

# Deep Dictionary Learning

Snigdha Tariyal, Angshul Majumdar, *Senior Member IEEE*, Richa Singh, *Senior Member IEEE*, and Mayank Vatsa, *Senior Member IEEE*

**Abstract**—Two popular representation learning paradigms are dictionary learning and deep learning. While dictionary learning focuses on learning “basis” and “features” by matrix factorization, deep learning focuses on extracting features via learning “weights” or “filter” in a greedy layer by layer fashion. This research focuses on combining the concepts of these two paradigms. In this paper, we propose deep dictionary learning and show how deeper architectures can be built using layers of dictionary learning. The proposed technique is compared with other deep learning approaches such as Stacked AutoEncoder, Deep Belief Network, and Convolutional Neural Network. Experiments on benchmark datasets show that the proposed technique achieves higher classification and clustering accuracies. On a real world problem of electrical appliance classification, we show that deep dictionary learning excels where others do not yield at-par performance. We postulate that the proposed formulation can pave the path for a new class of deep learning tools.

**Index Terms**—Deep Learning, Dictionary Learning, Feature Representation

## I. INTRODUCTION

IN representation learning paradigm, *dictionary learning* has received a lot of interest. Researchers applied the concept of dictionary learning in vision [1] and information retrieval [2] in late 90’s. In those early days, the term ‘dictionary learning’ had not been coined; researchers were using the term ‘matrix factorization’. The goal was to learn an empirical basis from the data. It basically required decomposing the data matrix to a basis / dictionary matrix and a feature matrix; hence the name ‘matrix factorization’.

The current popularity of dictionary learning owes to K-SVD [3, 4]. K-SVD is an algorithm to decompose a matrix (training data) into a dense *basis* and *sparse* coefficients. However, the concept of such a dense-sparse decomposition predates K-SVD [5]. Since the advent of K-SVD in 2006, there has been a plethora of work on this topic. Dictionary learning can be used both for unsupervised problems (mainly inverse problems in image processing) as well as for problems arising in supervised feature extraction. Furthermore, it (dictionary learning) has been used in virtually all inverse problems arising in image processing starting from simple

image [6, 7] and video [8] denoising, image inpainting [9], to more complex problems such as inverse half toning [10] and even medical image reconstruction [11]. Such inverse problems can also be solved using the Compressed Sensing (CS) [12, 13] framework. However, it has been seen that learning the basis (via dictionary learning) yields better (customized) representation compared to the fixed basis employed by Compressed Sensing.

Mathematical transforms such as Discrete Cosine Transform (DCT), wavelet, curvelet, and Gabor have been widely used in image classification problems [14-16]. Existing techniques have used these transforms as a sparsifying step followed by statistical feature extraction methods such as Principal Component Analysis (PCA) or Linear Discriminant Analysis (LDA) before providing the features to a classifier. Just as dictionary learning is replacing such fixed transforms in signal processing problems, it is also replacing them in feature extraction scenarios. Dictionary learning provides researchers the opportunity to design dictionaries to yield not only sparse representation (e.g., curvelet, wavelet, and DCT) but also discriminative information.

Initial techniques in discriminative dictionary learning have proposed naïve approaches, which learn specific dictionaries for each class [17-19]. Later, discriminative penalties are introduced in dictionary learning framework to improve the classification performance. One such technique is to include softmax discriminative cost function [20-22]; other discriminative penalties include Fisher discrimination criterion [23], linear predictive classification error [24, 25] and hinge loss function [26, 27]. In [28, 29], discrimination is introduced by forcing the learned features to map to corresponding class labels. Prior studies on dictionary learning (DL) are, generally, ‘shallow’ learning models just like a restricted Boltzmann machine (RBM) [30] and an autoencoder (AE) [31], the two popular deep learning architectures. In DL, the cost function is Euclidean distance between the data and the representation given by the learned basis; for RBM it is Boltzmann energy; whereas for AE, the cost is the Euclidean reconstruction error between the data and the decoded representation/features.

Almost at the same time, when dictionary learning started gaining popularity, researchers in machine learning observed that better (more abstract and compact) representation can be achieved by going deeper in a neural network architecture. Deep Belief Network (DBN) is formed by stacking one RBM after the other [32, 33]. Similarly, stacked autoencoder (SAE) are created by one AE followed by the other [34, 35].

Inspired from both the feature learning paradigms, in this paper, we propose to learn multi-level deep dictionaries.

The authors are with IIIT Delhi, New Delhi, India 110020. (e-mail: [snigdha1491@iiitd.ac.in](mailto:snigdha1491@iiitd.ac.in), [angshul@iiitd.ac.in](mailto:angshul@iiitd.ac.in), [mayank@iiitd.ac.in](mailto:mayank@iiitd.ac.in), [rsingh@iiitd.ac.in](mailto:rsingh@iiitd.ac.in)).

