Nearest Hyperdisk Methods for High-Dimensional Classification

Hakan Cevikalp HAKAN.CEVIKALP@GMAIL.COM
Eskisehir Osmangazi University, Eskisehir, Turkey

Bill Triggs BILL.TRIGGS@IMAG.FR
Laboratoire Jean Kuntzmann, Grenoble, France

Robi Polikar POLIKAR@ROWAN.EDU
Rowan University, Glassboro, NJ USA

Abstract

In high-dimensional classification problems it is infeasible to include enough training samples to cover the class regions densely. Irregularities in the resulting sparse sample distributions cause local classifiers such as Nearest Neighbors (NN) and kernel methods to have irregular decision boundaries. One solution is to "fill in the holes" by building a convex model of the region spanned by the training samples of each class and classifying examples based on their distances to these approximate models. Methods of this kind based on affine and convex hulls and bounding hyperspheres have already been studied. Here we propose a method based on the *bounding hyperdisk* of each class – the intersection of the affine hull and the smallest bounding hypersphere of its training samples. We argue that in many cases hyperdisks are preferable to affine and convex hulls and hyperspheres: they bound the classes more tightly than affine hulls or hyperspheres while avoiding much of the sample overfitting and computational complexity that is inherent in high-dimensional convex hulls. We show that the hyperdisk method can be kernelized to provide nonlinear classifiers based on non-Euclidean distance metrics. Experiments on several classification problems show promising results.

1. Introduction

Nearest neighbours (NN) – assigning the query to the class with the nearest training sample(s) under some suitable distance metric – is one of the simplest methods for multi-class classification. Asymptotically it makes at most twice as many errors as the optimal Bayes rule classifier, but this result assumes dense sampling which requires training sets that are exponentially large in the dimensionality of the underlying feature space class distributions. In high-dimensional problems such as text, gene or visual object classification, tractable training sets are necessarily much smaller than this, and the performance of NN can often be poor. The main problem is the sparse and irregular distribution of the training samples, which often leaves "holes" in the input space – regions that have few or no nearby training samples from the relevant class. Equivalently, local density estimates in high dimensions are intrinsically noisy because any region with a radius significantly smaller than that of the class has such a small volume relative to that of the class that it typically contains few or no samples. These effects make the inter-class decision boundaries of high dimensional NN and local kernel based methods erratic, thus leading to classification errors.

One way to circumvent this problem is to approximate each class with a point set that "fills in the holes" between the examples. In particular, any convex set containing the examples has this property. Several approximations of this kind have already been studied including the affine hulls, convex hulls, bounding hyperspheres and bounding hyperellipsoids of the examples (Gulmezoglu et al., 2001, Laaksonen, 1997, Nalbantov et al., 2007, Vincent & Bengio, 2001). Despite the simplicity of their geometry, such approximations are useful in high dimensions because in any case fine local details can not be resolved with practical numbers of samples. Queries are classified to the class whose convex approximation is closest to the query point – a convex nearest-point problem that can be solved reasonably efficiently with standard methods. This is equivalent to NN in which additional points are fantasized to fill in the set of each class.

Affine hulls (*i.e.* spanning linear subspaces that have

Appearing in *Proceedings of the 25^{th} International Conference on Machine Learning*, Helsinki, Finland, 2008. Copyright 2008 by the author(s)/owner(s).

been shifted to pass through the centroid of the class) were first used for global classifiers of isolated words and hand-written digits in (Gulmezoglu et al., 2001, Laaksonen, 1997), giving good classification performance. Similarly, (Nalbantov et al., 2007) used convex hulls for global classifiers on some of the UCI and SlatLog problems, comparing these to Support Vector Machines (SVMs) both theoretically and empirically. Such global convex approximations may fail to capture the decision boundaries of classes with nonlinear boundaries and one can also build more local approximations, or even build a separate approximation for each query sample based on convex approximations of its k nearest neighbours from each class. Again the query is classified to the (locally) nearest hull. Although this is not immune to the hole problem, (Vincent & Bengio, 2001) reported significant improvements over traditional NN for affine and convex hull methods of this kind in handwritten digit classification. Another way to handle complex boundaries is via nonlinear mapping to a high-dimensional feature space (*e.g*. via a kernel) followed by a global convex set approximation of the kind described below.

Besides classification, approximations based on affine or convex hulls have also been used for dimensionality reduction. Mixtures of Principal Component Analyzers can be used to approximate nonlinear data manifolds under local linearity assumptions (Hinton et al., 1997). Locally Linear Embedding (Roweis & Saul, 2000) approximates the nonlinear structure of high-dimensional data by exploiting local affine/convex reconstructions. (Verbeek, 2006) combined several locally valid linear manifolds to obtain a global nonlinear mapping between the high-dimensional sample space and a low-dimensional manifold. In (Cevikalp et al., 2008), we proposed a margin based discriminative dimensionality reduction method based on convex models of classes.

The current paper presents a new convex approximation based classifier that models each class with its bounding hyperdisk – the intersection of the affine hull and the minimal bounding hypersphere of its training examples. Hyperdisks are attractive primitives because they maintain the stability of the affine hull and hypersphere methods while providing better localization of the training samples and hence potentially better discrimination. Convex hull approximations tend to be unrealistically tight (for practical training set sizes, classes typically extend considerably beyond the convex hull of the training samples) while affine hull and hypersphere ones tend to be too loose in complementary senses (one too "broad", the other too "deep"). The hyperdisk approach to some extent captures the best aspects of each method. It can be applied both globally and locally and it is simple enough to be expressible in terms of dot products and hence to allow kernelization.

