Orthogonal Low-rank Transformation

Qiang Qiu, José Lezama, and Guillermo Sapiro

Abstract A convenient and often used assumption for high-dimensional data is that it approximately lies in a union of low-dimensional subspaces embedded in the high-dimensional space. Such low-dimensional intrinsic structures are often violated for real-world observations, as they can be corrupted by errors or deviate from ideal models. In this chapter, we discuss a learned geometric transformation on subspaces using nuclear norm as the modeling and optimization criteria. The learned transformation restores a low-rank structure for data from the same class, and, at the same time, enforces an orthogonal structure for data from different classes. In this way, we explicitly minimize intra-class variance and maximize inter-class margin simultaneously. We discuss both linear and non-linear versions of the Orthogonal Low-rank Transformation (OLT), resulting in a general framework that can be implemented using both linear models and deep neural networks. We demonstrate how such transformation can significantly enhance the performance of existing algorithms by accommodating the data into structures that better fit their hypothesis compared to the raw data. Experimental validation results are presented for various tasks including clustering, classification, domain adaptation and hashing.

1 Introduction

High-dimensional data often have a small underlying intrinsic dimension. For example, in the area of computer vision, face images of a subject Basri and Jacobs (2003); Wright et al. (2009), handwritten images of a digit Hastie and Simard

Qiang Qiu
Duke University, USA. e-mail: qiang.qiu@duke.edu

José Lezama
Universidad de la República, Uruguay. e-mail: jlezama@fing.edu.uy

Guillermo Sapiro
Duke University, USA. e-mail: guillermo.sapiro@duke.edu

(1998), and trajectories of a moving object Tomasi and Kanade (1992) can all be well-approximated by a low-dimensional subspace of the high-dimensional ambient space. Thus, multiple class data often lie in a union of low-dimensional subspaces. The ubiquitous subspace clustering problem is to partition high-dimensional data into clusters corresponding to their underlying subspaces.

Low-dimensional intrinsic structures are often violated for real-world data. For example, under the assumption of Lambertian reflectance, Basri and Jacobs (2003) shows that face images of a subject obtained under a wide variety of lighting conditions can be accurately approximated with a 9-dimensional linear subspace. However, real-world face images are often captured under pose variations; in addition, faces are not perfectly Lambertian, and exhibit cast shadows and specularities Candès et al. (2011). Therefore, it is critical for tasks such as clustering and classification to handle corrupted underlying structures of realistic data, and as such, deviations from ideal subspaces.

When data from the same low-dimensional subspace are arranged as columns of a single matrix, the matrix should be approximately low-rank. Thus, a promising way to handle corrupted data for subspace clustering is to restore such low-rank structure. Multiple efforts have been invested in seeking transformations such that the transformed data can be decomposed as the sum of a low-rank matrix component and a sparse error one Peng et al. (2010); Shen and Wu (2012); Zhang et al. (2011). These methods build on recent theoretical and computational advances in rank minimization.

In this chapter, we discuss learning a transformation on subspaces using matrix rank, via its nuclear norm convex surrogate, as the optimization criteria. The learned transformation recovers a low-rank structure for data from the same class, and, at the same time, enforces an orthogonal structure for data from different classes. In this way, we simultaneously reduce intra-class variations, and increase inter-class separations to improve performance for tasks such as clustering and classification. We show how this transformation can be achieved as a linear operation or as a non-linear one using deep neural networks.

2 Algorithms

2.1 Linear Orthogonal Low-rank Transformation (OLT)

Let $\{\mathcal{S}_c\}_{c=1}^{C}$ be C m-dimensional subspaces of $\mathbb{R}^d$ (not all subspaces are necessarily of the same dimension, this is only assumed here to simplify notation). A data set is denoted as $\mathbf{Y} = \{\mathbf{y}_i\}_{i=1}^{N} \subseteq \mathbb{R}^d$, with each data point $\mathbf{y}_i$ in one of the C subspaces and arranged as a column of $\mathbf{Y}$. $\mathbf{Y}_c$ denotes the set of points in the c-th subspace $\mathcal{S}_c$, arranged as columns of the matrix $\mathbf{Y}_c$. As data points in $\mathbf{Y}_c$ lie in a low-dimensional subspace, the matrix $\mathbf{Y}_c$ is expected to be *low-rank*, and such low-rank structure is critical for tasks such as subspace clustering and classification. However, as discussed above, this low-rank structure is often violated for real data.

We discuss here a global linear transformation on subspaces that restores a low-rank structure for data from the same class, and, at the same time, enforces a maximally separated (orthogonal) structure for data from different classes. In this way, we reduce the variation within each class and introduce separations between the classes for more robust clustering or classification.

For pedagogical reasons, we first adopt matrix rank as the key learning criterion, which is later replaced by the nuclear norm, and compute one global linear transformation on all subspaces as

$$\arg\min_{\mathbf{T}} \sum_{c=1}^{C} rank(\mathbf{TY}_c) - rank(\mathbf{TY}), \;\; \text{s.t.} ||\mathbf{T}||_2 = 1, \tag{1}$$

where $\mathbf{T} \in \mathbb{R}^{d \times d}$ is one global linear transformation on all data points, $||\cdot||_2$ denotes the matrix induced 2-norm. Intuitively, minimizing the first *representation* term $\sum_{c=1}^{C} rank(\mathbf{TY}_c)$ encourages a consistent representation for the transformed data from the same class; and minimizing the second *discrimination* term $-rank(\mathbf{TY})$ encourages a diverse representation for transformed data from different classes. The normalization $||\mathbf{T}||_2 = 1$ prevents the trivial solution $\mathbf{T} = 0$.

We now explain that the pedagogical formulation in (1) using rank is however not optimal to simultaneously reduce the variation within the same class and introduce separations between the different classes, motivating the use of the nuclear norm not only for optimization reasons but for modeling ones as well. Let $\mathbf{A}$ and $\mathbf{B}$ be matrices of the same dimensions, and $[\mathbf{A}, \mathbf{B}]$ be the concatenation of $\mathbf{A}$ and $\mathbf{B}$, we have

$$rank([\mathbf{A}, \mathbf{B}]) \leq rank(\mathbf{A}) + rank(\mathbf{B}), \tag{2}$$

with equality if and only if $\mathbf{A}$ and $\mathbf{B}$ are disjoint, i.e., their column spaces intersect only at the origin. It is easy to show that (2) can be extended for the concatenation of multiple matrices with equality if matrices are independent. Thus, the objective function (1) reaches the minimum 0 if matrices are independent after applying the learned transformation $\mathbf{T}$. However, independence does not infer maximal separation, an important goal for robust clustering and classification. For example, two lines intersecting only at the origin are independent regardless of the angle in between, and they are maximally separated only when the angle becomes $\frac{\pi}{2}$. With this intuition in mind, we now proceed to our proposed formulation based on the nuclear norm.

Let $||\mathbf{A}||_*$ denote the nuclear norm of the matrix $\mathbf{A}$, i.e., the sum of the singular values of $\mathbf{A}$. The nuclear norm $||\mathbf{A}||_*$ is the convex envelope of $rank(\mathbf{A})$ over the unit ball of matrices Fazel (2002). As the nuclear norm can be optimized efficiently, it is often adopted as the best convex approximation of the rank function in the literature on rank optimization, e.g., Candès et al. (2011) and Recht et al. (2010).

We replace the rank function in (1) with the nuclear norm,

$$\arg\min_{\mathbf{T}} \sum_{c=1}^{C} ||\mathbf{TY}_c||_* - ||\mathbf{TY}||_*, \;\; \text{s.t.} ||\mathbf{T}||_2 = 1. \tag{3}$$

Again, the normalization condition $||\mathbf{T}||_2 = 1$ prevents the trivial solution $\mathbf{T} = 0$ (we will later develop a more convenient technique to prevent the trivial solution).

It is important to note that (3) is not simply a relaxation of (1). Not only the replacement of the rank by the nuclear norm is critical for optimization considerations in reducing the variation within same class subspaces, but as we show next, the learned transformation $\mathbf{T}$ using the objective function (3) also maximizes the separation between different class subspaces (a missing property in (1)), leading to improved clustering and classification performance.

We start by presenting some basic norm relationships for matrices and their corresponding concatenations.

Theorem 1 *Let* $\mathbf{A}$ *and* $\mathbf{B}$ *be matrices of the same row dimensions, and* $[\mathbf{A}, \mathbf{B}]$ *be the concatenation of* $\mathbf{A}$ *and* $\mathbf{B}$*, we have*

$$||[\mathbf{A}, \mathbf{B}]||_* \leq ||\mathbf{A}||_* + ||\mathbf{B}||_*. \tag{4}$$

with equality if and only if the column spaces of $\mathbf{A}$ *and* $\mathbf{B}$ *are orthogonal.*

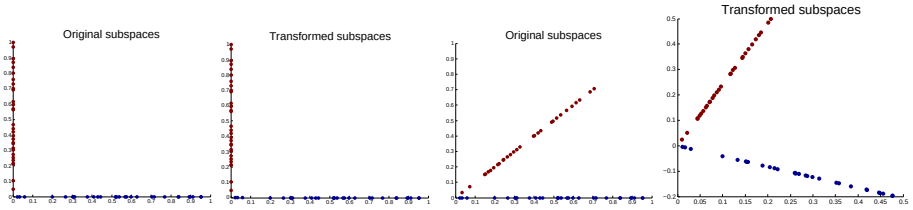
Proof See Appendix.