Analogous to what the researchers in deep learning community have been able to achieve by designing deeper architectures, it is our assertion that the proposed formulation of deep dictionary learning algorithms will inspire researchers to design more effective dictionary learning algorithms. One may think that multiple levels of dictionaries can be collapsed to a single level. However such a collapsed shallow dictionary will not be the same as the proposed deep dictionary learning. This is because dictionary learning is a bi-linear problem<sup>1</sup>. Had it been linear the architecture would have been collapsible; since it is not, the shallow and the deep architectures will not be equivalent. The remaining paper first discusses the literature review of dictionary learning, deep Boltzmann machine and auto-encoder in Section II. This is followed by mathematical formulation of the proposed deep dictionary learning in Section III and experimental evaluation in Section IV.

## II. LITERATURE REVIEW

We will briefly review prior studies on dictionary learning, stacked autoencoders, and deep Boltzmann machines.

### A. Dictionary Learning

Early studies in dictionary learning focused on learning a basis for representation. There were no constraints on the dictionary atoms or on the loading coefficients. The method of optimal directions [36] was used to learn the basis:

$$\min_{D,Z} \|X - DZ\|_F^2 \quad (1)$$

Here,  $X$  is the training data,  $D$  is the dictionary to be learnt and  $Z$  consists of the loading coefficients.

For problems in sparse representation, the objective is to learn a basis that can represent the samples in a sparse fashion, i.e.  $Z$  needs to be sparse. K-SVD [3, 4] is the most well-known technique for solving this problem. Fundamentally, it solves a problem of the form:

$$\min_{D,Z} \|X - DZ\|_F^2 \text{ such that } \|Z\|_0 \leq \tau \quad (2)$$

Here we have abused the notation slightly, the  $l_0$ -norm is defined on the vectorized version of  $Z$ .

K-SVD proceeds in two stages. In the first stage it learns the dictionary and in the next stage it uses the learned dictionary to sparsely represent the data. Solving the  $l_0$ -norm minimization problem is NP hard [37]. K-SVD employs the greedy (sub-optimal) orthogonal matching pursuit (OMP) [38] to solve the  $l_0$ -norm minimization problem approximately. In the dictionary learning stage, K-SVD proposes an efficient technique to estimate the atoms one at a time using a rank one update. The major disadvantage of K-SVD is that it is a relatively slow technique owing to its requirement of computing the SVD (singular value decomposition) in every

iteration. There are other efficient optimization based approaches for dictionary learning [39, 40] – these learn the full dictionary instead of updating the atoms separately.

The dictionary learning formulation in equation (2) is unsupervised. As mentioned before, there is a large volume of work on supervised dictionary learning problems. The first work on Sparse Representation based Classification (SRC) [41] is not much of a “dictionary learning technique”, but a simple dictionary design problem where all the training samples are concatenated in a large dictionary. Later, several improvements to the basic SRC formulation are proposed in [42-44]. In [42, 43] supervision is added in the form of group-sparsity. In [44] a non-linear extension to the SRC is proposed. Later works handled the non-linear extension in a smarter fashion using the kernel trick [45-47].

The SRC does not exactly fit into the dictionary learning paradigm. However, [48] proposed a simple extension of SRC – instead of using raw training samples as the basis, they learnt a separate basis for each class and used these dictionaries for classification. This approach is naïve; there is no guarantee that dictionaries from different classes would not be similar. Ramirez et al. have addressed this issue by applying an additional incoherency penalty on the dictionaries [49]. This penalty assures that the dictionaries from different classes look different from each other. The formulation is given as:

$$\min_{D_i, Z_i} \sum_{i=1}^C \left\{ \|X_i - DZ_i\|_F^2 + \lambda \|Z_i\|_1 \right\} + \eta \sum_{i \neq j} \|D_i^T D_j\|_F^2 \quad (3)$$

Unfortunately this formulation does not improve the overall results too much. It learns dictionaries that look different from each other but does not produce features that are distinctive; i.e. the feature generated for the test sample from dictionaries of all classes looked more or less the same. This issue was rectified in [50].

The label consistent KSVD is one of the more recent techniques for learning discriminative sparse representation. It is simple to understand and implement; it showed good results for face recognition [28, 29]. The first technique is termed as Discriminative K-SVD [28] or LC-KSVD1 [29]; it proposes an optimization problem of the following form:

$$\min_{D,Z,A} \|X - DZ\|_F^2 + \lambda_1 \|D\|_F^2 + \lambda_2 \|Z\|_1 + \lambda_3 \|Q - AZ\|_F^2 \quad (4)$$

Here,  $Q$  is the label of the training samples; it is a canonical basis with a one for the correct class and zeroes elsewhere.  $A$  is a parameter of the linear classifier.

In [29], a second formulation is proposed that adds another term to penalize the classification error. The LC-KSVD2 formulation is as follows:

$$\min_{D,Z,A,W} \|X - DZ\|_F^2 + \lambda_1 \|D\|_F^2 + \lambda_2 \|Z\|_1 + \lambda_3 \|Q - AZ\|_F^2 + \lambda_4 \|H - WZ\|_F^2 \quad (5)$$

$H$  is a ‘discriminative’ sparse code corresponding to an input signal sample, if the nonzero values of  $H_i$  occur at those indices where the training sample  $X_i$  and the dictionary item  $d_k$  share the same label. Basically this formulation imposes labels not only on the sparse coefficient vectors  $Z_i$ ’s but also on the

<sup>1</sup> Synthesis dictionary formulation  $X=DZ$  is bilinear; the variables are the dictionary  $D$  and feature  $Z$ . Bilinearity means that, it is linear in each of the variables ( $D$  and  $Z$ ) if the other one is constant ( $Z$  and  $D$  respectively).