The paper is organized as follows. In section 2 we recall the affine and convex hull based methods. Section 3 introduces the hyperdisk method. Section 4 describes our experiments and data sets. Finally, section 5 presents conclusions and future directions.

2. Background on Related Methods

2.1. Nearest Affine Hull (NAH) Classification

Let the training samples be $\mathbf{x}_{ci} \in \mathbb{R}^d$, where $c = 1, \ldots, C$ indexes the C classes and $i = 1, \ldots, N_c$ indexes the N_c samples of class c. We suppose that the affine hull of the samples from each class is a proper subset of $\mathbb{R}^d$ of dimension less than d (which certainly holds when $N_c \ll d$). The affine hull is the affine span of the training samples, *i.e*. the smallest affine subspace containing them

$$H_c^{\text{aff}} = \left\{ \mathbf{x} = \textstyle\sum_{i=1}^{N_c} \alpha_i \, \mathbf{x}_{ci} \;\middle|\; \sum_i \alpha_i = 1 \right\}. \qquad (1)$$

The affine hull gives a rather loose approximation to the class region because it does not constrain the position of the training points within the affine subspace. The distance from a query point $\mathbf{x}_q$ to an affine hull H_c^{aff} is the norm of the displacement from $\mathbf{x}_q$ to the closest point on the hull, which can be expressed as the orthogonal projection of $\mathbf{x}_q$ normal to the subspace (see, *e.g*., (Cevikalp et al., 2007) for derivations):

$$d(\mathbf{x}_q, H_c^{\text{aff}}) = \|(\mathbf{I} - \mathbf{P}_c)(\mathbf{x}_q - \boldsymbol{\mu}_c)\| = \|\mathbf{P}_c^{\perp} \mathbf{x}_q - \boldsymbol{\mu}_c^{\perp}\|. \quad (2)$$

Here: $\mathbf{I}$ is the identity matrix, $\mathbf{P}_c$ is the orthogonal projection onto the spanning subspace (the range of the covariance matrix) of the class-c training samples, and $\mathbf{P}_c^{\perp} = \mathbf{I} - \mathbf{P}_c$ is the orthogonal projection onto the null space of the covariance – *i.e*. the orthogonal complement of the spanning subspace, called the *indifference subspace* in (Gulmezoglu et al., 2001, Cevikalp et al., 2005). $\boldsymbol{\mu}_c$ can be any reference point in H_c^{aff} – *e.g*. one of the samples $\mathbf{x}_{ci}$, or their mean – and $\boldsymbol{\mu}_c^{\perp} = \mathbf{P}_c^{\perp} \boldsymbol{\mu}_c$, the residual of $\boldsymbol{\mu}_c$ under the projection, encodes the orthogonal displacement of H_c^{aff} from the origin.

As its name suggests, the NAH classifier assigns the query to the class whose affine hull is the closest:

$$g(\mathbf{x}_q) = \min_{c=1,\ldots,C} (d(\mathbf{x}_q, H_c^{\text{aff}})). \qquad (3)$$

Equivalently, NAH chooses the class that provides the best (smallest $\|\text{error}\|$) reconstruction of the query using an affine combination of training samples. The decision boundaries of NAH are piecewise quadratic. Numerically, point projections can be computed on the fly without explicitly evaluating and storing the $d \times d$ projection matrices $\mathbf{P}_c$ and $\mathbf{P}_c^{\perp}$ by using $\mathbf{P}_c = \mathbf{Q}_c \, \mathbf{Q}_c^{\top}$ where $\mathbf{Q}_c$ is the U matrix of the thin SVD (or equivalently the Q matrix of the

thin QR decomposition) of the matrix of centred class-c training examples $[\mathbf{x}_{c1} - \boldsymbol{\mu}_c, \ldots, \mathbf{x}_{cN_c} - \boldsymbol{\mu}_c]$.

In practice the training data is often somewhat noisy. This can harm the classification performance owing to the inclusion of spurious 'noise' dimensions in the affine hulls. To reduce this we suppress dimensions of the SVD (and hence of $\mathbf{Q}_c$) that correspond to overly small singular values.

For nonlinear classes that lie on smooth manifolds, NAH can also be applied locally by finding the k-nearest samples to the query from each class, building local affine hulls using these nearest neighbors, and assigning the query to the class with the closest hull (Vincent & Bengio, 2001). This can reproduce complex nonlinear decision boundaries.

