Deep Reconstruction Models for Image Set Classification

Munawar Hayat, Mohammed Bennamoun and Senjian An

Abstract—Image set classification finds its applications in a number of real-life scenarios such as classification from surveillance videos, multi-view camera networks and personal albums. Compared with single image based classification, it offers more promises and has therefore attracted significant research attention in recent years. Unlike many existing methods which assume images of a set to lie on a certain geometric surface, this paper introduces a deep learning framework which makes no such prior assumptions and can automatically discover the underlying geometric structure. Specifically, a Template Deep Reconstruction Model (TDRM) is defined whose parameters are initialized by performing unsupervised pre-training in a layer-wise fashion using Gaussian Restricted Boltzmann Machines (GRBMs). The initialized TDRM is then separately trained for images of each class and class-specific DRMs are learnt. Based on the minimum reconstruction errors from the learnt class-specific models, three different voting strategies are devised for classification. Extensive experiments are performed to demonstrate the efficacy of the proposed framework for the tasks of face and object recognition from image sets. Experimental results show that the proposed method consistently outperforms the existing state of the art methods.

Index Terms—Image Set Classification, Deep Learning, Auto-Encoders, Video based Face Recognition, Object Recognition

1 Introduction

Recognition problems in computer vision are mainly based on single images [1], [2]. With significant advances in imaging technology, multiple images of a person or an object are becoming readily available in a number of real-life scenarios. Examples include security and surveillance systems, personal albums acquired over a period of time and multi-view camera networks. Recognition from these multiple images is commonly formulated as an image set classification problem and has gained significant attention from the research community in recent years [3]–[13].

Compared with single image based classification, recognition from image sets is more appealing as it can effectively handle a wide range of appearance variations within images. These could be caused by changing illumination conditions, different backgrounds, viewpoint variations, non-rigid deformations, occlusions and disguise. Image set classification methods commonly model the appearance variability information within images of a set on a geometric surface such as an image set modeled by a subspace [3], [15], a combination of subspaces [4], [16], a point on the Grassmannian [7] or Lie Group [9] of Riemannian manifold. This requires prior assumptions in regards to the specific category of the geometric surface on which images of the set are believed to lie. In contrast, this paper introduces a deep learning framework which makes no such prior assumptions regarding the underlying geometry and can instead automatically discover the structure of the complex non-linear surface on which images of the set (under different variations) are present. The proposed framework (see block diagram in Fig. 1) first defines a Template Deep Reconstruction Model (TDRM) whose weights are initialized with an unsupervised layer-wise pre-training using Gaussian Restricted Boltzmann Machines (GRBMs). The initialized TDRM is then separately trained for each class (using all images of that class) to learn class-specific DRMs. The training is performed in a way that the DRM learns to reconstruct images of that class. A class-specific model is therefore made to learn the structure and the geometry of the complex non-linear surface on which images of that class are present. For classification of a given test image set, we first reconstruct each of its images from the learnt class-specific DRMs. The reconstruction errors from the respective DRMs are then computed and three different voting strategies are introduced to decide on the identity of the test image set. The proposed framework is extensively tested for the tasks of image set classification based face recognition on Honda/UCSD [17], CMU Mobo [18], YouTube Celebrities [19], a composite Kinect dataset [20], [21], PubFig [22], a subset of YouTube Faces [23] and COX [24] datasets; and object recognition on ETH-80 [25] dataset. The experimental evaluations and comparisons with existing methods show that the proposed method consistently achieves state of the art performance.

Followings are the major contributions of our work. 1) A novel deep learning based framework is introduced for image set classification (Sec. 3). Deep learning has recently gained significant success in a number of areas [26], [27], but its application to image set classification has not yet been explored. In this work, we propose the first deep learning based image set classification framework. The proposed

• *The authors are with the School of Computer Science and Software Engineering, The University of Western Australia, 35 Stirling Highway, 6009, Crawley, WA, Australia*
E-mail: munawar.hayat@research.uwa.edu.au,
{mohammed.bennamoun, senjian.an}@uwa.edu.au

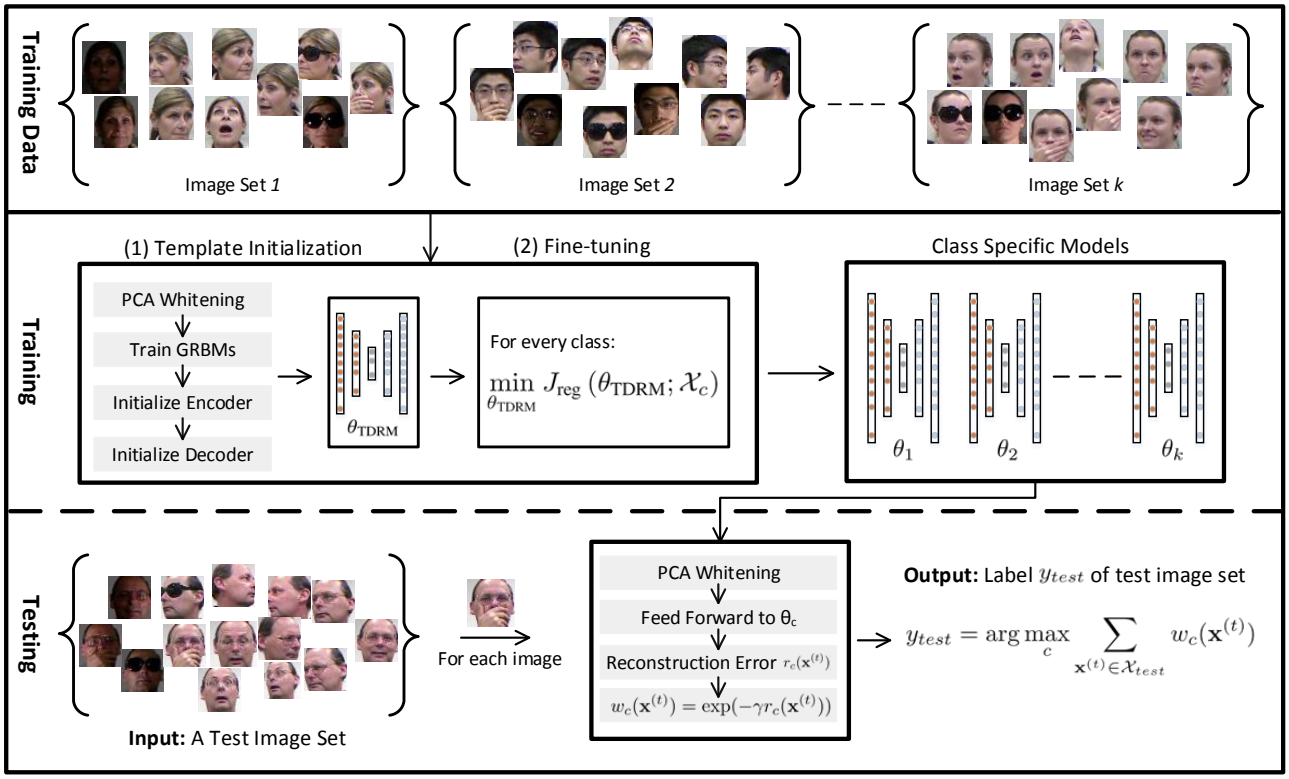


Fig. 1: Block diagram of the proposed Deep Reconstruction Models (DRMs) based image set classification framework. The framework constitutes *training* and *testing*. During *training*, we first define a Template DRM (TDRM) and initialize its parameters by unsupervised pre-training using Gaussian Restricted Boltzmann Machines (GRBMs). The initialized TDRM is then separately trained with training images of each class to learn class-specific DRMs. During *testing*, the learnt DRMs are used to reconstruct images of a test image set and a voting strategy is adopted for classification.

framework learns Deep Reconstruction Models (DRMs) which can automatically discover the underlying geometry of the data. 2) For classification, three different voting strategies are suggested (Sec. 4.4). These include majority voting, weighted voting and preferential weighted voting. These strategies effectively incorporate the reconstruction error information from the DRMs to make a decision regarding the class of a test image set. 3) To further refine the classification performance, a method for automatic pose group approximation is introduced (Sec. 4.4.3). 4) Face recognition from Kinect data is formulated as an RGB-D based image set classification problem (Sec. 5.2.4). Compared with single image based classification, this formulation produces a better performance and does not require any expensive pre-processing steps. 5) In order to evaluate the performance of the proposed framework, extensive experiments are done along with comparisons to existing state of the art methods for image set classification (Sec. 5). The experimental results show that the proposed method achieves the best classification performance and its computational time during testing is comparable or better than the existing methods. A preliminary version of this work appeared in [13]. This paper extends [13] by providing two alternative voting strategies, a method for pose group approximation, experimental evaluation on more real-life datasets and a more detailed description of the work.

2 Related Work

Image set classification involves two major steps: 1) to find a representation of the images in the set, and 2) to define suitable distance metrics for the computation of the similarity between these representations. Based on the used type of representation, existing image set classification methods can be categorized into parametric-model and non-parametric-model methods. The parametric-model methods [28] approximate an image set in terms of a certain statistical distribution model and then measure the similarity between two image sets (two distribution models) using *e. g.* KL-divergence. These methods fail to produce a desirable performance if there is no strong statistical relationship between the test and the training image sets. The other type of image set representation methods (non-parametric methods) do not model image sets in terms of statistical distributions. These methods have shown promising results and are being actively developed recently [3]–[14].

The non-parametric model based methods represent an image set either by its representative exemplars or on a geometric surface. Based upon the type of representation, different distance metrics have been developed to determine the between-set distance. For example, for image sets represented in terms of representative exemplars, the set-set distance can be defined as the Euclidean distance between the set representatives. These can simply be the set mean

[4] or adaptively learnt set samples [6], [8]. Cevikalp *et al.* [6] learn the set samples from the affine hull or convex hull models of the set images. The set to set distance is then termed as Affine Hull Image Set Distance (AHISD) or Convex Hull Image Set Distance (CHISD). Hu *et al.* [8] define the set-set distance as the distance between their Sparse Approximated Nearest Points (SANPs). The SANPs of two sets are first determined from the mean image and the affine hull model of the corresponding sets. The SANPs are then sparsely approximated from the set's sample images while simultaneously searching for the closest points between sets. As set representative based methods require the computation of a one-to-one set distance, these methods are capable of handling intra set variations very effectively. However, their performance is highly prone to outliers. They are also computationally very expensive as a one-to-one match of the query set with all sets in the galley is required. These methods could therefore be very slow in the case of a large gallery.

Unlike set representative based methods, the second category of non-parametric methods model a complete image set by a point on a geometric surface [4], [5], [7], [9] . The image set can be represented either by a subspace, mixture of subspaces or on a complex non-linear manifold. Principal angles have been very commonly used to determine the distance between image sets represented by a linear subspace. The d principal angles $0 \leq \theta_1 \leq \cdots \leq \theta_d \leq \frac{\pi}{2}$ between two subspaces are defined as the smallest angles between any vector in one subspace and any other vector in the second subspace. The similarity between subspaces is then defined as the sum of the cosines of the principal angles. For image set representations on manifolds, appropriate distance metrics have been adopted such as the geodesic distance [29], [30], the projection kernel metric [31] on the Grassmann manifold, and the log-map distance metric [32] on the Lie group of Riemannian manifold. In order to discriminate image sets on the manifold surface, different learning strategies have been developed. Mostly, a discriminant analysis method is contrived for different set representations. Examples include Discriminative Canonical Correlations (DCC) [3], Manifold Discriminant Analysis (MDA) [5], Graph Embedding Discriminant Analysis (GEDA) [7] and Covariance Discriminative Learning (CDL) [9].

