Nonlinear Subspace Feature Enhancement for Image Set Classification

Mohammed E. Fathy(✉), Azadeh Alavi, and Rama Chellappa

Center for Automation Research, University of Maryland,
College Park, MD 20742, USA
{mefathy,azadeh,rama}@umiacs.umd.edu

Abstract. While several methods have been proposed for modeling and recognizing image sets, the success of these methods relies heavily on how well the image data follows the assumptions of the underlying models. Among the models that have been utilized by many image set classification methods, the physically inspired subspace model assumes that the images of an object lie on a union of low-dimensional subspaces. Despite their successful performance in controlled environments, the performance of such subspace-based classifiers suffers in practical unconstrained settings, where the data may not strictly follow the assumptions necessary for the subspace model to hold. In this paper, we propose Nonlinear Subspace Feature Enhancement (NSFE), an approach for nonlinearly embedding image sets into a space where they adhere to a more discriminative subspace structure. In turn, this improves the performance of subspace-based classifiers such as sparse representation-based classification. We describe how the structured loss function of NSFE can be optimized in a batch-by-batch fashion by a two-step alternating algorithm. The algorithm makes very few assumptions about the form of the embedding to be learned and is compatible with stochastic gradient descent and back-propagation. This makes NSFE usable with deep, feed-forward embeddings and trainable in an end-to-end fashion. We experiment with two different types of features and nonlinear embeddings over three image set datasets and we show that our method compares favorably to state-of-the-art image set classification methods.

1 Introduction

Image set classification aims to compute a single label for a set of images that are assumed to belong to the same class. The interest in the use of image sets for visual recognition tasks, such as face recognition, has grown in line with the increasing prevalence of video-capable consumer devices and surveillance cameras [1–19]. A video is typically believed to have richer information (i.e. more frames) than in a still image and so can lead to improved classification performance. However, the improvement in performance is sometimes limited in practice due to the challenges videos share with still images (*e.g.* variations in pose, illumination, motion-induced artifacts and occlusion) in addition to the

C. V. Jawahar et al. (Eds.): ACCV 2018, LNCS 11364, pp. 142–158, 2019.
https://doi.org/10.1007/978-3-030-20870-7_9

low resolution at which videos are sometimes captured to reduce bandwidth and storage requirements.

As reviewed in Sect. 2, several methods for modeling and classifying image sets have been proposed. Many of these have utilized the *subspace assumption* which (informally) states that the instances from a particular class lie on (or close to) a union of low-dimensional linear subspaces (the property is illustrated in Fig. 1). The assumption is theoretically founded on the work of [20] which shows that the images of a static convex Lambertian object, taken under varying Lambertian illumination from a fixed viewpoint, approximately lie on a low-dimensional subspace [20].

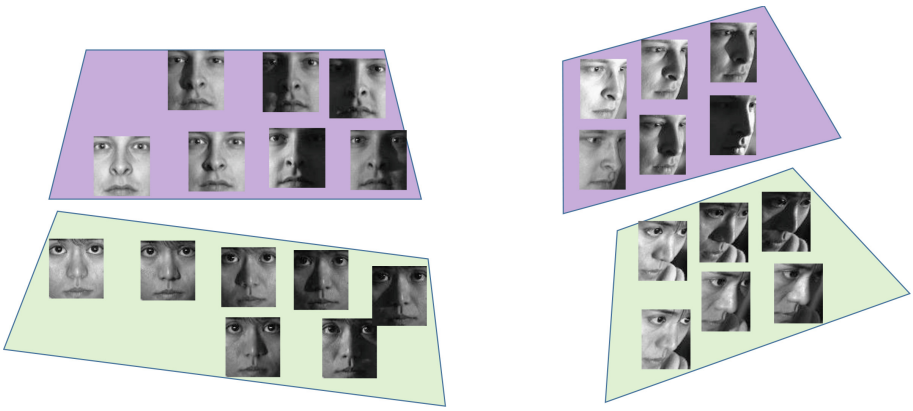

Fig. 1. An illustration of the discriminative subspace structure that is naturally exhibited by the *controlled* images of a visual object (*e.g.* a person's face) [20,21]. The example illustrates the property for the face images of two different subjects, taken under two different poses and varying illumination. The images in which the visual object (i.e. face) has the same pose and identity lead to raw intensity vectors that lie close to a low-dimensional subspace regardless of the variations in Lambertian illumination. Our goal is to learn a nonlinear embedding that improves the discriminative subspace layout of image sets and consequently enhance the performance of subspace-based image set classifiers.

Despite the theoretical foundations of the subspace model, the success of the associated algorithms relies on how well these assumptions are satisfied in practice (i.e. the convexity of the imaged object, the fixing of viewpoint, the Lambertian illumination, and the use of raw intensities to represent images). In practical unconstrained settings, these requirements may not be met and so the data may not strictly follow the subspace model in such scenarios (*e.g.* varying pose and/or use of image features nonlinearly derived from intensities).

To mitigate this, we propose an algorithm to learn a nonlinear embedding that enhances the low-dimensional discriminative subspace structure of the image sets. Under such an embedding, an instance from one class is more likely

to be closer to the subspace spanned by the samples of the same class than to the subspaces spanned by the samples from other classes. This can enhance the performance of subspace-based classifiers, such as Sparse-Representation-based Classification (SRC) [21], which essentially finds a low-dimensional subspace that is closest to the test sample and uses the labels in that subspace to decide a label for the particular test sample. Given a batch of samples, we formulate a novel structured loss function that encourages the distance between each sample and the subspace spanned by the same-class samples (within the batch) to be lower than the distances between the sample and the subspaces spanned by other classes (present in the batch). We then present a two-step alternating optimization algorithm to minimize the loss function in a way that is compatible with back-propagation. This allows the function to be minimized with Stochastic Gradient Descent (SGD)-based algorithms that are typically used to train deep networks [22,23]. At the end of training, the learned embedding is used to project the image sets and the Mean-Sequence SRC (MS-SRC) [10] is used to classify the test image sets.

