Deep Sparse Representation-based Classification

Mahdi Abavisani, *Student Member, IEEE* and Vishal M. Patel, *Senior Member, IEEE*

***Abstract*—We present a transductive deep learning-based formulation for the sparse representation-based classification (SRC) method. The proposed network consists of a convolutional autoencoder along with a fully-connected layer. The role of the autoencoder network is to learn robust deep features for classification. On the other hand, the fully-connected layer, which is placed in between the encoder and the decoder networks, is responsible for finding the sparse representation. The estimated sparse codes are then used for classification. Various experiments on three different datasets show that the proposed network leads to sparse representations that give better classification results than state-of-the-art SRC methods. The source code is available at: github.com/mahdiabavisani/DSRC.**

***Index Terms*—Deep learning, sparse representation-based classification, deep sparse representation-based classification.**

I. Introduction

Sparse coding has become widely recognized as a powerful tool in signal processing and machine learning with various applications in computer vision and pattern recognition [1]–[3]. Sparse representation-based classification (SRC) as an application of sparse coding was first proposed in [1], and was shown to provide robust performance on various face recognition datasets. Since then, SRC has been used in numerous applications [4]–[9]. In SRC, an unlabeled sample is represented as a sparse linear combination of the labeled training samples. This representation is obtained by solving a sparsity-promoting optimization problem. Once the representation is found, the label is assigned to the test sample based on the minimum reconstruction error rule [1].

The SRC method is based on finding a linear representation of the data. However, linear representations are almost always inadequate for representing non-linear structures of the data which arise in many practical applications. To deal with this issue, some works have exploited the kernel trick to develop non-linear extensions of the SRC-based methods [10]–[20]. Kernel SRC methods require the use of a pre-determined kernel function such as polynomial or Gaussian. Selection of the kernel function and its parameters is an important issue in training when kernel SRC methods are used for classification.

In this paper, we propose a deep neural network-based framework that finds an explicit nonlinear mapping of data, while simultaneously obtaining sparse codes that can be used for classification. Learning nonlinear mappings with neural networks has been shown to produce remarkable improvements in subspace clustering tasks [21], [22]. We introduce a transductive model, which accepts a set of training and test samples, learns a mapping that is suitable for sparse representation, and recovers the corresponding sparse codes. Our model consists of an encoder that is responsible for learning the mapping, a sparse coding layer which mimics the task of constructing the mapped test samples by a combination of the mapped training samples, and a decoder that is used for training the networks.

M. Abavisani is with the department of Electrical and Computer Engineering at Rutgers University, Piscataway, NJ USA. email: mahdi.abavisani@rutgers.edu. Vishal M. Patel is with the department of Electrical and Computer Engineering at Johns Hopkins University, Baltimore, MD USA. email: vpatel36@jhu.edu. This work was partially supported by the NSF grant 1618677 and by US Office of Naval Research (ONR) Grant YIP N00014-16-1-3134.

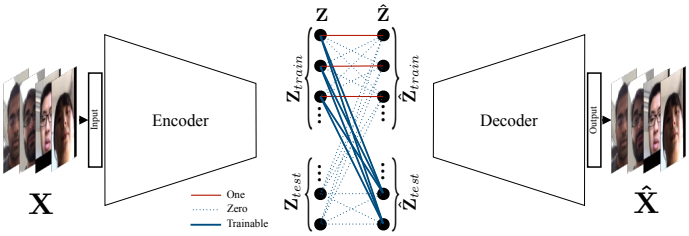


Fig. 1. An overview of the proposed deep SRC network. The trainable parameters of sparse coding layer are depicted with solid blue lines. Note that $\mathbf{Z}_{train} = \hat{\mathbf{Z}}_{train}$, and $\mathbf{Z}_{test} \approx \hat{\mathbf{Z}}_{test} = \mathbf{Z}_{train}\mathbf{A}$.

