Mixture Distribution Modeling on the Tangent Space of Hyper-Spherical Reproducing Kernel Hilbert Spaces

John L. Karlen\textsuperscript{a}, Shraddha Singh\textsuperscript{b} and Georgios C. Anagnostopoulos\textsuperscript{c}

\textsuperscript{a} University of Massachusetts Amherst, Amherst, Massachusetts; jkarlen120@gmail.com
\textsuperscript{b} The University of Texas at Austin, Austin, Texas; shraddhas@utexas.edu
\textsuperscript{c} Electrical & Computer Engineering, Florida Institute of Technology, Melbourne, Florida; georgio@fit.edu

ABSTRACT

Non-linear transformations have been shown to often yield data distributions that are more useful for a given ML task or for visualization than the ones of the original data. In this work we present such a non-linear mapping, which is a novel, parametrized variant of Kernel Principal Component Analysis (KPCA) for some, possibly, infinite-dimensional Reproducing Kernel Hilbert Space (RKHS). We use this mapping as part of a model, which aims to transform data distributions into Gaussian finite mixtures. Subsequently, we use this formulation in conjunction with Quadratic Discriminant Analysis (QDA) to tackle recognition tasks, as well as in the context of Gaussian mixture modeling for clustering. Our formulations encompass $L_1$ regularization to promote sparse solutions and can be solved via a Block Coordinate Descent (BCD) that alternates between Newton steps for the mixture parameters and projected sub-gradient steps for the remaining parameters. The benefits of our approach are demonstrated by preliminary experimental results for clustering and classification tasks entailing benchmark data sets.

Keywords: block coordinate descent, classification & clustering, kernel PCA, logarithmic map, pivot point, projected sub-gradient, RKHS

1. INTRODUCTION

Use of the “kernel trick” has become common practice as a means of applying linear methods to non-linearly transformed data, after which Machine Learning (ML) tasks may become easier for the resultant distribution. The motivation for this work is a new kernel, for which the geometric interpretation of the Reproducing Kernel Hilbert Space (RKHS) is an infinite dimensional hyperplane tangent to a hyper-sphere. The implicit mapping of this kernel is, in part, a Logarithmic map from the hyper-sphere to the tangent plane, for which there exists, necessarily, a tangent or “pivot” point. The effect of different pivot points on the mapping is schematically illustrated in 1. How would one find the optimal pivot point for which the mapping and subsequent Principal Component Analysis (PCA) produces the best distribution for clustering, or other ML tasks? This is the salient problem in this research. We create an algorithm for pivot optimization to use for clustering and classification tasks on various real-world data sets. We chose a pivot arbitrarily for an initial mapping and optimize it with a block coordinate descend method and an incorporated projected sub-gradient method. At each pivot point, an approximate gradient is calculated for the employed ML task. Then the pivot is moved in the direction chosen, restrained to remain on the hypersphere. The paper is organized as follows: In Section 2 describes the new approach, with discussion of our algorithm, as well as some background on RKHS’s and Kernel Principal Component Analysis (KPCA). In section 3 the applications and experiments that are possible with the new approach are described, highlighting some topical data sets and showcasing the generality of the approach. Section 4 concludes the paper, giving a sense of the possible impact and directions in which the method could be expanded. Some potential future work will be discussed.

2. NEW APPROACH

This section introduces our approach of applying mixture distribution models on the tangent space of hyperspheres. After stating the problem, a description of the kernel used, and the method of modeling will follow
2.1 Problem Statement

Let \( \mathcal{X} = \{x_1, \ldots, x_N\} \) be a set of \( N \) separated training samples, each with \( P \) attributes. The inputs can be non-numeric, as long as they exist in what referred to as the input space, \( \mathcal{X} \). A kernel function is the inner product between any two input elements in \( \mathcal{X} \), which have been mapped to a corresponding RKHS - effectively defining a new measure of similarity. This RKHS is denoted as \( \mathcal{H} \) and the mapping to it is defined as \( \phi(x) \). The implicit function \( \phi() \) maps from the input space to the associated \( \mathcal{H} \), which is called the feature space. For the purpose of this paper it is noted that the resultant transformation can be very non-linear and can have dramatic effects. The kernel function can be expressed as follows:

\[
k(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{H}}
\]  