2.2. Nearest Convex Hull (NCH) Classification

The affine hull gives a rather loose approximation to the class region. Alternatively, we can take a maximally tight bound by approximating the class with the convex hull of its training samples. For this, we include non-negativity constraints $\alpha_i \geq 0$, $i = 1, \ldots, N_c$ in (1) and replace all of the affine hull distance computations with convex hull ones. The distance from a query $\mathbf{x}_q$ to the convex hull of class c is the norm of the displacement from $\mathbf{x}_q$ to the closest point on the hull. This reduces to solving the following quadratic programming problem

$$\begin{aligned} &\min_{\boldsymbol{\alpha}_c} \ \frac{1}{2}||\mathbf{x}_q - \mathbf{X}_c\,\boldsymbol{\alpha}_c||^2 \\ &s.t. \ \sum_{i=1}^{N_c} \alpha_{ci} = 1, \ \ \alpha_{ci} \geq 0, \ \ i = 1, \ldots, N_c, \end{aligned} \tag{4}$$

where $\mathbf{X}_c$ is a matrix whose columns are the class-c training samples. Given the optimal α^*_{ci} coefficients, the distance from $\mathbf{x}_q$ to the convex hull of the class c is $||\mathbf{x}_q - \mathbf{X}_c\,\boldsymbol{\alpha}^*_c||$. This is repeated for each class and the query is assigned to the class with the closest convex hull.

Finding the maximum margin between two classes is equivalent to finding the closest points on their convex hulls (Bennett & Bredensteiner, 2000) so convex distances can also be computed by using a classical hard-margin SVM algorithm to find the margin (convex distance) separating each class from the given query point.

NAH and NCH are "one class" methods in the sense that we do not explicitly calculate the decision boundaries during the training phase. Instead they remain implicit and the decisions are made on-line for each test sample. However both approaches can be viewed as large margin classifiers closely related to hard-margin linear SVM's. In particular, the piecewise linear/quadratic decision boundary of NCH contains the SVM boundary as one facet, and generalizes it to use distance to the convex hull rather than linear separation as the decision criterion.

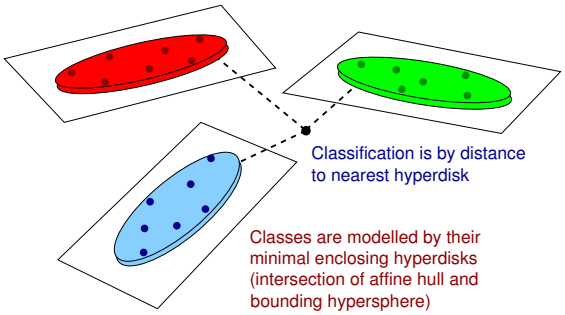

Figure 1. The principle of the proposed nearest bounding hyperdisk method. Classes are modelled by the bounding hyperdisk of their training examples and new examples are classified to the class with the closest hyperdisk.

3. Nearest Bounding Hyperdisk (NHD) Classification

In high-dimensional spaces, classes often extend well beyond the convex hulls of their training samples. For example, any individual simplex spanned by points sampled from a high-dimensional hypersphere can include only a negligible fraction of the volume of the sphere even if the vertices themselves are well spaced and close to the surface of the sphere. Conversely, affine hulls often give a rather loose approximation to the class as they do not constrain the positions of the training points within the affine subspace. This is problematic if the classes have similar or intersecting affine hulls but very different distributions of samples within their hulls. In such cases the classification performance will be poor if the affine projections of the queries onto the affine hulls are too far from training samples (*e.g.* as indicated by large values of the α_i coefficients for the constructed affine projections). The "soft margin" approach to handling this is to allow negative weights in (4) but to penalize over-large values by including upper and lower bounds in the quadratic program. However this deteriorates the run-time efficiency of NAH because the affine hull parameters of classes can no longer be computed in advance.

Instead, we can keep both a simpler geometric interpretation and good run-time efficiency by approximating the class samples with their bounding hyperdisk, *i.e.* the intersection of their affine hull and their minimal bounding hypersphere.

3.1. Global Nearest Hyperdisk Method

We will only describe the basic global Nearest Hyperdisk (NHD) classifier, but local application is also possible in the same way as for NAH and NCH. NHD approximates each class with the smallest bounding hyperdisk of its training samples – the set formed by inter-

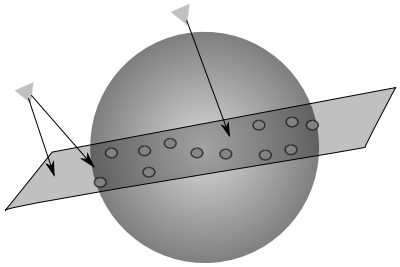

Figure 2. Computing distances from queries to a hyperdisk. The affine projection of the query on the left lies outside the hypersphere so it needs to be projected along the affine hull onto the hypersphere before the query-hyperdisk distance can be calculated. The affine projection of the query at the top already lies within the hypersphere so no adjustment is necessary.

secting their affine hull and their smallest bounding hypersphere. Such hyperdisks can be computed economically and they support rapid nearest point computations. There are already a number of methods based on affine hulls or bounding hyperspheres – for example hyperspheres have been used for outlier detection (Tax & Duin, 2004, Shawe-Taylor & Cristianini, 2004) and binary classification (Wang et al., 2005) – but we are not aware of any previous machine learning method based on hyperdisks. The bounding hypersphere of class c is characterized by its center $\mathbf{s}_c$ and radius r_c. These can be found by solving the following quadratic program

$$\begin{aligned} &\min_{\gamma, r_c, \xi} \; r_c^2 + \gamma \sum_{i=1}^{N_c} \xi_i \\ &\quad s.t. \;\; ||\mathbf{x}_{ci} - \mathbf{s}_c||^2 \leq r_c^2 + \xi_i, \;\; i = 1, \ldots, N_c, \end{aligned} \tag{5}$$

or its dual

$$\begin{aligned} &\min_{\alpha} \;\; \sum_{i,j} \alpha_i \alpha_j \langle \mathbf{x}_{ci}, \mathbf{x}_{cj} \rangle - \sum_i \alpha_i \langle \mathbf{x}_{ci}, \mathbf{x}_{ci} \rangle \\ &\quad s.t. \;\; \sum_{i=1}^{N_c} \alpha_i = 1, \;\; 0 \leq \alpha_i \leq \gamma, \;\; i = 1, \ldots, N_c. \end{aligned} \tag{6}$$