For multi-level dictionary (say 2) the formulation is  $X=D_1D_2Z$ . This is a trilinear formulation and is a different problem altogether. Solving the trilinear 2-level dictionary learning will not yield the same features as that of a collapsed bilinear ( $D=D_1D_2$ ) dictionary learning problem.

dictionary atoms.

### B. Deep Boltzmann Machine

Restricted Boltzmann Machines are undirected models that use stochastic hidden units to model the distribution over the stochastic visible units. The hidden layer is symmetrically connected with the visible unit and the architecture is “restricted” as there are no connections between units of the same layer. Traditionally, RBMs are used to model the distribution of the input data  $p(x)$ .

The schematic diagram of RBM is shown in Fig. 1. The objective is to learn the network weights ( $W$ ) and the representation ( $H$ ). This is achieved by optimizing the Boltzmann cost function given by:

$$p(W, H) = e^{-E(W, H)} \quad (6)$$

where,  $E(W, H) = -H^T W X$  including the bias terms. Assuming independence, the conditional distributions are given by

$$p(X | H) = \prod p(x | h) \quad (7)$$

$$p(H | X) = \prod p(h | x) \quad (8)$$

Assuming a binary input variable, the probability that a node will be active can be given as follows,

$$p(x = 1 | h) = \text{sigm}(W^T h)$$

$$p(h = 1 | x) = \text{sigm}(W x)$$

Computing the exact gradient of this loss function is almost intractable. However, there is a stochastic approximation to approximate the gradient termed as contrastive divergence gradient. A sequence of Gibbs sampling based reconstruction, produces an approximation of the expectation of joint energy distribution, using which the gradient can be computed.

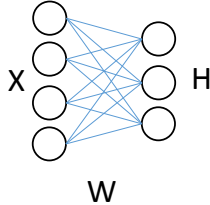


Fig. 1. Restricted Boltzmann Machine

Usually RBM is unsupervised, but there are studies where discriminative RBMs are trained by utilizing the class labels [51]. There are also RBMs which are sparse [52]; the sparsity is controlled by firing the hidden units only if they are over some threshold. Supervision can also be achieved using sparse RBMs by extending it to have similar sparsity structure within the group / class [53].

Deep Boltzmann Machines (DBM) [54] is an extension of RBM created by stacking multiple hidden layers on top of each other (Fig. 2). DBM is an undirected learning model and thus it is different from the other stacked network architectures in which each layer receives feedback from both the top-down and bottom-up layer signals. This feedback mechanism helps in managing uncertainty in learning models. While the

traditional RBM can model logistic units, a Gaussian-Bernoulli RBM [55] can be used as well with real valued (between 0 and 1) visible units.

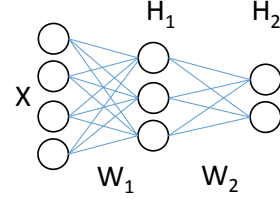


Fig. 2. Deep Boltzmann Machine

### C. Stacked Autoencoder

An autoencoder consists (as seen in Fig. 3) of two parts – the encoder maps the input to a latent representation and the decoder maps the latent representation back to the data. For a given input vector (including the bias term)  $x$ , the latent space is expressed as:

$$h = Wx \quad (9)$$

Here, the rows of  $W$  are the link weights from all the input nodes to the corresponding latent node. Usually a non-linear activation function is used at the output of the hidden nodes leading to:

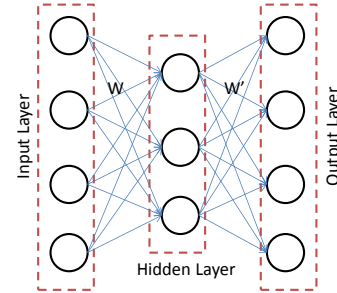


Fig. 3. Single Layer Autoencoder

$$h = \phi(Wx) \quad (10)$$

Although the sigmoid function is popularly used, other non-linear activation functions such as  $\tanh$  can also be used. Rectifier units and large neural networks employ linear activation; owing to this linearity, the training process is considerably faster. The decoder reverse maps the latent variables to the data space.

$$x = W' \phi(Wx) \quad (11)$$

Since the data space is assumed to be the space of real numbers, there is no sigmoid function here. During training, the problem is to learn the encoding and decoding weights –  $W$  and  $W'$ . These are learnt by minimizing the Euclidean cost:

$$\arg \min_{W, W'} \|X - W' \phi(WX)\|_F^2 \quad (12)$$

Here  $X = [x_1 | \dots | x_N]$  consists of all the training samples stacked as columns where total number of training samples are  $N$ . The problem in Equation (12) is clearly non-convex, but is smooth and hence can be solved by gradient descent

techniques; the activation function needs to be smooth and continuously differentiable.