A. Sparse representation-based classification

In SRC, given a set of labeled training samples, the goal is to classify an unseen set of test samples. Suppose that we collect all the vectorized training samples with the label i in the matrix $\mathbf{X}^i_{train} \in \mathbb{R}^{d_0 \times n_i}$, where d_0 is the dimension of each sample and n_i is the number of samples in class i, then the training matrix can be constructed as

$$\mathbf{X}_{train} = [\mathbf{X}^1_{train}, \mathbf{X}^2_{train}, \cdots, \mathbf{X}^K_{train}] \in \mathbb{R}^{d_0 \times n} \quad (1)$$

where $n_1 + n_2 + \cdots + n_K = n$ and we have a total of K classes.

In SRC, it is assumed that an observed sample $\mathbf{x}_{test} \in \mathbb{R}^{d_0}$ can be well approximated by a linear combination of the samples in $\mathbf{X}^i_{train}$ if $\mathbf{x}_{test}$ is from class i. Thus, it is possible to predict the class of a given unlabeled data by finding a set of samples in the training set that can better approximate $\mathbf{x}_{test}$. Mathematically, these samples can be found by solving the following optimization problem

$$\min_{\boldsymbol{\alpha}} \|\boldsymbol{\alpha}\|_0 \ \text{ s.t. } \ \mathbf{x}_{test} = \mathbf{X}_{train}\boldsymbol{\alpha}, \quad (2)$$

where $\|\boldsymbol{\alpha}\|_0$ counts the number of non-zero elements in $\boldsymbol{\alpha}$. The minimization problem (2) finds a sparse solution for the linear system. However, since the optimization problem (2) is an NP-hard problem, in practice, a sparsity constraint is enforced by the ℓ_1-norm of $\boldsymbol{\alpha}$ which is a convex relaxation of the above problem [23], [24]. Thus, in practice the following minimization problem is solved to obtain the sparse codes

$$\min_{\boldsymbol{\alpha}} \|\mathbf{x}_{test} - \mathbf{X}_{train}\boldsymbol{\alpha}\|_2^2 + \lambda_0\|\boldsymbol{\alpha}\|_1, \quad (3)$$

where λ_0 is a positive regularization parameter. Once $\boldsymbol{\alpha}$ is found, one can estimate the class label of $\mathbf{x}_{test}$ as follows

$$\text{class}(\mathbf{x}_{test}) = \arg\min_k \|\mathbf{x}_{test} - \mathbf{X}_{train}\delta_k(\boldsymbol{\alpha})\|_2^2, \quad (4)$$

where $\delta_k(\cdot)$ is the characteristic function that selects the coefficients associated with the class i.

II. Deep sparse representation-based classification network

We develop a transductive classification model based on sparse representations. In a transductive model, as opposed to inductive models, both training and test sets are observed, and the learning process pursues reasoning from the specific training samples to a specific set of test cases [25]. We build our method based on convolutional autoencoders. In particular, our network contains an encoder, a sparse coding layer, and a decoder. The encoder receives both the training and test sets as raw data inputs and extracts abstract features from them. The sparse coding layer recovers the test cases by a sparse linear combination of the training samples, and concatenates them along with the training features which are then fed to the decoder. The decoder maps both the training embeddings and the recovered test embeddings back to the original representation of the data. Figure 1 gives an overview of the proposed deep SRC (DSRC) framework.

Sparse representation: Let $\mathbf{X}_{train} \in \mathbb{R}^{d_0 \times n}$ and, $\mathbf{X}_{test} \in \mathbb{R}^{d_0 \times m}$ be the given vectorized training and testing data, respectively. We feed $\mathbf{X} = [\mathbf{X}_{train}, \mathbf{X}_{test}]$ to the encoder, where it develops the corresponding embedding features $\mathbf{Z} = [\mathbf{Z}_{train}, \mathbf{Z}_{test}] \in \mathbb{R}^{d_z \times (m+n)}$. The minimization problem (3) for a single test observation can be re-written for a matrix of testing embedding features as

$$\min_{\mathbf{A}} \|\mathbf{Z}_{test} - \mathbf{Z}_{train}\mathbf{A}\|_F^2 + \lambda_0 \|\mathbf{A}\|_1, \quad (5)$$

where $\mathbf{A} \in R^{n \times m}$ is the coefficient matrix that contains the sparse codes in its columns, and λ_0 is a positive regularization parameter. Note that the first penalty term in equation (5) is equivalent to the penalty term used for a fully-connected neural network layer with the input of $\mathbf{Z}_{train}$, the output of $\mathbf{Z}_{test}$ and trainable parameters of $\mathbf{A}$. As a result, considering the sparsity constraint, one can model the optimization problem (5) in a neural network framework with a fully-connected layer with sparse parameters which have no non-linearity activation or bias nodes. We use such a model inside our sparse coding layer to find the sparse codes for the observed test set.