The requirements for a valid kernel function are:

- \( k(x, x') = k(x', x) \)
- \( k(x, x) \geq 0, \forall x \in \mathcal{X} \)
- if \( \{x_1, x_2, \ldots, x_N\} \in \mathcal{X} \), \( \exists K, K_{ij} = k(x_i, x_j), \forall x \in \mathcal{X} \)

These properties guarantee symmetric form, and a positive semi-definite matrix, \( K \). \( K \) is the gram matrix of the set of mapped elements \( \phi(\mathcal{X}) \). If \( \{\phi_1, \phi_2, \ldots\} \) is an orthonormal basis of the RKHS, then

\[
k(x, x') = \sum_{k=1}^{\infty} \phi_k(x)\phi_k(x')
\]  

2.2 Radial Basis Functions (RBFs) and tangent hyperplanes

An RBF kernel is a kernel with the property: \( k(x, x) = 1 \), which forces the images of the original data to lie on a hypersphere RKHS of infinite dimension. Gaussian and multiquadric kernels are examples of RBF kernels. For this work we have chosen to use, primarily, the Gaussian Kernel as an RBF:

\[
k_g(x_1, x_2) = \exp\left(-\frac{\|x_1 - x_2\|^2}{2\sigma^2}\right)
\]

with variance parameter \( \sigma^2 \in \mathbb{R}_+ \). The seminal aspect of such a kernel, in the context of this work, is that a hyper-spherical distribution, \( \phi(\mathcal{X}) \), that spans some subspace of \( \mathcal{H} \), can be Logarithm mapped on to a hyperplane.
Algorithm 1 QDA on RKHS hypersphere tangent space

Input: $X, X_t, \ell, \mu \geq 0, \alpha_{\text{init}}, \lambda_{\text{max}} > 0$

Output: $\{p_i, \mu_i, C_i\}_{i=1, \ldots, C}$

1: $\alpha \leftarrow \alpha_{\text{init}}$
2: $\lambda \leftarrow \lambda_{\text{max}}$
3: $t \leftarrow 0$

// compute class priors
4: $p_i \leftarrow \frac{N}{N}$

5: converged $\leftarrow$ false
6: while not converged do

7: $t \leftarrow t + 1$

// compute projections
8: compute $\tilde{K}_{TS}(\alpha)$ as in Eq. (6)
9: $\tilde{K}_{TS} \overset{EVD}{\rightarrow} W\Gamma W^T$
10: $A_P \leftarrow \left[ \frac{w_1}{\sqrt{\gamma_1}}, \ldots, \frac{w_d}{\sqrt{\gamma_d}} \right]$
11: $y_n \leftarrow A_P^T \left[ K_{TS}(x_n) - \frac{1}{N} K_{TS} 1_N \right]$  

// update remaining mixture parameters
12: for $i = 1$ to $C$ do
13: $\mu_i \leftarrow \frac{1}{N_i} \sum_{n: \ell_n = i} y_n$
14: $C_i \leftarrow \frac{1}{N_i} \sum_{n: \ell_n = i} (y_n - \mu_i)(y_n - \mu_i)^T$

// calculation of $\tau$ values gives an estimation of the membership of each element to each class.
15: $\tau_{in} \leftarrow \frac{p_i(\alpha)}{\sum_{i=1}^{C} p_i(\alpha)}$
16: end for

// reduce step size
17: $\lambda \leftarrow \frac{\lambda}{t}$

// move pivot in direction of gradient
18: $\alpha_{\text{temp}} \leftarrow \alpha - \lambda \frac{\partial \varepsilon(\alpha)}{\partial \alpha}$ as in Eq. (10), Eq. (11)

// $\alpha$ must be retracted to the hypersphere
19: $\alpha \leftarrow \frac{\alpha}{\sqrt{\alpha^T K_{TS} \alpha}}$