The rest of this paper is organized as follows. A brief review of related work is presented in Sect. 2. We then describe in Sect. 3 the structured loss function and the optimization procedure of the NSFE algorithm. We experimentally evaluate NSFE in Sect. 4 where the results show the superiority of NSFE compared to several existing image set classification methods. We conclude the paper in Sect. 5.

2 Related Work

The image set classification problem has been formulated in various ways. One popular formulation is to compute the distance, either over a vector space or a manifold, between the probe set and each gallery set and then associate the probe with the class of its nearest gallery set. These include discriminative [2,5,8, 16,18,19,24,25] and non-discriminative methods [1,3,4,7,11,12]. There are also other formulations that do not rely on nearest neighbor-based classification such as the binary SVM reverse-training approach of [15], the neural network-based methods [14,17], linear representation/coding methods [10,26] and clustering methods [6,13]. In what follows, we give a brief description of these methods.

Vector Space Methods: Several methods treat the whole image set as a subspace and measure the distance between subspaces by finding the pair of closest points inside them. Such methods include Affine (or Convex) Hull Image Set Distance (AHISD/CHISD) [3], Sparse-Approximated Nearest Points (SANP) [4], and Dual Linear Regression-based Classification (DLRC) [12]. The Sparse-Approximated Nearest Subspaces (SANS) [11] applies sparse coding to subspace-cluster each gallery image set and measures the distance from the gallery set to the probe set by finding the average distance of each cluster in the gallery set to its nearest subspace approximation from the probe set. Dictionary-based Face Recognition from Videos (DFRV) [7] learns a dictionary consisting of K sub-dictionaries for each gallery image set after clustering its images by appearance

into K groups. The probe set is associated with the class whose gallery dictionaries result in the lowest reconstruction error for the majority of the images in the probe set. Simultaneous Feature and Dictionary Learning (SFDL) [16] discriminatively learns dictionaries for the different classes in addition to learning a linear projection $\mathbf{W}$ to improve the separation between the instances of the different classes. The classification algorithm is identical to DFRV except that the probe images are first transformed using $\mathbf{W}$. Hierarchical subspace clustering of the combined set of faces of the gallery and the probe has been proposed using either sparse codes [6] or Grassmann manifolds [13]. The probe set is associated with the class with the most similar distribution of elements over the clusters to the distribution of the probe elements.

Manifold Methods: Another approach is to represent the image sets as manifolds (or points on a manifold) and use the distance $d(\mathcal{P}, \mathcal{G})$ between the probe $\mathcal{P}$ and each gallery set $\mathcal{G}$ to label $\mathcal{P}$. Methods based on this general idea differ on how they represent an image set as/on a manifold and the way the distance between the sets is measured. Examples of methods that represent each image set as a separate manifold include the Manifold-Manifold Distance (MMD) method [1] and the Manifold Discriminant Analysis (MDA) method [2]. Other manifold methods have represented the subspace approximately spanning an image set as a point on a Grassmann manifold (as opposed to representing each set as a separate manifold). Kernels for Grassmann manifolds are then utilized to perform Discriminant Analysis (DA) [24] or graph-based DA [5] and the distances in the embedded space are used for classification. Kernel dictionary learning and sparse coding on Grassmann manifold have also been considered for image set classification [9]. Instead of using kernels, Projection Metric Learning (PML) [25] discriminatively learns a mapping into another, lower dimensional Grassmann manifold where the projection distance between a pair of points is used for nearest neighbor classification. Covariance Discriminant Learning (CDL) [8] treats the covariance of the image set as a point on a Riemannian manifold that is mapped to a Euclidean space via the logarithmic map. Partial Least Squares (PLS) is then used to learn the mapping from the gallery points to their labels and the resulting mapping is used to classify the probe point. Another related method learns a discriminative, geometry-preserving Mahalanobis metric over the logarithm of the mean-modified covariance matrices and is shown to outperform CDL in [18]. Discriminant Analysis on the Riemannian manifold of Gaussian distributions (DARG) models each image set as a Mixture of Gaussians (MoG) and then runs kernel discriminant analysis based on a combined kernel for Gaussians [19]. Kernel Density Estimation (KDE) has also been used to model image sets as probability density functions in [27] where kernel Fisher discriminant analysis is subsequently applied on the statistical manifold.

Neural Network Methods: With recent successes of deep networks in many vision tasks, different neural network architectures have been recently proposed for image set classification. Two such examples are the generative, per-class five-layer model proposed in [28] and the discriminative, per-class two-layer model

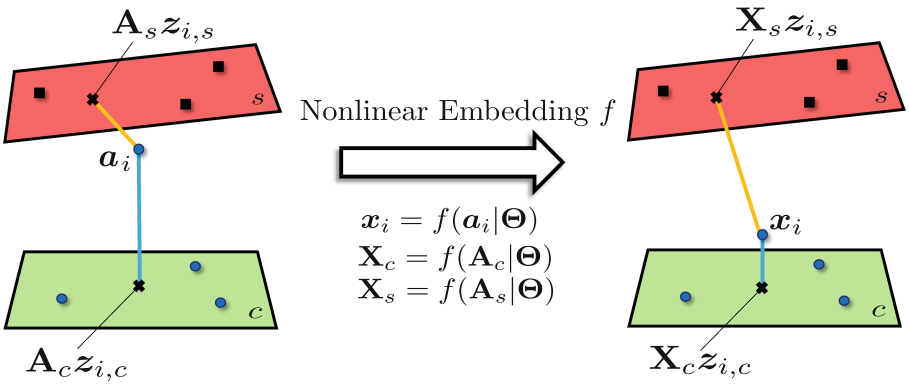

Fig. 2. An illustration of images and class-specific subspaces before and after the embedding. NSFE aims to improve the discriminative subspace arrangement of the data such that the images of a particular class c lie closer to the subspace $\mathbf{X}_c = f(\mathbf{A}_c)$ spanned by that class than any subspace $\mathbf{X}_s = f(\mathbf{A}_s)$ spanned by any other class s.

proposed in [17]. As we describe in Sects. 3.3 and 4, the two example embeddings we train with NSFE in this work are also based on neural networks.