It is easy to see that Theorem 1 can be extended for the concatenation of multiple matrices by induction. Thus, for (3), we have,

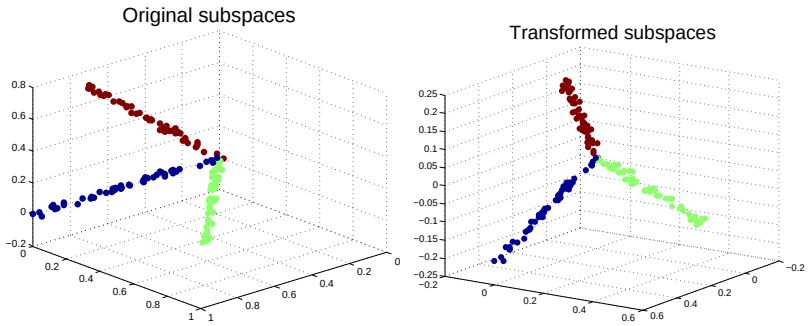
$$\sum_{c=1}^{C} ||\mathbf{T}\mathbf{Y}_c||_* - ||\mathbf{T}\mathbf{Y}||_* \geq 0. \tag{5}$$

Based on (5) and Theorem 1, the proposed objective function (3) reaches the minimum 0 if the column spaces of every pair of matrices are orthogonal after applying the learned transformation $\mathbf{T}$; or equivalently, (3) reaches the minimum 0 when the separation between every pair of subspaces is maximized after transformation, i.e., the smallest principal angle between subspaces equals $\frac{\pi}{2}$ (see Miao and Ben-Israel (1992); Elhamifar and Vidal (2013) for a definition of principal angles). Note that such improved separation is not obtained if the rank is used in the second term in (3), thereby further justifying the use of the nuclear norm instead.

We have then, both intuitively and theoretically, justified the selection of the criteria (3) for learning the transform $\mathbf{T}$. We now illustrate the properties of the learned transformation $\mathbf{T}$ using synthetic examples in Figure 1 (real examples are presented in Section 3). Here we adopt a projected subgradient method described in Section 2.3.1 to search for the transformation matrix $\mathbf{T}$ that minimizes (3). As shown in Figure 1, the learned transformation $\mathbf{T}$ via (3) maximizes the separation between every pair of subspaces towards $\frac{\pi}{2}$, and reduces the deviation of the data points to the true subspace when noise is present. Note that, comparing Figure 1c to Figure1d, the learned transformation using (3) maximizes the angle between subspaces, and the nuclear norm changes from $|[\mathbf{A}, \mathbf{B}]|_* = 1.41$ to $|[\mathbf{A}, \mathbf{B}]|_* = 1.95$ to make $|\mathbf{A}|_* + |\mathbf{B}|_* - |[\mathbf{A}, \mathbf{B}]|_* \approx 0$; However, in both cases, where subspaces are independent, $rank([\mathbf{A}, \mathbf{B}]) = 2$, and $rank(\mathbf{A}) + rank(\mathbf{B}) - rank([\mathbf{A}, \mathbf{B}]) = 0$.



(a) $\theta_{AB} = \frac{\pi}{2} = 1.57$. (b) $\theta_{AB} = 1.57$. (c) $\theta_{AB} = \frac{\pi}{4} = 0.79$, $|\mathbf{A}|_* = 1$, $|\mathbf{B}|_* = 1$, $|[\mathbf{A}, \mathbf{B}]|_* = 1.41$ (d) $\theta_{AB} = 1.57$, $|\mathbf{A}|_* = 1$, $|\mathbf{B}|_* = 1$, $|[\mathbf{A}, \mathbf{B}]|_* = 1.95$



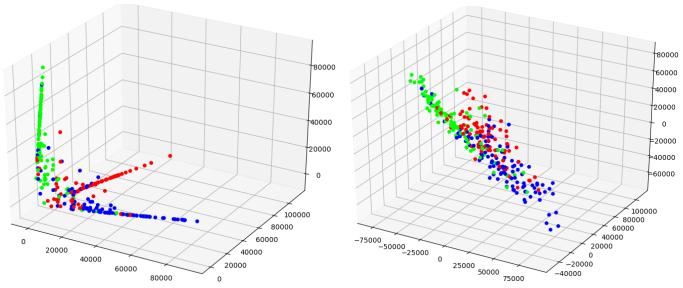
(e) $\big[\theta_{AB} = 0.79,\ \theta_{AC} = 0.79,\ \theta_{BC} = 1.05\big]$, $|\mathbf{A}|_* = 4.06,\ |\mathbf{B}|_* = 4.08,\ |\mathbf{C}|_* = 4.16$. (f) $\big[\theta_{AB} = 1.51,\ \theta_{AC} = 1.49,\ \theta_{BC} = 1.57\big]$, $|\mathbf{A}|_* = 1.93,\ |\mathbf{B}|_* = 2.37,\ |\mathbf{C}|_* = 1.20$.

Fig. 1: The learned transformation $\mathbf{T}$ using (3) with the nuclear norm as the key criterion. Three subspaces in $\mathbb{R}^3$ are denoted as $\mathbf{A}$(red), $\mathbf{B}$(blue), $\mathbf{C}$(green). We denote the angle between subspaces Miao and Ben-Israel (1992); Elhamifar and Vidal (2013) $\mathbf{A}$ and $\mathbf{B}$ as θ_{AB} (and analogous for the other pairs of subspaces). Using (3), we transform $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ in (a),(c),(e) to (b),(d),(f) respectively. We observe that the learned transformation $\mathbf{T}$ maximizes the distance between every pair of subspaces towards $\frac{\pi}{2}$.

2.2 Non-linear Orthogonal Low-rank Transformation

In this section, we extend the linear transformation in (3) into a learnable non-linear transformation, that can be modeled using neural networks, forming a non-linear OLT. In the context of classification, we consider a neural network as a deep feature extractor followed by a linear classifier, then by imposing the orthogonality and low-rank constraints at the deep feature level, the inter-class margin of the features is increased while the intra-class variability is decreased. This geometrical conditioning of the deep features using the non-linear version of the transform leads to improved generalization Lezama et al. (2018).

Consider a given training data batch $\mathbf{Y}$ of n samples, and $\mathbf{X} = \Phi(\mathbf{Y}; \theta)$ as the $n \times d$ embedding Φ of the datapoints, obtained by a neural network with parameters



(a) Non-linear OLT: 78.38% accuracy (b) Softmax: 76.69% accuracy

Fig. 2: Comparison between the deep feature embedding obtained by the non-linear OLT and a standard softmax loss using real images from 3 classes of the CIFAR-100 dataset. We show the actual 3D deep feature vectors for the validation images. The non-linear OLT produces intra-class compactness and inter-class orthogonality, and is able to achieve better classification performance than the softmax loss. A 4 layer, 100 hidden units MLP was used.

θ. Let $\mathbf{Y}_c$, $\mathbf{X}_c$ be the data and the sub-matrix of deep features belonging to class c, respectively, and $\mathbf{X}$ the matrix of deep features for the entire batch $\mathbf{Y}$. The rewriting of (3) using a learnable non-linear transformation can be posed as:

$$\mathbf{L}_o(\mathbf{X}) := \sum_{c=1}^{C} \max(\Delta, ||\mathbf{X}_c||_*) - ||\mathbf{X}||_* \tag{6}$$

$$= \sum_{c=1}^{C} \max(\Delta, ||\Phi(\mathbf{Y}_c; \theta)||_*) - ||\Phi(\mathbf{Y}; \theta)||_* \tag{7}$$

Note the normalization restriction $||T|| = 1$ is dropped. Instead, a bound $\Delta \in \mathbb{R}$ on the intra-class nuclear loss is also added, so that after a certain point the intra-class norm reduction is no longer enforced, thus avoiding the trivial solution of collapsing the features to zero. In practice $\Delta = 1$ is typically used. Note that the norm of the obtained features can be also be controlled by standard normalization losses in deep learning, or any other loss that ensures the feature vectors are not collapsed to zero.

As before, the global minimum of (6) is reached when each of the $\mathbf{X}_c$ matrices are orthogonal to each other (Theorem 1). We describe in the next section a simple descent direction for optimizing θ (6) that can be computed via backpropagation, and show that this direction vanishes only when the orthogonalization is achieved.

Figure 2 shows two simple illustrative examples of the result of applying non-linear OLT (6) as the objective function of a simple multi-layer perceptron (MLP). We compare with the result of applying the traditional softmax loss (softmax function plus cross-entropy loss). We observe a successful orthogonalization of the learned features when using non-linear OLT and a slightly better classification performance. Further experiments with the same observations can be found in Lezama et al. (2018).

2.3 Optimization

In this section we discuss methods for learning the OLT in its linear and non-linear versions.

2.3.1 Linear Transform

To optimize (3), we use a simple projected subgradient method to search for the optimal transformation matrix $\mathbf{T}$. Before describing it, we should note that the problem is non-differentiable and non-convex, and it deserves in its own right a proper study of efficient optimization techniques, which is not the focus of this chapter. Instead, a simple subgradient-based approach is used to present the framework, and it already leads to very fast convergence and excellent performance as detailed in the experimental section.

The objective function (3) is a D.C. (difference of convex functions) program, and the concave-convex procedure (CCCP) is a majorization-minimization algorithm often adopted to solve D.C. programs as a sequence of convex programs Dinh and An (1997); Sriperumbudur and Lanckriet (2012); Yuille and Rangarajan (2003). CCCP is used in many machine learning algorithms such as transductive SVMs Collobert et al. (2006), sparse PCA Sriperumbudur et al. (2007), and SVM feature selection Neumann et al. (2005).

Initialize $\mathbf{T}^{(0)}$ with the identity matrix ;
repeat

$$\begin{aligned}\mathbf{T}^{(t+1)} &= \arg\min_{\mathbf{T}} \mathcal{J}_{vex}(\mathbf{T}) + \partial\mathcal{J}_{cav}(\mathbf{T}^{(t)})\mathbf{T} \\ &= \arg\min_{\mathbf{T}} \sum_{c=1}^{C} ||\mathbf{T}\mathbf{Y}_c||_* - \text{trace}(\partial||\mathbf{T}^{(t)}\mathbf{Y}||_*\mathbf{Y}'\mathbf{T}').\end{aligned} \tag{8}$$

until *convergence or stopping criteria*;

Algorithm 1: The Concave-Convex Procedure (CCCP).