Here α_i are Lagrange multipliers and $\gamma \in [0, 1]$ is a ceiling parameter that can be set to a finite value to eliminate overdistant points as outliers. Given the solution, the center of the hypersphere is $\mathbf{s}_c = \sum_{i=1}^{N_c} \alpha_i \mathbf{x}_{ci}$ and the radius is $r_c = ||\mathbf{x}_{ci} - \mathbf{s}_c||$ for any $\mathbf{x}_{ci}$ with $0 < \alpha_i < \gamma$.

To compute the distance from a query to the hyperdisk of a class, we find the affine projection of the query onto the affine hull by $\mathbf{x}_q^{\text{aff}} = \mathbf{P}_c(\mathbf{x}_q - \boldsymbol{\mu}_c) + \boldsymbol{\mu}_c = \mathbf{P}_c \mathbf{x}_q + \boldsymbol{\mu}_c^{\perp}$. If the projection lies outside the bounding hypersphere we move it along the line joining it to the center of the sphere until it touches the sphere. The distance from the query to the disk is the distance from it to the (possibly moved) projection – see fig. 2. Formally, the distance is

$$d(\mathbf{x}_q, H_c^{\text{disk}}) = \sqrt{\max(\|\mathbf{x}_q^{\text{aff}} - \mathbf{s}_c\| - r_c, 0)^2 + \|\mathbf{x}_q - \mathbf{x}_q^{\text{aff}}\|^2}.$$

3.2. Kernelization of the Hyperdisk Method

We now show that the hyperdisk method can be kernelized, allowing it to be used in implicit high dimensional feature spaces induced by Mercer kernels. This brings all of the usual advantages and disadvantages of kernelization, notably scope for a richer choice of distance functions and highly nonlinear decision boundaries that can aid data separability in return for the need to work with an implicit model defined by a large set of training samples.

The kernel trick can be used to map the data into an implicit feature space as in Kernel PCA (Schölkopf et al., 1998). Let $\boldsymbol{\phi}(\cdot)$ be the implicit feature space embedding and $k(\mathbf{x}, \mathbf{y}) = \boldsymbol{\phi}^\top(\mathbf{x})\boldsymbol{\phi}(\mathbf{y})$ be the corresponding kernel function. Suppose that we want to project a sample $\mathbf{x}$ onto the affine hull of a given set of samples $\{\mathbf{x}_i | i = 1, \ldots, m\}$. Let $\boldsymbol{\Phi} = [\boldsymbol{\phi}(\mathbf{x}_1), \ldots, \boldsymbol{\phi}(\mathbf{x}_m)]$ be their feature space embedding matrix, $\mathbf{K} = \boldsymbol{\Phi}^\top \boldsymbol{\Phi} = [k(\mathbf{x}_i, \mathbf{x}_j)]$ be their $m \times m$ kernel matrix and $\mathbf{k}_{\mathbf{x}} = \boldsymbol{\Phi}^\top \boldsymbol{\phi}(\mathbf{x}) = [k(\mathbf{x}_i, \mathbf{x})]$ be the $m \times 1$ kernel vector of $\mathbf{x}$ against the samples. The feature space mean of the samples is $\boldsymbol{\mu} = \frac{1}{m} \boldsymbol{\Phi}\, \mathbf{1}_m$ where $\mathbf{1}_m$ is an m-vector of 1's. The explicit approach detailed below (3) is based on the thin SVD $\mathbf{U}\mathbf{D}\mathbf{V}^\top$ of the matrix of centered sample features $[\boldsymbol{\phi}(\mathbf{x}_1) - \boldsymbol{\mu}, \ldots, \boldsymbol{\phi}(\mathbf{x}_m) - \boldsymbol{\mu}] = \boldsymbol{\Phi}\,\boldsymbol{\Pi}$, where $\boldsymbol{\Pi} = \mathbf{I} - \frac{1}{m} \mathbf{1}_m \mathbf{1}_m^\top$ is an orthogonal projection in sample space that implements subtraction of the mean on $\boldsymbol{\Phi}$. Given this, the projection of $\mathbf{x}$ onto the affine hull of the mapped samples is then $\mathbf{U}\,\mathbf{U}^\top(\boldsymbol{\phi}(\mathbf{x}) - \boldsymbol{\mu}) + \boldsymbol{\mu}$, and the squared residual of this projection is $\|\boldsymbol{\phi}(\mathbf{x}) - \boldsymbol{\mu}\|^2 - \|\mathbf{U}^\top(\boldsymbol{\phi}(\mathbf{x}) - \boldsymbol{\mu})\|^2$. Also, we are free to use any origin and linear basis that we choose for computations within the affine hull so long as we do so consistently. In particular, if we choose the orthogonal basis given by $\mathbf{U}$ centred at $\boldsymbol{\mu}^\perp = (\mathbf{I} - \mathbf{U}\mathbf{U}^\top)\boldsymbol{\mu}$, the projection of $\mathbf{x}$ onto the affine hull is represented simply by $\mathbf{U}^\top \boldsymbol{\phi}(\mathbf{x})$.