Linear Representation (Coding) Methods: An effective approach, proposed in [21], for utilizing the subspace assumption for recognizing a given feature vector is to first compute its linear representation with respect to the gallery samples (i.e. project it on the gallery) then associate it with the class contributing the most to the representation. SRC [21], proposed for recognition of still face images, adopts this idea and casts the recognition problem as that of solving a convex Lasso optimization for the representation of the probe instance with respect to the gallery. It was then shown that replacing the l_1 regularization term with an l_2 term can yield similar performance with less processing time [29], resulting in the CRC method. Methods utilizing SRC and CRC for image set classification such as the Mean Sequence SRC (MS-SRC) [10] and Image Set CRC (ISCRC) [30] have also been developed.

Our goal in this paper is to learn nonlinear (or deep) features that can improve the performance of subspace-based classifiers like SRC. While some methods have been proposed previously with similar goals [16,31–33], they have been restricted to learning linear embeddings. In contrast, the proposed learning algorithm can be used with any embedding $\boldsymbol{x} = f(\boldsymbol{a}|\boldsymbol{\Theta})$, including deep ones, as long as the parameter subgradients $\boldsymbol{\partial} f / \boldsymbol{\partial \Theta}$ are defined.

3 Nonlinear Subspace Feature Enhancement (NSFE)

We assume that there is a mapping $f : \mathcal{A} \rightarrow \mathbb{R}^m$ that maps every input image $\boldsymbol{a}$ from the vector space $\mathcal{A}$ (i.e. the space of raw intensity images) to $\boldsymbol{x} = f(\boldsymbol{a})$ in some feature space $\mathbb{R}^m$. We further assume that the mapping f is parameterized

by a real tensor $\boldsymbol{\Theta}$ and that the parameter subgradients of $f : \boldsymbol{\partial} f / \boldsymbol{\partial \Theta}$ are defined. For example, the mapping f could be a neural network and $\boldsymbol{\Theta}$ could be the network weights. Assuming that during training labeled samples arrive in batches, our goal is to learn a value of the parameter tensor $\boldsymbol{\Theta}$ that would make an embedded sample $\boldsymbol{x}$ from a particular class c closer to the subspace spanned by batch samples from c than to any subspaces spanned by batch samples from any other class $s \neq c$.

Definitions and Notations: In the following discussion, we use $\mathbf{B}$ to denote the current batch of samples and $|\mathbf{B}|$ to denote the number of samples in the batch. Furthermore, we use n_c to denote the number of samples from class c present in batch $\mathbf{B}$ while $\mathbf{X}_c = \left[\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{n_c}\right] \in \mathbb{R}^{m \times n_c}$ is the matrix (dictionary) containing these samples along its columns. We use $C(\mathbf{B})$ to denote the set of class indices present in $\mathbf{B}$. In all our experiments, we sample each batch to contain nearly the same number of samples n_c from each class (the maximum difference between n_c and n_s is 1 for $c, s \in C(\mathbf{B})$). The sampling procedure ignores the boundaries between sets belonging to the same class and thus the subset drawn from a given class can contain samples from different sets within that class. In subsequent derivations, we assume $n_c > 1$ for all $c \in C(\mathbf{B})$ although in our experiments we have $6 \leq n_c \leq 20$. We also assume that the ith coordinate of a vector $\boldsymbol{z}$ is given by $\boldsymbol{z}^{(i)}$, the (i,j)th entry of a matrix $\mathbf{J}$ is given by $\mathbf{J}^{(i,j)}$, and the ith column is given by $\mathrm{col}_i(\mathbf{J})$.

3.1 Structured Loss Function

Before describing the loss function to be minimized, we need to formulate some measures of distance between a sample and different subspaces. Assuming the ith sample $\boldsymbol{x}_i$ is associated with class c (i.e. $c(i) = c$), we let $\boldsymbol{z}_{i,c}$ denote the linear representation of $\boldsymbol{x}_i$ with respect to the dictionary $\mathbf{X}_c$ (which is formed by the batch samples of class c present in $\mathbf{B}$). The representation $\boldsymbol{z}_{i,c}$ is obtained by solving the optimization problem

$$\boldsymbol{z}_{i,c} = \underset{\boldsymbol{z} \in \mathbb{R}^{n_c}}{\operatorname{argmin}} \left\|\boldsymbol{x}_i - \mathbf{X}_c \boldsymbol{z}\right\|_2^2 + \lambda \left\|\boldsymbol{z}\right\|_2^2, \text{s.t. } z^{(i)} = 0 \tag{1}$$

where we use l_2-norm instead of the sparsity inducing l_1-norm for efficiency purposes and also because n_c is typically small. It can be shown that

$$\boldsymbol{z}_{i,c} = \boldsymbol{u}_{i,c} - w_i \mathrm{col}_i(\mathbf{J}_c^{-1}) \tag{2}$$

where $\mathbf{J}_c = \mathbf{X}_c^T \mathbf{X}_c + \lambda \mathbf{I}$, $\boldsymbol{u}_{i,c} = \mathbf{J}_c^{-1} \mathbf{X}_c^T \boldsymbol{z}_{i,c}$, and $w_i = \boldsymbol{u}_{i,c}^{(i)} / \mathbf{J}_c^{-1(i,i)}$. Similarly, we define the linear representation $\boldsymbol{z}_{i,s}$ of the sample $\boldsymbol{x}_i$ with respect to the dictionary $\mathbf{X}_s$ formed by the batch samples of a different class $s \neq c = c(i)$ as a solution to the following optimization problem

$$\boldsymbol{z}_{i,s} = \underset{\boldsymbol{z} \in \mathbb{R}^{n_s}}{\operatorname{argmin}} \left\|\boldsymbol{x}_i - \mathbf{X}_s \boldsymbol{z}\right\|_2^2 + \lambda \left\|\boldsymbol{z}\right\|_2^2 \tag{3}$$

which has the closed form

$$\boldsymbol{z}_{i,s} = \mathbf{J}_s^{-1}\mathbf{X}_s^T\boldsymbol{x}_i \tag{4}$$