Our D.C. cost function in (3), which we will denote $\mathcal{J}(\mathbf{T})$, can be rewritten as the sum of a convex part $\mathcal{J}_{vex}(\mathbf{T})$ and a concave part $\mathcal{J}_{cav}(\mathbf{T})$, i.e.,

$$\mathcal{J}(\mathbf{T}) = \mathcal{J}_{vex}(\mathbf{T}) + \mathcal{J}_{cav}(\mathbf{T}) = [\sum_{c=1}^{C} ||\mathbf{T}\mathbf{Y}_c||_*] + [-||\mathbf{T}\mathbf{Y}||_*].$$

In each iteration of the CCCP procedure, Algorithm 1, we approximate the concave part using its subgradient $\partial\mathcal{J}_{cav}$, and minimize the resulting convex sub-problem. Note that the first term in (8) is the convex term in (3), and the added second term

Input: An $m \times n$ matrix $\mathbf{A}$, a small threshold value δ
Output: A subgradient of the nuclear norm $\partial||\mathbf{A}||_*$.
begin

1. Perform singular value decomposition:
$\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}$;

2. $s \leftarrow$ the number of singular values smaller than δ ,
3. Partition $\mathbf{U}$ and $\mathbf{V}$ as
$\mathbf{U} = [\mathbf{U}_1, \mathbf{U}_2]$, $\mathbf{V} = [\mathbf{V}_1, \mathbf{V}_2]$;
where $\mathbf{U}_1$ and $\mathbf{V}_1$ have $(n - s)$ columns.

4. Generate a random matrix $\mathbf{B}$ of the size $(m - n + s) \times s$,
$\mathbf{B} \leftarrow \frac{\mathbf{B}}{||\mathbf{B}||}$;

5. $\partial||\mathbf{A}||_* \leftarrow \mathbf{U}_1\mathbf{V}_1' + \mathbf{U}_2\mathbf{B}\mathbf{V}_2'$;

6. Return $\partial||\mathbf{A}||_*$;

end

Algorithm 2: An approach to evaluate a subgradient of matrix nuclear norm.

is a linear term on $\mathbf{T}$ using a subgradient of the concave term in (3) evaluated at the current iteration. $\partial||\cdot||_*$ is a subgradient of the nuclear norm $||\cdot||_*$, which can be evaluated using the simple approach shown in Algorithm 2 Watson (1992). Formal convergence analysis of CCCP for differentiable cases can be found in Yuille and Rangarajan (2003) and Sriperumbudur and Lanckriet (2012). Though the objective function (3) is non-differentiable, we still observe empirical convergence in our experiments Qiu and Sapiro (2015).

We provide here more details about Algorithm 1. During each CCCP iteration, we solve the convex sub-objective (8) using the subgradient method, iteratively taking a step in the negative direction of subgradient, where the subgradient is evaluated as

$$\sum_{c=1}^{C} \partial||\mathbf{T}\mathbf{Y}_c||_*\mathbf{Y}_c' - \partial||\mathbf{T}^{(\mathbf{t})}\mathbf{Y}||_*\mathbf{Y}'. \tag{9}$$

The subgradient method is guaranteed to converge to the optimal value by using a diminishing step size with an infinite travel condition, and to converge within some range of the optimal value if using a constant step size Boyd et al. (2003).

Therefore, given $\mathbf{T}^{(t+1)}$ as the minimizer found for the convex sub-problem (8) using the subgradient method, we have for (8),

$$\begin{aligned} &\sum_{c=1}^{C} ||\mathbf{T}^{(t+1)}\mathbf{Y}_c||_* - \text{trace}(\partial||\mathbf{T}^{(t)}\mathbf{Y}||_*\mathbf{Y}'\mathbf{T}^{(t+1)'}) \\ &\leq \sum_{c=1}^{C} ||\mathbf{T}^{(t)}\mathbf{Y}_c||_* - \text{trace}(\partial||\mathbf{T}^{(t)}\mathbf{Y}||_*\mathbf{Y}'\mathbf{T}^{(t)'}), \end{aligned} \tag{10}$$

and from the concavity of the second term in (3), we have

$$-||\mathbf{T}^{(t+1)}\mathbf{Y}||_* \leq -||\mathbf{T}^{(t)}\mathbf{Y}||_* - \text{trace}(\partial||\mathbf{T}^{(t)}\mathbf{Y}||_*\mathbf{Y}'(\mathbf{T}^{(t+1)} - \mathbf{T}^{(t)})'). \quad (11)$$

By summing (10) and (11), we obtain

$$\sum_{c=1}^{C} ||\mathbf{T}^{(t+1)}\mathbf{Y}_c||_* - ||\mathbf{T}^{(t+1)}\mathbf{Y}||_* \leq \sum_{c=1}^{C} ||\mathbf{T}^{(t)}\mathbf{Y}_c||_* - ||\mathbf{T}^{(t)}\mathbf{Y}||_*. \quad (12)$$

Thus, the objective (3) is non-increasing after each CCCP iteration, and is bounded from below by 0 (shown in Section 2.1) for our non-differentiable case. For efficiency considerations, while solving the convex sub-objective function (8), we perform only one iteration of the subgradient method to obtain a simplified method, and still observe empirical convergence in all experiments.

The norm constraint $||\mathbf{T}||_2 = 1$ is adopted in this formulation to prevent the trivial solution $\mathbf{T} = 0$. By initializing $\mathbf{T}^{(0)}$ with the identity matrix, we observed no trivial solution convergence in all experiments Qiu and Sapiro (2015).

As shown in Douglas et al. (2000), the norm constraint $||\mathbf{T}||_2 = 1$ can be incorporated to a gradient-based algorithm using various alternatives, e.g., Lagrange multipliers, coefficient normalization, and gradients in the tangent space. We implement the coefficient normalization method, i.e., after obtaining $\mathbf{T}^{(t+1)}$ from (8), we normalize $\mathbf{T}^{(t+1)}$ via $\frac{\mathbf{T}^{(t+1)}}{||\mathbf{T}^{(t+1)}||}$. In other words, we normalize the length of $\mathbf{T}^{(t+1)}$ without changing its direction. As discussed in Douglas et al. (2000), the problem of minimizing a cost function subject to a norm constraint forms the basis for many important tasks, and gradient-based algorithms are often used along with the norm constraint. Though it is expected that a norm constraint does not change the convergence behavior of a gradient algorithm Douglas et al. (2000); Fuhrmann and Liu (1984), to the best of our knowledge, a formal analysis of these issues is still pending. One alternative to this normalization, presented in Section 2.2 for the non-linear case, is to use a lower-bound on the intra-class matrix nuclear norm to prevent collapsing of the transformed datapoints.

2.3.2 Non-linear Transform

To optimize the non-linear version of the transform in (6) we will rely, as in the linear case, on the subdifferential of the nuclear norm of a matrix, whose computation is detailed in Algorithm 2 Watson (1992). In this case, however, we will use $\mathbf{B} = 0$ for the random matrix, obtaining the following subgradient for the nuclear norm minimization problem:

$$\partial_{||A||_*}(A) = \mathbf{U}_1\mathbf{V}_1', \quad (13)$$

where $\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}'$ is the standard SVD decomposition and the subscript 1 indicates the first columns of $\mathbf{U}$ and $\mathbf{V}$, corresponding to singular values greater than a given threshold δ (see Algorithm 2).

Intuitively, to avoid numerical issues, we are dropping the directions of the subgradient onto which the data matrix has no or very low energy already (i.e., their

corresponding singular values are already close to 0). This improves upon the formulation in the linear version of the transform, where all the directions of the space were used.

Suppose $\mathbf{X} = [\mathbf{X}_1 \mid \mathbf{X}_2 \mid \ldots \mid \mathbf{X}_C]$ is the deep feature matrix of one data batch. For $\mathbf{X}_c$, the feature submatrix of each class $c \in \{1, \ldots, C\}$, let $\mathbf{U}_{1c}$ and $\mathbf{V}_{1c}$ be its principal left and right singular vectors. Let $\mathbf{U}_1$ and $\mathbf{V}_1$ be the principal left and right singular vectors of $\mathbf{X}$, the deep feature matrix of all the classes combined. (By principal we mean those whose corresponding singular value is greater than the threshold δ.) Then, we propose the following descent direction for (6):

$$g_{L_o}(\mathbf{X}) := \sum_{c=1}^{C} \left[\mathbf{Z}_c^{(l)} \mid \mathbf{U}_{c1}\mathbf{V}'_{c1} \mid \mathbf{Z}_c^{(r)} \right] - \mathbf{U}_1\mathbf{V}'_1. \tag{14}$$

Here, $\mathbf{Z}_c^{(l)}$ and $\mathbf{Z}_c^{(r)}$ are fill matrices of zeros to complete the dimensions of $\mathbf{X}$. The first term in (14) reduces the variance of the principal components of the per-class features. The second term increases the variance of all the features together, projecting the feature matrix onto its closest orthogonal form.

The following proposition states that for sufficiently small δ, this direction vanishes only when the objective reaches the global minimum of zero.

Proposition 1 *If* $g_{L_o}(\mathbf{X}) = 0$ *and* $||X_c||_* > \Delta$, *then* $L_o(\mathbf{X}) = 0$.