Noting that the $\mathbf{D}$ matrices of thin SVDs (*i.e.* taking only the significantly non-zero singular values) are invertible, we have $\mathbf{U} = \boldsymbol{\Phi}\,\boldsymbol{\Pi}\,\mathbf{V}\,\mathbf{D}^{-1} = \boldsymbol{\Phi}\,\mathbf{A}^\top$ where $\mathbf{A} = \mathbf{D}^{-1}\mathbf{V}^\top\boldsymbol{\Pi}$. In the kernelized case we can not evaluate the SVD of $\boldsymbol{\Phi}\,\boldsymbol{\Pi}$ explicitly because this would require numerical computations in feature space, but we can work implicitly in sample-space in terms of the eigendecomposition $\mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^\top$ of the centred kernel matrix $\tilde{\mathbf{K}} = (\boldsymbol{\Phi}\,\boldsymbol{\Pi})^\top(\boldsymbol{\Phi}\,\boldsymbol{\Pi}) = \boldsymbol{\Pi}\,\mathbf{K}\,\boldsymbol{\Pi}$. Here, $\mathbf{V}$ is the same matrix as in the SVD of $\boldsymbol{\Phi}\,\boldsymbol{\Pi}$ and $\boldsymbol{\Lambda} = \mathbf{D}^2$ so that $\mathbf{A} = \boldsymbol{\Lambda}^{-1/2}\,\mathbf{V}^\top\,\boldsymbol{\Pi}$.

Putting all of these pieces together and noting that $\|\boldsymbol{\phi}(\mathbf{x})\|^2 = k(\mathbf{x}, \mathbf{x})$, we find that the squared residual error of the projection of $\mathbf{x}$ onto the affine hull of the examples

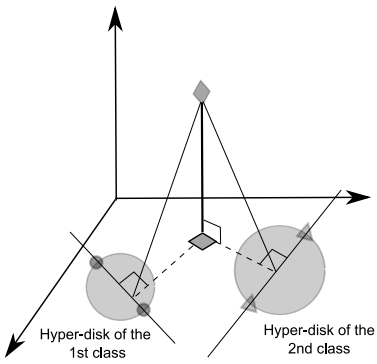

Figure 3. The kernelized NAH/NCH/NHD classifiers are based on distances between query samples and affine, *etc*., hulls of classes within the subspace spanned by the complete training data. These distances produce the same class assignments as the original classifiers.

is

$$k(\mathbf{x},\mathbf{x}) - \mathbf{k}_\mathbf{x}^\top \mathbf{A}^\top \mathbf{A} \mathbf{k}_\mathbf{x} - (2\,\mathbf{k}_\mathbf{x} - \mathbf{K}\tfrac{\mathbf{1}_m}{m})^\top (\mathbf{I} - \mathbf{A}^\top \mathbf{A} \mathbf{K}) \tfrac{\mathbf{1}_m}{m} \quad (7)$$

and the sample-space representative of the feature-space projection of $\mathbf{x}$ into the affine hull of the samples is simply $\mathbf{A}\,\mathbf{k}_\mathbf{x}$. We can use the representation vectors $\mathbf{A}\,\mathbf{k}_\mathbf{x}$ for any affine computation within the feature space affine hull, including calculations of hyperspheres and convex hulls, projections of new samples onto these, and within-hull distance computations. To calculate the overall squared distance from the example to the desired convex set within the hull, the squared residual error of the projection onto the hull (7) needs to be added to the squared within-hull distance.

In retrospect the obvious way to perform the above computations would be to use a separate feature subspace ($\mathbf{\Phi}$, $\mathbf{K}$, $\mathbf{k}_\mathbf{x}$, $\mathbf{A}$, *etc*.) for each class, but in the experiments below we actually worked in a global feature subspace based on the combined training samples of all classes. This subspace contains the affine hulls of all of the classes so the projections of test samples onto classes can be done in two stages, first projecting the sample onto the global affine hull, then projecting the result onto the class hull within the global one. The first projection is class-independent so it simply adds a sample-dependent constant residual to all of the sample-class distances. For decisions based on relative sample-class distances, these constants can be ignored. As a result, it suffices to perform all computations with the global $\mathbf{A}\,\mathbf{k}_\mathbf{x}$ vectors as though they were the original affine input points. In particular, the kernelized versions of NAH, NHD and NCH simply apply the corresponding linear method to the $\mathbf{A}\,\mathbf{k}_\mathbf{x}$ vectors of the global feature subspace. This process is illustrated in fig. 3. It only provides relative distances, so $k(\mathbf{x},\mathbf{x})$ is never needed.

