Nearest Archetype Hull Methods for Large-Scale Data Classification

Christian Thurau
Fraunhofer IAIS
christian.thurau@iais.fraunhofer.de

Abstract
This paper introduces an efficient geometric approach for data classification that can build class models from large amounts of high dimensional data. We determine a convex model of the data as the outcome of convex hull non-negative matrix factorization, a large-scale variant of Archetypal Analysis. The resulting convex regions or archetype hulls give an optimal (in a least squares sense) bounding of the data region and can be efficiently computed. We classify based on the minimum distance to the closest archetype hull. The proposed method offers (i) an intuitive geometric interpretation, (ii) single as well as multi-class classification, and (iii) handling of large amounts of high dimensional data. Experimental evaluation on common benchmark data sets shows promising results.

1. Introduction

Class models based on large amounts of high dimensional data are often difficult to handle and as such are seldom dealt with. While it was found that nearest neighbor methods work surprisingly well in these cases [12], it is often desirable to provide a more compact class representation that also offers a certain robustness against outliers. In high dimensional classification problems, it is often difficult to cover the class regions densely. Local classifiers, e.g. nearest neighbor or kernel methods are likely to have irregular decision boundaries [1]. One solution to this problem is to first build a convex model of the data region spanned by the training samples and then classify based on the distances to the closest model. For building these models several methods were recently proposed [1, 10]. They are usually based on computations or approximations of the affine hull or convex hull of the training data. For example, Cevikalp et al. [1] model a class with its bounding hyperdisk. The bounding hyperdisk is the intersection of the affine hull and the minimal bounding hypersphere of the training data. In [10], a classification method based on distances to the closest convex hull is proposed. While this simple approach offers an intuitive geometric interpretation, it does not scale well with the number of data points.

In this paper, we propose an alternative approach to nearest convex hull classification that is based on an optimized convex set approximation of the data convex hull. More precisely, we extend convex hull non-negative matrix factorization (CH-NMF) [11] to data classification problems. CH-NMF is a recently introduced sparse data representation that extends Archetypal Analysis (AA) [2] (convex-NMF [5]) to large-scale data analysis problems. The main idea of CH-NMF/AA is to approximate a data region by a convex set of basis vectors which usually manifest as the most extreme data samples. CH-NMF handles large data sets efficiently, as it restricts the data analysis to a candidate set of subsampled convex hull data points (note that the number of vertices on the convex hull grows much slower than the data set, e.g. \( \Omega(\sqrt{\log n}) \) for \( n \) random Gaussian points in the plane [8]). For small sets of high-dimensional data the proposed method shares the same favorable properties as other convex set approaches as it also deals with holes of only sparsely covered boundaries. The resulting data model is compact and usually more robust against outliers than other nearest convex set methods.

The remainder of this paper is organized as follows. In the following Section, we introduce the nearest archetype hull method (AHM). Section 3 presents experimental results, and Section 4 concludes this paper.

2. Nearest Archetypal Hull Method

The nearest archetypal hull method approximates a convex set containing the training examples. This main idea for building a data model, resembles other approaches, including nearest convex hull, affine hull, or nearest hyperdisks [10, 1]. The convex set approximation differs from the aforementioned methods and
is directly derived from Archetypal Analysis (AA) [2] and convex hull non-negative matrix factorization (CH-NMF) [11]. Based on the derived data model, we classify by means of nearest archetype hull distance computations. Next, we provide a more formal description of the method:

Given a data matrix \( X = [x_1, x_2, \ldots, x_n] \) whose \( n \) column vectors \( x_i \) are \( m \) dimensional data points. AA aims at minimizing the residual sum of squares

\[
\text{RSS}(p) = \min_{A,B} \| X - XBA \|^2
\]

where the unknown coefficient matrices \( A \in \mathbb{R}^{p \times n} \) and \( B \in \mathbb{R}^{n \times p} \) are constrained to column convexity \((A \geq 0, \sum_{i=1}^{p} a_{il} = 1, l = 1, \ldots, n, B \geq 0, \sum_{j=1}^{n} b_{jl} = 1, l = 1, \ldots, p)\). Informally, AA reconstructs a data set by a convex combination of convex combinations of data samples.