Proof We give the proof for two classes, its extension to multiple classes is straightforward. Let $\mathbf{X} = [\mathbf{A} \mid \mathbf{B}]$ with $\mathbf{A}$ and $\mathbf{B}$ corresponding to the feature matrices of two classes. Let $\mathbf{A} = \mathbf{U}_{A1}\Sigma_{A1}\mathbf{V}'_{A1} + \mathbf{U}_{A2}\Sigma_{A2}\mathbf{V}'_{A2}$, $\mathbf{B} = \mathbf{U}_{B1}\Sigma_{B1}\mathbf{V}'_{B1} + \mathbf{U}_{B2}\Sigma_{B2}\mathbf{V}'_{B2}$, and $\mathbf{X} = \mathbf{U}_1\Sigma_1\mathbf{V}'_1 + \mathbf{U}_2\Sigma_2\mathbf{V}'_2$ be their SVD decomposition, where the subscript 1 corresponds to the singular values larger than zero and the subscript 2 to the remaining singular values. Let $\mathbf{0}$ be a generic matrix of zeroes, whose size is determined by context, for simplicity. Then,

$$L_o(\mathbf{X}) = ||A||_* + ||B||_* - |||[A \mid B]|||_*, \tag{15}$$

$$g_{L_o}(\mathbf{X}) = \left[\mathbf{U}_{A1}\mathbf{V}'_{A1} \middle| \mathbf{0} \right] + \left[\mathbf{0} \middle| \mathbf{U}_{B1}\mathbf{V}'_{B1} \right] - \mathbf{U}_1\mathbf{V}'_1. \tag{16}$$

Then, $g_{L_o}(\mathbf{X}) = 0$ implies

$$\mathbf{U}_1\mathbf{V}'_1 = \left[\mathbf{U}_{A1}\mathbf{V}'_{A1} \middle| \mathbf{0} \right] + \left[\mathbf{0} \middle| \mathbf{U}_{B1}\mathbf{V}'_{B1} \right] \tag{17}$$

$$= \left[\mathbf{U}_{A1} \middle| \mathbf{U}_{B1} \right] \left[\begin{array}{c|c} \mathbf{V}'_{A1} & \mathbf{0} \\ \mathbf{0} & \mathbf{V}'_{B1} \end{array} \right]. \tag{18}$$

Since $\mathbf{U}_1$ and $\mathbf{V_1}$ are orthogonal matrices, and the rightmost matrix in (18) is also orthogonal, then $[\mathbf{U}_{A1} \mid \mathbf{U}_{B1}]$ must be orthogonal. Since $\mathbf{U}_{A1}$ and $\mathbf{U}_{B1}$ are orthogonal submatrices, this implies that their columns must be orthogonal to each other. Then, $\mathbf{A}$ and $\mathbf{B}$ are orthogonal to each other and thus $L_o(\mathbf{X}) = 0$ (Theorem 1).

3 Applications

3.1 Subspace Clustering

In this section we showcase a first application of the linear low-rank transform for a clustering task. The adopted clustering framework, termed LRSC (Learned Robust Subspace Clustering), detailed in Algorithm 3, iterates between two stages: In the first assignment stage, we obtain clusters using any subspace clustering methods, e.g., SSC Elhamifar and Vidal (2013), LSA Yan and Pollefeys (2006), LBF Zhang et al. (2012). In particular, we also discuss a subspace clustering technique that fully exploits low-rank structures. In the second update stage, based on the current clustering result, we compute the optimal transformation that minimizes (3). The algorithm is repeated until the clustering assignments stop changing.

Input: A set of data points $\mathbf{Y} = \{\mathbf{y}_i\}_{i=1}^{N} \subseteq \mathbb{R}^d$ in a union of C subspaces.
Output: A partition of $\mathbf{Y}$ into C disjoint clusters $\{\mathbf{Y}_c\}_{c=1}^{C}$ based on underlying subspaces.
begin
1. Initial a transformation matrix $\mathbf{T}$ as the identity matrix ;
repeat
Assignment stage:
2. Assign points in $\mathbf{TY}$ to clusters with any subspace clustering methods, e.g., the proposed R-SSC;
Update stage:
3. Obtain transformation $\mathbf{T}$ by minimizing (3) based on the current clustering result ;
until *assignment convergence*;
4. Return the current clustering result $\{\mathbf{Y}_c\}_{c=1}^{C}$;
end

Algorithm 3: Learning a robust subspace clustering (LRSC) framework.

LRSC (Algorithm 3) is a general procedure to enhance the performance of any subspace clustering method. To fully exploit the low-rank structure of the learned transformed subspaces, we further discuss the following specific technique for the clustering step in the above framework, called Robust Sparse Subspace Clustering (R-SSC):

1. For the transformed subspaces, we first recover their low-rank representation $\mathbf{L}$ by performing a low-rank decomposition (19), e.g., using RPCA Candès et al. (2011),[1]

[1] Note that while the learned transform $\mathbf{T}$ encourages low-rank in each sub-space, outliers might still exist. Moreover, during the iterations in Algorithm 3, the intermediate learned $\mathbf{T}$ is not yet the desired one. This justifies the incorporation of this further low-rank decomposition.

$$\arg\min_{\mathbf{L},\mathbf{S}} ||\mathbf{L}||_* + \beta||\mathbf{S}||_1 \text{ s.t. } \mathbf{TY} = \mathbf{L} + \mathbf{S}. \tag{19}$$

2. Each transformed point $\mathbf{Ty}_i$ is then sparsely decomposed over $\mathbf{L}$,

$$\arg\min_{\mathbf{x}_i} \|\mathbf{Ty}_i - \mathbf{L}\mathbf{x}_i\|_2^2 \text{ s.t. } \|\mathbf{x}_i\|_0 \leq K, \tag{20}$$

   where K is a predefined sparsity value ($K > d$). As explained in Elhamifar and Vidal (2013), a data point in a linear or affine subspace of dimension d can be written as a linear or affine combination of d or $d+1$ points in the same subspace. Thus, if we represent a point as a linear or affine combination of all other points, a sparse linear or affine combination can be obtained by choosing d or $d+1$ nonzero coefficients.
3. As the optimization process for (20) is computationally demanding, we further simplify (20) using Local Linear Embedding Roweis and Saul (2000); Wang et al. (2010). Each transformed point $\mathbf{Ty}_i$ is represented using its K Nearest Neighbors (NN) in $\mathbf{L}$, which are denoted as $\mathbf{L}_i$,

$$\arg\min_{\mathbf{x}_i} \|\mathbf{Ty}_i - \mathbf{L}_i\mathbf{x}_i\|_2^2 \text{ s.t. } \mathbf{1}'\mathbf{x}_i = 1. \tag{21}$$

   Let $\bar{\mathbf{L}}_i = \mathbf{L}_i - \mathbf{1}\mathbf{Ty}_i'$. $\mathbf{x}_i$ can then be efficiently obtained in closed form Saul and Roweis (2000),

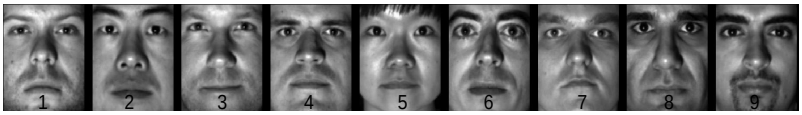
$$\mathbf{x}_i = \bar{\mathbf{L}}_i\bar{\mathbf{L}}_i' \setminus \mathbf{1},$$

   where $\mathbf{x} = \mathbf{A} \setminus \mathbf{B}$ solves the system of linear equations $\mathbf{Ax} = \mathbf{B}$, and then we rescale $\mathbf{x}_i$ so that $\mathbf{1}'\mathbf{x}_i = 1$. As suggested in Roweis and Saul (2000), if the correlation matrix $\bar{\mathbf{L}}_i\bar{\mathbf{L}}_i'$ is nearly singular, it can be conditioned by adding a small multiple of the identity matrix. From experiments, we observe this simplification step dramatically reduces the running time, without sacrificing the accuracy.
4. Given the sparse representation $\mathbf{x}_i$ of each transformed data point $\mathbf{Ty}_i$, we denote the sparse representation matrix as $\mathbf{X} = [\mathbf{x}_1 \dots \mathbf{x}_N]$. It is noted that $\mathbf{x}_i$ is written as an N-sized vector with no more than $K << N$ non-zero values (N being the total number of data points). The pairwise affinity matrix is now defined as $\mathbf{W} = |\mathbf{X}| + |\mathbf{X}'|$, and the subspace clustering is obtained using spectral clustering Luxburg (2007).

Based on our experimental results, the proposed R-SSC outperforms the compared subspace clustering techniques, in both accuracy and running time, e.g., about 500 times faster than the original SSC using the implementation provided in Elhamifar and Vidal (2013). Performance is further enhanced when R-SCC is used as an internal step of the LRSC framework in Algorithm 3.

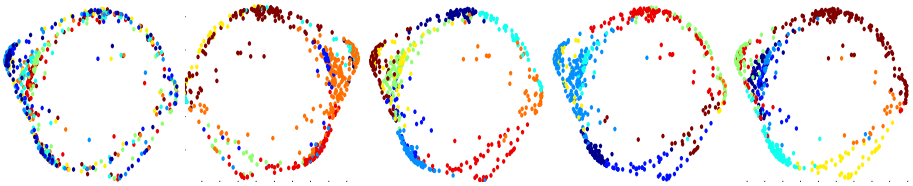
In the Extended YaleB data set, 38 different subjects are imaged under 64 lighting conditions, shown in Figure 3a. Under the assumption of Lambertian reflectance, face images of each subject under different lighting conditions can be accurately approximated with a 9-dimensional linear subspace Basri and Jacobs (2003). We conduct the face clustering experiments on the first 9 subjects shown in Figure 3b.

(a) Example illumination conditions.



(b) Example subjects.

Fig. 3: The extended YaleB face data set.