Our literature review suggests that most existing methods represent images of a set on some geometric surface. For example, AHISD [6], CHISD [6], SANP [8] and Regularized Nearest Point (RNP) [10] model images of the set by their geometric structure (affine hull or convex hull); DCC [3], Mutual Subspace Method (MSM) [15] model an image set by a subspace; Manifold to Manifold Distance (MMD) [4], MDA [5] and GEDA model an image set as a point on Grassmannian manifold and CDL [9] models an image set on Lie group of Riemannian manifold. These methods therefore make prior assumptions about the underlying geometry on which images of a set are believed to lie. In contrast, our proposed method defines a TDRM which can automatically learn the underlying geometric structure. Our TDRM has multiple layers stacked together through non-linear activation functions. The TDRM therefore becomes capable of discovering the complex geometric surface on which images of a set are present.

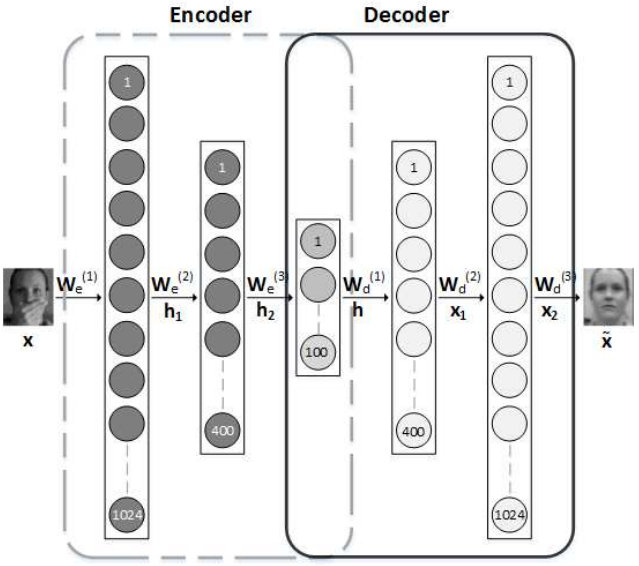


Fig. 2: Structure of the Template Deep Reconstruction Model (TDRM). The TDRM is based on an auto-encoder and has two parts: an encoder and a symmetric decoder. The encoder finds a low dimensional meaningful representation of the input data which is then used by the decoder to reconstruct the original input

3 Deep Reconstruction Models

We first define a Template Deep Reconstruction Model (TDRM) which will be used to learn the underlying structure of the data. The architecture of our TDRM is summarized in Fig 2 and the details are presented in Sec 3.1. For such a deep network to perform well, an appropriate initialization of the weights is required. We initialize the weights of the TDRM by performing pre-training in a greedy layer-wise fashion using Gaussian Restricted Boltzmann Machines (see details in Sec 3.2). The TDRM with the initialized weights is then separately fine-tuned for each of the k classes of the training image sets (see details in Sec. 3.3). We therefore end up with a total of k fine-tuned class-specific Deep Reconstruction Models (DRMs). The fine-tuned models are then used for image set classification (see details in Sec 4).

3.1 The Template Deep Reconstruction Model

As depicted in Fig 2, our TDRM is based on an Auto-Encoder (AE) and consists of two parts: an encoder and a decoder. Both the encoder and the decoder have three hidden layers each, with a shared third layer (the central hidden layer). The encoder part of the TDRM finds a compact low dimensional meaningful representation of the input data. We can formulate the encoder as a combination of layers connected by a non-linear activation function $s(\cdot)$ which maps the input data $\mathbf{x}$ to a representation $\mathbf{h}$ as follows

$$\begin{aligned}\mathbf{h} &= s(\mathbf{W}_e^{(3)}\mathbf{h}_2 + \mathbf{b}_e^{(3)}),\\ \mathbf{h}_2 &= s(\mathbf{W}_e^{(2)}\mathbf{h}_1 + \mathbf{b}_e^{(2)}),\\ \mathbf{h}_1 &= s(\mathbf{W}_e^{(1)}\mathbf{x} + \mathbf{b}_e^{(1)}),\end{aligned} \tag{1}$$

where $\mathbf{W}_e^{(i)} \in \mathbb{R}^{d_{i-1}\times d_i}$ is the encoder weight matrix for layer i with d_i nodes, $\mathbf{b}_e^{(i)} \in \mathbb{R}^{d_i}$ is the bias vector and $s(\cdot)$ is the element-wise non-linear activation function[1]. The encoder parameters are learnt by combining the encoder with the decoder and jointly training the encoder-decoder structure to reconstruct the input data by minimization of a cost function (Sec. 3.3). The decoder can therefore be defined as a combination of layers joined together by a non-linear activation function which reconstruct the input $\mathbf{x}$ from the encoder output $\mathbf{h}$. The reconstructed output $\tilde{\mathbf{x}}$ of the decoder is given by

$$\begin{aligned}\tilde{\mathbf{x}} &= s(\mathbf{W}_d^{(3)}\mathbf{x}_2 + \mathbf{b}_d^{(3)}),\\ \mathbf{x}_2 &= s(\mathbf{W}_d^{(2)}\mathbf{x}_1 + \mathbf{b}_d^{(2)}),\\ \mathbf{x}_1 &= s(\mathbf{W}_d^{(1)}\mathbf{h} + \mathbf{b}_d^{(1)}).\end{aligned} \tag{2}$$

Hereafter we will represent the complete encoder-decoder structure (the TDRM) by its parameters $\theta_{\text{TDRM}} = \{\theta_{\mathbf{W}}, \theta_{\mathbf{b}}\}$, where $\theta_{\mathbf{W}} = \left\{\mathbf{W}_e^{(i)}, \mathbf{W}_d^{(i)}\right\}_{i=1}^{3}$ and $\theta_{\mathbf{b}} = \left\{\mathbf{b}_e^{(i)}, \mathbf{b}_d^{(i)}\right\}_{i=1}^{3}$. Later (in Sec. 3.3) this template will be separately fine-tuned for all classes of the training image sets.

3.2 TDRM's Parameter Initialization

The above defined TDRM is used to learn class specific DRMs. This is accomplished by separate training of the TDRM with images of each class of the training data. The training is performed with stochastic gradient descent through back propagation [33]. The training may fail if the TDRM is initialized with inappropriate weights. More specifically, if the initialized weights are too large, the network gets stuck in local minima. On the other hand, if the initialized weights are too small, the vanishing gradient problem is encountered during back propagation in the initial layers and the network becomes infeasible to train [33]. The weights of the template are therefore initialized by performing unsupervised pre-training. For that, a greedy layer-wise approach is adopted and Gaussian RBMs are used. Below, we first present a brief overview of RBMs (for the sake of completion) and then explain their usage for our TDRM's parameter initialization.

An RBM [34] is a generative undirected graphical model with a bipartite structure of two sets of binary stochastic nodes termed as the visible ($\{v_i\}_1^{N_v}, v_i \in \{0,1\}$) and the hidden layer nodes ($\{h_j\}_1^{N_h}, h_j \in \{0,1\}$). The nodes of the visible layer are symmetrically connected with the nodes of the hidden layer through a weight matrix $\mathbf{W} \in \mathbb{R}^{N_v\times N_h}$ but there are no intra layer node connections. The joint probability $p(\mathbf{v},\mathbf{h})$ of the RBM structure is given by

$$p(\mathbf{v},\mathbf{h}) = \frac{1}{Z}\exp(-E(\mathbf{v},\mathbf{h})), \tag{3}$$

where Z is the partition function (used as a normalization constant) and $E(v,h)$ is the energy function of the model defined as

$$E(\mathbf{v},\mathbf{h}) = \sum_i b_i v_i - \sum_j c_j h_j - \sum_{ij} w_{ij} v_i h_j, \tag{4}$$

where $\mathbf{b}$ and $\mathbf{c}$ are the biases of the visible and hidden layer nodes respectively. The goal of an RBM is to learn the model parameters $(\mathbf{W},\mathbf{b},\mathbf{c})$ in order to generate data similar to the training data. This is achieved by maximizing the likelihood of the training data. Due to the restriction that there are no connections between nodes of the same layer, inference becomes readily tractable for RBMs unlike most directed graphical models. This has rendered RBMs very successful in a wide range of applications [35], [36]. The parameter of an RBM are learnt by a numerical method called Contrastive Divergence (CD) [37].

The standard RBM developed for binary stochastic data can be generalized to real valued data by appropriate modifications of its energy function. Gaussian RBM (GRBM) [36] is one such popular extension whose energy function is defined by modifying the bias term of the visible units as

$$E_{\text{GRBM}}(\mathbf{v},\mathbf{h}) = \sum_i \frac{(v_i - b_i)^2}{2\sigma_i^2} - \sum_j c_j h_j - \sum_{ij} w_{ij}\frac{v_i}{\sigma_i} h_j, \tag{5}$$

where σ_i is the standard deviation of the real valued Gaussian distributed inputs to the visible node v_i. It is possible to learn σ_i for each visible unit but this becomes difficult when using CD for GRBM parameter learning. We instead adopt an alternative approach and fix σ_i to a unit value in the data pre-processing stage (see Sec 4.2). The conditional probability distributions needed for inference and generation are given by

$$\begin{aligned} p(h_j = 1|\mathbf{v}) &= s\left(\textstyle\sum_i w_{ij} v_i + c_j\right),\\ p(v_i|\mathbf{h}) &= \tfrac{1}{\sigma_i\sqrt{2\pi}}\exp\left(\tfrac{-(v_i-u_i)^2}{2\sigma_i^2}\right),\end{aligned} \tag{6}$$

where

$$u_i = b_i + \sigma_i^2 \sum_j w_{ij} h_j. \tag{7}$$

Since our data is real valued, we use GRBMs to initialize the weights of our TDRM. Two layers are considered at a time and the GRBM parameters are learnt. Initially, the nodes of the input layer are considered to be visible units $\mathbf{v}$ and the nodes of the first hidden layer as the hidden units $\mathbf{h}$ of the first GRBM and its parameters are learnt. The activations of the first GRBM's hidden units are then used as an input to train the second GRBM. The process is repeated for all three hidden layers of the encoder part of the TDRM structure. The weights learnt for the encoder layers are then tied to the corresponding decoder layers *i. e.* $\mathbf{W}_d^{(3)} = \mathbf{W}_e^{(1)^T}, \mathbf{W}_d^{(2)} = \mathbf{W}_e^{(2)^T}, \mathbf{W}_d^{(1)} = \mathbf{W}_e^{(3)^T}$ (See Fig. 2 for notations).