The sparse coding layer is located between the encoder and decoder networks. Its task for $\mathbf{Z}_{train}$ is to pass them to the decoder, and for the test features $\mathbf{Z}_{test}$ it will pass their reconstructions that are found from (5), as $\mathbf{Z}_{train}\mathbf{A}$, to the decoder. Thus, assuming that $\hat{\mathbf{Z}}_{train}$ and $\hat{\mathbf{Z}}_{test}$ are the outputs of the sparse coding layer for training and testing features, we have

$$\hat{\mathbf{Z}}_{train} = \mathbf{Z}_{train}\mathbf{I}_n, \quad \hat{\mathbf{Z}}_{test} = \mathbf{Z}_{train}\mathbf{A}, \quad (6)$$

Algorithm 1 Deep sparse representation-based classification

1: **procedure** DSRC($\mathbf{X}_{train}, \mathbf{X}_{test}, \lambda_0, \lambda_1$).
2: Construct $\mathbf{X} = [\mathbf{X}_{train}, \mathbf{X}_{test}]$.
3: Find $\mathbf{A}$ via $\boldsymbol{\Theta}$ by solving the optimization problem (8).
4: Classify the test samples using (9) .
5: **end procedure**

where $\mathbf{I}_n \in \mathbb{R}^{n \times n}$ is the identity matrix. Therefore, if the decoder's input is $\hat{\mathbf{Z}} = [\hat{\mathbf{Z}}_{train}, \hat{\mathbf{Z}}_{test}]$, from (6) we can calculate $\hat{\mathbf{Z}}$ as $\hat{\mathbf{Z}} = \mathbf{Z}\boldsymbol{\Theta}_{sc}$, where

$$\boldsymbol{\Theta}_{sc} = \begin{bmatrix} \mathbf{I}_n & \mathbf{A} \\ \mathbf{0}_{n \times m} & \mathbf{0}_m \end{bmatrix}. \quad (7)$$

In equation (7), $\mathbf{0}_{n \times m} \in \mathbb{R}^{n \times m}$ and $\mathbf{0}_m \in \mathbb{R}^{m \times m}$ are zero matrices. One can write an end-to-end training objective that includes sparse coding and training of the encoder-decoder as follows

$$\min_{\boldsymbol{\Theta}} \|\mathbf{Z} - \mathbf{Z}\boldsymbol{\Theta}_{sc}\|_F^2 + \lambda_0 \|\boldsymbol{\Theta}_{sc}\|_1 + \lambda_1 \|\mathbf{X} - \hat{\mathbf{X}}\|_F^2, \quad (8)$$

where $\boldsymbol{\Theta}$ is the union of all the trainable parameters including encoder and decoder's parameters and $\mathbf{A}$. Here, $\hat{\mathbf{X}} = [\hat{\mathbf{X}}_{train}, \hat{\mathbf{X}}_{test}]$ is the output of the decoder (i.e. reconstructions), and λ_0 and λ_1 are positive regularization parameters. Note that the optimization problem (8) simultaneously finds sparse codes $\mathbf{A}$ and a set of desirable embedding features $\mathbf{Z}$ that are especially suitable for providing efficient sparse codes.

Classification: Once the sparse coefficient matrix $\mathbf{A}$ is found, it can be used for associating the class labels to the test samples. For each test sample $\mathbf{x}_{test}^i$ in $\mathbf{X}_{test}$, its embedding features $\mathbf{z}_{test}^i$, and the corresponding sparse code column $\boldsymbol{\alpha}^i$ in $\mathbf{A}$ are used to estimate the class labels as follows

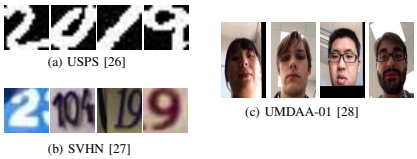
$$\text{class}(\mathbf{x}_{test}^i) = \arg\min_k \|\mathbf{z}_{test}^i - \mathbf{Z}_{train}\delta_k(\boldsymbol{\alpha}^i)\|_2^2. \quad (9)$$

The proposed DSRC method is summarized in Algorithm 1.