Our goal is to learn the embedding f such that we have

$$\|\boldsymbol{x}_i - \mathbf{X}_c\boldsymbol{z}_{i,c}\|_2^2 < \|\boldsymbol{x}_i - \mathbf{X}_s\boldsymbol{z}_{i,s}\|_2^2 \tag{5}$$

for all valid choices i, c, and s (Fig. 2). If such a discriminative subspace property is achieved for all choices of $c, s, \mathbf{X}_c$, and $\mathbf{X}_s$, a test sample $f(\boldsymbol{q})$ can be reconstructed using the samples of the true class more accurately compared to the samples of other classes. Applying a subspace classifier (like SRC) is thus more likely to associate $f(\boldsymbol{q})$ with its true class.

The proposed structured loss function, which we call Large-Margin Subspace Loss (LMSL), considers for every valid sample-to-subspaces-based triplet within the batch how well (5) is met. More specifically, LMSL is defined as

$$L = \frac{1}{T}\sum_{c\in C(\mathbf{B})}\sum_{\substack{i=1,\\ c(i)=c}}^{|\mathbf{B}|}\sum_{\substack{s\in C(\mathbf{B}),\\ s\neq c}}\Big[\|\boldsymbol{x}_i - \mathbf{X}_c\boldsymbol{z}_{i,c}\|_2^2 + m - \|\boldsymbol{x}_i - \mathbf{X}_s\boldsymbol{z}_{i,s}\|_2^2\Big]_+ \tag{6}$$

where m is the margin and the above sum is normalized by the number of terms/triplets T included the sum, which is $T = |\mathbf{B}|\,(|C(\mathbf{B})| - 1)$. It should be noted that the actual objective function being minimized is the sum of L and any other parameter regularization on $\boldsymbol{\Theta}$. LMSL can be thought of a kind of sample-to-subspace triplet loss [34,35]. The loss function treats as an anchor every sample $\boldsymbol{x}_i$ in the batch $\mathbf{B}$. For each anchor $\boldsymbol{x}_i$, LMSL considers as its corresponding positive point the class projection $\mathbf{X}_c\boldsymbol{z}_{i,c}$ and as a negative point its projection on one of the other-class subspaces $\mathbf{X}_s\boldsymbol{z}_{i,s}$. Thus, we have a total of $|C(B)| - 1$ triplets that have the sample $\boldsymbol{x}_i$ as the anchor.

3.2 Learning Algorithm

The LMSL function L can be difficult to optimize jointly with respect to both the sparse codes and $\boldsymbol{\Theta}$. Accordingly, we follow an alternating optimization approach. In this approach, we evaluate the sparse codes of all batch anchors using Eqs. (2, 4). Then, we treat the sparse codes as constants and use the chain rule and back-propagation to compute the parameter gradients of the loss function $\frac{\partial L}{\partial \theta_k}$, which are necessary for updating $\boldsymbol{\Theta}$ (see Fig. 3):

$$\frac{\partial L}{\partial \theta_k} = \sum_{b=1}^{|\mathbf{B}|}\left(\frac{\partial L}{\partial \boldsymbol{x}_b}\right)^T\frac{\partial \boldsymbol{x}_b}{\partial \theta_k} \tag{7}$$

If we assume $\boldsymbol{x}_b$ is associated with class s, b is its index within the batch, and r is its column index within $\mathbf{X}_s$, the left factor $\frac{\partial L}{\partial \boldsymbol{x}_b}$ in the above inner product is given by:

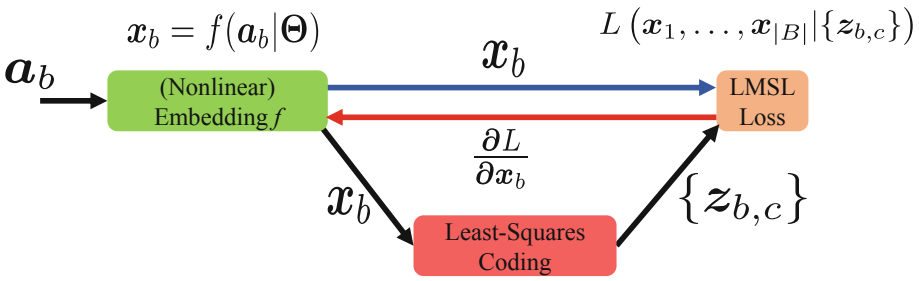

Fig. 3. An illustration of the alternating learning algorithm. After embedding the samples in the forward pass, the sparse codes $\boldsymbol{z}_{b,c}$ are computed $\forall(b,c)$ and substituted into the loss function. The sparse codes are held constant, the loss function is evaluated, and the derivatives of loss function with respect to $\boldsymbol{x}_b, \forall b$ are back-propagated. The chain rule (7) is then applied to evaluate the parameter subgradients $\boldsymbol{\partial} L/\boldsymbol{\partial}\theta_k$ of the loss function, which can then be used to update the parameters by an SGD-like algorithm.

$$\begin{aligned}\frac{\boldsymbol{\partial} L}{\boldsymbol{\partial} \boldsymbol{x}_b} = \frac{2}{T} \sum_{c \in C(\mathbf{B}), c \neq s} \{ & \Delta_{b,c} \left(\boldsymbol{x}_b - \mathbf{X}_s \boldsymbol{z}_{b,s} \right) \\ & - \Delta_{j,c} \sum_{j=1, c(j)=s, j \neq b}^{|\mathbf{B}|} \boldsymbol{z}_{j,s}^{(r)} \left(\boldsymbol{x}_j - \mathbf{X}_s \boldsymbol{z}_{j,s} \right) \} \\ & + \frac{2}{T} \sum_{i=1, c(i) \neq s}^{|\mathbf{B}|} \Delta_{i,s} \boldsymbol{z}_{i,s}^{(r)} \left(\boldsymbol{x}_i - \mathbf{X}_s \boldsymbol{z}_{i,s} \right) \end{aligned} \tag{8}$$

where $\Delta_{i,s}$ is a binary variable that is 1 iff the loss term corresponding to anchor sample i and negative class s is non-zero. The loss gradient in (8) is computed for each sample $\boldsymbol{x}_b$ in the batch and back-propagated for computing parameter updates. A summary of the learning algorithm of NSFE is given in Algorithm 1.