1. In our case we use a sigmoid defined as $s(z) = \frac{1}{1+e^{-z}}$

3.3 Learning Class Specific Models

The TDRM structure with the initialized weights is trained to learn class-specific DRMs. The training of a DRM is carried out for minimization of the reconstruction error over all m training examples of that class

$$J\left(\theta_{\text{TDRM}}\right) = \frac{1}{m}\sum_{t=1}^{m}\left\|\mathbf{x}^{(t)} - \tilde{\mathbf{x}}^{(t)}\right\|^2. \quad (8)$$

In order to avoid over-fitting and improve generalization of the learnt DRM to unknown test data, we introduce regularization terms into the cost function of TDRM. A weight decay penalty term J_{wd} and a sparsity constraint J_{sp} are added and the modified cost function becomes

$$J_{\text{reg}}\left(\theta_{\text{TDRM}}\right) = \frac{1}{m}\sum_{t=1}^{m}\left\|\mathbf{x}^{(t)} - \tilde{\mathbf{x}}^{(t)}\right\|^2 + \lambda_{\text{wd}} J_{\text{wd}} + \lambda_{\text{sp}} J_{\text{sp}}, \quad (9)$$

where λ_{wd} and λ_{sp} are regularization parameters. J_{wd} ensures small values of weights for all hidden units. It is defined as the summation of the squared Frobenius norm of all weight matrices

$$J_{\text{wd}} = \sum_{i}^{3}\left\|\mathbf{W}_e^{(i)}\right\|_F^2 + \sum_{i}^{3}\left\|\mathbf{W}_d^{(i)}\right\|_F^2. \quad (10)$$

J_{sp} enforces that the mean activation $\bar{\rho}_{i,j}$ (over all m training examples) of the jth unit of the ith hidden layer is as close as possible to a sparsity target ρ. It is defined in terms of the KL divergence as

$$\begin{aligned} J_{\text{sp}} &= \sum_{i}^{5}\sum_{j}\text{KL}\ \left(\rho||\bar{\rho}_{i,j}\right) \quad (11)\\ &= \sum_{i}^{5}\sum_{j}\rho\log\frac{\rho}{\bar{\rho}_{i,j}} + (1-\rho)\log\frac{1-\rho}{1-\bar{\rho}_{i,j}}. \end{aligned}$$

Here the KL divergence is computed between two distributions with means ρ and $\bar{\rho}_{i,j}$. The sparsity target ρ is a constant (typically a small value, set to 10^{-3} in our experiments in Sec. 5), whereas $\bar{\rho}_{i,j}$ is determined by taking the mean of the activation (see Eqs. (1) and (2) for activations) of a hidden unit over all training examples.

Synthetic Examples for Minority Classes

While training TDRM to learn class-specific DRMs, sub-optimal models might be learnt for classes with only a few training examples. To overcome this, we oversample the minority class (a class with few training examples) and create synthetic training examples. Specifically, we use the Synthetic Minority Oversampling TEchnique (SMOTE) [38]. For each training example of the minority class, we take the difference between the example and its nearest neighbor. The difference is then multiplied with a random number between $0-1$ and added to the original example. This gives a synthetic example which lies on the line joining the original example and its nearest neighbor. The total number of required synthetic examples can be controlled by two parameters *i. e.* the number of nearest neighbors considered for each example and the number of points generated on the line joining the original example and its nearest neighbor.

4 Image Set Classification Algorithm

We are now ready to describe our reconstruction error based image set classification algorithm. As shown in Figure 1, the algorithm comprises two parts: *training* to learn class-specific DRMs and *testing* (using the learnt DRMs) to decide the identity of a query image set. The algorithms for training and testing are summarized in Alg 1 and Alg 2 respectively. The details are presented below.

4.1 Problem Formulation

Given k training image sets $\mathcal{X}_1, \mathcal{X}_2, \cdots \mathcal{X}_k$ and their corresponding class labels $y_c \in [1, 2, \cdots k]$, where the image set $\mathcal{X}_c = \{\mathbf{x}^{(t)} | y^{(t)} = c; t = 1, 2, \cdots N_c\}$ has N_c images $\mathbf{x}^{(t)}$ belonging to class c, the problem of image set classification is formulated as follows: given a test image set $\mathcal{X}_{test}$, find the class y_{test} to which $\mathcal{X}_{test}$ belongs to. Note that in our formulation, if a class has multiple training image sets, we combine them into a single set.

4.2 Data Pre-Processing

The data is first pre-processed by encoding each image in terms of a Local Binary Pattern (LBP) [39] feature vector and performing PCA whitening. Specifically, each image is divided into 4×4 distinct non-overlapping uniformly spaced rectangular blocks and LBP histograms are computed for every block. Histograms from all blocks are concatenated into a single vector $\mathbb{R}^d$ which is used as a feature vector. There exists a strong correlation between adjacent patches of an image causing a redundancy in the encoded features. PCA whitening un-correlates the features while simultaneously reduces their dimensionality by discarding the redundant information. The procedure for PCA whitening is described below for completeness.

The images from all training image sets are encoded in terms of their LBP features and organized into columns of a matrix $X \in \mathbb{R}^{d\times n}$, where d is the dimensions of the feature vector and n is the total number of images. The covariance matrix of X is given by $\Sigma = \hat{X}^T\hat{X}$, where $\hat{X}$ is computed by subtracting the mean *i. e.* $\hat{X} = X - \frac{1}{n}\sum X$. Singular Value Decomposition of the covariance matrix results $\Sigma = USV$. The column vectors in U are orthogonal to each other and are arranged in descending order of their significance. S is a diagonal matrix with singular values $\sigma_i = S_{i,i}$ on the diagonal. Let $\hat{U}$ contain the top k column vectors from U corresponding to the k largest singular values in S. Given any image $\mathbf{x}^{(t)} \in \mathbb{R}^d$, we project it onto the matrix $\hat{U}$

$$\hat{\mathbf{x}}^{(t)} = \hat{U}^T\mathbf{x}^{(t)}. \quad (12)$$

The above procedure reduces the dimensionality of $\mathbf{x}^{(t)}$ from $\mathbb{R}^d$ to $\mathbb{R}^k$. The value of k[2] determines the amount of energy to be retained. The adjacent values of $\hat{\mathbf{x}}^{(t)}$ are un-correlated as it has been projected onto a matrix whose

2. In our case, $d = 944$ (the dimensions of LBP feature vector) and $k = 400$ (retains about 85% energy)

vectors are orthogonal to each other. Next, we divide each entry of $\hat{\mathbf{x}}^{(t)}$ by $\sqrt{\sigma_i}+\epsilon$, where σ_i is the covariance of each feature (determined by the diagonal matrix S) and ϵ is a small value (10^{-5} in our experiments) used to avoid zero division and reduce the effect of noise. Thus the final output of PCA whitening is

$$\hat{\hat{\mathbf{x}}}^{(t)} = \frac{\hat{\mathbf{x}}^{(t)}}{\sqrt{\sigma_i}+\epsilon}. \tag{13}$$

This ensures unit variance for each of the input feature $\hat{\hat{\mathbf{x}}}^{(t)}$. For the sake of notational simplicity, we will denote a pre-processed image $\hat{\hat{\mathbf{x}}}^{(t)}$ by $\mathbf{x}^{(t)}$ here and onwards.

4.3 Training of DRMs

Algorithm 1 Learning Deep Reconstruction Models

Input: Training image sets: $\mathcal{X}_1, \mathcal{X}_2, \cdots \mathcal{X}_k$
1: Compute LBP features
2: Perform PCA whitening
3: $\hat{\mathcal{X}}$: Randomly selected small subset from $\mathcal{X} = \bigcup_c \mathcal{X}_c$
4: Train GRBMs using $\hat{\mathcal{X}}$ to initialize $\theta_{\text{TDRM}} = \{\theta_{\mathbf{W}}, \theta_{\mathbf{b}}\}$
5: **for** $c = 1 \cdots k$ **do**
6: $\quad \theta_c \leftarrow \min_{\theta_{\text{TDRM}}} J_{\text{reg}}\left(\theta_{\text{TDRM}}; \mathcal{X}_c\right)$
7: **end for**
Output: Class-specific DRMs $\theta_1, \theta_2, \cdots \theta_k$

After pre-processing the training data, we gather images from all training sets into a single data set $\mathcal{X} = \bigcup_c \mathcal{X}_c$. Next, a subset $\hat{\mathcal{X}}$ is generated from $\mathcal{X}$. $\hat{\mathcal{X}}$ contains a small fraction (500 in our experiments) of randomly picked images from $\mathcal{X}$, drawn equally from all classes. $\hat{\mathcal{X}}$ is used for layer-wise GRBM training of all layers of the encoder part of the template. The weights of the decoder layers are then initialized with their corresponding tied weights of the encoder layers. Note that using $\hat{\mathcal{X}}$ (instead of $\mathcal{X}$) for TDRM's parameter's initialization ensures that the parameters are not biased towards the majority classes (the classes with more images). Furthermore, the time required to train GRBMs reduces significantly with a fewer number of images in $\hat{\mathcal{X}}$.

Now that we have the TDRM structure with the initialized weights, we separately fine tune its parameters θ_{TDRM} for each of the k training image sets. We therefore learn k class-specific DRMs *i. e.* $\theta_1, \theta_2, \cdots \theta_k$. A class-specific model θ_c is achieved by optimization of the regularized cost function J_{reg} over all images of the set $\mathcal{X}_c$ *i. e.*

$$\theta_c = \min_{\theta_{\text{TDRM}}} J_{\text{reg}}\left(\theta_{\text{TDRM}}; \mathcal{X}_c\right). \tag{14}$$

Since the model is being trained to reconstruct the input data, it might achieve perfect reconstruction simply by learning an identity function. This is not desirable as the model would not learn any useful representations of the input data and would therefore not generalize to the unknown test data. Appropriate settings in the configurations of the TDRM are therefore required to ensure that a class specific model learns the underlying structure of the data and produces useful representations. For our TDRM, since the number of nodes in the first hidden layer is larger than the dimensions of the input data, we first learn an over-complete representation of the data by mapping it to a high dimensional space. This high dimensional representation is then followed by a bottleneck *i. e.* the data is mapped back to a compact, abstract and low dimensional representation in the subsequent layers of the encoder. With such a mapping, the redundant information in the data is discarded and only the required useful content of the data is retained. It can be shown that if we reduce our TDRM structure to have only a single hidden layer connected through a linear activation function, the weights learnt by the structure would be similar to a PCA subspace. However, in our case, since the activation functions used are non-linear and a number of hidden layers are stacked together, the TDRM becomes capable of adapting itself to very complex non-linear manifold structures.