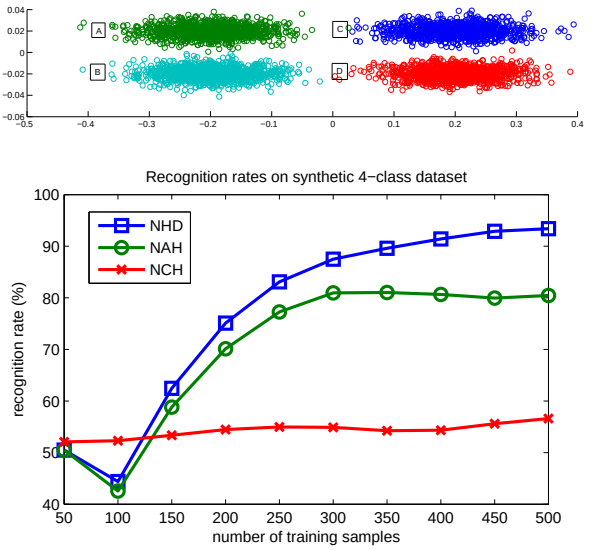

Figure 4. Top: the first two dimensions of the four disk dataset. Bottom: overall test-set recognition rates for NHD, NAH and NCH on this dataset for varying numbers of training points.

4. Experiments

We compared the proposed hyperdisk method (NHD) to Nearest Neighbour (NN), Nearest Affine Hull (NAH), Nearest Convex Hull (NCH) and Nearest Sphere Center[1] (NSC) classifiers in two regimes: high-dimensional problems where the dimensionality of the input space is much larger than the size of the training sets and the native (unkernelized) classifier is used; and low-dimensional problems where the training sets are larger than the dimensionality of the input space and a kernelized classifier is needed. We tested the methods on three tasks from multi-class visual recognition in the high-dimensional regime, and on five tasks from the UCI collection in the low-dimensional one. In each case, we optimized the algorithm parameters using global coarse-to-fine search, with random partitions of the training data into training and validation sets.

4.1. Experiments on Synthetic Data

Before starting, we illustrate some properties of the methods on a simple synthetic data set with four classes. This was produced by creating four unit-radius spheres (one for each class) in 300 dimensions with centres $(\pm 0.2, \pm 0.2, 0, \ldots, 0)$, sampling test and training points uniformly within each sphere, then compressing 200 of the dimensions including the second one by a factor of 10 – see

[1]NSC computes bounding hyperspheres for each class and assigns the query to the class whose sphere *center* is nearest.

fig. 4 (top). This produces a high dimensional data set with 100D-disk like classes and many irrelevant variables. The classes are fairly well separable but the data has somewhat suboptimal scaling. For the NAH and NHD methods we estimated the affine dimension using an eigenvalue gap detector that reliably gave the correct result (100) for all runs with more than about 120 training points.

Fig. 4 (bottom) shows the resulting recognition rates for NAH, NHD and NCH with varying numbers of training samples. The hyperdisk method predominates, particularly for larger numbers of training samples. The example is somewhat idealized – the data is quite clean and the classes have a form that is well adapted to the hyperdisk model – but it illustrates several advantages of the hyperdisk method. Firstly, NCH performs poorly. It separates classes $\{A, B\}$ from $\{C, D\}$ almost perfectly, but it is not much better than random (around 55-60% correct) at separating A from B and C from D. This happens because the interclass spacing is small and the convex hulls of the training samples fill so little of the volume of the 100-D class disks that test samples are almost as likely to lie close to the hull of the wrong class as to that of the right one – *i.e.* even though the hulls "fill in the gaps" between the training samples, they are still very poor estimates of the actual class boundaries. NCH is also much slower than NAH and NHD at run time because it needs to solve a quadratic program for each test sample to find the nearest point on the hull. Both problems are endemic to the convex hull formulation.

Secondly, NAH does surprisingly well, especially when one considers that it has an asymptotic error rate of 50%: for exact estimates of the 100-D affine hulls of the classes, A and C (and similarly, B and D) are indistinguishable because they have identical affine hulls. Empirically NAH does much better than this because the estimates of the affine hulls are noisy: being estimated from examples of class A, the hull for class A always passes close to the centre of class A, but its random tilt typically makes it pass somewhat further from the centre of class C, and vice versa. Hence, empirical NAH estimates indirectly incorporate some information about the relative positions of the classes within their affine hyperplanes. This may explain why the performances of NAH and NHD are often similar in the below experiments on real data. However, as the above results suggest, it is often advisable to incorporate the position information explicitly by using NHD.

4.2. Experiments on Image Datasets

ORL Face Dataset.[2] The Olivetti-Oracle Research Lab face dataset contains 10 upright 92×112 frontal face images per person of $C = 40$ individuals, taken at different times with slightly different lighting conditions, image positions, facial expressions and facial details. For this experiment we used the raw image pixels as input features without applying any visual preprocessing. For training we randomly selected $N = 3, 5, 7$ images of each individual, keeping the remaining $10 - N$ for testing. The results are summarized in table 4.1 (top left). The NHD and NAH classifiers were equal best among the methods tested, followed by NCH, then NN, with NSC coming last.

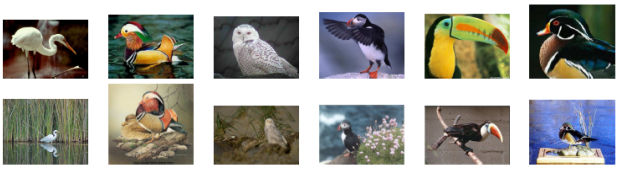

Figure 5. Some examples from the Birds dataset.