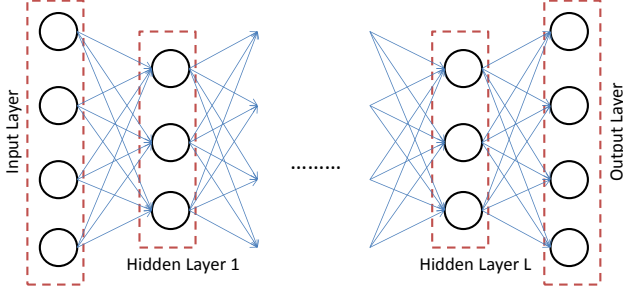


Fig. 4. Stacked Autoencoder

There are several extensions to the basic autoencoder architecture. Stacked autoencoders have multiple hidden layers – one inside the other (see Fig. 4). The corresponding cost function is expressed as follows:

$$\arg \min_{W_1 \dots W_L, W'_1 \dots W'_L} \|X - g \circ f(X)\|_F^2 \quad (13)$$

where,  $g = W'_1 \phi(W'_2 \dots W'_L (f(X)))$  and

$$f = \phi(W_L \phi(W_{L-1} \dots \phi(W_1 X)))$$

Solving the complete problem (13) is computationally challenging. Also learning so many parameters (network weights) lead to over-fitting. To address both these issues, the weights are usually learned in a greedy layer-by-layer fashion [32, 34].

Stacked denoising autoencoder [35] is a variant of the basic autoencoder where the input consists of noisy samples and the output consists of clean samples. Here the encoder and decoder are learnt to denoise noisy input samples. Another variation for the basic autoencoder is to regularize it, i.e.

$$\arg \min_{(W)_s} \|X - g \circ f(X)\|_F^2 + R(W, X) \quad (14)$$

The regularization can be a sparsity promoting term [56, 57] or a weight decay term (Frobenius norm of the Jacobian) as used in the contractive autoencoder [58]. The regularization term is usually chosen so that they are differentiable and hence minimizable using gradient descent techniques.

### III. DEEP DICTIONARY LEARNING

In this section, we describe the main contribution of this research. A single / shallow level of dictionary learning yields a latent representation of data and the dictionary atoms. Here, we propose to learn latent representation of data by learning multi-level dictionaries. The idea of learning deeper levels of dictionaries stems from the success of deep learning. In this section, for ease of understanding, we first explain the concept with two-layer deep dictionary learning and then extend it to multi-level dictionary.

The schematic diagram for dictionary learning is shown in Fig. 5. Let  $X$  be the data,  $D_l$  be the dictionary and  $Z$  be the feature/representation of  $X$  in  $D_l$ . Dictionary learning follows a synthesis framework (15), i.e. the dictionary is learnt such

that the features synthesize the data along with the dictionary.

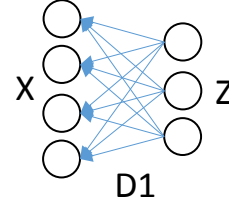
$$X = D_l Z \quad (15)$$


Fig. 5. Schematic Diagram for Dictionary Learning

There is a dictionary learning approach termed as analysis K-SVD, but it cannot be used for feature extraction. Analysis K-SVD can only produce a ‘clean’ version of the data and hence, is only suitable for inverse problems.

We propose to extend the shallow (Fig. 5) dictionary learning into multiple layers – leading to deep dictionary learning (Fig. 6). Mathematically, the representation at the second layer can be written as:

$$X = D_1 D_2 Z_2 \quad (16)$$

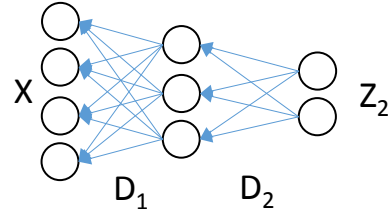


Fig. 6. Schematic Diagram for Deep Dictionary Learning

It must be noted that learning two-levels of dictionaries along with the coefficients (16) is not the same as learning a single (collapsed) dictionary and its corresponding features. Problem (15) (single level) is a bi-linear problem and (16) is a tri-linear problem; they are not the same. Hence one cannot expect to get the same features from a single level dictionary learning and a collapsed two level dictionary learning.

The challenges of learning multiple levels of dictionaries in one go are the following:

- 1) Recent studies have proven convergence guarantees for single level dictionary learning [59-63]. These proofs would be very hard to replicate for multiple layers.
- 2) Moreover, the number of parameters required to be solved increases when multiple layers of dictionaries are learnt simultaneously. With limited training data, this could lead to over-fitting.

Here we propose to learn the dictionaries in a greedy manner which is in sync with other deep learning techniques [32-34]. Moreover, layer-wise learning will guarantee the convergence at each layer. The diagram illustrating layer-wise learning is shown in Fig. 7.