III. Experimental results

In this section, we evaluate our method against state-of-the-art SRC methods. The USPS handwritten digits dataset [26], the street view house numbers (SVHN) dataset [27], and the UMDAA-01 face recognition dataset [28] are used in our experiments. Figure 2 (a), (b), and (c) show sample images from these datasets. Since the number of parameters in the sparse coding layer scales with the multiplication of training and testing sizes, we randomly select a smaller subset of the used datasets and perform all the experiments on the selected subset. In all the experiments, the input images are resized to 32×32.

We compare our method with the standard SRC method [1], Kernel SRC (KSRC) [14], SRC on features extracted from an autoencoder with similar architecture to our network (AE-SRC), and SRC on features extracted from the state-of-the-art pre-trained networks. In our experiment with the pre-trained networks, the networks are pre-trained on the Imagenet dataset [29]. For this purpose, we use the following



(a) USPS [26]

(b) SVHN [27]

(c) UMDAA-01 [28]

Fig. 2. Sample images from (a) USPS [26], (b) SVHN [27], and (c) UMDAA-01 [28].

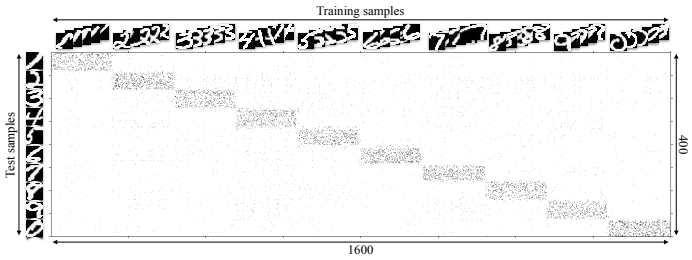


Fig. 3. Visualization of the sparse coding matrix ($\mathbf{A}$) in the experiment with the USPS dataset. Note that for better visualization the absolute value of the transposed $\mathbf{A}$ (i.e. $|\mathbf{A}^T|$) is shown.

four popular network architectures: VGG-19 [30], Inception-V3 [31], Resnet-50 [32] and Densenet-169 [33]. We feed these networks with our datasets, extract the features corresponding to the last layer before classification, and pass them to the classical SRC algorithm. Note these networks accept 224×224 inputs. Thus, as a preprocessing step, we resample the input images to 224×224 images before feeding them to the pre-trained networks.

We compare different methods in terms of their five-fold averaged classification accuracy. In all the experiments, unless otherwise stated, we randomly split the datasets into sets of training and testing samples, where 20% of the samples are used for testing, and 80% of the samples are used as the training set.

Network structure: The encoder network of our model consists of stacked four convolutional layers, and the decoder has three fractionally-strided convolution layers (also known as deconvolution layers). Each plugged in convolution or fractionally-strided convolution is coupled with a ReLu nonlinearity as well, but does not have a batch-norm layer. Table I gives the details of the network, including the kernel sizes and the number of filters.

TABLE I
DETAILS OF OUR NETWORKS. NOTE THAT THE NUMBER OF PARAMETERS IN THE SPARSE CODING LAYER RELY ON THE SIZE OF DATASET INCLUDING THE n TRAINING AND m TEST SAMPLES.