4.4 Classification

Algorithm 2 Image Set Classification from DRMs

Input: Test image set $\mathcal{X}_{test} = \{\mathbf{x}^{(t)}; t = 1, 2, \cdots N_{test}\}$
Class-specific DRMs: $\theta_1, \theta_2, \cdots \theta_k$
1: Compute LBP features
2: Perform PCA whitening
3: **for** each image $\mathbf{x}^{(t)} \in \mathcal{X}_{test}$ **do**
4: $\quad$ **for** $\theta_c = \theta_1 \cdots \theta_k$ **do**
5: $\quad\quad \{\mathbf{W}_e^{(i)}, \mathbf{W}_d^{(i)}, \mathbf{b}_e^{(i)}, \mathbf{b}_d^{(i)}\}_{i=1}^{3} \leftarrow \theta_c$
6: $\quad\quad \mathbf{h}^{(t)} \leftarrow s(\mathbf{W}_e^{(3)} s(\mathbf{W}_e^{(2)} s(\mathbf{W}_e^{(1)} \mathbf{x}^{(t)} + \mathbf{b}_e^{(1)}) + \mathbf{b}_e^{(2)}) + \mathbf{b}_e^{(3)})$
7: $\quad\quad \tilde{\mathbf{x}}_c^{(t)} \leftarrow s(\mathbf{W}_d^{(3)} s(\mathbf{W}_d^{(2)} s(\mathbf{W}_d^{(1)} \mathbf{h}^{(t)} + \mathbf{b}_d^{(1)}) + \mathbf{b}_d^{(2)}) + \mathbf{b}_d^{(3)})$
8: $\quad\quad r_c(\mathbf{x}^{(t)}) \leftarrow \left\| \mathbf{x}^{(t)} - \tilde{\mathbf{x}}_c^{(t)} \right\|_2$
9: $\quad$ **end for**
10: **end for**
11: **MV:** $y_{test} = \arg\max_c \sum_t \delta_c(v^{(t)})$ Refer to Eq (17)
12: **WV:** $y_{test} = \arg\max_c \sum_t w_c(\mathbf{x}^{(t)})$ Refer to Eq (18)
13: **PWV:** $y_{test} = \arg\max_c \sum_t p\left(c|\mathbf{x}^{(t)}\right)$ Refer to Eqs (20)-(26)
Output: Label y_{test} of $\mathcal{X}_{test}$

Given a test image set $\mathcal{X}_{test} = \left\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \cdots \mathbf{x}^{(N_{test})}\right\}$, we separately reconstruct (using Eqs. (1) and (2)) each of its image $\mathbf{x}^{(t)} \in \mathcal{X}_{test}$ from all class specific DRMs $\theta_c, c = 1 \cdots k$. If $\tilde{\mathbf{x}}_c^{(t)}$ is the reconstruction of the image $\mathbf{x}^{(t)}$ from model θ_c, then the reconstruction error is given by

$$r_c(\mathbf{x}^{(t)}) = \left\| \mathbf{x}^{(t)} - \tilde{\mathbf{x}}_c^{(t)} \right\|_2. \tag{15}$$

Given reconstruction errors $r_c(\mathbf{x}^{(t)})$ from all class specific models, we propose three different strategies to determine the class (y_{test}) of the test image set ($\mathcal{X}_{test}$).

4.4.1 Majority Voting (MV)

Each image $\mathbf{x}^{(t)} \in \mathcal{X}_{test}$ can cast only one vote. The vote $v^{(t)}$ is cast to the class whose model reconstructs the image with the least reconstruction error *i. e.*

$$v^{(t)} = \arg\min_c r_c(\mathbf{x}^{(t)}). \tag{16}$$

The votes casted by all images of $\mathcal{X}_{test}$ are then counted and the candidate class which achieves the maximum number of votes is declared as the class (y_{test}) of the test image set ($\mathcal{X}_{test}$)

$$y_{test} = \arg\max_c \textstyle\sum_t \delta_c(v^{(t)}), \text{ where}$$
$$\delta_c(v^{(t)}) = \begin{cases} 1, & v^{(t)} = c, \\ 0, & \text{otherwise.} \end{cases} \tag{17}$$

4.4.2 Weighted Voting (WV)

An image $\mathbf{x}^{(t)} \in \mathcal{X}_{test}$ casts a vote to all candidate classes. The vote casted to each class is given a weight which is determined by the reconstruction error from the respective class-specific model. Specifically, the weight $w_c(\mathbf{x}^{(t)})$ of the vote casted by an image $\mathbf{x}^{(t)}$ to class c is given by

$$w_c(\mathbf{x}^{(t)}) = \exp(-\gamma r_c(\mathbf{x}^{(t)})). \tag{18}$$

See Eq. (15) for $r_c(\mathbf{x}^{(t)})$. γ is a parameter adjusted by performing experiments on a cross validation set (see Sec 5.1.1). The candidate class which achieves the maximum accumulated weight from all images of $\mathcal{X}_{test}$ is then declared as the class (y_{test}) of the test image set ($\mathcal{X}_{test}$) *i. e.*

$$y_{test} = \arg\max_c \sum_{\mathbf{x}^{(t)} \in \mathcal{X}_{test}} w_c(\mathbf{x}^{(t)}). \tag{19}$$

4.4.3 Preferential Weighted Voting (PWV)

PWV is an extension of WV where preference is given to the vote casted by an image based upon its pose. PWV is designed for face datasets to give more preference to the vote of a face image with a pose which is commonly present in the training data. The class label y_{test} of the image set $\mathcal{X}_{test}$ using PWV is determined by

$$y_{test} = \arg\max_c \sum_{\mathbf{x}^{(t)} \in \mathcal{X}_{test}} w_c(\mathbf{x}^{(t)}) d(\mathbf{x}^{(t)}). \tag{20}$$

Where $w_c(\mathbf{x}^{(t)})$ (see Eq. 18) is a measure of the similarity of the image from the cth class-specific model and $d(\mathbf{x}^{(t)})$ is the preferential weight given to the vote casted by the image $\mathbf{x}^{(t)}$. $d(\mathbf{x}^{(t)})$ is determined in terms of the head pose of the face image $\mathbf{x}^{(t)}$. Specifically, to determine $d(\mathbf{x}^{(t)})$, each image is first assigned to a pose group by a *pose group approximation method*. $d(\mathbf{x}^{(t)})$ is then computed in terms of the presence of the pose group of the image $\mathbf{x}^{(t)}$ in the training data. Here, we first describe our pose group approximation method and then define $d(\mathbf{x}^{(t)})$ in Eqn. 26.

Pose Group Approximation: An image is said to belong to a pose group $g \in \{1, 2, \cdots G\}$, if its pose along the pitch direction (y-axis) is within $\theta_g \pm 15°$. For our purpose, we define $G = 5$ and $\theta = \begin{bmatrix} -60, & -30, & 0, & 30, & 60 \end{bmatrix}$. The process of pose group approximation constitutes two steps: training and testing.

Training: Let $X_g \in \mathbb{R}^{d \times n_g}$ contain n_g images $\mathbf{x}^{(t)} \in \mathbb{R}^d$ whose pose is within $\theta_g \pm 15°$. We automatically select these images from a Kinect data set (see Sec 5.2.4). The pose of Kinect images can be determined by the random regression forest based method of [21]. From X_g, we want to extract the directions of major data orientation. To do that, we first subtract the mean image from X_g and compute its covariance matrix Σ_g *i. e.*

$$\hat{X}_g = X_g - \frac{1}{n_g} \sum_t \mathbf{x}^{(t)}, \tag{21}$$

$$\Sigma_g = \hat{X}_g \hat{X}_g^T. \tag{22}$$

Performing singular value decomposition of the covariance matrix Σ_g results, $\Sigma_g = U_g S_g V_g$. U_g contains eigenvectors arranged in the descending order of their significance. From U_g, we select the top k eigenvector corresponding to the k largest eigenvalues and represent them as columns of a matrix $\mathcal{S}_g \in \mathbb{R}^{d \times k}$. $\mathcal{S}_g$ is therefore a subspace whose columns represent the predominant data structure in the images of X_g. Next, during the testing phase of pose group approximation, $\mathcal{S}_g$ is used for a linear regression based classification strategy [40].

Testing: The pose group $\mathcal{P}(\mathbf{x}^{(t)})$ of the image $\mathbf{x}^{(t)}$ is determined by

$$\mathcal{P}(\mathbf{x}^{(t)}) = \arg\min_g \left\| \mathbf{x}^{(t)} - \tilde{\mathbf{x}}_g^{(t)} \right\|_2, \tag{23}$$

where $\tilde{\mathbf{x}}_g^{(t)}$ is linearly reconstructed from $\mathcal{S}_g$ as

$$\tilde{\mathbf{x}}_g^{(t)} = \mathcal{S}_g \alpha_g^{(t)}. \tag{24}$$

The above equation has an analytical solution given by

$$\alpha_g^{(t)} = (\mathcal{S}_g^T \mathcal{S}_g)^{-1} \mathcal{S}_g^T \mathbf{x}^{(t)}. \tag{25}$$

The above described method gives us the pose group of an image. We use this method to determine the pose group of all images of the training data as well as the images of the query image set. Let g be the pose group of the image $\mathbf{x}^{(t)} \in \mathcal{X}_{test}$ of the test image set. The preferential weight $d(\mathbf{x}^{(t)})$ for the image $\mathbf{x}^{(t)} \in \mathcal{X}_{test}$ is then given by

$$d(\mathbf{x}^{(t)}) = \frac{1}{\sum_c N_c} \sum_{\mathcal{X}_c} \sum_{\mathbf{x}^{(t)} \in \mathcal{X}_c} \delta_g \mathcal{P}(\mathbf{x}^{(t)}), \tag{26}$$

where N_c is the number of training images for cth class and $\delta_g \mathcal{P}(\mathbf{x}^{(t)})$ is the delta function given by

$$\delta_g \mathcal{P}(\mathbf{x}^{(t)}) = \begin{cases} 1, & \mathcal{P}(\mathbf{x}^{(t)}) = g \\ 0, & \text{otherwise.} \end{cases} \tag{27}$$

5 EXPERIMENTS

We evaluate and compare performance of the proposed method with existing methods for the tasks of face and object recognition. For face recognition, the performance evaluation is presented for six RGB data sets (Honda/UCSD [17], CMU Mobo [18], YouTube Celebrities (YTC) [19], PubFig [22], a subset of YouTube Faces (YTF) [23] & COX [24]) and an RGB-D Kinect dataset (obtained by combining three Kinect datasets). For object recognition, we use ETH-80 dataset [25]. The detailed description of each of these datasets and the performance evaluation of our method and the existing methods is presented in Sec 5.2. Here, the common experimental settings are presented first.

5.1 Experimental Settings

The face from each frame in the videos of Honda/UCSD and Mobo datasets is automatically detected using Viola and Jones face detection algorithm [41]. However, in case of YTC dataset, face detection by [41] fails in a significant number of frames due to the poor image resolution and the large head rotations. We therefore used the method in [42] to track the face region across each video of YTC dataset. For PubFig, YTF and COX datasets, we used the already cropped face images provided with the respective datasets. In the case of Kinect face datasets, the random regression forest based classifier proposed in [21] is used to automatically detect faces from depth images. As depth data is pre-aligned with RGB, the same location of the detected face in the depth image is used for the corresponding RGB image. After a successful detection, the face region is cropped and all colored images are converted to gray scale levels. The cropped gray scale images are then resized to 20×20, 40×40 and 30×30 for Honda/UCSD, Mobo and YTC datasets respectively. The provided cropped faces of PubFig, YTF and COX are resized to 20×24, 30×30 and 30×30 respectively. The depth and the gray scale images of the Kinect datasets are resized to 20×20. In the case of the object dataset (ETH-80), the 128×128 cropped images[3] are resized to 32×32. Histogram equalization is applied on all images to minimize illumination variations.

5.1.1 Settings of Our Method

For TDRM's parameter initialization and learning class specific models, we adjust appropriate hyper-parameters by following the guidelines in [43], [44]. The optimal parameters are searched by doing grid search while performing experiments on a cross validation set. Specifically, for TDRM's initialization using GRBMs, initial weights for layer-wise GRBM training are drawn from a uniform random distribution in the range $\begin{bmatrix} -.005 & .005 \end{bmatrix}$. Contrastive Divergence [37] is used to train GRBMs on 500 randomly selected images from the training data. Mini-batches of 100 images are used and the training is carried out for 20 epochs. A fixed learning rate of 10^{-3} is used. In order to train the initialized TDRM to learn class-specific models, we use an annealed learning rate (started with 2×10^{-3} and multiplied by a factor of 0.6/epoch), an L_2-weight decay (λ_{wd} in Eq. (9)) of 10^{-2}, a sparsity target (ρ in Eq. (11)) of 10^{-3} and non-sparsity penalty term (λ_{sp} in Eq. (9)) of 0.5. The training is performed by considering a mini-batch size of 5 images for 30 epochs. For classification, we use $\gamma = 3$ in Eq. (18) for WV and Eq. (20) for PWV. Note that the mentioned hyper-parameters for TDRM's parameter initialization and learning class-specific models are consistent across all datasets.