By substituting \( Z = XB \), we can interpret \( Z \) as a set of basis vector that are combined by the coefficients \( A \) for approximating the data. The basis vectors are also referred to as archetypes. The convexity constraint leads to basis vectors that share some interesting properties that are not found within other common matrix factorization methods (e.g., SVD or NMF). First, the coefficient matrix \( B \) is sparse and often enforces specific data points \( x \in X \) as archetypes (the basis vectors are part of the data set). Second, the basis vectors or archetypes are located on the data convex hull. Third, AA provides an (locally) optimal approximation of the data convex hull using a finite convex point set.

Unfortunately, optimizing the AA cost function (Equation 1) becomes more and more demanding with an increasing sample size \( n \) (this starts with relatively small sample sizes \( \approx 5000 \)). Therefore, we introduced an approach that allows application of AA to large data sets [11]. It builds on the observation that the resulting archetypes will always reside on the data convex hull. Thus, if we would know the points on the convex hull in advance, we could restrict the optimization process to just these preselected points. However, computation of the convex hull has a worst case complexity of \( \Omega(n^{d/2}) \) [4]. Thus, a direct estimation of convex hull data points seems ill posed for most practical data analysis tasks. Consequently, we proposed a smart sub-sampling \( X^H \) of points on the convex hull of \( X \). While it could be argued that by subsampling we sacrifice some (possibly very good) solutions to AA, it is important to note that joint estimation of \( A \) and \( B \) is a highly non convex problem with many local minima [2]. Thus, reducing the optimization problem to a subsampling of \( X \) only narrows the choice of possible solutions.

Our strategy for producing \( X^H \) exploits the fact that the original data matrix \( X \) contains finitely many data points so that its convex hull forms a polytope in \( \mathbb{R}^m \). From the main theorem of polytope theory we know that every image of a polytope \( P \) under an affine map \( \pi : x \rightarrow Mx + t \) is a polytope [13]. In particular, every vertex of an affine image of \( P \) corresponds to a vertex of \( P \). This allows us to sample the convex hull of \( X \) by means of considering convex hulls of different 2D projections of the data. We project the data onto the \((h-1)/2\) 2D subspaces spanned by all pairwise combinations of the first \( h \) eigenvectors of the covariance matrix of \( X \) (for the experiments we select \( h \) such that we account for 95% of the energy of the eigenvalue spectrum). It has recently been shown that in extremely high dimensions all data come to lie on the hull [6, 7]. Interestingly, for the examples presented in the next section, we found that the total amount of points \( n' \) in \( X^H \) obtained from several 2D projections is considerably smaller than the actual number of points \( n \) in \( X \).

The archetype hull consists of a set of \( p \) convex hull data points \( Z^m \times p \) (archetypes) of class \( c \). The archetypes result from application of AA to the sub-sampled selection of candidates \( X^H_c \). New data is classified based on its distance to the nearest archetype hull. The distance \( d \) of a data point \( x \) to the closest archetype hull \( Z \) associated with a specific class \( c \) is the norm of the displacement to the closest point on the hull. Thus, \( d \) can be computed by solving the following quadratic programming problem

\[
d(x,c) = \min_{\alpha} \| x - Z_c \alpha \|^2 = 1, \ s.t. \| \alpha \|_1 = 1, \alpha \geq 0
\] (2)

Following the nearest convex hull method, we assign the data sample \( x \) to the class with the closest archetype hull.
\[ l(k) = \arg\min_{c=1, \ldots, N} d(Z_c, x) \quad (3) \]