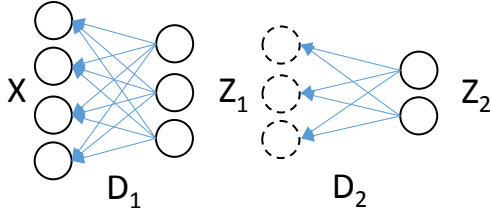


Fig. 7. Greedy Layer-wise Learning

Extending this idea, a multi-level dictionary learning problem with non-linear activation ( $\varphi$ ) can be expressed as,

$$X = D_1 \varphi(D_2 \varphi(\dots \varphi(D_N Z))) \quad (17)$$

Ideally, we would have to solve the following problem.

$$\min_{D_1, \dots, D_N, Z} \|X - D_1 \varphi(D_2 \varphi(\dots \varphi(D_N Z)))\|_F^2 + \mu \|Z\|_1 \quad (18)$$

However, such a problem is highly non-convex and requires solving huge number of parameters. With limited amount of data, it will lead to over-fitting. To address these issues, as mentioned before, we propose a greedy approach where, we learn one layer at a time – similar to pre-training in the deep learning paradigm. With the substitution  $Z_1 = \varphi(D_2 \varphi(\dots \varphi(D_N Z)))$ , Equation (17) can be written as  $X = D_1 Z_1$  such that it can be solved as single layer dictionary learning. The representation  $Z_1$  is not sparse. Hence it can be solved using alternating minimization –

$$\min_{D_1, Z_1} \|X - D_1 Z_1\|_F^2 \quad (19)$$

Optimality of solving (19) by alternating minimization has been proven in [56]. Therefore we follow the same approach. The dictionary  $D$  and the basis  $Z$  is learnt by:

$$Z_1 \leftarrow \min_Z \|X - D_1 Z_1\|_F^2 \quad (20a)$$

$$D_1 \leftarrow \min_D \|X - D_1 Z_1\|_F^2 \quad (20b)$$

This is the method of optimal directions [36] and both (20a) and (20b) are simple least square problems having closed form solutions.

For the second layer, we substitute  $Z_2 = \varphi(\dots \varphi(D_N Z))$ , which leads to  $Z_1 = \varphi(D_2 Z_2)$ , or alternately,  $\varphi^{-1}(Z_1) = D_2 Z_2$ ; this too is a single layer dictionary learning. Since the representation is dense, it can be solved using

$$\min_{D_2, Z_2} \|\varphi^{-1}(Z_1) - D_2 Z_2\|_F^2 \quad (21)$$

This too is can be solved by alternating minimization as in the case of first layer (20). Continuing in this fashion till the penultimate layer, in the final layer we have  $Z_{N-1} = \varphi(D_N Z)$  or  $\varphi^{-1}(Z_{N-1}) = D_N Z$ . In the last level the coefficient  $Z$  can be sparse. For learning sparse features, one needs to regularize by applying  $l_1$ -norm on the features. This is given by:

$$\min_{D_N, Z} \|\varphi^{-1}(Z_{N-1}) - D_N Z\|_F^2 + \lambda \|Z\|_1 \quad (22)$$

This too is solved using alternating minimization.

$$Z \leftarrow \min_Z \|\varphi^{-1}(Z_{N-1}) - D_N Z\|_2^2 + \lambda \|Z\|_1 \quad (23a)$$

$$D_N \leftarrow \min_{D_N} \|\varphi^{-1}(Z_{N-1}) - D_N Z\|_F^2 \quad (23b)$$

As before, (23b) is a least square problem having a closed form solution. Although not analytic, the solution to (23a) can be solved using the Iterative Soft Thresholding Algorithm (ISTA) [64]. The ISTA solution for (23a) is given by:

$$\text{Initialize: } Z \leftarrow \min_Z \|\varphi^{-1}(Z_{N-1}) - D_N Z\|_2^2$$

Iterate till convergence

$$B = Z + \frac{1}{\alpha} D_N^T (\varphi^{-1}(Z_{N-1}) - D_N Z)$$

$$Z \leftarrow \text{signum}(B) \max\left(0, |B| - \frac{\lambda}{2\alpha}\right)$$

It is important to note that learning multiple dictionaries cannot be collapsed into a single one even if the activation function is linear. This is because dictionary learning is bi-linear. For example, if the dimensionality of the sample is  $m$  and the first dictionary is of size  $m \times n_1$  and the second one is  $n_1 \times n_2$ , it is not possible to learn a single dictionary of size  $m \times n_2$  and expect the same results as a two-stage dictionary.

In general, for non-linear activation functions it is not possible to collapse the multiple levels of dictionaries into a single level for testing. However, for the linear activation function, the multiple levels of dictionaries can be collapsed into a single stage by matrix multiplication of the different dictionaries and the sparse code / features computed from the thus formed single level dictionary by standard  $l_1$ -minimization.