Coil100 Objects Dataset.[3] The Coil100 dataset includes 72 views each of 100 different objects taken on a turntable at orientations spaced at 5 degree intervals. We chose 40 objects randomly for the experiments. We used the raw grayscale pixels of the 128×128 images as input features, without applying any further visual preprocessing. For training we randomly selected $N = 18, 36, 54$ images of each object, keeping the remaining $72 - N$ for testing. The results are given in table 4.1 (top right). NHD and NAH again give very similar results with NHD having a slight edge. NHD achieves the best accuracy for $N = 18, 54$ while for $N = 36$ NCH is preferred to NHD and NAH. NSC again produced the worst results.

Birds Dataset. This contains six categories, each with 100 images (Lazebnik et al., 2005). It is a challenging visual object recognition task with the birds appearing against highly cluttered backgrounds and the images having large intra-class, scale, and viewpoint variability. Some example images are shown in fig. 5. We use a "bag of features" representation for the images as they are too diverse to allow simple geometric alignment of their objects. In this method, patches are sampled from the image at many different positions and scales, either densely, randomly or based on the output of some kind of salient region detector. Here we used a dense grid of patches. Each patch was described using the robust visual descriptor SIFT (Lowe, 2004) and vector quantized using nearest neighbor assignment against a 2000 word visual dictionary learned from the complete set of training patches. For training we randomly selected $N = 25, 50, 75$ images of each class, keeping the remaining $100 - N$ for testing.

The results are given in table 4.1 (bottom left). For $N = 50, 75$, NCH achieves the best recognition rates whereas

[2] *www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html*

[3] *www1.cs.columbia.edu/CAVE/software/softlib/coil-100.php*

Table 1. Classification Rates (%) and their standard deviations on respectively the ORL Face data set (top left), the COIL data set (top right), and the Birds data set (bottom left). The recognition rates are averages over 15 random training/test splits. (Bottom right) Classification Rates (%) on selected UCI data sets.

ORL	$N = 3$	$N = 5$	$N = 7$
NHD	**88.50** $\pm$ 2.2	95.30 $\pm$ 1.5	**97.00** $\pm$ 1.8
NAH	**88.50** $\pm$ 2.2	**95.30** $\pm$ 1.5	**97.00** $\pm$ 1.8
NCH	88.47 $\pm$ 2.2	94.97 $\pm$ 1.5	96.72 $\pm$ 1.6
NSC	86.50 $\pm$ 2.8	91.77 $\pm$ 1.5	93.61 $\pm$ 2.0
NN	87.74 $\pm$ 2.3	94.30 $\pm$ 1.5	96.11 $\pm$ 1.7

COIL	$N = 18$	$N = 36$	$N = 54$
NHD	**97.38** $\pm$ 0.3	99.35 $\pm$ 0.4	**99.93** $\pm$ 0.1
NAH	97.33 $\pm$ 0.3	99.32 $\pm$ 0.5	**99.93** $\pm$ 0.1
NCH	97.35 $\pm$ 0.3	**99.41** $\pm$ 0.3	99.41 $\pm$ 0.4
NSC	82.81 $\pm$ 3.3	82.94 $\pm$ 1.2	83.98 $\pm$ 1.2
NN	96.56 $\pm$ 0.5	98.84 $\pm$ 0.4	99.72 $\pm$ 0.3

Birds	$N = 25$	$N = 50$	$N = 75$
NHD	**86.62** $\pm$ 1.6	90.51 $\pm$ 1.2	92.14 $\pm$ 1.8
NAH	**86.62** $\pm$ 1.6	90.51 $\pm$ 1.2	92.14 $\pm$ 1.8
NCH	86.60 $\pm$ 1.6	**90.91** $\pm$ 1.4	**92.67** $\pm$ 1.7
NSC	84.43 $\pm$ 2.3	87.82 $\pm$ 1.8	87.85 $\pm$ 1.8
NN	53.38 $\pm$ 4.1	60.51 $\pm$ 8.3	64.05 $\pm$ 2.6

UCI	Iris	IS	MF	Wine	WDBC
NHD	**96.7**	96.0	**98.4**	96.7	96.3
NAH	**96.7**	95.7	**98.4**	96.7	95.3
NCH	96.0	95.7	98.2	**97.8**	**97.7**
NSC	96.0	93.5	97.9	96.1	95.1
NN	96.0	**96.3**	97.6	94.5	96.0

Table 2. The key parameters of the low-dimensional datasets selected from the UCI Repository.

Data set	# Classes	# Examples	Dim.
Iris	3	150	4
IS	7	2310	19
MF	10	2000	256
Wine	3	178	13
WDBC	2	569	30

NHD and NAH are equal best for $N = 25$. All of the convex approximation based methods significantly outperform Nearest Neighbours.

4.3. Experiments with UCI Datasets

In the second group of experiments we tested the kernelized versions of the methods on five lower-dimensional datasets from the UCI repository: Iris, Image Segmentation (IS), Multiple Features (MF) - pixel averages, Wine, and Wisconsin Diagnostic Breast Cancer (WDBC). The key parameters of the datasets are summarized in table 2 and the results are presented in table 4.1 (bottom right).

In each case the dimensionality of the input space is smaller than the number of samples in each class. It follows that the native NAH classifier cannot be used directly because the affine hull of each class typically spans the entire input space. However kernelized versions of all of the classifiers can still be applied. The NCH and NSC formulations directly support kernelization while for NAH and NHD we used the Kernel PCA projection method described in section 3.2. We used Gaussian kernels and 5-fold cross-validation for all experiments.