5.1.2 Settings of Compared Methods

We compare our proposed method with a number of recently proposed state of the art image set classification methods. The compared methods include Discriminant Canonical Correlation Analysis (DCC) [3], Manifold-to-Manifold Distance (MMD) [4], Manifold Discriminant Analysis (MDA) [5], the Linear version of the Affine Hull-based Image Set Distance (AHISD) [6], the Convex Hull-based Image Set Distance (CHISD) [6], Sparse Approximated Nearest Points (SANP) [8], Graph Embedding Discriminant Analysis (GEDA) [7], Covariance Discriminant Learning (CDL) [9], Regularized Nearest Points (RNP) [10], Mean Sequence Sparse Representation Classification (MSSRC) [11] and Set to Set Distance Metric Learning (SSDML) [12]. The implementations provided by the respective authors are used for all methods except CDL. We carefully implemented CDL as it is not publicly available. The parameters for all methods are optimized for best performance. Specifically, for MSM, we apply PCA to retain 90% of the total energy. For DCC, we set the dimensions of the embedding space to 100. The number of retained dimensions for a subspace are set to 10 (90% energy is preserved) and the corresponding 10 maximum canonical correlations are used to compute set-set similarity. For datasets with one training set per class (Honda/UCSD, CMU, Kinect and PubFig), we randomly divide the training set into two subsets to construct the within class sets as in [3]. The parameters for MMD and MDA are adopted from [4] and [5] respectively. The number of connected nearest neighbors to compute the geodesic distance is either set to 12 or to the number of images in the smallest image set of the dataset. The ratio between Euclidean distance and geodesic distance is optimized for all data sets. The distance in case of MMD is computed in terms of maximum canonical correlation. No parameter settings are required for AHISD. For CHISD, the same error penalty term ($C = 100$) as in [6] is adopted. For SANP, the same weight parameters as in [8] are adopted for convex optimization. For GEDA, we set $k^{[\text{cc}]} = 1, k^{[\text{proj}]} = 100$ and $v = 3$ (the value of v is searched over a range of 1-10 for best performance). The number of eigenvectors r used to represent an image set is set to 9 and 6 respectively for Mobo and YouTube Celebrities and 10 for all other datasets. No parameter settings are required for CDL. For RNP [10], PCA is applied to preserve 90% of the energy and the same weight parameters as in [10] are adopted. No parameter configurations are required for MSSRC and

3. Available at http://www.d2.mpi-inf.mpg.de/Datasets/ETH80

TABLE 1: Performance on Honda/UCSD dataset

Methods	All-All	100-100	50-50	50-25	50-15	25-50	All-1
MSM [15]	88.21 ± 3.86	85.64 ± 4.39	83.08 ± 1.73	82.25 ± 4.32	80.67 ± 4.99	79.24 ± 4.97	57.69 ± 12.34
DCC [3]	92.56 ± 2.25	89.28 ± 2.46	82.05 ± 3.30	80.81 ± 8.81	80.23 ± 3.38	78.43 ± 2.71	3.84 ± 3.02
MMD [4]	92.05 ± 2.25	85.59 ± 2.16	83.12 ± 4.49	82.44 ± 5.19	80.38 ± 3.64	81.17 ± 3.75	–
MDA [5]	94.36 ± 3.38	91.79 ± 1.62	85.64 ± 5.82	84.97 ± 4.02	83.67 ± 5.77	84.74 ± 4.47	–
AHISD [6]	91.28 ± 1.79	90.77 ± 3.24	89.85 ± 2.16	90.79 ± 3.93	90.32 ± 1.62	$\mathbf{89.54 \pm 1.64}$	$\mathbf{76.67 \pm 7.69}$
CHISD [6]	93.62 ± 1.63	91.09 ± 1.78	90.56 ± 2.05	89.23 ± 4.32	86.69 ± 2.99	86.17 ± 2.49	$\mathbf{75.81 \pm 7.81}$
GEDA [7]	91.28 ± 5.82	88.21 ± 9.06	82.82 ± 6.05	83.25 ± 3.24	81.36 ± 3.53	80.37 ± 3.47	–
SANP [8]	95.13 ± 3.07	94.10 ± 3.21	91.90 ± 2.76	90.84 ± 4.65	89.61 ± 4.09	88.54 ± 3.84	54.87 ± 11.35
CDL [9]	98.97 ± 1.32	96.23 ± 1.24	93.90 ± 2.24	91.34 ± 2.53	89.31 ± 5.54	$\mathbf{88.72 \pm 4.84}$	5.13 ± 0.0
MSSRC [11]	97.95 ± 2.65	96.97 ± 1.32	94.35 ± 1.46	90.86 ± 4.15	91.82 ± 2.42	$\mathbf{90.77 \pm 2.49}$	72.8 ± 4.84
SSDML [12]	86.41 ± 3.64	84.36 ± 2.25	83.41 ± 1.73	84.69 ± 4.26	80.26 ± 3.59	80.21 ± 3.45	70.77 ± 10.05
RNP [10]	95.90 ± 2.16	92.33 ± 3.24	90.23 ± 3.26	89.34 ± 6.61	85.38 ± 2.02	84.81 ± 2.72	40.0 ± 7.37
DRM-MV	$\mathbf{100.00 \pm 0.0}$	$\mathbf{99.23 \pm 1.24}$	$\mathbf{96.92 \pm 2.91}$	$\mathbf{94.87 \pm 4.35}$	$\mathbf{88.46 \pm 6.42}$	$\mathbf{83.94 \pm 4.55}$	$\mathbf{73.59 \pm 7.55}$
DRM-WV	$\mathbf{100.00 \pm 0.0}$	$\mathbf{100.0 \pm 0.0}$	$\mathbf{100.0 \pm 0.0}$	$\mathbf{98.20 \pm 1.23}$	$\mathbf{96.15 \pm 3.25}$	$\mathbf{85.33 \pm 3.66}$	$\mathbf{73.59 \pm 7.55}$
DRM-PWV	$\mathbf{100.00 \pm 0.0}$	$\mathbf{100.0 \pm 0.0}$	$\mathbf{100.0 \pm 0.0}$	$\mathbf{98.32 \pm 1.14}$	$\mathbf{96.26 \pm 1.43}$	$\mathbf{85.62 \pm 3.12}$	$\mathbf{73.59 \pm 7.55}$

Average identification rates and standard deviations of different methods on Honda/UCSD dataset. Experiments are performed by considering different sizes of the training and testing image sets. For example, $50 - 25$ means an experiment was performed by restricting an upper limit of 50 and 25 frames on the total number of images in the training and testing sets respectively. A "–" in the last column means that the respective method could not be evaluated by using only one image in the testing set.

SSDML.

5.2 Results and Analysis

5.2.1 Honda/UCSD Dataset

The Honda/UCSD dataset [17] contains 59 video sequences of 20 different subjects. The number of frames for each video sequence varies from 12 to 645. For performance evaluation and comparison with existing state of the art methods, we perform experiments following the standard evaluation configuration provided in [17]. Each video is considered as an image set. 20 video sequences are used for training and the remaining 39 sequences are used for testing. In order to achieve a consistency in the results, we run experiments 10 times for different random selections of training and testing image sets.

The performance of different methods is evaluated by reducing the size of the training and testing sets. Specifically, we set an upper limit N_{train} and N_{test} on the total number of images in the training and testing sets respectively. Experiments are performed by considering $N_{train} = \{All, 100, 50, 25\}$ and $N_{test} = \{All, 100, 50, 25, 15, 1\}$. The experimental results for different combinations of the training and the testing set lengths are presented in Table 1. The Cumulative Match Characteristics (CMC) curves for the top performing methods (for all frames of a video considered as an image set) are shown in Figure 3a. The results suggest that the proposed method outperforms the other methods and achieves the best average identification rates for most of the configurations of the training and testing set lengths. The results also show that the performance of all methods degrades for a smaller number of images in a set. Affine hull or convex hull based methods (AHISD, CHISD, RNP, SANP) perform more consistently over different set lengths. For the methods which represent an image set either by a linear subspace or a combination of multiple linear subspaces (MSM, DCC, MMD, MDA, GEDA), their performance degrades gracefully with a reduced set length. Our proposed method achieves perfect classification when the number of images in a set is reduced to 50 images. Reducing the total number of images to less than 50 degrades the performance of our method. However, compared with other methods, the proposed method is the least affected by the reduced set size. The performance of the proposed method degrades more for reduced training set size whereas the effect of the reduced testing set size on the performance is not very noticeable.

If the size of the testing image set is further reduced to as low as one image, the performance of all methods drops sharply and our proposed method achieves an average identification rate of $73.59 \pm 7.55\%$. Note that image set classification methods are primarily developed for identification from very low resolution images (20×20 in this case). Correct identification using a single image therefore becomes very challenging and the presence of multiple images in a set is required to complement each other's appearance.

5.2.2 CMU Mobo Dataset

The Mobo (Motion of Body) dataset [18] was originally created for the pose identification of the human body. The dataset contains a total of 96 sequences of 24 subjects walking on a treadmill. For our experiments, we randomly select one sequence of a subject for training and the remaining three sequences are used for testing. The experiments are repeated 10 times for different random selections of the training and the testing sets. The average identification rates of our proposed method along with a comparison with other methods are provided in Table 2. The CMC curves for different methods are shown in Fig. 3b. The results show that the proposed method achieves the highest average identification rate along with the lowest standard deviation which suggests that the proposed method is both accurate and reliable. Amongst the existing methods, SANP, MSSRC and RNP show a comparable performance. SANP achieves the highest rank 2 and rank 3 identification rates. However, our method outperforms all methods for rank 1 and rank 4-10 identification rates.