// check for convergence
20: if $\varepsilon(\alpha_{\text{temp}}) < \varepsilon(\alpha)$ then
21: $\alpha \leftarrow \alpha_{\text{temp}}$
22: end if
23: if $\|\alpha - \alpha_{\text{temp}}\|_2 < 1$ then
24: converged $\leftarrow$ true
25: end if
26: end while

tangent to the hypersphere, defined by a pivot element in the hypersphere: $\phi(x_0)$, where $x_0$ is the pre-image of the pivot. Thus, a new kernel can be constructed from the RBF, using the Logarithmic mapping as the implicit mapping, and it can be shown that this kernel is valid.\footnote{\[ k(x, x') = \langle \log_{\phi(x_0)}(x), \log_{\phi(x_0)}(x') \rangle \] (4)}
\[
k_{TS}(x, x') = \frac{\arccos k_g(x, x_0) \arccos k_g(x', x_0)}{\sqrt{1 - k_g(x, x_0)^2} \sqrt{1 - k_g(x', x_0)^2}} \tag{5}
\]

The tangent or pivot point that results is the important variant of the algorithm, which, when optimized, will produce a more favorable mapping for the ML technique employed. The implementation tasks will be identifying clusters as Gaussian Mixture Models (GMMs) and classifying with QDA. The implemented algorithm’s iterations will choose a more optimal pivot point based on an approximated gradient in the \( P \) dimension subspace of \( \mathcal{H} \).

### 2.3 Algorithm Implementation

#### 2.3.1 KPCA

The notations and relevant formulas for the KPCA process are outlined in this section. Let \( \mathbf{I} \) be an identity matrix and let \( \mathbf{1}_N \in \mathbb{R}^N \) be the vector of all ones. An \( N \times N \) kernel matrix can be centered using a projection matrix \( \mathbf{P} = \mathbf{I} - \frac{1}{N} \mathbf{1}_N \mathbf{1}_N^T \). The centered matrix is denoted as \( \tilde{\mathbf{K}} \).

\[
\tilde{\mathbf{K}} := \mathbf{P} \mathbf{K} \mathbf{P} \tag{6}
\]

PCA provides an orthonormal basis in the RKHS, onto which the elements can be projected, allowing for the modeling of the distribution according to our choice of Probability Density Function (PDF). In the infinite dimensional RKHS, PDFs are not possible, making PCA a necessity in the new approach. Given a set of \( M \) test elements \( \{x_{t1}, \ldots, x_{t2}\} = \mathcal{X}_t \), a test kernel matrix, \( \mathbf{K}_t \), of test vs training set can be defined:

\[
\mathbf{K}_t := [k(x_{ti}, x_j)]_{i=1, \ldots, M, j=1, \ldots, N} \in \mathbb{R}^{M \times N} \tag{7}
\]

#### 2.3.2 Pivot Point Optimization

Any point on the RKHS can be expressed as a linear combination of the \( n \) mapped elements. It is possible to construct any pivot as a linear combination of every element in the set.

\[
\phi(x_0) = \sum_{n=1}^{N} \alpha_n \phi(x_n) \tag{8}
\]

The tangent space kernel can now be redefined as such:

\[
k_{TS}(x, x_0) = \sum_{n=1}^{N} \alpha_n k_{TS}(x, x_n) \tag{9}
\]

It is convenient to pick an arbitrary combination and optimize it to find a local extrema of the objective function. GMM clustering with EM, and QDA classification will provide a test of concept. This construction defines a vector \( \mathbf{\alpha} = \{\alpha_1, \alpha_2, \cdots, \alpha_N\} \). Our objective function is the complete Negative Log Likelihood (NLL), denoted as \( \varepsilon \).

\[
\log(\varepsilon(\mathbf{y})) = \sum_{n=1}^{N} \sum_{i=1}^{C} \left[ \tau(n, i) \log(p(i)) + \tau(n, i) \log \left( \frac{1}{\sqrt{2\pi} \sqrt{\det(\mathbf{C}_i)}} \right) - \frac{1}{2} (y_n - \mu_i)^T \mathbf{C}_i^{-1} (y_n - \mu_i) \right] \tag{10}
\]