	Layer	Input	Output	Kernel	(stride, pad)
Encoder	Conv 1	$\mathbf{X}$	Conv 1	$1\times5\times5\times10$	(2,1)
	Conv 2	Conv 1	Conv 2	$1\times3\times3\times20$	(2,1)
	Conv 3	Conv 2	Conv 3	$1\times3\times3\times30$	(1,0)
	Conv 4	Conv 3	$\mathbf{Z}$	$1\times3\times3\times30$	(1,0)
Sparse coding layer	$\mathbf{\Theta}_{sc}$	$\mathbf{Z}$	$\hat{\mathbf{Z}}$	$m\times n$ Parameters	-
Decoder	deconv 1	$\hat{\mathbf{Z}}$	deconv 1	$1\times3\times3\times30$	(1,0)
	deconv 2	deconv 1	deconv 2	$1\times3\times3\times20$	(2,1)
	deconv 3	deconv 2	$\hat{\mathbf{X}}$	$1\times5\times5\times10$	(2,1)

Training details: We implemented our method with Tensorflow-1.4 [34]. We use the adaptive momentum-based gradient descent method (ADAM) [35] to minimize the loss function, and apply a learning rate of 10^{-3}. Before we start training on our objective function, in each experiment, we pre-train our encoder and decoder on the dataset without the sparse coding layer. In particular, we pre-train the encoder-decoder for $20k$ epochs with the objective of $\min_{\hat{\mathbf{\Theta}}} \|\mathbf{X}-\hat{\mathbf{X}}\|_F^2$, where $\hat{\mathbf{\Theta}}$ indicates the union of parameters in the encoder and decoder networks. We use a batch size of 100 for this stage. However, in the actual stage of training, we feed all the samples including the training and testing samples as a single large batch. We set the regularization parameters as $\lambda_0 = 1$ and $\lambda_1 = 8$ in all the experiments.

A. USPS digits

The first set of experiments is conducted on the USPS handwritten digits dataset [26]. This dataset contains 7291 training and 2007 test grayscale images of ten digits (0-9). Figure 2 (a), shows example images from this dataset. We perform the experiments on a subset with a total size of 2000 samples. In particular, we randomly select 160 and 40 samples per digit from the training and testing sets, respectively. The first row of Table II shows the performance of various SRC methods. As can be observed from this table, the proposed method performs significantly better than the other methods including the classical and deep learning-based methods.

Figure 3 shows the coefficient matrix $\mathbf{A}$, extracted from $\mathbf{\Theta_{sc}}$, the matrix of the network trained for this experiment. For better visualization, we show the absolute value of the transposed $\mathbf{A}$ (i.e. $|\mathbf{A}^T|$). Thus, each row of the matrix in Figure 3 corresponds to the sparse codes for one of the test samples. Similarly, columns in this figure are coefficients related to the training samples. This matrix is sparse and shows a block diagram pattern, where most of the non-zero coefficients for each test sample are those that correspond to the training samples with the same class as the observed test sample.

Analysis of the network: To understand the effects of some of our model choices, we compare the performance of our DSRC method with variations of it by changing the regularization norm on $\mathbf{\Theta_{sc}}$ in the loss function (8). We replace the term $\|\mathbf{\Theta}_{sc}\|_1$ in (8) by $\|\mathbf{\Theta}_{sc}\|_p$, where $p = 0.5, 1.5$ and 2, and report their performances by *DSRC*$_{0.5}$, *DSRC*$_{1.5}$ and *DSRC*$_2$, respectively.

In addition, if we do not follow the specific structure described in equation (7), and instead have a fully connected layer with $(m+n)^2$ parameters which receives $\mathbf{Z}$ and reconstructs $\hat{\mathbf{Z}}$, the architecture of the network will be similar to the deep subspace clustering networks (DSC) proposed in [21] for the task of subspace clustering. As an ablation study, we use this method to extract sparse codes and then apply the same classification rule as in (9) to estimate class labels for the test set. We call this method *DSC-SRC*.

Table III reveals that while the regularization norm on the coefficient matrix is selected between ℓ_1 and ℓ_2, it does not have much effect on the performance of the classification task. However, in our experiments, we observed that for norms smaller than 1, the problem is not stable and often does not converges. In addition, *DSC-SRC* cannot provide a

Dataset	SRC	KSRC	AE-SRC	VGG19-SRC	InceptionV3-SRC	Resnet50-SRC	Denesnet169-SRC	DSRC (ours)
USPS	87.78	91.34	88.65	91.27	93.51	95.75	95.26	**96.25**
SVHN	15.71	27.42	18.69	52.86	41.14	47.88	37.65	**67.75**
UMDAA-01	79.00	81.37	86.70	82.68	86.15	91.84	86.35	**93.39**

TABLE II
SPARSE REPRESENTATION-BASED CLASSIFICATION ACCURACY OF DIFFERENT METHODS.