TABLE 2: Performance Evaluation of All Methods on Different Datasets

Methods	Mobo	YTC	Kinect	PubFig	YTF	COX	ETH
MSM FG'98 [15]	96.81 ± 1.97	50.21 ± 3.59	89.29 ± 4.11	56.95 ± 2.64	30.73 ± 3.24	26.38 ± 10.89	75.50 ± 4.83
DCC TPAMI'07 [3]	88.89 ± 2.45	51.42 ± 4.95	92.52 ± 2.00	34.90 ± 7.67	34.55 ± 1.70	43.30 ± 12.11	$\mathbf{91.75 \pm 3.74}$
MMD CVPR'08 [4]	92.50 ± 2.87	54.04 ± 3.69	93.90 ± 2.25	36.21 ± 6.86	36.65 ± 2.52	54.86 ± 10.25	77.50 ± 5.00
MDA CVPR'09 [5]	80.97 ± 12.28	55.11 ± 4.55	93.46 ± 3.57	34.25 ± 6.39	39.82 ± 2.83	$\mathbf{73.05 \pm 10.37}$	77.25 ± 5.46
AHISD CVPR'10 [6]	92.92 ± 2.12	61.49 ± 5.63	91.60 ± 2.18	62.05 ± 2.04	36.18 ± 1.49	64.10 ± 11.33	78.75 ± 5.30
CHISD CVPR'10 [6]	$\mathbf{96.52 \pm 1.18}$	60.42 ± 5.95	92.73 ± 1.91	64.81 ± 2.13	39.81 ± 2.51	63.13 ± 10.42	79.53 ± 5.32
GEDA CVPR'11 [7]	84.86 ± 3.24	52.48 ± 4.45	91.43 ± 6.28	35.45 ± 26.17	35.64 ± 3.49	53.15 ± 15.78	79.50 ± 5.24
SANP TPAMI'12 [8]	$\mathbf{97.64 \pm 0.94}$	$\mathbf{65.60 \pm 5.57}$	93.83 ± 3.12	80.41 ± 2.45	36.59 ± 5.62	$\mathbf{66.23 \pm 13.38}$	77.75 ± 7.31
CDL CVPR'12 [9]	90.00 ± 4.38	56.38 ± 5.31	$\mathbf{94.59 \pm 0.96}$	51.05 ± 3.98	36.00 ± 3.25	56.05 ± 16.31	77.75 ± 4.16
MSSRC CVPR'13 [11]	$\mathbf{97.50 \pm 0.88}$	59.36 ± 5.70	$\mathbf{95.51 \pm 2.30}$	85.55 ± 2.76	$\mathbf{50.64 \pm 2.23}$	$\mathbf{69.40 \pm 15.69}$	$\mathbf{90.50 \pm 3.07}$
SSDML ICCV'13 [12]	$\mathbf{95.14 \pm 2.20}$	$\mathbf{66.24 \pm 5.21}$	86.88 ± 3.39	$\mathbf{88.80 \pm 1.60}$	38.55 ± 2.85	65.33 ± 10.81	81.00 ± 6.58
RNP FG'13 [10]	96.11 ± 1.43	$\mathbf{65.82 \pm 5.39}$	$\mathbf{96.23 \pm 2.50}$	$\mathbf{88.60 \pm 0.99}$	34.55 ± 3.15	$\mathbf{66.20 \pm 12.83}$	81.00 ± 3.16
DRM-MV	$\mathbf{97.92 \pm 0.73}$	$\mathbf{71.35 \pm 4.83}$	$\mathbf{98.11 \pm 1.68}$	$\mathbf{88.55 \pm 1.45}$	$\mathbf{46.91 \pm 2.92}$	$\mathbf{66.08 \pm 14.69}$	$\mathbf{98.12 \pm 1.69}$
DRM-WV	$\mathbf{98.15 \pm 0.68}$	$\mathbf{72.23 \pm 4.79}$	$\mathbf{98.26 \pm 1.65}$	$\mathbf{88.67 \pm 1.54}$	$\mathbf{48.42 \pm 2.98}$	$\mathbf{69.85 \pm 12.93}$	$\mathbf{98.25 \pm 1.69}$
DRM-PWV	$\mathbf{98.33 \pm 0.65}$	$\mathbf{72.55 \pm 4.74}$	$\mathbf{98.26 \pm 1.65}$	$\mathbf{89.90 \pm 0.86}$	$\mathbf{51.45 \pm 3.06}$	$\mathbf{66.45 \pm 12.31}$	–

Experimental performance of different methods in terms of average identification rates and standard deviations on CMU/Mobo, YouTube Celebrities (YTC), Kinect, PubFig, YouTube Faces (YTF), COX and ETH-80 datasets. The proposed method achieves the best performance on all datasets. The increase in performance is more significant for YTC, ETH-80 and YTF datasets.

5.2.3 YouTube Celebrities Dataset

YouTube Celebrities (YTC) dataset [19] contains 1910 videos of 47 celebrities collected from YouTube. The face images of the dataset exhibit a large diversity and variations in the form of pose, illumination and expressions. Moreover, the quality and resolution of the images is very low due to the high compression rate. Since the face regions in the videos are cropped by tracking [42], the low image quality introduces many tracking errors and the region of the cropped face is not uniform across frames of even the same video.

For performance evaluation, we use five fold cross validation experimental settings as followed in [4], [5], [8], [9]. The complete dataset is equally divided into five folds with 9 image sets per subject in each fold. Three of these image sets are randomly selected for training, whereas the remaining six sets are used for testing. Table 2 summarizes the average identification rates and the standard deviations of different methods. The CMC curves are shown in Fig. 3c. It can be observed that the achieved identification rates for all methods are low for this dataset compared with the Honda/UCSD and Mobo dataset. This is owing to the challenging nature of the dataset. The videos have been captured in real life scenarios and they exhibit a wide range of appearance variations. The results suggest that our proposed method significantly outperforms the existing methods and achieves a relative performance improvement of 9.5% over the second best method. Moreover, the proposed method consistently achieves the best identification rates from rank 1 to 10. Note that the results of some methods *e. g.* [5], [9] are relatively lower than as reported in their respective papers because of our more challenging experimental setup, large tracking errors in the automatically cropped faces and the presence of faces under a wide range of appearance variations. These methods only use the successfully detected faces using [41]. We observe that face detection by Viola and Jones [41] fails in a significant number of frames (specially those with large head rotations and low image quality). In our case, we extract faces from videos by tracking [42]. Our experiments therefore include faces under a wide range of variations.

5.2.4 Kinect Dataset

Face recognition from RGB-D data acquired by a Kinect sensor is still in its infancy and only few works [20] have addressed this problem. The method in [20] first pre-processes Kinect depth images to produce a canonical frontal view for faces with profile and non-frontal views. The sparse representation based classification method of [45] is then used for recognition. The method is evaluated on CurtinFaces dataset and achieves a classification rate of 91.1% for RGB , 88.7% for D and 96.7% for fusion of RGB-D data. The proposed method is single frame based and does not make use of the plentitude of data which can be instantly acquired from a Kinect sensor (30 frames per second). Here, we formulate face recognition from Kinect data as an RGB-D based image set classification problem. Our formulation avoids expensive pre-processing steps (such as hole filling, spike removal and canonical view estimation; otherwise required for single image based classification) and effectively makes use of the abundant and readily available Kinect data.

The method in [20] is evaluated on CurtinFaces (a Kinect RGB-D database of 52 subjects). For our image set classification experiments, we combine three Kinect datasets: CurtinFaces [20], Biwi Kinect [21] and an in-house dataset acquired at our laboratory at UWA. The number of subjects in each of these datasets is 52 (5000 RGB-D images), 20 (15,000 RGB-D images) and 48 (15000 RGB-D images) respectively. These datasets are combined into a single dataset of 120 subjects. The images in the joint dataset have a large range of variations in the form of changing illumination conditions, head pose rotations, expression deformations, sunglass disguise, and occlusions by hand. For performance evaluation, RGB-D images of each subject are randomly divided into five uniform folds. Considering each fold as an image set, we select one set for training and the remaining sets for testing. All experiments are repeated five times for different selections of training and testing sets. The results averaged over five iterations are summarized in Table 2. The results show that the proposed method achieves the highest identification rate (98.26%). The existing methods show a good performance and the

Fig. 4: Example images of a person from PubFig dataset.

achieved identification rates are 89.29% and higher. The results suggest that image set classification proves to be a better choice for Kinect based face recognition. The achieved performance by image set classification methods is better or comparable to single image based technique (96.7%) of [20]. Moreover, the image set classification techniques avoid computationally expensive pre-processing steps. Note that compared with our experiments on a large dataset of 120 subjects, the results reported for [20] are on a small dataset of 52 subjects only.

5.2.5 Public Figures Face Database (PubFig)

PubFig [22] is a real-life dataset of 200 people collected from the internet. The images of the dataset have been acquired in uncontrolled situations without any user cooperation. The sample images of a subject in Fig. 4 show large variations in the images caused by pose, lighting, expressions, backgrounds and camera positions. For our experiments, we divide equally the images of each subject into three folds. Considering each fold as an image set, we use one of them for training and the remaining two are used for testing. Experiments are repeated five times for different random selections of images of the training and testing folds. The experimental results in Table 2 show that the proposed method achieves an average rank-1 identification rate of $89.90 \pm 0.86\%$ and outperforms all compared methods. The CMC curves in Fig 3g show that the proposed method consistently achieves the best identification rates over all ranks.

5.2.6 YouTube Faces Dataset

Fig. 5: Sample frames from videos of a person in the YTF database.

YouTube Faces (YTF) dataset [23] contains real-life videos of 1595 persons downloaded from YouTube. The videos of the dataset have been acquired in unconstrained environment and exhibit a wide range of appearance variations. Few sample images of a person from the dataset are shown in Fig. 5. The YouTube faces database and its evaluation protocol was originally developed for face verification. In order to evaluate our method and the other image set classification methods for the task of face identification, we select a subset of the dataset with four or more videos per person. Considering each video as an image set, we randomly choose three videos for training and the remaining videos are used for testing. Experiments are repeated five times for different random selections of the training and testing videos. The identification rates averaged over five iterations are reported in Table 2. The results suggest that due to the challenging nature of this dataset, the identification rates achieved by all methods are lower compared to all the other evaluated datasets. Our proposed method achieves an average identification rate of $51.45 \pm 3.06\%$, which is superior to the other methods. MSSRC [11] also achieves a very good identification rate of $50.64 \pm 2.23\%$ on this dataset. The achieved performance by our method and MSSRC [11] is significantly better than the other methods.

5.2.7 COX Dataset

Fig. 6: Example images of a person from COX dataset.

The COX [24] dataset contains 4000 uncontrolled low resolution video sequences of 1000 subjects. The videos have been captured inside a gymnasium with subjects walking naturally and without any restriction on expression and head orientation. The dataset contains four videos per subject. The face resolution, head orientation and lighting conditions in each video are significantly different from the others. Sample images of a subject from this dataset are shown in Fig. 6. For our experiments, we consider images of each video as an image set and follow a leave-one-out strategy. Specifically, one image set per subject is used for testing whereas the remaining are used for training. For consistency, four runs of experiments are performed by swapping the training and testing image sets. The average identification rates of all methods in Table 2 show that the proposed method, MDA [5], MSSCR [11], SANP [8] and RNP [10] achieve good performance on this dataset. The CMC curves for the top performing methods in Fig 3f show that MDA [5] achieves the best rank $1-3$ identification rates, whereas the proposed method outperforms others for rank $4-10$ identification rates.