#### Training Algorithm (for any activation function $\varphi$ )

Initialize:  $D_i, i = 1 \dots N$

For first level; repeat until convergence –

$$Z_1 \leftarrow \min_Z \|X - D_1 Z_1\|_F^2$$

$$D_1 \leftarrow \min_D \|X - D_1 Z_1\|_F^2$$

From 2<sup>nd</sup> to penultimate level; repeat until convergence –

$$Z_l \leftarrow \min_{Z_l} \|\varphi^{-1}(Z_{l-1}) - D_l Z_l\|_F^2$$

$$D_l \leftarrow \min_{D_l} \|\varphi^{-1}(Z_{l-1}) - D_l Z_l\|_F^2$$

For final level; repeat until convergence –

$$Z_N \leftarrow \min_{Z_N} \|\varphi^{-1}(Z_{N-1}) - D_N Z_N\|_F^2 + \lambda \|Z_N\|_1$$

$$D_N \leftarrow \min_{D_N} \|\varphi^{-1}(Z_{N-1}) - D_N Z_N\|_F^2$$

#### Testing Algorithm (for linear activation function)

Collapse multiple levels of dictionaries into a single one

$$D = D_1 D_2 \dots D_N$$

Compute sparse code / features for the test sample  $x_{\text{test}}$ .

$$z_{\text{test}} = \min_{z_{\text{test}}} \|x_{\text{test}} - D z_{\text{test}}\|_2^2 + \lambda \|z_{\text{test}}\|_1$$



### Testing Algorithm (for non-linear activation function)

Generate features for first level

$$z_{1,test} = \min_{z_{1,test}} \|x_{test} - D_1 z_{1,test}\|_2^2$$

From 2<sup>nd</sup> to penultimate level

$$z_{l,test} = \min_{z_{l,test}} \|\phi^{-1}(z_{l-1,test}) - D_l z_{l,test}\|_2^2$$

For final level

$$z_{test} = \min_{z_{N-1,test}} \|z_{N-1,test} - D_N z_{test}\|_2^2 + \lambda \|z_{test}\|_1$$

#### A. Connection with Existing Algorithms

In this section, we compare and contrast the proposed deep dictionary approach with some popular deep learning algorithms, namely RBM and Autoencoder, and hierarchical dictionary learning approaches. From Figures 1 and 5, it is evident that in both RBM and dictionary learning, the task is to learn the network weights / atoms and the representation given the data. They differ from each other in the cost functions used. For RBM it is the Boltzmann function whereas for dictionary learning, instead of maximizing similarity, we minimize the Euclidean distance between the data ( $X$ ) and the synthesis ( $DZ$ ). Further, RBM has a stochastic formulation whereas dictionary learning is deterministic. Moreover, RBM typically uses binary or real values as input. On the other hand, deep dictionary learning can work both on real and complex inputs.

Similar to RBM, a comparison of the proposed deep dictionary learning with autoencoders can be performed. The synthesis dictionary learning model is expressed as:  $X = D_S Z$  where  $X$  is the data,  $D_S$  is the learnt synthesis dictionary and  $Z$  are the sparse coefficients. Usually one promotes sparsity in the features and the learning requires minimizing the following,

$$\|X - D_S Z\|_F^2 + \lambda \|Z\|_1 \quad (25)$$

This is the well-known synthesis prior formulation where the task is to find a dictionary that can synthesize / generate signals from sparse features. There is an alternate co-sparse analysis prior dictionary learning paradigm [65] where the goal is to learn a dictionary such that when it is applied on the data the resulting coefficient is sparse. The model is represented as  $D_A \hat{X} = Z$ . The corresponding learning problem is framed by minimizing:

$$\|X - \hat{X}\|_F^2 + \lambda \|D_A \hat{X}\|_1 \quad (26)$$

If we combine analysis and synthesis, using  $\hat{X} = D_S Z$ ,  $D_A \hat{X} = Z$  and impute it in (25) we get:

$$\|X - D_S D_A \hat{X}\|_F^2 + \lambda \|D_A \hat{X}\|_1 \quad (27)$$

If we drop the sparsity term, it becomes

$$\|X - D_S D_A \hat{X}\|_F^2 \quad (28)$$

This is similar to the expression of a sparse denoising autoencoder [54] with linear activation at the hidden layer. Further, we can express autoencoder in the terminology of dictionary learning – autoencoder is a model that learnt the analysis and the synthesis dictionaries. To the best of our knowledge, this is the first work that shows the architectural similarity between autoencoders and dictionary learning.

We also briefly discuss the differences with recently proposed dictionary learning algorithms. First, we would like to emphasize that the proposed approach is not related to hierarchical or structured dictionary learning techniques [66, 67]. In these studies, unlike the proposed approach, the goal is to learn dictionary atoms that are related to each other. These studies, generally, follow shallow learning techniques (single level) and the relationships are between atoms of dictionary within the same level.

We next discuss the work on “Double Sparsity” [68] where the authors decompose the dictionary into a product of a fixed basis (e.g. wavelet, and DCT) and a sparse set of coefficients; instead of learning the full dictionary (as is done in standard dictionary learning), they learn a dictionary that can be represented as a linear combination of fixed basis. The learning mechanism is formulated as follows:

$$\min_{T,Z} \|X - \Phi T Z\|_F^2 \text{ s.t. } \|T\|_0 \leq s \text{ and } \|Z\|_0 \leq s \quad (29)$$

In this formulation,  $\Phi T = D$ ; i.e. instead of learning the full dictionary, the authors learn the coefficients required to synthesize the dictionary  $D$  from a fixed basis  $\Phi$ . The advantage of this approach is that the learnt dictionary has fast forward and ad-joint operators (since it is synthesized from an efficient operator) and hence is useful for solving large-scale inverse problems. Clearly the proposed algorithm is different from Equation (29) [68]. We learn the full dictionary in each level and continue the process for multiple levels with a goal to learn abstract representations.