(a) Ground truth. (b) SSC, $e = 71.25\%$, $t = 714.99\ sec.$ (c) LBF, $e = 76.37\%$, $t = 460.76\ sec.$ (d) LSA, $e = 71.96\%$, $t = 22.57\ sec.$ (e) R-SSC, $e = \mathbf{67.37}\%$, $t = \mathbf{1.83}\ sec.$

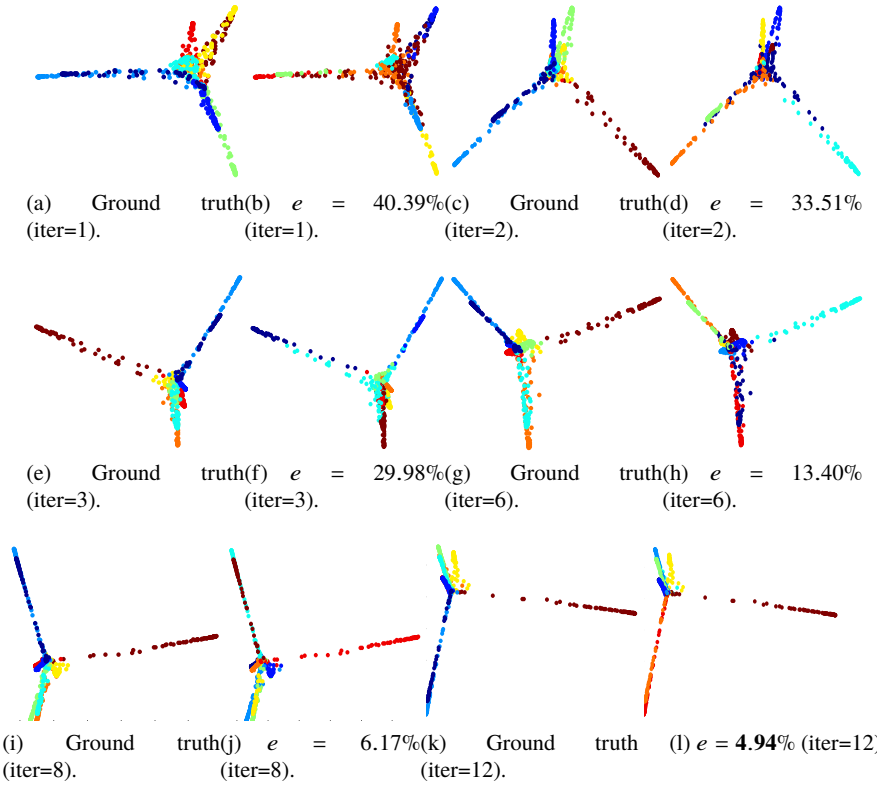
Fig. 4: Misclassification rate (e) and running time (t) on clustering 9 subjects using different subspace clustering methods. The proposed R-SSC outperforms state-of-the-art methods both in accuracy and running time. This is further improved using the learned transform, LRSC reduces the error to 4.94%, see Figure 5.

We set the sparsity value $K = 10$ for R-SSC, and perform 100 iterations for the subgradient descent updates while learning the transformation.

Figure 4 shows error rate (e) and running time (t) on clustering subspaces of 9 subjects using different subspace clustering methods. The proposed R-SSC techniques outperforms state-of-the-art methods both in accuracy and running time. As shown in Figure 5, using the LRSC algorithm (that is, learning the transform), the misclassification errors of R-SSC are further reduced significantly, for example, from 67.37% to 4.94% for the 9 subjects.

3.2 Classification

In this section, we demonstrate the use of the non-linear OLT in the deep learning framework for improved generalization. We highlight the plug-and-play nature of the proposed loss for several popular deep network architectures and different standard



(a) Ground truth(b) e = 40.39%(c) Ground truth(d) e = 33.51% (iter=1). (iter=1). (iter=2). (iter=2).

(e) Ground truth(f) e = 29.98%(g) Ground truth(h) e = 13.40% (iter=3). (iter=3). (iter=6). (iter=6).

(i) Ground truth(j) e = 6.17%(k) Ground truth (l) e = **4.94**% (iter=12). (iter=8). (iter=8). (iter=12).

Fig. 5: Misclassification rate (e) on clustering 9 subjects using the proposed LRSC framework. We adopt the proposed R-SSC technique for the clustering step. With the proposed LRSC framework, the clustering error of R-SSC is further reduced significantly, e.g., from 67.37% to 4.94% for the 9-subject case. Note how the classes are clustered in clean subspaces in the transformed domain.

visual classification datasets. We will also further analyze the effect the proposed embedding has on the deep features.

In all experiments, we seek to minimize the combination of the classification loss and the non-linear OLT loss:

$$\min_{\theta} L_s(\mathbf{X}, \mathbf{y}, \theta) + \lambda \cdot L_o(\mathbf{X}, \mathbf{y}, \theta^*) + \mu \cdot ||\theta||^2, \tag{22}$$

where L_S is the softmax loss. The second term L_o is the objective function of non-linear OLT (6).

Here θ^* means every weight in the network except the weights of the last fully-connected layer, which is the linear classifier. This is because the non-linear OLT is applied to the deep features at the penultimate layer. The third term represents the

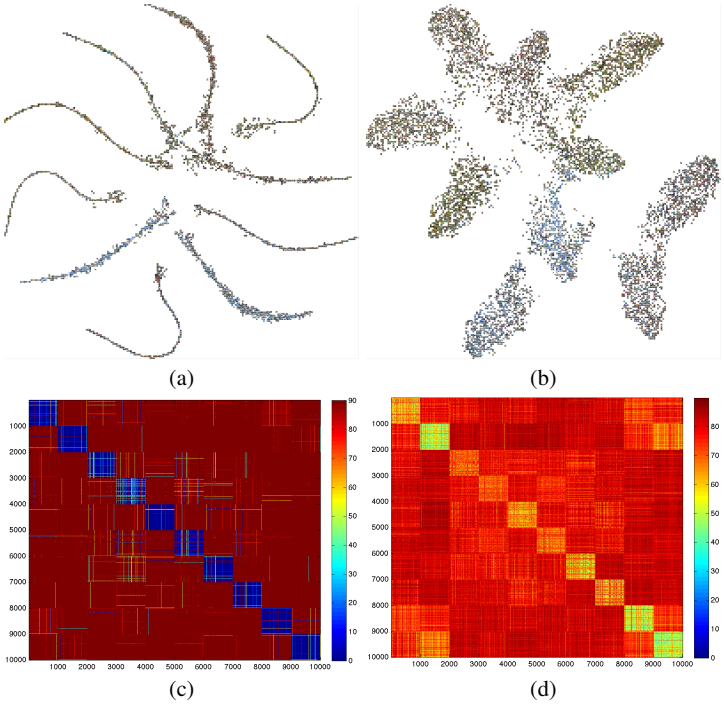
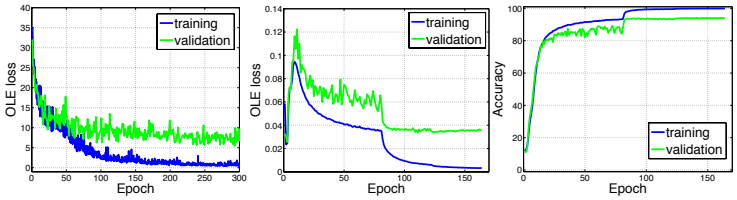


Fig. 6: Barnes-Hut-SNE Van Der Maaten (2013) visualization of the deep feature embedding learned for the validation set of CIFAR10, using VGG-16 Simonyan and Zisserman (2014). **(a)** With softmax loss and non-linear OLT . **(b)** With softmax loss only. The separation between classes is increased, and a low-rank structure is recovered for each class. **(c)** Angle between the features of the 10,000 validation samples, ordered by class, with non-linear OLT. **(d)** Without non-linear OLT. With the proposed transformation the angle between features is collapsed inside each class and inter-class features are orthogonal.

standard weight decay. The parameter λ controls the weight of the non-linear OLT loss; $\lambda = 0$ corresponds to standard network training. This parameter was adjusted with a held-out validation set of 10% of the training set Lezama et al. (2018). Note that the magnitude of the non-linear OLT objective depends on the size and norm of the features matrices.

The values of the hyperparameters, as well as other details on the network architectures, training procedures and datasets can be found in the original publication Lezama et al. (2018)[2]. The additional computation time incurred by using the non-linear OLT term is between 10% and 33% during training, using an unoptimized

[2] Source code for the non-linear OLT can be found for the Caffe and PyTorch libraries in `https://github.com/jlezama/OrthogonalLowrankEmbedding`.



(a) (b) (c)

Fig. 7: Learning curves. **(a)** non-linear OLT objective when used standalone. Data and model from Fig 2c. **(b) & (c)** non-linear OLT objective and accuracy when used in combination with softmax loss for a ResNet-110 on CIFAR10+ He et al. (2016a). Learning rate drops by 0.1 at 81 and 122 epochs.