5.2.8 ETH-80 Dataset

ETH-80 contains images of eight object categories which include apples, cars, cows, cups, dogs, horses, pears and tomatoes. Each object category further includes ten subcategories such as different brands of cars or different breeds of dogs. Each subcategory has images under 41 orientations. For performance evaluation, we follow an experimental setup similar to [3], [5], [9]. Images of an

TABLE 3: Equal Error Rates of different methods

Methods	Honda	Mobo	YTC	Kinect	YTF	COX	PubFig
MSM [15]	7.58 ± 0.23	6.55 ± 1.61	23.90 ± 3.04	7.99 ± 0.28	38.93 ± 0.50	18.78 ± 4.73	19.01 ± 1.50
DCC [3]	6.18 ± 1.21	7.90 ± 1.14	21.68 ± 2.35	6.67 ± 1.30	31.45 ± 0.78	11.52 ± 3.98	31.34 ± 4.15
MMD [4]	5.20 ± 0.36	8.62 ± 2.15	20.53 ± 2.12	5.21 ± 0.45	32.28 ± 1.12	6.15 ± 0.85	31.25 ± 3.38
MDA [5]	5.60 ± 1.33	21.62 ± 9.9	19.26 ± 2.06	5.87 ± 1.39	33.72 ± 0.68	6.32 ± 0.90	36.96 ± 0.79
AHISD [6]	10.23 ± 0.09	9.90 ± 1.37	21.46 ± 2.14	9.27 ± 0.10	36.57 ± 0.75	8.99 ± 3.09	20.93 ± 0.67
CHISD [6]	8.62 ± 1.21	7.63 ± 1.25	21.23 ± 2.56	7.21 ± 0.4	34.31 ± 0.59	8.12 ± 2.91	19.15 ± 0.71
GEDA [7]	5.68 ± 1.35	8.68 ± 1.80	23.42 ± 3.51	6.08 ± 1.37	37.23 ± 0.77	19.51 ± 4.61	23.90 ± 1.29
SANP [8]	6.23 ± 1.21	4.11 ± 0.86	19.49 ± 2.44	6.73 ± 1.23	23.31 ± 3.12	10.25 ± 2.13	7.14 ± 0.59
CDL [9]	0.13 ± 0.16	3.91 ± 0.75	18.40 ± 2.04	2.17 ± 0.16	27.85 ± 0.83	8.31 ± 4.14	9.95 ± 0.93
MSSRC [11]	0.78 ± 0.67	1.44 ± 0.17	20.44 ± 2.43	4.25 ± 0.68	16.43 ± 1.65	5.76 ± 2.25	3.71 ± 0.21
SSDML [12]	17.71 ± 2.16	8.18 ± 3.25	17.34 ± 2.49	7.71 ± 2.22	37.57 ± 0.98	7.66 ± 3.18	12.05 ± 0.86
RNP [10]	12.50 ± 0.73	5.91 ± 0.73	24.55 ± 2.55	2.84 ± 0.82	37.59 ± 0.52	8.34 ± 1.86	10.79 ± 0.83
DRM-WV	$\mathbf{0.00 \pm 0.00}$	$\mathbf{1.46 \pm 0.51}$	$\mathbf{11.68 \pm 2.14}$	$\mathbf{0.39 \pm 0.04}$	$\mathbf{15.29 \pm 3.28}$	$\mathbf{4.27 \pm 3.32}$	$\mathbf{2.80 \pm 0.57}$

Comparison of methods for image set classification based face verification. The proposed DRMs based method achieves the lowest EER on all datasets

object in a subcategory are considered as an image set. For each object, five subcategories are selected for training and the the remaining five are used for testing. 10 runs of experiments are performed for different random selections of the training and testing sets. The average identification rates and standard deviations are presented in Table 2 and CMC curves for different methods are given in Fig. 3h. The proposed DRM based method achieves the highest identification rate and achieves a significant improvement over other methods.

5.3 Face Verification Experiments

We also present a comparison of different methods for image set classification based face verification. The performance is reported in terms of Equal Error Rate (EER), a rate where false acceptance rate becomes equal to false rejection rate. The average EER on face datasets for different methods are summarized in Table 3. The results suggest that the proposed method consistently achieves the lowest EER for all datasets. Specifically, the difference with other methods is more pronounced in case of YouTube Celebrities dataset where the proposed DRM based method achieves an EER of 11.68% compared to the second lowest error rate of 17.34% by SSDML. For the YouTube Celebrities datasets, we also present the Receiver Operating Characteristic (ROC) curves in Fig. 3i for different methods. The proposed method clearly outperforms the others by producing the highest true positive rates against all false positive rates.

5.4 Ablative Analysis

Here we present an ablative analysis in order to study the effect of the different components on the overall performance of the proposed method. Specifically, the following aspects of the proposed framework are explored:

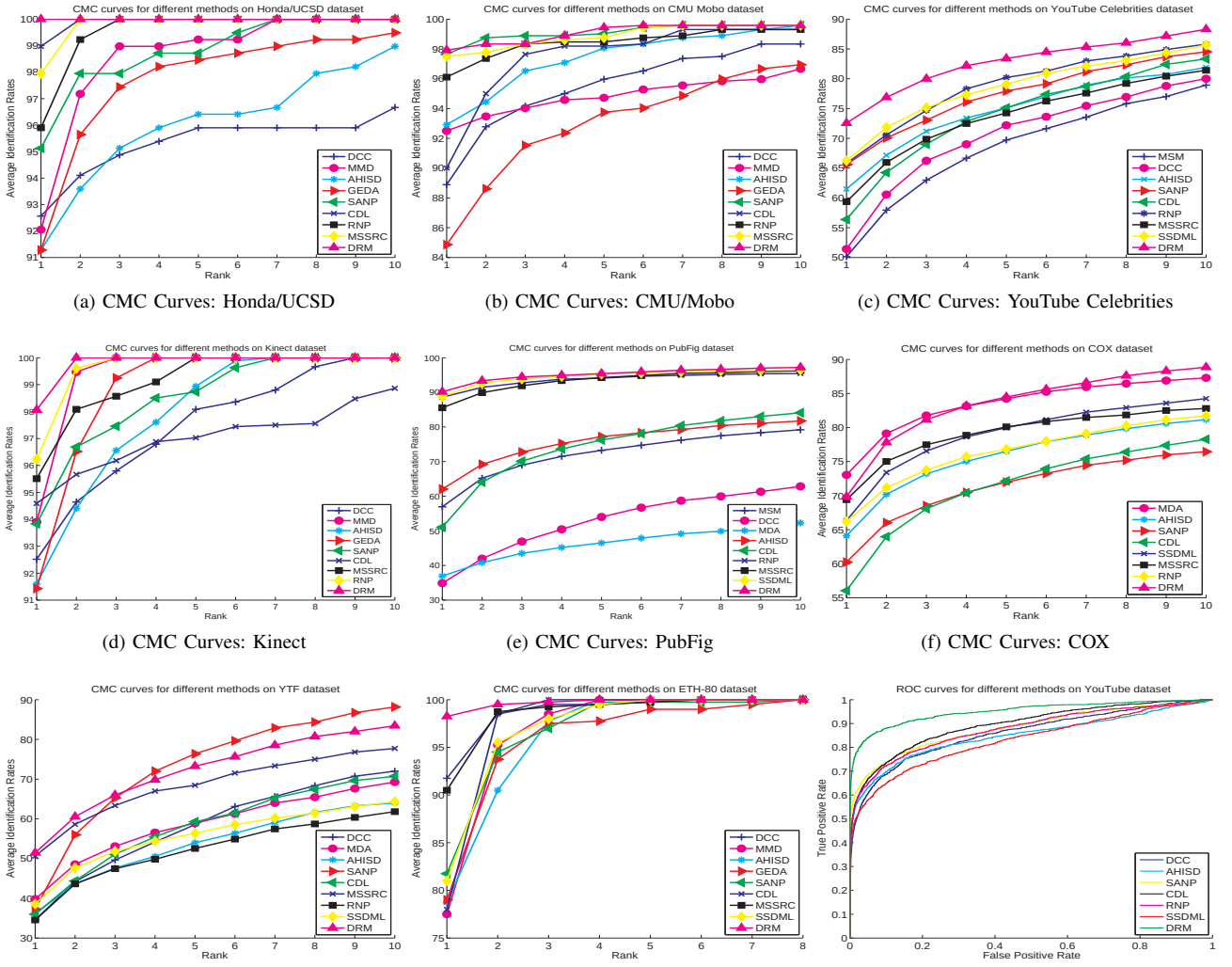
Effect of Template Initialization: We perform experiments on the YouTube Celebrities dataset without initialization *i. e.* the class-specific DRMs are learnt from a randomly initialized TDRM. The achieved performance is $63.26 \pm 4.10\%$, which is significantly lower than the performance achieved by the class-specific DRMs learnt from the initialized TDRM. This suggests that the template initialization is an important element of our proposed method.

Effect of Network Depth: Experiments are performed by changing the depth of the network to $1024-400-1024$ and $1024 - 400 - 100 - 40 - 100 - 400 - 1024$. The method achieves an identification rate of $66.67 \pm 4.87\%$ and $72.58 \pm 4.20\%$ respectively on the YouTube Celebrities dataset. The results suggest that reducing the depth of the network causes a significant performance degradation, whereas the gain in performance due to an increased network depth is very small. The selected network depth of $1024-400-100-400-1024$ is therefore a good tradeoff between computational complexity and performance.

Importance of Class-Specific Adaptation: We performed experiments by omitting the class-specific adaptation component of our method. Specifically, experiments are performed by only considering the initialized encoder part of the TDRM and adding a softmax layer of k nodes (where k is the total number of classes) at the end of the network. The achieved performance on the YouTube Celebrities dataset is $68.74 \pm 6.75\%$. In addition to its good performance, the class-specific adaptation makes our method easily scalable. Specifically, since our method learns a model for each class by using images of that class only, enrolling a new class would not require retraining on the complete training data. Instead, the class-specific model for the newly added class can be learnt completely independently by only using the images of that class.

5.5 Timing Analysis

A comparison of the computational complexity of all methods on the benchmark ETH-80 dataset using a Core 2 Quad Machine is presented in Table 4. The training time (in seconds) is given in Table 4 (a) while the time (in seconds) required to identify an image set from the training data is given in Table 4 (b). The proposed method requires comparatively more time for training. The training however is performed offline. The proposed method requires .026 seconds to identify an image set. This time is comparable to the fastest methods (MSM, MDA, GEDA and RNP).



(a) CMC Curves: Honda/UCSD (b) CMC Curves: CMU/Mobo (c) CMC Curves: YouTube Celebrities

(d) CMC Curves: Kinect (e) CMC Curves: PubFig (f) CMC Curves: COX

(g) CMC Curves: YouTube Faces (h) CMC Curves: ETH-80 (i) ROC Curves: YouTube Celebrities

Fig. 3: Performance Curves for different methods on all datasets. The CMC curves (a-h) show that the proposed method achieves the highest identification rates for all ranks on most of the dataset. The ROC curves for face verification experiments on YTC dataset in (i) show that the proposed method significantly outperforms the others. Figure best seen in colors.

The efficient testing is attributed to the fact that the proposed method only requires a few matrix multiplications (using Eqs 1 and 2) during testing. We note that the methods which do not require any training and instead directly compute a one-one set distance are very slow during testing. These methods could be computationally infeasible for larger gallery sizes. Unlike those methods, our proposed DRMs based method is easily scalable. Enrolling new classes would not require re-training on the complete dataset. Instead the class specific models for the added classes can be learnt independently of the existing classes. Furthermore, the training time for the proposed method does not increase with increased image resolution. Images of different resolutions are pre-processed (see Sec. 4.2) to fixed low dimensional input features and used by the proposed DRMs based method.

6 Conclusion

In this paper, we have proposed a novel deep learning framework for image set classification. Specifically, an adaptive multi-layer neural network structure has been introduced which is first pre-trained for appropriate parameter initialization and then fine-tuned for learning class-specific Deep Reconstruction Models (DRMs). By automatically discovering the underlying non-linear complex geometric surface, the DRMs can effectively model appearance variations within images of each class. The learnt DRMs are then used for a minimum reconstruction error based classification strategy during testing. The proposed framework has been extensively evaluated on a number of benchmark video datasets, an RGB-D Kinect dataset and an object dataset and state of the art performance has been achieved.