	DSRC	DSC-SRC	$\text{DSRC}_{0.5}$	$\text{DSRC}_{1.5}$	DSRC_{2}
USPS	**96.25**	78.25	N/C	95.75	**96.25**

TABLE III
THE CLASSIFICATION ACCURACY CORRESPONDING TO THE ABLATION STUDY. N/C REFERS TO THE CASES WHERE THE LEARNING PROCESS DID NOT CONVERGE.

desirable performance. Note that the fully-connected layer in this method (counterpart to our sparse coding layer) does not limit the testing features to be reconstructed with only the training features. As a result, it is possible that testing features shape an isolated group that does not have a strong connection to the training features. This makes it more difficult to estimate a label for the test samples.

B. Street view house numbers

The SVHN dataset [27] contains 630,420 color images of real-world house numbers collected from Google Street View images. This dataset has three splits as the training set with 73,257 images, the testing set with 26,032 images and an extra set containing 531,131 additional samples. In this experiment, similar to our experiments on MNIST, we randomly select 160 images per digit from the training split and 40 per digit from the test split. This dataset is much more challenging than MNIST. This is in part due to the large variations of data. Furthermore, many samples in this dataset contain multiple digits in an image. The task is to classify the center digit.

The second row in Table II compares the performance of different SRC methods. This table demonstrates the advantage of our method. While the classification task is much more challenging on SVHN than MNIST, the gap between the performance of our method and the second best performance is even more. The next best performing method is *VGG19-SRC* which performs 14.86% behind the accuracy of our method.

C. UMD mobile faces

The UMD mobile face dataset (UMDAA-01) [28] contains 750 front-facing camera videos of 50 users captured while using a smartphone. This dataset has been collected over three different sessions. This dataset was originally collected for the active authentication task, but since its frames include challenging facial image instances with various illumination and pose conditions it has also been used for other tasks [36], [37]. In this experiment, we randomly select 50 facial images per subjects from the data in Session 1. Figure 2 shows some sample images from this dataset.

The performance of various SRC methods on the UMDAA-01 dataset are tabulated in the third row of Table II. As can be seen, our proposed DSRC method similar to the experiments with SVHN provides remarkable improvements as compared to the other SRC methods. This clearly shows that more challenging datasets are better represented by our method. This is because our method not only efficiently finds the sparse codes, but also it seeks for a representation of data (the output of the encoder) that is especially suitable for sparse representation.

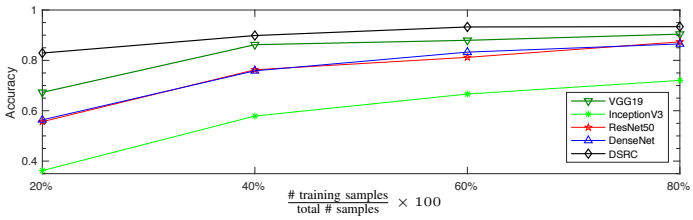


Fig. 4. Effect of the number of training samples on the performance of different classification networks. The figure shows five-fold averaged classification accuracies of the methods trained on varying number of training samples in the UMDAA-01 dataset.

Comparison to state-of-the-art classification networks: While deep neural networks perform very well when they are trained on large datasets, in the case of limited number of labeled training samples, they often tend to overfit to the training samples. The objective of this experiment is to analyze the performance of our approach in such circumstances. We compare our method to the following classification networks: VGG-19 [30], Inception-V3 [31], Resnet-50 [32] and Densenet-169 [33]. We first pre-train the networks on the Imagenet dataset [29], and then fine-tune them on the available training samples in UMDAA-01.

Figure 4 shows the performance of the classification networks on four different versions of UMDAA-01 dataset with varying number of training samples. The four versions are created by randomly splitting the dataset into sets of training and testing samples that respectively contain 20%, 40%, 60% and 80% of the total number of samples as training samples and use the rest of samples as the testing set. As the figure suggests, accuracy improves by increasing the number of training samples in all the cases. However, the results show better performances for DSRC even when less training data is available.