The gradient in terms of the NLL can be expressed as:

\[
\frac{\partial \varepsilon(\mathbf{\alpha})}{\partial \mathbf{\alpha}} = \frac{1}{\delta} \begin{bmatrix} \varepsilon(\mathbf{\alpha} + \delta \epsilon_1) - \varepsilon(\mathbf{\alpha}) \\ \vdots \\ \varepsilon(\mathbf{\alpha} + \delta \epsilon_n) - \varepsilon(\mathbf{\alpha}) \end{bmatrix} \tag{11}
\]
QDA implementation is shown in Algorithm 1. EM implementation is shown in Algorithm 2. Each iteration of the algorithm calculates an approximate gradient and takes some step length $\lambda$ in the direction of the gradient. Due to the possibility that such a step may overstep the local minimum of the NLL function, the algorithm keeps track of the lowest value so far, while reducing the $\lambda$ at each iteration.

3. EXPERIMENTS

In this section the possible applications of our technique are assessed. The choice of possible clustering or classification algorithm as well as the choice of model PDF gives a plethora of options. For the sake of simplicity, Gaussian model is used, as well as the Gaussian Kernel. To showcase applications in clustering and classification, EM and QDA are used. Data sets were used, which made use of for a k-means hyper-spherical clustering algorithm. The performances of the two algorithms are compared.

3.1 Experimental Setting

3.1.1 Iris Data-set

The iris data set contains 3 classes with 50 instances of each where each class refers to a type of an iris plant. Each data point has four attributes associated to it — sepal length, sepal width, petal length, and petal width. For clustering, we obtained 3 unique sets of 48 points each — a training set, a test set and a validation set. The classes are equal in size. After validation of algorithm parameters such as the kernel spread $\sigma$ and regularization term, the test set was used. For classification tasks two subsets of 60 and 90 points were constructed for the training and test sets respectively.

3.1.2 E. coli Data Set

The Ecoli data set contains 7 predictive attributes to classify the ecoli according to protein localization sites. The original data set contains 8 classes, but for the sake of visually illustrating the dramatic transformations that can be achieved by optimizing mapping for a Gaussian model we have only clustered on the 3 largest classes, and eliminated 2 of the 7 predictive attributes which exhibited no significant variance for the remaining data. Although the EM algorithm is running on 5 dimensional data, we only depict projection of the cluster means and data on to the two largest components of the orthonormal basis, produced by PCA. It can be noted that after several iterations, the approximated optimal pivot point gives rise to clearly Gaussian clusters, for which the EM easily clusters with good accuracy.

3.1.3 Ionosphere

The Ionosphere set contains 351 instances of radar data, which probe the ionosphere for areas with free electrons. Each datum has 34 attributes, which can be characterized into two classes — good and bad. The two classes signify areas with structure and areas without.

3.1.4 USPS handwritten Digits Data Set

The USPS handwritten digits data set is a collection of 16x16 images (gray level pixel intensities) of handwritten digits (0-9). Each image is a 256 dimensional data point and there are 7291 images in total. This data-set contains 10 classes, with each class referring to a single digit. To test our classification algorithm, we used a small subset of the USPS handwritten data-set, with only three of the ten original classes. The numbers 0, 4, and 7 were chosen for classification. The data is in split into 2 unique sets of 150 points each for training and validation. The first set, or the test set, contained the first 50 occurrences of each class. This set was further split to form a training set with 60 points, 20 instances of each class. The next 50 occurrences of each class formed the validation set. The error was measured by comparing the labels provided in the data-set to the ones obtained after classification.

3.1.5 Vertebral Column Data Set

The Vertebral column data set contains the data of 310 orthopaedic patients. Each patient is characterized by six biomechanical features. The task was to classify patient’s condition as normal or abnormal. An abnormal condition meant that a patient had either disk hernia or spondylolithesis. We obtained a train set of 40 points and a test set of the remaining (270) points from this data set to test our classification algorithm.
3.1.6 Abalone Data set

The Abalone data set contains 4177 instances of abalones with 8 attributes each. The attributes are physical measurements of the abalones. The task associated to this data set is to classify the abalones based on their age in order to be able to replace the cumbersome method of measuring age that currently exists. For our experiment, we obtained a training set of 60 abalones and a test set of 150 abalones. There were 29 possible age predictions in the original set, but we tweaked it to obtain 3 classes by replacing age with age groups. All the abalones under the age of 9 formed the first class. The second class contained abalone between the ages of 10 and 17. Abalones 18 years or older formed the third class. We classified the data set into these three classes.