This is essentially the same classification scheme as used in NCH [10] and differs only in the selection of data samples within \( Z_c \) (for NCH the complete data set \( X_c \) is used). In contrast to NCH, we (i) prespecify the accuracy of the convex hull approximation by using \( p \) convex points (more archetypes lead to a better approximation), (ii) greatly reduce the computational complexity for large data sets \( (p \ll n) \), (iii) are less sensitive to outliers as the convex hull subselection obeys the data variance. A simple 2D example for archetype hull classification can be seen in Fig. 1. AHM approximates the data convex hull using 4 archetypes per class (Fig. 1a also shows the residuals of training data points outside the archetype hull). Class labels for novel data points (Fig. 1b) are assigned based on the minimal distance to the closest point on the respective archetype hull.

It is important to note that if a data sample resides within the archetype hull (similar to the definition of the convex hull), \( d \) reduces to zero. For practical problems in low dimensional spaces it can easily happen that multiple convex hulls subsume each other. In that case, it becomes impossible to assign a correct class label, also see [10] for further details on this problem. However, for practical problems in high dimensional spaces this is seldom the case as we know that most data samples lie on the convex hull, thus \( d \) will never reduce to zero.

### 3. Experiments

We evaluated the proposed archetype hull method (AHM) on a broad set of publicly available data. In addition to the ORL face recognition data which was also used in [1], we also used the popular MNIST handwritten digits database, and the Inria pedestrian detection data-set [3]. The data sets highlight the classifiers performance for different kinds of problems: (a) ORL - underrepresented high dimensional data, (b) MNIST - well represented high dimensional data and multiple classes, (c) INRIA pedestrian detection - well represented high dimensional data and two classes.

**ORL Face Dataset:** The ORL face dataset consists of 400 labeled face images of 40 individuals. Each image has a resolution of 92 × 112 gray-value pixels. Following [1], we randomly split the data into varying numbers of training and test sets. For training we used either 3, 5, or 7 images for each individual and the remaining images for testing. Then, we measured the average precision of 15 random splits per different number of training samples. We compared the results against a variety of related methods (see [1] for more details). The used methods are Nearest Hyperdisk Method (NHM), Nearest Affine Hull (NAH), Nearest Convex Hull (NCH), and Nearest Sphere Center (NSC). The results can be seen in Table 1. Overall, AHM shows better or similar performance as other geometric classifiers. It is important to note that if the number of training samples \( n \) (per class) is smaller or equal to the number of basis vectors \( N \), AHM will usually result in a simple NCH classifier. The reason is simply that an optimal solution for \( Z_c \) usually converges exactly to the supplied training samples. For approximating the ORL face dataset we used 5 basis vectors.

**MNIST handwritten digits:** The MNIST database of handwritten digits is a well known benchmark in machine learning. It consists of 60000 (gray valued) training images (approx. 6000 per class) and 10000 test images. Each image has a size of 28 × 28 pixels and shows a single digit. Various methods have been evaluated on this data-set [9]. A summary of the results using varying numbers of basis vectors \( N \) can be seen in Table 2.

### Table 1. Results for the ORL face dataset.

<table>
<thead>
<tr>
<th>Method</th>
<th>3</th>
<th>5</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>AHM</td>
<td>89.85 ± 2.3</td>
<td>95.70 ± 1.5</td>
<td>97.88 ± 1.3</td>
</tr>
<tr>
<td>NN</td>
<td>89.04 ± 2.2</td>
<td>94.43 ± 1.2</td>
<td>97.05 ± 1.2</td>
</tr>
<tr>
<td>NHM[1]</td>
<td>88.50 ± 2.2</td>
<td>95.30 ± 1.5</td>
<td>97.00 ± 1.8</td>
</tr>
<tr>
<td>NAH[1]</td>
<td>88.50 ± 2.2</td>
<td>95.30 ± 1.5</td>
<td>97.00 ± 1.8</td>
</tr>
<tr>
<td>NCH[1]</td>
<td>88.47 ± 2.2</td>
<td>94.97 ± 1.5</td>
<td>97.62 ± 1.6</td>
</tr>
<tr>
<td>NSC[1]</td>
<td>86.50 ± 2.8</td>
<td>91.77 ± 1.5</td>
<td>93.61 ± 2.0</td>
</tr>
<tr>
<td>NN[1]</td>
<td>87.74 ± 2.3</td>
<td>94.30 ± 1.5</td>
<td>96.11 ± 1.7</td>
</tr>
</tbody>
</table>