## IV. EXPERIMENTAL EVALUATION

The effectiveness of the proposed deep dictionary learning is evaluated on multiple benchmark databases from different areas such as images, text, and signals. The results are compared with related state-of-the-art algorithms. In this work we use a linear activation function for all the experiments.

### A. Datasets

We have evaluated the performance on several benchmarks datasets. The first one is the MNIST dataset that consists of 28x28 images of handwritten digits ranging from 0 to 9. The dataset has 60,000 images for training and 10,000 images for testing. It should be noted that we have not performed any preprocessing on this dataset.

Related to MNIST database, MNIST variations datasets are also used. These are more challenging databases, primarily due to fewer training samples (10,000) and larger number of test samples (50,000). The validation set of 2000 samples are not used in this work since our method does not require tuning and SAE as well as DBN are already optimized for MNIST. Here is the listing of these databases.

1. basic (smaller subset of MNIST)
2. basic-rot (smaller subset with random rotations)
3. bg-rand (smaller subset with uniformly distributed noise in background)
4. bg-img (smaller subset with random image background)
5. bg-img-rot (smaller subset with random image background plus rotation)

These datasets are primarily created to empirically benchmark deep learning algorithms [69]. Samples for each of the datasets are shown in Fig. 8.

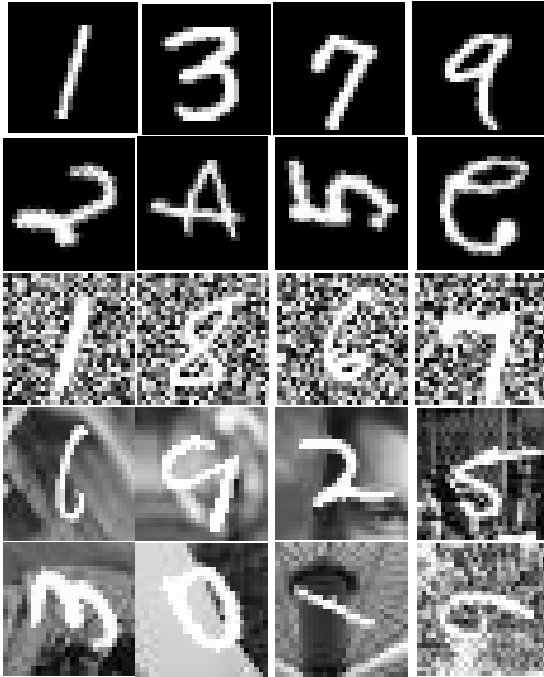


Fig. 8. Top to bottom. basic, basic-rot, bg-rand, bg-img, bg-img-rot

The second database is related to text documents and relates to the problem of classifying documents into their corresponding newsgroup topic. We have used a version of the commonly used 20-newsgroup dataset [70] for which the training and test sets contain documents collected at different times, a setting that is more reflective of a practical application. The training set consists of 11,269 samples and the test set contains 7,505 examples. We have used 5000 most frequent words for the binary input features and follow the same protocol as outlined in [51].

The third dataset is the GTZAN music genre dataset [71]. The dataset contains 10,000 three-second audio clips, equally distributed among 10 musical genres: blues, classical, country, disco, hip-hop, pop, jazz, metal, reggae and rock. 592 Mel-Phon Coefficient (MPC) features represent each example in the set. These are a simplified formulation of the Mel-frequency Cepstral Coefficients (MFCCs) that have been shown to yield better classification performance in literature. Since there is no predefined standard split and fewer examples, we have used 10-fold cross validation (procedure mentioned in [35]), where each fold consisted of 9000 (we do

not require validation examples unlike [35]) training examples and 1000 test examples.

### B. Effect of Layers on Dictionary Learning

We first analyze the results of the proposed deep dictionary learning and the effect of increasing the number of layers. The objective of this experiment is to show that the representation learnt from a single level of dictionary and multi-level dictionary are different and multi-level dictionaries yield better classification performance. With just one level of dictionary, i.e., a shallow dictionary, we learn 50 atoms for MNIST, 625 atoms for the 20-newsgroup, and 148 for GTZAN; these correspond to the last level of dictionary atoms. For deep dictionary, we performed experiments with up to three levels of dictionary. For the MNIST database and its variations, the number of basis in the multi-level dictionaries is: 300-150-50. For the 20-newsgroup and the GTZAN (music genre classification) datasets, the number of atoms in every layer is halved from that of the previous layer. The classification is performed with a simple K-Nearest Neighbor ( $K = 1$ ). The classification accuracies are reported in Table 1, Column 2 reports the results for shallow dictionary and columns 3-5 report the results with different layers of deep dictionary. The results show that for all the databases, deep dictionary (3-layer) learning offers improvements over shallow dictionary learning. The improvement in accuracy is possibly owing to more abstract representation learnt from these layers. Depending on the complexity of the dataset, the difference in performance varies from 0.40% to more than 9%. Fig. 9 illustrates first layer dictionary on MNIST database.