Input: A batch of samples $[\boldsymbol{a}_1, ..., \boldsymbol{a}_{|\mathbf{B}|}]$ and their labels.

1 Group batch samples by class.

2 Embed and l_2-normalize each sample in the batch: $\boldsymbol{x}_b = f(\boldsymbol{a}_b|\boldsymbol{\Theta}_t)$.

3 For each class $c \in C(\mathbf{B})$, use Cholesky-Factorization to invert $\mathbf{J}_c = \mathbf{X}_c^T \mathbf{X}_c + \lambda \mathbf{I}$.

4 For each class $c \in C(\mathbf{B})$, use Eq. (2) to compute the code vector of its batch samples with respect to $\mathbf{X}_c$.

5 For each class $c \in C(\mathbf{B})$, use Eq. (4) to compute the code vector of other-class samples in the batch with respect to $\mathbf{X}_c$.

6 Compute the LMSL loss L using Eq. (6). Compute and back-propagate the LMSL gradient $\frac{\partial L}{\partial x_b}$, for $b = 1, \ldots, |\mathbf{B}|$.

7 Use the chain rule and Eq. (7) to compute the loss gradients $\frac{\partial L}{\partial \theta_k}$ of the parameters which can then be used to update these parameters.

Algorithm 1. NSFE Learning Algorithm Summary

3.3 Concrete Embeddings

Our method can work with any vector-space inputs and can easily utilize any nonlinear embeddings for which the parameter subgradients of $f : \boldsymbol{\partial} f / \boldsymbol{\partial \Theta}$ are defined, including feed-forward neural networks. We test the proposed method with two types of vector-space inputs: raw intensity images and the hand-crafted Log-Euclidean Grid of Region Covariance Matrices (LE-GRCM) features proposed in [33]. With intensity images as inputs, we use the 32-layer deep fully convolutional residual network proposed in [36] for the CIFAR-10 dataset. The network has the following configuration:

(a) An initial $3 \times 3 \times 16$ convolutional layer. The notation specifies 16 filters, each has a weight kernel of dimensions 3×3. The stride is always 1 in both directions.
(b) A first block of ten $3 \times 3 \times 16$ convolutional layers, with residual connections made every two layers. The last layer is followed by a 2×2 average pooling with a stride of 2 in both directions.
(c) A second block of ten $3 \times 3 \times 32$ convolutional layers. Residual connections and a final average pooling are defined for this block.
(d) A third block of ten $3 \times 3 \times 64$ convolutional layers. It uses residual connections in a similar fashion but does not have a subsequent pooling layer.
(e) A final $1 \times 1 \times 10$ convolutional layer that is not followed by any nonlinearities or batch normalization [37]. The output of that layer is reshaped as a vector and l_2-normalized to produce the final embedded feature vector.

The final layer replaces the global average pooling operation used in [36] in an attempt to retain spatial information in the computed features. Unless otherwise stated, we add batch normalization and ReLU nonlinearities according to the architecture in [36]. The total number of parameters in this architecture is 463,856, which is less than 0.5 million.

Since an LE-GRCM vector input is not a 2D image, we cannot use a conventional CNN for the embedding to process such hand-crafted features. Instead, we use a very basic, fully-connected 2-layer network with the following architecture: FC-3600 $\rightarrow$ ReLU $\rightarrow$ FC-406 $\rightarrow$ l_2-normalization, where FC-k is a linear fully-connected layer with k units.

3.4 Classification

After training, the learned embedding f is used to map the training data then we use the Online Dictionary Learning (ODL) algorithm described in [38] to compute a dictionary $\mathbf{D}_c$ for each class c. Given a test set, we use the learned embedding f to map it and we follow the MS-SRC approach [10] by computing the mean vector $\bar{\boldsymbol{y}}$ of the embedded test set and then using SRC to find a label for $\bar{\boldsymbol{y}}$. The details of ODL and MS-SRC algorithms can be found in [38] and [10], respectively.

It is worth noting that ODL is an unsupervised algorithm and enhanced performance can be further achieved by using any of the discriminative dictionary learning algorithms instead of ODL. However, we only use ODL in the next section to objectively and more precisely evaluate the effect of NSFE on accuracy.

4 Experiments

We experimentally compare the performance of NSFE against several existing algorithms for image-set classification. The compared methods include Affine Hull-based Image Set Distance (AHISD) [3], its convex variant (CHISD) [3], Sparse-Approximated Nearest Points (SANP) [4], Dictionary-based Face Recognition from Videos (DFRV) [7], Mean Sequence Sparse Representation-based Classification (MS-SRC) [10], Set to Set Distance Metric Learning (SSDML) [26], Deep Reconstruction Models (DRM) [14], Covariance Discriminative Learning (CDL) [8], Log-Euclidean Metric Learning (LEML) [18], and the shallow subspace Feature Learning+SRC (FL+SRC) approach of [33] both with intensity images as inputs (FL+SRC) as well as LE-GRCM features (LE-FL+SRC). We show the results of our method with both intensity features as inputs (NSFE) and LE-GRCM features (LE-NSFE). For comparability, the results of other Log-Euclidean methods (i.e. CDL and LEML) are obtained using LE-GRCM features.