NHD and NAH were the equal best classifiers for the Iris and MF databases while NCH came first for Wine and WDBC, and NN for IS. In all of the cases tested the proposed NHD classifier either matches or outperforms the NAH classifier. The convex approximation based approaches typically outperformed NN, but the difference was not as high as in the Birds database.

4.4. Discussion

The NHD and NAH classifiers often had almost identical performance but when there were differences NHD usually dominated. This suggests that NHD's tighter bounds on the classes are sometimes useful, but that they are often inactive, either because the affine hull projections of most queries already lie within the class hyperspheres or because the additional projections onto the hyperspheres do not add useful new discriminant information.

NHD and NAH often outperformed NCH in both the high-dimensional native experiments and the low-dimensional kernelized ones. As mentioned above, in high dimensions the convex hulls of the training samples typically significantly underestimate the extents of the classes unless the number of samples is exponential in the dimension of the class. Thus, despite the simplicity of their underlying approximations, the affine hulls and hyperdisks may often turn out to be better guides to the region spanned by the class than the convex hulls.

In the low-dimensional problems, NN (and related kernel methods) often perform relatively well, perhaps because hole artifacts are not so prevalent in low dimensions. Similarly, as the dimension decreases, NCH progressively improves relative to NAH because it provides tighter bounds on the class regions. NHD seems to offer a useful compro-

mise here.

In terms of run-time efficiency NAH and NHD are to be preferred as the affine hull and bounding hyperdisk parameters can be computed off-line. When there are large numbers of training samples, NCH often becomes prohibitively slow at run-time because it needs to solve a quadratic program for each sample-hull distance computation.

5. Summary and Conclusions

We have introduced a new method for high-dimensional classification based on approximating each class with the minimal bounding hyperdisk of its training samples – the intersection of their affine hull and their bounding hypersphere – and assigning test samples to the class with the nearest hyperdisk. For robustness, the algorithm uses PCA to suppress over-small "noise" dimensions in the affine hull and it removes outliers from the hypersphere calculation by bounding their Lagrange multipliers. In practice the hyperdisk approximation offers a useful middle ground between the loose approximation provided by the affine hull of the samples and the over-tight one given by their convex hull. It can also be kernelized to allow it to be used in lower-dimensional problems that require complex decision boundaries.

Future work. We are currently working on large-margin classifiers that calculate explicit decision boundaries during the training phase by maximizing the separation between the affine hull or hyperdisk approximations of the classes. These may be useful alternatives to SVMs, which maximize the separation between the convex hulls of the classes. Given that the affine hull and hyperdisk methods were often more accurate than the convex hull ones in the experiments, the new methods may yield more efficient classifiers than SVM in terms of both accuracy and computational complexity.

References

Bennett, K. P., & Bredensteiner, E. J. (2000). Duality and geometry in svm classifiers. *ICML*.

Cevikalp, H., Larlus, D., Douze, M., & Jurie, F. (2007). Local subspace classifiers: Linear and nonlinear approaches. *IEEE Workshop on Machine Learning for Signal Processing*. Thessaloniki, Greece.

Cevikalp, H., Neamtu, M., Wilkes, M., & Barkana, A. (2005). Discriminative common vectors for face recognition. *IEEE Transactions on PAMI*, *27*, 4–13.

Cevikalp, H., Triggs, B., Jurie, F., & Polikar, R. (2008). Margin-based discriminant dimensionality reduction for visual recognition. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.

Gulmezoglu, M. B., Dzhafarov, V., & Barkana, A. (2001). The common vector approach and its relation to principal component analysis. *IEEE Trans. Speech Audio Proc.*, *9*, 655–662.

Hinton, G. E., Dayan, P., & Revow, M. (1997). Modeling the manifolds of images of handwritten digits. *IEEE Transactions on Neural Networks*, *18*, 65–74.

Laaksonen, J. (1997). *Subspace classifiers in recognition of handwritten digits*. Doctoral dissertation, Helsinki University of Technology.

Lazebnik, S., Schmid, C., & Ponce, J. (2005). A maximum entropy framework for part-based texture and object recognition. *International Conference on Computer Vision*.

Lowe, D. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, *60*, 91–110.

Nalbantov, G. I., Groenen, P. J. F., & Bioch, J. C. (2007). *Nearest convex hull classification* (Technical Report). Econometric Institute and Erasmus Research Institute of Management.

Roweis, S. T., & Saul, L. K. (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science*, *290*, 2323–2326.

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

Shawe-Taylor, J., & Cristianini, N. (2004). *Kernel methods for pattern analysis*. Cambridge University Press.

Tax, D. M. J., & Duin, R. P. W. (2004). Support vector data description. *Machine Learning*, *54*, 45–66.

Verbeek, J. (2006). Learning non-linear image manifolds by global alignment of local linear models. *IEEE Transactions on PAMI*, *28*, 1236–1250.

Vincent, P., & Bengio, Y. (2001). K-local hyperplane and convex distance nearest neighbor algorithms. *NIPS*.

Wang, J., Neskovic, P., & Cooper, L. N. (2005). Pattern classification via single spheres. *Discovery Science* (pp. 241–252).