### Table 2. Results for the MNIST dataset.

<table>
<thead>
<tr>
<th>Method</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>k-NN[9]</td>
<td>95.0</td>
</tr>
<tr>
<td>SVM Gaussian kernel [9]</td>
<td>98.6</td>
</tr>
<tr>
<td>1000 RBF + linear classifier [9]</td>
<td>96.4</td>
</tr>
<tr>
<td>linear classifier [9]</td>
<td>88.0</td>
</tr>
</tbody>
</table>

(a) Precision for varying number of basis vectors

(b) Precision achieved by other methods
<table>
<thead>
<tr>
<th>Method</th>
<th>k</th>
<th>Det. rate</th>
<th>FPPW</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k$-NN</td>
<td>1</td>
<td>65.13</td>
<td>$1.7 \times 10^{-3}$</td>
</tr>
<tr>
<td>$k$-NN</td>
<td>3</td>
<td>67.22</td>
<td>$2.2 \times 10^{-3}$</td>
</tr>
<tr>
<td>ker. HOG SVM</td>
<td>-</td>
<td>99.0</td>
<td>$10^{-2}$</td>
</tr>
<tr>
<td>lin. HOG SVM</td>
<td>-</td>
<td>98.5</td>
<td>$10^{-2}$</td>
</tr>
</tbody>
</table>

(a) Precision for varying number of basis vectors

(b) Results achieved by other methods

Table 3. Results for the INRIA pedestrian detection dataset.

Inria pedestrian detection: We used the INRIA pedestrian dataset [3] for performance evaluation of high dimensional two class problems. The dataset is frequently used for benchmarking pedestrian detection approaches. It is generally considered difficult due to pose articulations, occlusion, clutter and viewpoint changes. It contains 2416 positive training samples and 1218 negative training images, and 1132 positive testing samples and 453 negative testing images. The samples are normalized to $128 \times 64$. Each positive sample shows a centered person. From the negative training images we sampled approx. 6000 windows showing various backgrounds, from the negative testing images we sampled approx. 2200 windows. We use a Histogram of oriented Gradients (HOG) representations [3]. The implementation varies slightly from [3] and omits certain computations for efficiency. For training, we tested different numbers of AHM basis vectors. We compared the results to a conventional NN classifier and results for Support Vector Machines reported on in [3]. A summary of the results can be seen in Table 3. It should be noted that more recent results on this dataset provide better detection rates given a lower false positives per window (FPPW) rate. However, a fair comparison is impossible as other features were used.

4. Conclusion

We presented a novel approach for building class models given large amount of high dimensional data. The approach is based on an optimized approximation of the data convex hull [11]. It resides in between the nearest convex hull method [10] and other geometric methods, for example the nearest hyperdisk method [1]. Compared to the loose hull approximations of affine hull or nearest hyperdisk, the resulting archetype hull offers a more accurate approximation of the data boundaries given a sufficiently large number of archetypes. Compared to the nearest convex hull methods, it offers a loose approximation of the data hull and is therefore less sensitive to outliers and is also applicable to very large amounts of data. Also, the approach is efficient, straightforward to implement, and offers an intuitive geometric interpretation. For more demanding setups large parts of the computation can be parallelized.

For large number of classes and massive datasets, we currently investigate additional tree-like structuring of found class models. As the presented approach does not optimize decision boundaries, it allows for the seamless integration of novel classes without having to retrain existing class models.

References