For future research, we plan to incorporate appropriate modifications into the proposed method to make it further

TABLE 4: Timing Analysis

Methods	Time	Methods	Time	Methods	Time
MSM	N/A	DCC	13.36	MMD	N/A
MDA	1.22	AHISD	N/A	CHISD	N/A
GEDA	2.7	SANP	N/A	CDL	76.21
RNP	N/A	MSSRC	N/A	SSDML	21.92
DRM-MV	278.8	DRM-WV	278.8		

(a) Training time (in seconds) for different methods. N/A means the method does not require training. Although the proposed method requires more time for training, it is easily scalable to new enrollments and the training time is independent of the image resolution.

Methods	Time	Methods	Time	Methods	Time
MSM	.045	DCC	.311	MMD	8.43
MDA	.005	AHISD	.095	CHISD	.213
GEDA	.068	SANP	105.7	CDL	1.40
RNP	.027	MSSRC	4.78	SSDML	.577
DRM-MV	.026	DRM-WV	.026		

(b) Testing time (the time in seconds required to identify an image set from the gallery) for different methods. The testing time for the proposed method is comparable to the fastest methods.

robust to noisy image data, outliers and diverse within-set data variations. We will also explore convolutional deep reconstruction models for image set classification.

ACKNOWLEDGMENTS

This work is supported by SIRF scholarship from the University of Western Australia (UWA) and ARC grant DPI10102166.

REFERENCES

[1] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld, "Face recognition: A literature survey," *ACM Comput. Surv.*, vol. 35, pp. 399–458, December 2003.

[2] A. Mian, M. Bennamoun, and R. Owens, "An efficient multimodal 2d-3d hybrid approach to automatic face recognition," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 29, no. 11, pp. 1927–1943, 2007.

[3] T.-K. Kim, J. Kittler, and R. Cipolla, "Discriminative learning and recognition of image set classes using canonical correlations," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 29, no. 6, pp. 1005–1018, 2007.

[4] R. Wang, S. Shan, X. Chen, and W. Gao, "Manifold-manifold distance with application to face recognition based on image set," in *Computer Vision and Pattern Recognition, CVPR 2008. IEEE Conference on*, pp. 1–8.

[5] R. Wang and X. Chen, "Manifold discriminant analysis," in *Computer Vision and Pattern Recognition, CVPR 2009. IEEE Conference on*. IEEE, 2009, pp. 429–436.

[6] H. Cevikalp and B. Triggs, "Face recognition based on image sets," in *Computer Vision and Pattern Recognition, CVPR 2010. IEEE Conference on*. IEEE, 2010, pp. 2567–2573.

[7] M. T. Harandi, C. Sanderson, S. Shirazi, and B. C. Lovell, "Graph embedding discriminant analysis on grassmannian manifolds for improved image set matching," in *Computer Vision and Pattern Recognition, CVPR 2011. IEEE Conference on*, pp. 2705–2712.

[8] Y. Hu, A. S. Mian, and R. Owens, "Face recognition using sparse approximated nearest points between image sets," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 34, no. 10, pp. 1992–2004, 2012.

[9] R. Wang, H. Guo, L. Davis, and Q. Dai, "Covariance discriminative learning: A natural and efficient approach to image set classification," in *Computer Vision and Pattern Recognition, CVPR 2012. IEEE Conference on*, pp. 2496–2503.

[10] M. Yang, P. Zhu, L. V. Gool, and L. Zhang, "Face recognition based on regularized nearest points between image sets," *Automatic Face and Gesture Recognition, FG 2013. 10th IEEE International Conference and Workshops on*, vol. 0, pp. 1–7, 2013.

[11] E. Ortiz, A. Wright, and M. Shah, "Face recognition in movie trailers via mean sequence sparse representation-based classification," in *Computer Vision and Pattern Recognition, CVPR 2013. IEEE Conference on*, 2013, pp. 3531–3538.

[12] P. Zhu, L. Zhang, W. Zuo, and D. Zhang, "From point to set: Extend the learning of distance metrics," in *International Conference on Computer Vision, ICCV 2013. IEEE Conference on*.

[13] M. Hayat, M. Bennamoun, and S. An, "Learning non-linear reconstruction models for image set classification," in *Computer Vision and Pattern Recognition, CVPR 2014. IEEE Conference on*, 2014, pp. 1915–1922.

[14] M. Hayat, M. Bennamoun, and S. An, "Reverse Training: An efficient Approach for Image Set Classification," in *Computer Vision–ECCV 2014*. Springer, 2014.

[15] O. Yamaguchi, K. Fukui, and K.-i. Maeda, "Face recognition using temporal image sequence," in *Automatic Face and Gesture Recognition, FG 1998. Third IEEE International Conference on*, pp. 318–323.

[16] M. Nishiyama, M. Yuasa, T. Shibata, T. Wakasugi, T. Kawahara, and O. Yamaguchi, "Recognizing faces of moving people by hierarchical image-set matching," in *Computer Vision and Pattern Recognition, CVPR 07. IEEE Conference on*, pp. 1–8.

[17] K.-C. Lee, J. Ho, M.-H. Yang, and D. Kriegman, "Video-based face recognition using probabilistic appearance manifolds," in *Computer Vision and Pattern Recognition, CVPR 2003. IEEE Computer Society Conference on*, vol. 1, pp. I–313.

[18] R. Gross and J. Shi, "The cmu motion of body (mobo) database," Tech. Rep., 2001.

[19] M. Kim, S. Kumar, V. Pavlovic, and H. Rowley, "Face tracking and recognition with visual constraints in real-world videos," in *Computer Vision and Pattern Recognition, CVPR 2008. IEEE Conference on*

[20] B. Y. Li, A. S. Mian, W. Liu, and A. Krishna, "Using kinect for face recognition under varying poses, expressions, illumination and disguise," in *Applications of Computer Vision, WACV 2013. IEEE Workshop on*, pp. 186–192.

[21] G. Fanelli, J. Gall, and L. Van Gool, "Real time head pose estimation with random regression forests," in *Computer Vision and Pattern Recognition, CVPR 2011. IEEE Conference on*. IEEE, 2011, pp. 617–624.

[22] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar, "Attribute and Simile Classifiers for Face Verification," in *IEEE International Conference on Computer Vision (ICCV)*, Oct 2009.

[23] L. Wolf, T. Hassner, and I. Maoz, "Face recognition in unconstrained videos with matched background similarity," in *Computer Vision and Pattern Recognition, CVPR 2011. IEEE Conference on*. IEEE, 2011, pp. 529–534.

[24] Z. Huang, S. Shan, H. Zhang, S. Lao, A. Kuerban, and X. Chen, "Benchmarking still-to-video face recognition via partial and local linear discriminant analysis on COX-S2V dataset," in *Computer Vision–ACCV 2012*. Springer, 2013, pp. 589–600.

[25] B. Leibe and B. Schiele, "Analyzing appearance and contour based methods for object categorization," in *Computer Vision and Pattern Recognition, CVPR 2003. IEEE Conference on*, vol. 2. IEEE, 2003, pp. II–409.

[26] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, no. 8, pp. 1798–1828, 2013.

[27] S. H. Khan, M. Bennamoun, F. Sohel, R. Togneri, "Automatic Feature Learning for Robust Shadow Detectionin" in *Computer Vision and Pattern Recognition, CVPR 2014. IEEE Conference on*, 2014, pp. 1939–1946.

[28] O. Arandjelovic, G. Shakhnarovich, J. Fisher, R. Cipolla, and T. Darrell, "Face recognition with image sets using manifold density divergence," in *Computer Vision and Pattern Recognition, CVPR 2005. IEEE Conference on*, vol. 1. IEEE, 2005, pp. 581–588.

[29] P. Turaga, A. Veeraraghavan, A. Srivastava, and R. Chellappa, "Statistical computations on grassmann and stiefel manifolds for image and video-based recognition," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 33, no. 11, pp. 2273–2286, 2011.

[30] M. Hayat, M. Bennamoun, "An Automatic Framework for Textured 3D Video-based Facial Expression Recognition" *Affective Computing, IEEE Transactions on*, 2014.

[31] J. Hamm and D. D. Lee, "Grassmann discriminant analysis: a unifying view on subspace-based learning," in *25th international conference on Machine learning, ICML 2008*. ACM, 2008, pp. 376–383.

[32] M. T. Harandi, C. Sanderson, A. Wiliem, and B. C. Lovell, "Kernel analysis over riemannian manifolds for visual recognition of actions, pedestrians and textures," in *Applications of Computer Vision (WACV), 2012 IEEE Workshop on*. IEEE, 2012, pp. 433–439.

[33] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[34] P. Smolensky, "Information processing in dynamical systems: Foundations of harmony theory," 1986.

[35] G. W. Taylor, R. Fergus, Y. LeCun, and C. Bregler, "Convolutional learning of spatio-temporal features," in *Computer Vision–ECCV 2010*. Springer, 2010, pp. 140–153.

[36] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[37] G. Hinton, S. Osindero, M. Welling, and Y.-W. Teh, "Unsupervised discovery of nonlinear structure using contrastive backpropagation," *Cognitive science*, vol. 30, no. 4, pp. 725–731, 2006.

[38] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, no. 1, pp. 321–357, 2002.

[39] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, no. 7, pp. 971–987, 2002.

[40] I. Naseem, R. Togneri, and M. Bennamoun, "Linear regression for face recognition," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 32, no. 11, pp. 2106–2112, 2010.

[41] P. Viola and M. J. Jones, "Robust real-time face detection," *International Journal of Computer Vision, IJCV*, vol. 57, no. 2, pp. 137–154, 2004.

[42] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, "Incremental learning for robust visual tracking," *International Journal of Computer Vision, IJCV*, vol. 77, no. 1-3, pp. 125–141, 2008.

[43] G. Hinton, "A practical guide to training restricted boltzmann machines," *Momentum*, vol. 9, no. 1, 2010.

[44] Y. Bengio, "Practical recommendations for gradient-based training of deep architectures," in *Neural Networks: Tricks of the Trade*. Springer, 2012, pp. 437–478.

[45] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 31, pp. 210–227, 2009.

Munawar Hayat Munawar Hayat received his Bachelor of Engineering degree from National University of Science and Technology (NUST) in 2009. Later, he was awarded Erasmus Mundus Scholarship for a joint European Masters degree program. He is currently a PhD candidate at The University of Western Australia (UWA) sponsored by the Scholarship for International Research Fees (SIRF). His research interests include computer vision, signal and image processing, pattern recognition and machine learning.

Mohammed Bennamoun Mohammed Bennamoun received his MSc degree in control theory from Queens University, Kingston, Canada, and his Ph.D. degree in computer vision from Queens/QUT in Brisbane, Australia. He is currently a Winthrop Professor at the University of Western Australia, Australia. He served as a guest editor for a couple of special issues in International journals such as the International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI). He was selected to give conference tutorials at the European Conference on Computer Vision (ECCV) and the International Conference on Acoustics Speech and Signal Processing (ICASSP). He organized several special sessions for conferences, e.g., the IEEE International Conference in Image Processing (ICIP). He also contributed in the organization of many local and international conferences. His research interests include control theory, robotics, obstacle avoidance, object recognition, artificial neural networks, signal/image processing, and computer vision. He published more than 200 journal and conference publications.

Senjian An received his B.S degree from Shandong University, the M.S. degree from the Chinese Academy of Sciences, and the Ph.D. degree from Peking University, China. He is currently a Research Assistant Professor at the School of Computer Science and Software Engineering, The University of Western Australia. His research interests include machine learning, image processing, object detection and recognition.