In all experiments, each method is given a set of labeled image sets for training and is required to classify (or more specifically identify) a number of test image sets. For performance comparison, we use the classification accuracy (i.e. recognition rate) as a metric by measuring the percentage of test image sets that are correctly classified.

For existing methods, we have used the source code provided by the original authors and set the parameters according to the recommendations made in their respective papers.

NSFE Parameter Settings: In all experiments, we use SGD with momentum to update the weights of the embedding network in each iteration for a total of 50 K iterations. The momentum is set to 0.9 and we use a learning rate schedule of 0.1 for the first 20 K iterations then we divide it by 10 for each subsequent 10 K iterations. For the 2-layer fully-connected network, we train for 20 K iterations with a learning rate of 0.01 that we decrease to 0.001 after 10 K iterations. We use a batch of size 128. We also set the representation regularization parameter λ of NSFE to 0.01, the margin $m = 0.5$, and the desired number of atoms in each class-specific dictionary computed by ODL to 50.

To guarantee a fair comparison with other methods and to accurately measure the ability of our method to learn effective features, we do not perform any pre-training on any external data and we initialize the weights of our embeddings randomly.

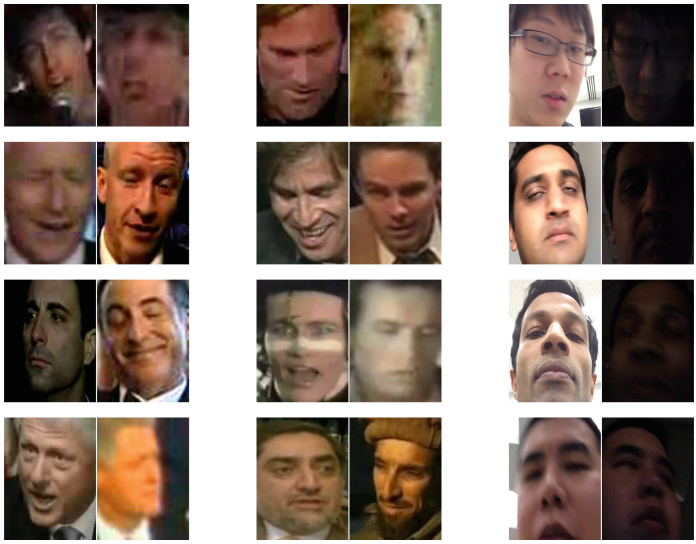

Fig. 4. Sample face pairs from YTC, YTF and MobFaces. Each pair of faces in each column belong to the same subject. YTC and YTF photos reveal large intra-class appearance variations and low resolution. MobFaces photos are relatively frontal but they reveal some challenges such as blur and intra-class variations in illumination and context due to the change in sessions.

The datasets used in our experiments are described below. Figure 4 shows examples from each dataset.

4.1 YouTube Celebrities (YTC)

The YTC dataset contains 1,910 YouTube-downloaded videos of 47 subjects [39]. For a given subject, the videos are short segments clipped from three longer, parent videos downloaded from YouTube. YTC has been built to be very challenging for face tracking and recognition by choosing very low resolution videos with wild variations in pose, scale, hair style, make-up, illumination, motion and number of people per frame.

We perform ten-fold cross-validation experiment. Each fold contains nine distinct videos for each subject: three for training and six for testing, randomly drawn in the same manner of previous works [28,33]. The $9 \times 47 = 423$ videos in each fold are randomly selected from the complete dataset while minimizing the overlap between different folds as much as possible.

Feature Extraction: We use the Viola-Jones (VJ) detector [40] as in prior works [18,33] to locate the faces in each video. Then we use the eye locations

detected using the method of [41] to align the subject's face to a standard, 30×36 pixel frame. The intensities of each frame are histogram equalized and we use the faces detected in each video to define the corresponding image set. To test the robustness of the compared methods to outliers, we have not cleaned any of the bad detections or misaligned faces.

4.2 YouTube Faces (YTF)

The YTF dataset contains 3,425 videos of 1,595 subjects with diverse ethnicities [42]. Similar to YTC, YTF videos are downloaded from YouTube and are very challenging for face recognition. Since our method is used for identification rather verification, we adopt the experimental protocol of [33] which is more suitable for testing identification: We use those subjects with four or more videos available. This results in 226 subjects. After randomly dropping one subject, we randomly split the remaining 225 subjects into five mutually exclusive groups, with 45 subjects each. We run the experiment on each group where we use the first three videos of each subject as gallery sets and the remaining videos for testing. Since the dataset provides aligned face images, we extract intensity features from each image by cropping the central 100×100 box, resizing it to 30×36, and histogram-equalizing it.

4.3 Mobile Faces (MobFaces)

The MobFaces dataset has 750 videos of 50 subjects recorded by a smartphone's front camera during usage [43]. Each subject provides 3 sessions $\times$ 5 videos/session (one enrollment + four tasks) where each session is taken under a different illumination and/or in a different place. The dataset includes some mobile camera-specific such as wild variations in illumination and context due to the mobility of the device. We compute the features as for the YTC dataset. We adopt the two evaluation protocols suggested by [43] by dividing the task videos into ten-second long clips and treating each clip as a separate query set. In the first protocol (MobFaces-I), training is done using only the 50 enrollment videos from one session and testing is performed on the ten-second long task video clips from the two other sessions. In the second protocol (MobFaces-II), training is done on the 100 enrollment videos of two sessions and testing is done on the task video clips of the remaining session. Results are reported for each of the six scenarios possible with these protocols. The clipping of the 600 task videos results in 1065 ten-second clips for the first session, 587 the second, and 666 for the third. Note that training sets under these protocols contain relatively very few images with limited appearance (*e.g.* some classes have a single image set with nine almost-identical images only available for training) which makes it challenging.

Table 1. The mean recognition rates obtained with the compared methods on YTC and YTF.

