernel can be defined. Another possible extension of this work is to apply our method recursively to sets of sets. Finally, it might be possible to integrate each step of our procedure, including estimating $r$ and $\eta$, into a single consistent Bayesian operation.

## Acknowledgments

We would like to thank Lior Wolf, Patrick Haffner and the anonymous referees for corrections and several important comments which have been integrated into the paper.

## References

Bhattacharyya, A. (1943). On a measure of divergence between two statistical populations defined by their probability distributions. *Bull. Calcutta Math Soc.*, *35*, 99–110.

Collins, M., & Duffy, N. (2002). Convolution kernels for natural language. *Advances in Neural Information Processing Systems 14* (pp. 625–632). Cambridge, MA: MIT Press.

Cortes, C., Haffner, P., & Mohri, M. (2003). Rational kernels. *Advances in Neural Information Processing Systems 15*. Cambridge, MA: MIT Press.

Dirac, P. A. M. (1930). *The principles of quantum mechanics*. Oxford University Press.

Haussler, D. (1999). *Convolution kernels on discrete structures* (Technical Report UCSC-CRL-99-10). Department of Computer Science, University of California at Santa Cruz.

Jaakkola, T., & Haussler, D. (1999). Exploiting generative models in discriminative classifiers. *Advances in Neural Information Processing Systems 11*. Cambridge, MA: MIT Press.

Jebara, T. (2003). Convex invariance learning. *Ninth International Workshop on Artificial Intelligence and Statistics*.

Jebara, T., & Kondor, R. (2003). Bhattacharyya and expected likelihood kernels. *Proceedings of the Sixteenth Annual Conference on Learning Theory and Seventh Kernel Workshop*. In press.

Kondor, R., & Lafferty, J. (2002). Diffusion kernels on graphs and other discrete input spaces. *Machine Learning: Proceedings of the Nineteenth International Conference (ICML '02)*.

Lafferty, J., & Lebanon, G. (2003). Information diffusion kernels. *Advances in Neural Information Processing Systems 15*. Cambridge, MA: MIT Press.

Leslie, C., Eskin, E., Weston, J., & Noble, W. S. (2003). Mismatch string kernels for SVM protein classifiaction. *Advances in Neural Information Processing Systems 15*. Cambridge, MA: MIT Press.

Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., & Watkins, C. (2002). Text classification using string kernels. *Journal of Machine Learning Research*, *2*, 419–444.

Mackay, D. J. C. (1997). Gaussian processes: A replacement for neural networks? *Tutorial at the Tenth Annual Conference on Neural Information Processing Systems*. Available from `http://wol.ra.phy.cam.ac.uk/pub/mackay/`.

Schölkopf, B., & Smola, A. J. (2001). *Learning with kernels: Support vector machines, regularization, optimization and beyond*. Cambridge, MA: MIT Press.

Schölkopf, B., Smola, A. J., & Müller, K.-R. (1998). Nonlinear principal component analysis as a kernel eigenvalue problem. *Neural Computation*, *10*, 1299–1319.

Vishwanathan, S. V. N., & Smola, A. J. (2003). Fast kernels for string and tree matching. *Advances in Neural Information Processing Systems 15*. Cambridge, MA: MIT Press.

Watkins, C. (2000). Dynamic alignment kernels. In A. J. Smola, B. Schölkopf, P. Bartlett, and D. Schuurmans (Eds.), *Advances in kernel methods*. Cambridge, MA: MIT Press.

Wolf, L., & Shashua, A. (2003). Kernel principal angles for classification machines with applications to image sequence interpretation. *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Zhu, H., Williams, C. K. I., Rohwer, R., & Morciniec, M. (1997). *Gaussian regression and optimal finite dimensional linear models* (Technical Report NCRG/97/011). Aston University, Neural Computing Research Group.