Dataset	Architecture	λ	% Error ($L_o + \lambda \cdot L_s$)	% Error (L_s only)
SVHN	DenseNet-40-12 Huang et al. (2016)	1/2	**3.62** ± 0.04	3.93 ± 0.08
MNIST	DenseNet-40-12	1/2	**0.78** ± 0.04	0.88 ± 0.03
CIFAR10+	DenseNet-40-12	1/8	**5.30** ± 0.26	5.54 ± 0.13
CIFAR10+	ResNet-110 He et al. (2016a)	1/4	**5.39** ± 0.25	6.05 ± 0.8
CIFAR10+	VGG-19 Simonyan and Zisserman (2014)	1/4	**7.13** ± 0.2	7.37 ± 0.11
CIFAR10+	VGG-11	1/2	**7.73** ± 0.14	8.06 ± 0.22
CIFAR10	VGG-16 Liu et al. (2016)	1/2	**7.22** ± 0.14	8.23 ± 0.13
CIFAR100+	PreResNet-110 He et al. (2016b)	1/20	**22.8** ± 0.34	23.01 ± 0.19
CIFAR100+	VGG-19	1/10	**27.54** ± 0.11	28.04 ± 0.42
CIFAR100	VGG-19	1/10	**37.25** ± 0.33	38.15 ± 0.28
FaceScrub-500	VGG-FACE Parkhi et al. (2015a)	250	**1.55** ± 0.02	2.49 ± 0.01
STL-10	5-layer CNN	1/16	**25.42** ± 0.20	28.68 ± 0.67
STL-10+	5-layer CNN	1/4	**16.68** ± 0.24	18.22 ± 0.27

Table 1: Visual classification results. L_o is the non-linear OLT objective, L_s is the standard softmax loss. When using non-linear OLT, networks generalize better than with softmax loss alone.

implementation that performs the SVD in CPU. Faster runtimes could be achieved by performing the SVD on the GPU.

3.2.1 Visual Classification Results

Table 1 shows the resulting classification performance, with and without application of the non-linear OLT. In all the experiments, a value of λ was found through validation such that the generalization of the network is improved.

Compared to a state-of-the-art intra-class compactness method Liu et al. (2016) using VGG-16 on CIFAR10, the lowest classification error obtained using non-linear OLT was 7.08%, compared to 7.58% reported in Liu et al. (2016). Compared to the same network with only the standard softmax loss, a relative reduction in the error of more than 12% is obtained.

The improvement in generalization performance is more important when only scarce training data is available. In the Facescrub-500 experiment, where less than 100 samples are available per class on average, the error is reduced by 40%. Figure **??** illustrates how the advantage of using non-linear OLT is more significant when fewer training data is available. We fixed $\lambda = 0.25$ and trained a 5-layer CNN on STL-10 without data augmentation. We varied the number of samples from just 50 to 500 training samples per class, repeating each experiment 5 times.

3.3 Cross-modal Embedding

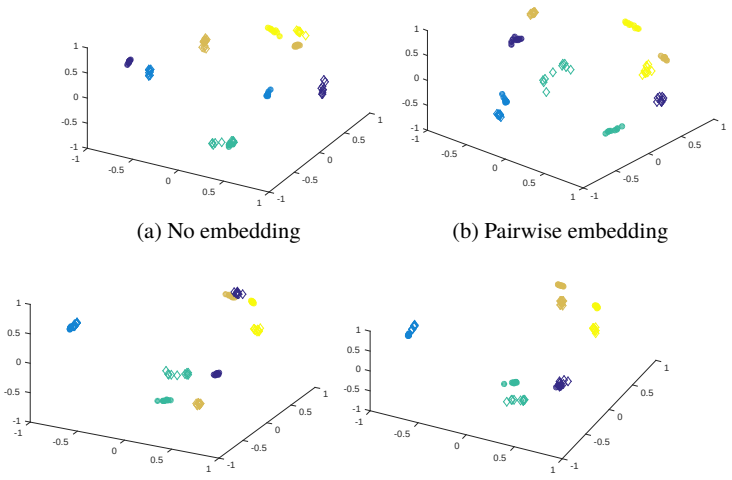
In this section, we propose a simple way to extend a deep neural network (DNN) model pre-trained on face images in the visible spectrum (VIS) to the near-infrared spectrum (NIR), using the linear low-rank embedding at the output layer. With the low-rank embedding layer appended at the end, the deep neural network that sees only VIS images produces deep features for VIS and NIR images in a common space. More details of this approach can be found in Lezama et al. (2017).

In traditional single-spectrum VIS face recognition, DNN models adopt the classification objective such as softmax at the final layer. Deep features produced at the second-to-last layer are often l_2-normalized, and then compared using the cosine similarity to perform face recognition Parkhi et al. (2015b); Sun et al. (2014). Thus, successful face DNN models expect deep features generated for VIS faces from the same subject to reside in a low-dimensional subspace.

In Figure 8a we illustrate the following, which motivates the proposed embedding: Face images from five subjects in VIS and NIR are input to VGG-face Parkhi et al. (2015b), one of the best publicly available DNN face models. The generated deep features are visualized in two dimensions using PCA, with filled circle for VIS, unfilled diamond for NIR, and one color per subject. We observe that VIS and NIR faces from the same subject often form two different clusters respectively. Such observation indicates that a successful DNN face model pre-trained on VIS faces is able to generate discriminative features for NIR faces; however, when a subject is imaged under a different spectrum, the underlying low-rank structure assumption is often violated.

Our finding is that the low-rank transform $\mathbf{T}$ in (3) can still effectively restore for the same subject a low-rank structure, even when $\mathbf{Y}_c$ contains mixed NIR and VIS training data from the c-th subject. As no DNN retraining is required, a very important advantage of our approach in practice, the learned low-rank transform can be simply appended as a linear embedding layer after the DNN output layer to allow a VIS model accepting both VIS and NIR images. As shown in Figure 8d, low-rank embedding effectively unifies cross-spectral deep features in Figure 8a.

In DNN-based face recognition, deep feature embeddings with pairwise or triplet constraints are commonly used Parkhi et al. (2015b); Saxena and Verbeek (2016); Schroff et al. (2015). Two popular DNN embedding schemes, pair-wise (ITML Davis et al. (2007)) and triplet (LMNN Weinberger and Saul (2009)) embeddings,



(a) No embedding (b) Pairwise embedding

(c) Triplet embedding (d) Low-rank embedding

Fig. 8: Deep features generated using the VGG-face model Parkhi et al. (2015b) for VIS (filled circle) and NIR (unfilled diamond) face images from five subjects, one color per subject. Data are visualized in two dimensions using PCA. In (a), without embedding, VIS and NIR faces from the same subject often form two different clusters respectively. In (d), the low-rank embedding successfully restores a low-rank structure for multi-spectrum faces from the same subject. In (b) and (c), popular pair-wise and triplet embeddings still show significant intra-class variations across spectrums (best viewed zooming on screen).

are shown in Figure 8b and Figure 8c respectively, and contrary to our approach, significant intra-class variations, i.e., the distance between same color clusters, are still observed across spectrums. Table 2 shows quantitatively the result of applying this method to NIR-VIS matching, when using different VIS DNNs as black boxes.

3.4 Hashing

We discuss how to use OLT to design a simple random forest hashing scheme, where independently trained random trees act as hashing functions by setting '1' for the visited tree leaf, and '0' for the rest.

Random forest Breiman (2001); Criminisi and Shotton (2013) is an ensemble of binary *decision trees*, where each tree consists of hierarchically connected *split* (internal) nodes and *leaf* (terminal) nodes. Each split node corresponds to a *weak learner*, and evaluates each arriving data point sending it to the left or right child based on the weak learner binary outputs. Following the random forest literature

	Accuracy (%)
VGG-S	75.04
VGG-S + Hallucination	80.65
VGG-S + Low-rank	89.88
VGG-S + Hallucination + Low-rank	95.72
VGG-face	72.54
VGG-face + Hallucination	83.10
VGG-face + Low-rank	82.26
VGG-face + Hallucination + Low-rank	91.01
COTS	83.84
COTS + Hallucination	93.02
COTS + Low-rank	91.83
COTS + Hallucination + Low-rank	**96.41**

Table 2: Cross-spectral rank-1 identification rate on CASIA NIR-VIS 2.0 (see text for protocol). We evaluate three pre-trained single-spectrum (VIS) DNN models: VGG-S Chatfield et al. (2014), VGG-face Parkhi et al. (2015b), and a commercial of-the-shelf model(COTS). Hallucination refers to a pre-processing step where a VIS image is hallucinated from a NIR one Lezama et al. (2017). When both schemes are used together, we observe significant further improvements, e.g., 75.04% to 95.72% for the VGG-S model. The proposed framework gives state-of-the-art (96.41%) without touching at all the VIS recognition system.

Criminisi and Shotton (2013), we specify here a maximum tree depth d to limit the size of a tree. Thus, a tree of depth d consists of 2^{d-1} tree leaf nodes, indexed in the breadth-first order.

Traditional random forests fail as hashing functions mainly because of two issues: their inability to preserve the underlying similarity due to the inconsistency of hash codes generated in each tree for the same class data; and the lack of principled way of aggregating hash codes produced by individual trees into a single longer code. Here we address both problems with a novel method based on the OLT. First, a random class grouping scheme is used to enforce code uniqueness for each class. Second, the OLT is used for weak learners inside each node to encourage code consistency.

The proposed method is termed ForestHash Qiu et al. (2018). It consists of shallow random trees in a forest, usually of depth 2 or 3. At each tree split node, arriving classes are randomly partitioned into two groups for a significantly simplified two-class classification problem, which can be sufficiently handled by a light-weight CNN weak learner, usually of 2 to 4 layers. We set '1' for the visited tree leaf, and '0' for the rest. By simultaneously pushing each data point through M trees of the depth d, we obtain $M(2^{d-1})$-bit hash codes. The random grouping of the classes enables code uniqueness by enforcing that each class shares code with different classes in different trees. The low-rank loss adopted for CNN weak learners encourages code consistency by minimizing intra-class variations and maximizing inter-class distance for the two random class groups. The obtained codes serve as efficient and compact image representation for both image retrieval and classification.

3.4.1 Random class grouping

A random class grouping scheme is first introduced to randomly partition arriving classes into two groups at each tree split node. Random class grouping serves two main purposes: First, a multi-class problem is significantly reduced to a two-class classification problem at each split node, which can be sufficiently handled by a very light-weight CNN weak learner. Second, random class grouping enforces each class to share its code with different classes in different trees, which allows the information-theoretic aggregation developed in the sequel to later produce a near-optimal unique hash code for each class.