Methods	YTC	YTF
AHISD	57.27	17.18
CHISD	64.79	32.99
SANP	66.99	31.62
DFRV	66.70	36.77
SSDML	69.22	34.02
DRM	70.35	43.99
MS-SRC	74.68	45.02
FL+SRC	75.71	45.36
NSFE (ours)	***78.23**	**54.91**
Methods with LE-GRM input		
CDL	67.62	41.92
LEML	73.26	48.45
LE-FL+SRC	76.28	53.26
LE-NSFE (ours)	**76.42**	***56.66**

4.4 Results

Table 1 shows the mean recognition rate of the compared methods for the YTC and YTF datasets where we group the methods by the type of input features (raw images vs LE-GRCM). For each group, we highlight in **bold** the highest performance under each setting and we place an asterisk $*$ next to the single highest overall performance for that setting. For both datasets and types of input features, the proposed method, NSFE/LE-NSFE, achieves the highest mean recognition rate. Table 2 shows the recognition rates for the six different splits for the MobFaces dataset where we use the same grouping and highlighting adopted by Table 1. The training image sets of this dataset contain very limited visual variations (namely once short video per subject for each setting in MobFaces-I and two such videos in MobFaces-II) while the test image sets are captured under ambient conditions different from those of training. Meanwhile, our method (NSFE/LE-NSFE) ranks among the top two best performing methods under each individual setting and it is the best performing on average for each of the two protocols, with a significant margin on MobFaces-II due to the availability of more images and visual variations during training in that protocol. This shows that our method achieves relatively higher gain in performance as more deta and variations become available for training.

Table 2. The recognition rates obtained on the MobFaces dataset under the different protocols. The setting ($1 \rightarrow \{2,3\}$) involves training on session 1 (i.e. the lit session) and testing on sessions 2 and 3 (i.e. the unlit and day-lit sessions). The other five settings are defined in a similar manner. Each 'avg' column contains the average of the rates obtained under the three settings to its left. Since each session has a different number of test video clips, the average column weighs the rate of each setting by its number of test sets.

Methods	MobFaces-I			
	$1 \rightarrow \{2,3\}$	$2 \rightarrow \{1,3\}$	$3 \rightarrow \{1,2\}$	Avg
AHISD	15.00	31.14	29.30	26.12
CHISD	10.61	26.57	25.73	21.96
SANP	9.34	27.09	26.15	21.96
DFRV	19.39	32.29	30.87	28.30
SSDML	10.53	28.89	26.15	22.95
DRM	33.28	38.94	37.95	37.06
MS-SRC	32.40	46.56	42.49	41.29
FL+SRC	32.88	**46.97**	42.25	41.48
NSFE (ours)	**47.01**	46.27	**46.49**	**46.55**
Methods with LE-GRM input				
CDL	41.66	36.78	42.68	40.21
LEML	42.70	45.93	44.07	44.39
LE-FL+SRC	48.20	***56.21**	54.90	53.58
LE-NSFE (ours)	***49.08**	49.68	***61.38**	***53.69**
Methods	MobFaces-II			
	$\{2,3\} \rightarrow 1$	$\{1,3\} \rightarrow 2$	$\{1,2\} \rightarrow 3$	Avg
AHISD	24.41	51.28	52.85	39.39
CHISD	23.29	44.97	47.60	35.76
SANP	20.38	48.89	45.95	34.94
DFRV	32.11	50.60	52.40	42.62
SSDML	21.31	50.09	54.95	38.27
DRM	**53.62**	70.53	69.37	62.42
MS-SRC	43.29	71.89	75.53	59.79
FL+SRC	44.98	72.40	76.58	61.00
NSFE (ours)	52.11	***81.43**	**83.63**	**68.59**
Methods with LE-GRM input				
CDL	63.57	67.12	65.32	64.97
LEML	49.39	66.95	74.62	61.09
LE-FL+SRC	62.72	75.64	86.19	72.74
LE-NSFE (ours)	***68.92**	**76.15**	***87.84**	***76.19**

5 Conclusion

We presented NSFE, an approach for discriminatively learning a nonlinear embedding that can improve the subspace structured representation of image sets, and thus improve the performance of subspace-based classifiers such as MS-SRC. Since the proposed structured loss function LMSL is minimized in an online fashion, the proposed approach can be used to train existing feed-forward architectures via back-propagation. The minimization algorithm can also utilize the capabilities of modern GPUs, which provide APIs for solving batches of small linear systems of equations. In fact, all the linear systems solved in our batch processing algorithm are small, ranging from 6×6 to 22×22 systems of equations, depending on the number of samples from a certain class available in the batch. Consequently, we were able to train and test many copies of our model for the different experiments described above without facing any unusual delays.

Acknowledgment. This research is based upon work supported by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via IARPA R&D Contract No. 2014-14071600012. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon.

References

1. Wang, R., Shan, S., Chen, X., Gao, W.: Manifold-manifold distance with application to face recognition based on image set. In: CVPR, pp. 1–8 (2008)
2. Wang, R., Chen, X.: Manifold discriminant analysis. In: CVPR, pp. 429–436 (2009)
3. Cevikalp, H., Triggs, B.: Face recognition based on image sets. In: CVPR, pp. 2567–2573 (2010)
4. Hu, Y., Mian, A.S., Owens, R.: Sparse approximated nearest points for image set classification. In: CVPR, pp. 121–128 (2011)
5. Harandi, M.T., Sanderson, C., Shirazi, S., Lovell, B.C.: Graph embedding discriminant analysis on Grassmannian manifolds for improved image set matching. In: CVPR, pp. 2705–2712 (2011)
6. Mahmood, A., Mian, A.: Hierarchical sparse spectral clustering for image set classification. In: BMVC, pp. 1–11 (2012)
7. Chen, Y.-C., Patel, V.M., Phillips, P.J., Chellappa, R.: Dictionary-based face recognition from video. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012. LNCS, vol. 7577, pp. 766–779. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33783-3_55
8. Wang, R., Guo, H., Davis, L.S., Dai, Q.: Covariance discriminative learning: a natural and efficient approach to image set classification. In: CVPR, pp. 2496–2503 (2012)
9. Harandi, M., Sanderson, C., Shen, C., Lovell, B.C.: Dictionary learning and sparse coding on Grassmann manifolds: an extrinsic solution. In: ICCV, pp. 3120–3127 (2013)