IV. CONCLUSION

We presented an autoencoder-based sparse coding network for SRC. In particular, we introduced a sparse coding layer that is located between the conventional encoder and decoder networks. This layer recovers sparse codes of embedding features that are received from the encoder. The spare codes are later used to estimate the class labels of testing samples. We discussed a framework that allows an end-to-end training. Various experiments on three different image classification datasets showed that the proposed network leads to sparse representations that give better classification results than state-of-the-art SRC methods.

REFERENCES

[1] J. Wright, A.Y. Yang, A. Ganesh, S.S. Sastry, and Yi Ma, "Robust face recognition via sparse representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp. 210 –227, feb. 2009.

[2] Michal Aharon, Michael Elad, Alfred Bruckstein, et al., "K-svd: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Transactions on signal processing*, vol. 54, no. 11, pp. 4311, 2006.

[3] E. Elhamifar and R. Vidal, "Sparse subspace clustering: Algorithm, theory, and applications," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 11, pp. 2765–2781, 2013.

[4] Xue Mei and Haibin Ling, "Robust visual tracking and vehicle classification via sparse representation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 11, pp. 2259–2272, 2011.

[5] Sumit Shekhar, Vishal M Patel, Nasser M Nasrabadi, and Rama Chellappa, "Joint sparse representation for robust multimodal biometrics recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 1, pp. 113–126, 2014.

[6] Mahdi Abavisani, Mohsen Joneidi, Shideh Rezaeifar, and Shahriar Baradaran Shokouhi, "A robust sparse representation based face recognition system for smartphones," in *2015 IEEE Signal Processing in Medicine and Biology Symposium (SPMB)*. IEEE, 2015, pp. 1–6.

[7] Mohsen Joneidi, Alireza Zaeemzadeh, Shideh Rezaeifar, Mahdi Abavisani, and Nazanin Rahnavard, "Lfm signal detection and estimation based on sparse representation," in *2015 49th Annual Conference on Information Sciences and Systems (CISS)*. IEEE, 2015, pp. 1–5.

[8] Pramuditha Perera and Vishal M Patel, "Face-based multiple user active authentication on mobile devices," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 5, pp. 1240–1250, 2019.

[9] Mohsen Ghassemi, Zahra Shakeri, Anand D Sarwate, and Waheed U Bajwa, "Learning mixtures of separable dictionaries for tensor data: Analysis and algorithms," *arXiv preprint arXiv:1903.09284*, 2019.

[10] A. Shrivastava, V. M. Patel, and R. Chellappa, "Multiple kernel learning for sparse representation-based classification," *IEEE Transactions on Image Processing*, vol. 23, no. 7, pp. 3013–3024, July 2014.

[11] H. Van Nguyen, V. M. Patel, N. M. Nasrabadi, and R. Chellappa, "Design of non-linear kernel dictionaries for object recognition," *IEEE Transactions on Image Processing*, vol. 22, no. 12, pp. 5123–5135, Dec 2013.

[12] V. M. Patel and R. Vidal, "Kernel sparse subspace clustering," in *2014 IEEE International Conference on Image Processing (ICIP)*, Oct 2014, pp. 2849–2853.

[13] V. M. Patel, H. V. Nguyen, and R. Vidal, "Latent space sparse subspace clustering," in *2013 IEEE International Conference on Computer Vision*, Dec 2013, pp. 225–232.

[14] Li Zhang, Wei Da Zhou, Pei Chann Chang, Jing Liu, Zhe Yan, Ting Wang, and Fan Zhang Li, "Kernel sparse representation-based classifier," *IEEE Transactions on Signal Processing*, vol. 60, no. 4, pp. 1684 –1695, april 2012.

[15] S. Gao, I. W. Tsang, and L. T. Chia, "Kernel sparse representation for image classification and face recognition," in *European Conference on Computer Vision*, 2010, vol. 6314.

[16] X. T. Yuan and S. Yan, "Visual classification with multi-task joint sparse representation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[17] Hanchao Qi and Shannon Hughes, "Using the kernel trick in compressive sensing: Accurate signal recovery from fewer measurements," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2011, pp. 3940 –3943.