3.2 Experimental Results

On comparing the percent error of our approach to other classification algorithms on the USPS Data set we inferred that our approach performed better than the three other approaches, as listed in the Table 1, in three out of the four data sets. The error was calculated by comparing predicted labels to the actual labels provided in the data sets. The USPS, Abalone, and the Vertebral Column data sets showed less errors when classified through our approach with the 1-Nearest Neighbor placing second for all the three data sets. These results illustrated the benefits of using our approach for classification tasks. The Quadratic Discriminant Analysis gave better results when classifying the Iris data set and our approach placed third on that particular set. Even so, our approach gave a 5.6% error which was small and still better than the 1-Nearest Neighbor with a 6.7% error.

<table>
<thead>
<tr>
<th>Classifier Type</th>
<th>USPS</th>
<th>Abalone</th>
<th>Vertebral Column</th>
<th>Iris</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our Approach</td>
<td>4</td>
<td>37.3</td>
<td>27.04</td>
<td>5.6</td>
</tr>
<tr>
<td>1-Nearest Neighbor</td>
<td>6</td>
<td>41.3</td>
<td>63.3</td>
<td>6.7</td>
</tr>
<tr>
<td>Naive Bayes</td>
<td>70.6</td>
<td>40.6</td>
<td>68.9</td>
<td>4</td>
</tr>
<tr>
<td>Quadratic Discriminant Analysis</td>
<td>30</td>
<td>47.3</td>
<td>65.6</td>
<td>2</td>
</tr>
<tr>
<td>Better</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>NO</td>
</tr>
</tbody>
</table>

Table 1: Percent error of four classifiers on four data sets

On comparing the accuracy of our approach to the hyper-spherical clustering on the E. coli, Iris and the Ionosphere data sets in order to assess the quality of the obtained clustering results, we observed that our approach provided better accuracies than the hyper-spherical clustering on all three data sets, as shown in the next three tables (Table 2 - Table 4) hence illustrating its benefits in accomplishing clustering tasks.

<table>
<thead>
<tr>
<th>Cluster Type</th>
<th>E. coli</th>
<th>Confidence Interval</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our approach</td>
<td>94.54</td>
<td>90.18, 97.35</td>
</tr>
<tr>
<td>Hyper-spherical clustering</td>
<td>91.26</td>
<td>86.19, 94.92</td>
</tr>
<tr>
<td>Better</td>
<td>YES</td>
<td>-</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cluster Type</th>
<th>Iris</th>
<th>Confidence Interval</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our approach</td>
<td>95.83</td>
<td>85.75, 99.49</td>
</tr>
<tr>
<td>Hyper-spherical clustering</td>
<td>89.33</td>
<td>80.06, 95.28</td>
</tr>
<tr>
<td>Better</td>
<td>YES</td>
<td>-</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cluster Type</th>
<th>Ionosphere</th>
<th>Confidence Interval</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our approach</td>
<td>78.99</td>
<td>73.70, 83.64</td>
</tr>
<tr>
<td>hyper-spherical clustering</td>
<td>69.72</td>
<td>63.63, 75.34</td>
</tr>
<tr>
<td>Better</td>
<td>YES</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 2: Accuracy of two clustering algorithms with confidence region on three data sets

4. CONCLUSIONS

A new parametrized variant of Kernel PCA was introduced that is able to influence the distributional characteristics of the non-linearly projected data. Its projections can be fine-tuned to the Machine Learning task at hand or for visualization purposes. In this work we demonstrated its usefulness in the context of Gaussian mixture
Figure 2: E. coli data set: The plots above depict our clustering algorithm in action for 3 types of E. coli protein site localizations. A basis is produced by KPCA given the current mapping. Black markers and ellipses represent the estimated means and 80% Gaussian confidence regions respectively. It is evident that the final mapping is much more conducive to a useful Gaussian Mixture Model approximation than the mapping which produced the original distribution. The results were obtained using a Gaussian kernel with spread $= 0.25$. 
modeling for classification (via Quadratic Discriminant Analysis) and clustering. In a limited-scope experimental study, the results we obtained reflect that our non-linear projection method was meritorious in classifying and clustering a collection of data sets, when compared to a few standard alternatives.