3.4.2 Low-rank loss

The low-rank loss is adopted for the weak learners in each node of the forest. Consider s-dimensional data points belonging to two classes after random class grouping, which for simplicity are denoted as positive and negative. We stack the points as columns of the matrices $\mathbf{X}^+$ and $\mathbf{X}^-$, respectively. At each tree split node, we learn the transformation matrix $\mathbf{T}$ for a binary problem:

$$\min_{\mathbf{T}} ||\mathbf{T}\mathbf{X}^+||_* + ||\mathbf{T}\mathbf{X}^-||_* - ||\mathbf{T}[\mathbf{X}^+, \mathbf{X}^-]||_*. \tag{23}$$

Based on Theorem 1, (23) reaches its minimum 0 if the column spaces of the two classes become orthogonal after applying the learned transformation $\mathbf{T}$.

Splitting functions. With random class grouping, we have a two-class classification problem at each split node. We stack the training data points from each class as columns of the matrices $\mathbf{X}^+$ and $\mathbf{X}^-$, respectively. During training, at the i-th split node, we denote the arriving training samples as $\mathbf{X}^+$ and $\mathbf{X}^-$. After a weight matrix $\mathbf{T}$ is successfully learned by minimizing (23), it is reasonable to assume that each of the classes will belong to a low-dimensional subspace, the distance from which can be used to classify previously unseen points. We use k-SVD Aharon et al. (2006) to learn a pair of dictionaries $\mathbf{D}^\pm$, for each of the two classes, by minimizing

$$\min_{\mathbf{D}_\pm, \mathbf{Z}^\pm} \|\mathbf{T}\mathbf{X}^\pm - \mathbf{D}^\pm\mathbf{Z}^\pm\| \text{ s.t. } \|\mathbf{z}^\pm\|_0 \leq l, \tag{24}$$

where the ℓ_0 pseudonorm $\|\mathbf{z}^\pm\|_0$ counts the number of non-zero elements in each column of $\mathbf{Z}^\pm$, and l controls the subspace dimension.

At testing, given a data point $\mathbf{x}$, the splitting function is evaluated by first projecting $\mathbf{x}$ onto both dictionaries and evaluating the projection errors

$$e^\pm(\mathbf{x}) = \arg\min_{\mathbf{z}^\pm} \|\mathbf{D}^\pm\mathbf{z}^\pm - \mathbf{W}\mathbf{x}\|_2 \;=\; \|\mathbf{P}^\pm\mathbf{x}\|_2, \tag{25}$$

where $\mathbf{P}^\pm = \mathbf{D}^\pm(\mathbf{D}^{\pm\mathrm{T}}\mathbf{D}^\pm)^{-1}\mathbf{D}^{\pm\mathrm{T}}\mathbf{T}$ are the $n \times n$ projection matrices. The point is sent to the left subtree if $e^-(\mathbf{x}) < e^+(\mathbf{x})$, and to the right subtree otherwise. In practice, we only store the projection matrices $\mathbf{P}^\pm$ at each split node.

Note that the non-linear version of the orthogonal low-rank transform is the ultimate way in handling intricate data. As in Figure 6, the intra-class variations among features are collapsed and inter-class features are orthogonal. Such property is particularly beneficial at each tree split node.

3.4.3 Experimental Evaluation

		6,000 samples per class			100 samples per class			30 samples per class		
	Test time (μs)	Train time (s)	Prec.	Rec.	Train time (s)	Prec.	Rec.	Train time (s)	Prec.	Rec.
HDML Norouzi et al. (2012)	10	93780	92.94	60.44	1505	62.52	2.19	458	24.28	0.21
FastHash Lin et al. (2014)	115	865	84.70	76.60	213	73.32	33.04	151	57.08	11.77
TSH Lin et al. (2013)	411	164325	86.30	3.17	21.08	74.00	5.19	2.83	56.86	3.94
ForestHash CNN2	13	81.6	**97.99**	**95.99**	7	**94.24**	**74.02**	2.69	**89.56**	**46.36**

Table 3: Experimental results on 36-bit retrieval performance (%) on MNIST (rejection hamming radius 0), using different training set sizes. Test time is the average binary encoding time in microseconds (μs).

Table 3 shows a comparison between ForestHash and the supervised hashing methods HDML Norouzi et al. (2012), TSH Lin et al. (2013), and FastHash Lin et al. (2014). All methods report excellent performance, where HDML is a deep learning based hashing method, and FastHash is a boosted trees based method. We adopt the experimental setting from Norouzi et al. (2012), i.e., a 60K training set and a disjoint 10K query set split on the MNIST data. Each hashing method is assessed by the retrieval precision and recall at radius 0. As shown in Table 3, ForestHash with a two-layer CNN significantly outperforms all compared methods. We refer the reader to Qiu et al. (2018) for futher quantitative performance results.

4 Conclusions

In this chapter, we introduced an orthogonal low-rank transformation approach OLT. Using nuclear norm as the optimization criteria, we learn a subspace transformation that reduces variations within the subspaces by enforcing a low-rank structure, and increases separations between the subspaces by enforcing orthogonality. Both linear and non-linear versions are presented and theoretical support for the proposed method is provided for each case. Experimentally, the proposed transformation successfully embeds the data into a convenient low-rank and orthogonal structure, improving the state-of-the-art performance for various tasks, such as subpace clustering, image classification, domain adaptation and hashing.

5 Proof of Theorem 1

It suffices to prove the case when the number of classes $K = 2$, as larger K by induction. This proof was taken from Qiu and Sapiro (2015); Zhu et al. (2019).

Proof Let $\begin{bmatrix} \mathbf{E} & \mathbf{G} \\ \mathbf{G}^* & \mathbf{F} \end{bmatrix} = \begin{bmatrix} \mathbf{A}^*\mathbf{A} & \mathbf{A}^*\mathbf{B} \\ \mathbf{B}^*\mathbf{A} & \mathbf{B}^*\mathbf{B} \end{bmatrix}^{\frac{1}{2}} = |[\mathbf{A}, \mathbf{B}]|$, we have

$$\begin{cases} |\mathbf{A}|^2 = \mathbf{A}^*\mathbf{A} = \mathbf{E}^2 + \mathbf{G}\mathbf{G}^* \\ |\mathbf{B}|^2 = \mathbf{B}^*\mathbf{B} = \mathbf{F}^2 + \mathbf{G}^*\mathbf{G} \\ \mathbf{A}^*\mathbf{B} = \mathbf{E}\mathbf{G} + \mathbf{G}\mathbf{F}. \end{cases} \tag{26}$$

Suppose $\{\boldsymbol{a}_i\}_{i=1}^m$, $\{\boldsymbol{b}_i\}_{i=1}^n$ are the orthonormal eigenvectors of $|\mathbf{A}|, |\mathbf{B}|$, then

$$\||\mathbf{A}|\boldsymbol{a}_i\|^2 = \left\langle |\mathbf{A}|^2\boldsymbol{a}_i, \boldsymbol{a}_i \right\rangle = \left\langle (\mathbf{E}^2 + \mathbf{G}\mathbf{G}^*)\boldsymbol{a}_i, \boldsymbol{a}_i \right\rangle = \|\mathbf{E}\boldsymbol{a}_i\|^2 + \|\mathbf{G}^*\boldsymbol{a}_i\|^2, \tag{27}$$

$$\||\mathbf{B}|\boldsymbol{b}_i\|^2 = \left\langle |\mathbf{B}|^2\boldsymbol{b}_i, \boldsymbol{b}_i \right\rangle = \left\langle (\mathbf{F}^2 + \mathbf{G}^*\mathbf{G})\boldsymbol{b}_i, \boldsymbol{b}_i \right\rangle = \|\mathbf{F}\boldsymbol{b}_i\|^2 + \|\mathbf{G}\boldsymbol{b}_i\|^2. \tag{28}$$

We thus have the following inequality chain that proves (4)

$$\begin{aligned} \|\mathbf{A}\|_* + \|\mathbf{B}\|_* &= \mathrm{Tr}(|\mathbf{A}|) + \mathrm{Tr}(|\mathbf{B}|) = \sum_{i=1}^m \langle |\mathbf{A}|\boldsymbol{a}_i, \boldsymbol{a}_i \rangle + \sum_{i=1}^n \langle |\mathbf{B}|\boldsymbol{b}_i, \boldsymbol{b}_i \rangle \\ &= \sum_{i=1}^m \||\mathbf{A}|\boldsymbol{a}_i\| + \sum_{i=1}^n \||\mathbf{B}|\boldsymbol{b}_i\| \\ &= \sum_{i=1}^m \left(\|\mathbf{E}\boldsymbol{a}_i\|^2 + \|\mathbf{G}^*\boldsymbol{a}_i\|^2 \right)^{\frac{1}{2}} + \sum_{i=1}^n \left(\|\mathbf{F}\boldsymbol{b}_i\|^2 + \|\mathbf{G}\boldsymbol{b}_i\|^2 \right)^{\frac{1}{2}} \\ &\geq \sum_{i=1}^m \|\mathbf{E}\boldsymbol{a}_i\| + \sum_{i=1}^n \|\mathbf{F}\boldsymbol{b}_i\| \geq \sum_{i=1}^m \langle \mathbf{E}\boldsymbol{a}_i, \boldsymbol{a}_i \rangle + \sum_{i=1}^n \langle \mathbf{F}\boldsymbol{b}_i, \boldsymbol{b}_i \rangle \\ &= \mathrm{Tr}(\mathbf{E}) + \mathrm{Tr}(\mathbf{F}) = \mathrm{Tr}(|[\mathbf{A}, \mathbf{B}]|) = \|[\mathbf{A}, \mathbf{B}]\|_* \end{aligned} \tag{29}$$

We next show that the equality holds if and only if $\mathbf{A}^*\mathbf{B} = \mathbf{0}$.