10. Ortiz, E.G., Wright, A., Shah, M.: Face recognition in movie trailers via mean sequence sparse representation-based classification. In: CVPR, pp. 3531–3538 (2013)
11. Chen, S., Sanderson, C., Harandi, M.T., Lovell, B.C.: Improved image set classification via joint sparse approximated nearest subspaces. In: CVPR, pp. 452–459 (2013)
12. Chen, L.: Dual linear regression based classification for face cluster recognition. In: CVPR, pp. 2673–2680 (2014)
13. Mahmood, A., Mian, A., Owens, R.: Semi-supervised spectral clustering for image set classification. In: CVPR, pp. 121–128 (2014)
14. Hayat, M., Bennamoun, M., An, S.: Learning non-linear reconstruction models for image set classification. In: CVPR, pp. 1915–1922 (2014)
15. Hayat, M., Bennamoun, M., An, S.: Reverse training: an efficient approach for image set classification. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8694, pp. 784–799. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10599-4_50
16. Lu, J., Wang, G., Deng, W., Moulin, P.: Simultaneous feature and dictionary learning for image set based face recognition. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8689, pp. 265–280. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10590-1_18
17. Lu, J., Wang, G., Deng, W., Moulin, P., Zhou, J.: Multi-manifold deep metric learning for image set classification. In: CVPR (2015) 1137–1145
18. Huang, Z., Wang, R., Shan, S., Li, X., Chen, X.: Log-Euclidean metric learning on symmetric positive definite manifold with application to image set classification. In: ICML, pp. 720–729 (2015)
19. Wang, W., Wang, R., Huang, Z., Shan, S., Chen, X.: Discriminant analysis on Riemannian manifold of Gaussian distributions for face recognition with image sets. In: CVPR, pp. 2048–2057 (2015)
20. Basri, R., Jacobs, D.W.: Lambertian reflectance and linear subspaces. PAMI **25**, 218–233 (2003)
21. Wright, J., Yang, A.Y., Ganesh, A., Sastry, S.S., Ma, Y.: Robust face recognition via sparse representation. PAMI **31**, 210–227 (2009)
22. Duchi, J., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. JMLR **12**, 2121–2159 (2011)
23. Sutskever, I., Martens, J., Dahl, G.E., Hinton, G.E.: On the importance of initialization and momentum in deep learning. In: ICML, pp. 1139–1147 (2013)
24. Hamm, J., Lee, D.D.: Grassmann discriminant analysis: a unifying view on subspace-based learning. In: ICML, pp. 376–383 (2008)
25. Huang, Z., Wang, R., Shan, S., Chen, X.: Projection metric learning on Grassmann manifold with application to video based face recognition. In: CVPR, pp. 140–149 (2015)
26. Zhu, P., Zhang, L., Zuo, W., Zhang, D.: From point to set: extend the learning of distance metrics. In: ICCV, pp. 2664–2671 (2013)
27. Harandi, M., Salzmann, M., Baktashmotlagh, M.: Beyond Gauss: image-set matching on the Riemannian manifold of PDFs. In: ICCV, pp. 4112–4120 (2015)
28. Hayat, M., Bennamoun, M., An, S.: Deep reconstruction models for image set classification. PAMI **37**, 713–727 (2015)
29. Zhang, L., Yang, M., Feng, X.: Sparse representation or collaborative representation: which helps face recognition? In: ICCV, pp. 471–478 (2011)

30. Zhu, P., Zuo, W., Zhang, L., Shiu, S.C.K., Zhang, D.: Image set-based collaborative representation for face recognition. IEEE Trans. Inf. Forens. Secur. **9**, 1120–1132 (2014)
31. Zhang, H., Zhang, Y., Huang, T.S.: Simultaneous discriminative projection and dictionary learning for sparse representation based classification. Pattern Recogn. **46**, 346–354 (2013)
32. Qiu, Q., Sapiro, G.: Learning transformations for clustering and classification. JMLR **16**, 187–225 (2015)
33. Fathy, M.E., Alavi, A., Chellappa, R.: Discriminative Log-Euclidean feature learning for sparse representation-based recognition of faces from videos, pp. 3359–3367 (2016)
34. Weinberger, K.Q., Blitzer, J., Saul, L.K.: Distance metric learning for large margin nearest neighbor classification. In: NIPS, pp. 1473–1480 (2005)
35. Schroff, F., Kalenichenko, D., Philbin, J.: FaceNet: a unified embedding for face recognition and clustering. In: CVPR, pp. 815–823 (2015)
36. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR, pp. 770–778 (2016)
37. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. In: ICML, pp. 448–456 (2015)
38. Mairal, J., Bach, F., Ponce, J., Sapiro, G.: Online dictionary learning for sparse coding. In: ICML, pp. 689–696 (2009)
39. Kim, M., Kumar, S., Pavlovic, V., Rowley, H.: Face tracking and recognition with visual constraints in real-world videos. In: CVPR, pp. 1–8 (2008)
40. Viola, P., Jones, M.J.: Robust real-time face detection. IJCV **57**, 137–154 (2004)
41. Asthana, A., Zafeiriou, S., Cheng, S., Pantic, M.: Robust discriminative response map fitting with constrained local models. In: CVPR, pp. 3444–3451 (2013)
42. Wolf, L., Hassner, T., Maoz, I.: Face recognition in unconstrained videos with matched background similarity. In: CVPR, pp. 529–534 (2011)
43. Fathy, M.E., Patel, V.M., Chellappa, R.: Face-based active authentication on mobile devices. In: ICASSP, pp. 1687–1691 (2015)