ACKNOWLEDGMENTS
The authors acknowledge support from National Science Foundation (NSF) grant No. 1263011. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF. Finally, the authors would like to thank Halley Weitzman of University of Maryland, College Park, Jean Michel Rouly of George Mason University, Rana Haber of Florida Institute of Technology and Mark Moyou of Florida Institute of Technology for providing advice, technical assistance and thoughtful feedback.
Algorithm 2: EM on RKHS hypersphere tangent space

Input: \( \mathcal{X}, \mathcal{X}_t, \mu \geq 0, \tau, \alpha_{\text{init}}, \lambda_{\text{max}} > 0 \)

Output: \( \{p_i, \mu_i, C_i\}_{i=1}^C \)

1: \( \alpha \leftarrow \alpha_{\text{init}} \)
2: \( \lambda \leftarrow \lambda_{\text{max}} \)
3: \( t \leftarrow 0 \)
   // compute class priors
4: \( p_i \leftarrow \frac{N_i}{N} \)
5: converged \( \leftarrow \) false
6: while not converged do

7: \( t \leftarrow t + 1 \)
   // compute projections
8: compute \( \tilde{K}_{TS}(\alpha) \) as in Eq. (6)
9: \( \tilde{K}_{TS} \xrightarrow{\text{EVD}} \mathbf{W} \Gamma \mathbf{W}^T \)
10: \( \mathbf{A}_P \leftarrow \left[ \frac{w_1}{\sqrt{\gamma_1}}, \ldots, \frac{w_d}{\sqrt{\gamma_d}} \right] \)
11: \( \mathbf{y}_n \leftarrow \mathbf{A}_P^T \left[ \mathbf{K}_{TS}(\mathbf{x}_n) - \frac{1}{N} \mathbf{K}_{TS} 1_N \right] \)
   // update remaining mixture parameters
12: for \( i = 1 \) to \( C \) do
   // M-step: update GMM parameters
13: \( p_i \leftarrow \frac{1}{N} \sum_{n=1}^{N} \tau_{in} \)
14: \( \mu_i \leftarrow \frac{\sum_{n=1}^{N} \tau_{in} \mathbf{x}_n}{\sum_{n=1}^{N} \tau_{in}} \)
15: \( C_i \leftarrow \frac{\sum_{n=1}^{N} \tau_{in} (\mathbf{x}_n - \mu_i) (\mathbf{x}_n - \mu_i)^T}{\sum_{n=1}^{N} \tau_{in}} \)
   // E-step: update memberships
16: \( \tau_{in} \leftarrow \frac{p_i \varepsilon(\alpha)}{\sum_{i=1}^{C} p_i \varepsilon(\alpha)} \)
17: end for
   // reduce step size
18: \( \lambda \leftarrow \frac{\lambda}{t} \)
   // move pivot in direction of gradient
19: \( \alpha_{\text{temp}} \leftarrow \alpha \leftarrow \alpha - \lambda \frac{\partial \varepsilon(\alpha)}{\partial \alpha} \) as in Eq. (10), Eq. (11)
   // \( \alpha \) must be retracted to the hypersphere
20: \( \alpha \leftarrow \frac{\alpha}{\sqrt{\alpha^T \mathbf{K}_{TS} \alpha}} \)
   // check for convergence
21: if \( \varepsilon(\alpha_{\text{temp}}) < \varepsilon(\alpha) \) then
22: \( \alpha \leftarrow \alpha_{\text{temp}} \)
23: end if
24: if \( \|\alpha - \alpha_{\text{temp}}\|_2 \ll 1 \) then
25: converged \( \leftarrow \) true
26: end if
27: end while