[18] Hien Van Nguyen, Vishal M Patel, Nasser M Nasrabadi, and Rama Chellappa, "Kernel dictionary learning," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2012*. IEEE, 2012, pp. 2021–2024.

[19] Jun Yin, Zhonghua Liu, Zhong Jin, and Wankou Yang, "Kernel sparse representation based classification," *Neurocomputing*, vol. 77, no. 1, pp. 120–128, 2012.

[20] V. M. Patel, H. V. Nguyen, and R. Vidal, "Latent space sparse and low-rank subspace clustering," *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 4, pp. 691–701, 2015.

[21] Pan Ji, Tong Zhang, Hongdong Lia, Mathieu Salzmann, and Ian Reid, "Deep subspace clustering networks," in *Advances in Neural Information Processing Systems*, 2017.

[22] M. Abavisani and V. M. Patel, "Deep multimodal subspace clustering networks," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 6, pp. 1601–1614, Dec 2018.

[23] Emmanuel J Candes and Terence Tao, "Decoding by linear programming," *IEEE transactions on information theory*, vol. 51, no. 12, pp. 4203–4215, 2005.

[24] David L Donoho, "For most large underdetermined systems of linear equations the minimal ??1-norm solution is also the sparsest solution," *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, vol. 59, no. 6, pp. 797–829, 2006.

[25] Alexander Gammerman, Volodya Vovk, and Vladimir Vapnik, "Learning by transduction," in *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc., 1998, pp. 148–155.

[26] Jonathan J. Hull, "A database for handwritten text recognition research," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 16, no. 5, pp. 550–554, 1994.

[27] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng, "Reading digits in natural images with unsupervised feature learning," in *NIPS workshop on deep learning and unsupervised feature learning*, 2011, vol. 2011, p. 5.

[28] Heng Zhang, Vishal M Patel, Sumit Shekhar, and Rama Chellappa, "Domain adaptive sparse representation-based classification," in *Automatic Face and Gesture Recognition (FG), 2015 11th IEEE International Conference and Workshops on*. IEEE, 2015, vol. 1, pp. 1–8.

[29] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. Ieee, 2009, pp. 248–255.

[30] Karen Simonyan and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[31] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.

[32] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[33] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[34] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al., "Tensorflow: a system for large-scale machine learning.," in *Operating Systems Design and Implementation*, 2016, vol. 16, pp. 265–283.

[35] Diederik P Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[36] Mahdi Abavisani and Vishal Patel, "Domain adaptive subspace clustering.," in *British Machine Vision Conference*. 2016, BMVA.

[37] Mahdi Abavisani and Vishal M Patel, "Adversarial domain adaptive subspace clustering," in *IEEE International Conference on Identity, Security, and Behavior Analysis*. IEEE, 2018, pp. 1–8.

APPENDIX: CONVERGENCE

To empirically show the convergence of our proposed method, in Figure 5, we plot the values of the objective function of DSRC in the experiment with the UMDAA-01 dataset and its classification accuracy in different iterations. The reported loss values in Figure 5 are scaled to have a maximum value of one. As can be seen from the figure, our algorithm converges in a few iterations.

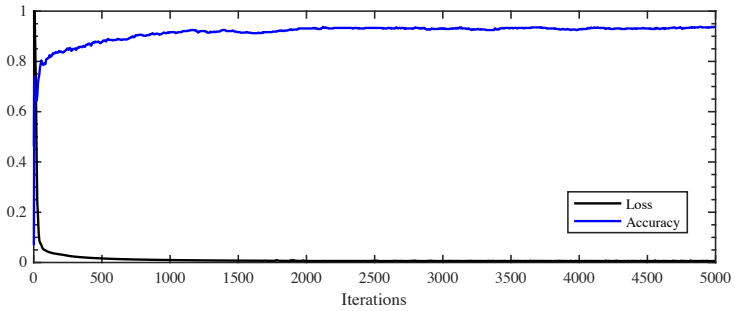


Fig. 5. Values of the DSRC's loss function in the experiment on the UMDAA-01 dataset vs iterations. The loss values are scaled to have the maximum value of one.