- If $\mathbf{A}^*\mathbf{B} = \mathbf{0}$, then

$$\begin{aligned} \|[\mathbf{A}, \mathbf{B}]\|_* &= \mathrm{Tr}(|[\mathbf{A}, \mathbf{B}]|) = \mathrm{Tr}\left(\begin{bmatrix} \mathbf{A}^*\mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{B}^*\mathbf{B} \end{bmatrix}^{\frac{1}{2}} \right) = \mathrm{Tr}\left(\begin{bmatrix} |\mathbf{A}| & \mathbf{0} \\ \mathbf{0} & |\mathbf{B}| \end{bmatrix} \right) \\ &= \mathrm{Tr}(|\mathbf{A}|) + \mathrm{Tr}(|\mathbf{B}|) = \|\mathbf{A}\|_* + \|\mathbf{B}\|_*. \end{aligned} \tag{30}$$

- If $\|[\mathbf{A}, \mathbf{B}]\|_* = \|\mathbf{A}\|_* + \|\mathbf{B}\|_*$, then both of the inequalities in the chain (29) must be equalities. Note that the first inequality is an equality only if $\mathbf{G} = \mathbf{0}$. This combined with the last equation in (26) implies

$$\mathbf{A}^*\mathbf{B} = \mathbf{E}\mathbf{G} + \mathbf{G}\mathbf{F} = \mathbf{0} \tag{31}$$

References

Aharon M, Elad M, Bruckstein A (2006) K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. IEEE Transactions on signal processing 54(11):4311–4322

Basri R, Jacobs DW (2003) Lambertian reflectance and linear subspaces. IEEE Trans on Patt Anal and Mach Intell 25(2):218–233

Boyd S, Xiao L, Mutapcic A (2003) Subgradient method. Notes for EE392o, Stanford University

Breiman L (2001) Random forests. Machine Learning 45(1):5–32

Candès EJ, Li X, Ma Y, Wright J (2011) Robust principal component analysis? J ACM 58(3):11:1–11:37

Chatfield K, Simonyan K, Vedaldi A, Zisserman A (2014) Return of the devil in the details: Delving deep into convolutional nets. In: British Machine Vision Conference (BMVC)

Collobert R, Sinz F, Weston J, Bottou L (2006) Large scale transductive svms. J Mach Learn Res 7:1687–1712

Criminisi A, Shotton J (2013) Decision Forests for Computer Vision and Medical Image Analysis. Springer

Davis JV, Kulis B, Jain P, Sra S, Dhillon IS (2007) Information-theoretic metric learning. In: ICML, Corvalis, Oregon, USA, pp 209–216

Dinh TP, An LTH (1997) Convex analysis approach to d.c. programming: Theory, algorithms and applications. Acta Mathematica Vietnamica 22(1):289–355

Douglas SC, Amari S, Kung SY (2000) On gradient adaptation with unit-norm constraints. IEEE Trans on Signal Processing 48(6):1843–1847

Elhamifar E, Vidal R (2013) Sparse subspace clustering: Algorithm, theory, and applications. IEEE Trans on Patt Anal and Mach Intell To appear

Fazel M (2002) Matrix Rank Minimization with Applications. PhD thesis, Stanford University

Fuhrmann DR, Liu B (1984) An iterative algorithm for locating the minimal eigenvector of a symmetric matrix. In: Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing, Dallas, TX

Hastie T, Simard PY (1998) Metrics and models for handwritten character recognition. Statistical Science 13(1):54–65

He K, Zhang X, Ren S, Sun J (2016a) Deep residual learning for image recognition. In: Proc. IEEE Computer Society Conf. on Computer Vision and Patt. Recn., pp 770–778

He K, Zhang X, Ren S, Sun J (2016b) Identity mappings in deep residual networks. In: Proc. European Conference on Computer Vision, Springer, pp 630–645

Huang G, Liu Z, Weinberger KQ, van der Maaten L (2016) Densely connected convolutional networks. arXiv preprint arXiv:160806993

Lezama J, Qiu Q, Sapiro G (2017) Not afraid of the dark: NIR-VIS face recognition via cross-spectral hallucination and low-rank embedding. In: CVPR, pp 6628–6637

Lezama J, Qiu Q, Musé P, Sapiro G (2018) OLÉ: Orthogonal low-rank embedding, a plug and play geometric loss for deep learning. In: Proc. IEEE Computer Society Conf. on Computer Vision and Patt. Recn.

Lin G, Shen C, Suter D, van den Hengel A (2013) A general two-step approach to learning-based hashing

Lin G, Shen C, Shi Q, van den Hengel A, Suter D (2014) Fast supervised hashing with decision trees for high-dimensional data. In: Proc. IEEE Computer Society Conf. on Computer Vision and Patt. Recn.

Liu W, Wen Y, Yu Z, Yang M (2016) Large-margin softmax loss for convolutional neural networks. In: International Conference on Machine Learning, pp 507–516

Luxburg U (2007) A tutorial on spectral clustering. Statistics and Computing 17(4):395–416

Marsaglia G, Styan GPH (1972) When does rank (a + b) = rank(a)+rank(b)? Canad Math Bull 15(3)

Miao J, Ben-Israel A (1992) On principal angles between subspaces in R_n. Linear Algebra and its Applications 171(0):81 – 98

Neumann J, Schnörr C, Steidl G (2005) Combined SVM-based feature selection and classification. Mach Learn 61(1-3):129–150

Norouzi M, Fleet DJ, Salakhutdinov R (2012) Hamming distance metric learning. In: Advances in Neural Information Processing Systems

Parkhi OM, Vedaldi A, Zisserman A (2015a) Deep face recognition. In: British Machine Vision Conference

Parkhi OM, Vedaldi A, Zisserman A (2015b) Deep face recognition. In: BMVC

Peng Y, Ganesh A, Wright J, Xu W, Ma Y (2010) RASL: Robust alignment by sparse and low-rank decomposition for linearly correlated images. In: Proc. IEEE Computer Society Conf. on Computer Vision and Patt. Recn., San Francisco, USA

Qiu Q, Sapiro G (2015) Learning transformations for clustering and classification. Journal of Machine Learning Research 16(187-225):2

Qiu Q, Lezama J, Bronstein A, Sapiro G (2018) Foresthash: Semantic hashing with shallow random forests and tiny convolutional networks. In: Proceedings of the European Conference on Computer Vision (ECCV), pp 432–448

Recht B, Fazel M, Parrilo PA (2010) Guaranteed minimum rank solutions to linear matrix equations via nuclear norm minimization. SIAM Review 52(3):471–501

Roweis ST, Saul LK (2000) Nonlinear dimensionality reduction by locally linear embedding. Science 290:2323–2326

Saul LK, Roweis ST (2000) An introduction to locally linear embedding URL `http://www.cs.nyu.edu/~roweis/lle/publications.html`

Saxena S, Verbeek J (2016) Heterogeneous face recognition with CNNs. In: ECCV TASK-CV 2016 Workshops

Schroff F, Kalenichenko D, Philbin J (2015) Facenet: A unified embedding for face recognition and clustering. In: Proc. IEEE Computer Society Conf. on Computer Vision and Patt. Recn., pp 815–823

Shen X, Wu Y (2012) A unified approach to salient object detection via low rank matrix recovery. In: Proc. IEEE Computer Society Conf. on Computer Vision and Patt. Recn., Rhode Island, USA

Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:14091556

Sriperumbudur BK, Lanckriet GRG (2012) A proof of convergence of the concave-convex procedure using zangwill's theory. Neural Computation 24(6):1391–1407

Sriperumbudur BK, Torres DA, Lanckriet GRG (2007) Sparse eigen methods by d.c. programming. In: International Conference on Machine Learning

Sun Y, Chen Y, Wang X, Tang X (2014) Deep learning face representation by joint identification-verification. In: NIPS, pp 1988–1996

Tomasi C, Kanade T (1992) Shape and motion from image streams under orthography: a factorization method. International Journal of Computer Vision 9:137–154

Van Der Maaten L (2013) Barnes-Hut-SNE. arXiv preprint arXiv:13013342

Wang J, Yang J, Yu K, Lv F, Huang T, Gong Y (2010) Locality-constrained linear coding for image classification. In: Proc. IEEE Computer Society Conf. on Computer Vision and Patt. Recn., San Francisco, USA

Watson GA (1992) Characterization of the subdifferential of some matrix norms. Linear Algebra and Applications 170:1039–1053

Weinberger K, Saul L (2009) Distance metric learning for large margin nearest neighbor classification. The Journal of Machine Learning Research 10:207–244

Wright J, Yang A, Ganesh A, Sastry S, Ma Y (2009) Robust face recognition via sparse representation. IEEE Trans on Patt Anal and Mach Intell 31(2):210–227

Yan J, Pollefeys M (2006) A general framework for motion segmentation: independent, articulated, rigid, non-rigid, degenerate and non-degenerate. In: Proc. European Conference on Computer Vision, Graz, Austria

Yuille AL, Rangarajan A (2003) The concave-convex procedure. Neural Computation 4:915–936

Zhang T, Szlam A, Wang Y, Lerman G (2012) Hybrid linear modeling via local best-fit flats. International Journal of Computer Vision 100(3):217–240

Zhang Z, Liang X, Ganesh A, Ma Y (2011) TILT: transform invariant low-rank textures. In: Proc. Asian conference on Computer vision, Queenstown, New Zealand

Zhu W, Qiu Q, Wang B, Lu J, Sapiro G, Daubechies I (2019) Stop memorizing: A data-dependent regularization framework for intrinsic pattern learning. SIAM Journal on Mathematics of Data Science 1:476–496