Generally, the dictionary atoms are initialized by randomly choosing samples from the training set; however, this leads to variability in results. In this research, we propose a deterministic initialization based on the QR decomposition of the training data matrix. Orthogonal vectors from  $Q$  (in order) are used to initialize the dictionary.

We next show that the multi-level dictionaries cannot be collapsed into a single one and should not be expected to yield the same results. The difference between the performance of multi-level dictionary learning and single level dictionary learning is evident in Table I; see columns 2 and 5. If the learning is linear, it is possible to collapse multiple dictionaries into one; but dictionary learning is inherently non-linear. Hence it is not possible to learn a single layer of dictionary in place of multiple levels and expect the same output.

### C. Comparison with Deep Learning Approaches

Since the proposed deep architecture is inspired from other currently existing deep learning approaches, we have compared our results with a stacked autoencoder (SAE) and deep belief network (DBN). The implementation for these have been obtained from [72] and [73] respectively. Similar to deep dictionary learning, SAE and DBN also have a three-layer architecture. The number of nodes is halved in every subsequent layer and comparison is performed using K-Nearest Neighbor (KNN) and Support Vector Machine

(SVM). To ensure fair comparison of representation techniques, we have kept the classifier fixed for these experiments. The results are shown in Tables II and III, respectively.

TABLE I: DEEP VS SHALLOW DICTIONARY LEARNING

Dataset	Shallow (50)	1 layer (300)	2 layer (300-150)	3 layer (300-150-50)
MNIST	97.35	96.52	97.30	97.75
basic	95.02	94.39	95.27	95.80
basic-rot	84.19	83.37	85.39	87.00
bg-rand	87.19	86.52	87.49	89.35
bg-img	78.86	77.56	79.20	81.00
bg-img-rot	54.40	54.06	55.93	57.77
20-newsgroup	60.96	65.22	68.93	70.48
GTZAN	76.57	75.70	79.68	83.31

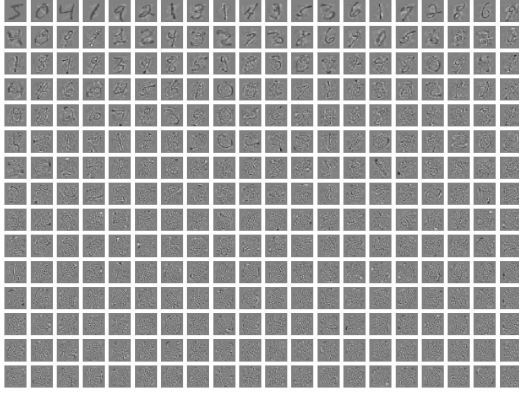


Fig. 9. First level dictionary for MNIST

We observe that apart from one case each in Tables II and III, the proposed algorithm yields better results than DBN and SAE. For KNN, the results of deep dictionary learning are slightly better; however, with SVM classifier, deep dictionary yields considerably better results.

We have also compared the performance of the proposed algorithm with dictionary learning techniques such as D-KSVD [28], LC-KSVD [29], and supervised dictionary learning [21]. These are individually fine-tuned to yield the best possible results. Comparison is also performed with stacked denoising autoencoder (SDAE), deep belief network (DBN) fine-tuned with soft-max classifier and convolutional neural network (CNN). The results on DBN and SDAE are from [35]; the results from CNN are from [74] – which is a baseline technique for CNN architectures. The results are summarized in Table IV.

It can be observed that the proposed deep dictionary learning techniques almost always yield better results than shallow dictionary learning (supervised DL, LC-KSVD and D-KSVD); only for two instances i.e., the simple MNIST dataset and bg-img-rot do the shallow learning techniques yield better results. In almost all other cases, the proposed algorithm is in the top two best algorithms and achieves better

accuracy than highly tuned models such as DBN, SDAE and CNN. CNN cannot be run on the 20-newsgroup dataset since it does not have local correlation and cannot be represented as a linear time invariant system – an aspect required for the convolution operation to hold.

TABLE II: COMPARING THE CLASSIFICATION ACCURACY OF DDL WITH DBN AND SAE WITH KNN (K = 1) CLASSIFICATION

Dataset	DDL	DBN	SAE
MNIST	<b>97.75</b>	97.05	97.33
basic	<b>95.80</b>	95.37	95.25
basic-rot	<b>87.00</b>	84.71	84.83
bg-rand	<b>89.35</b>	77.16	86.42
bg-img	81.00	<b>86.36</b>	77.16
bg-img-rot	<b>57.77</b>	50.47	52.21
20-newsgroup	<b>70.48</b>	70.09	69.78
GTZAN	<b>83.31</b>	80.99	82.79

TABLE III: COMPARING THE CLASSIFICATION ACCURACY OF DDL WITH DBN AND SAE WITH SVM CLASSIFICATION

Dataset	DDL	DBN	SAE
MNIST	<b>98.64</b>	98.53	98.50
basic	97.28	88.44	<b>97.40</b>
basic-rot	<b>90.34</b>	76.59	79.83
bg-rand	<b>92.38</b>	78.59	85.34
bg-img	<b>86.17</b>	75.22	74.99
bg-img-rot	<b>63.85</b>	48.53	49.14
20-newsgroup	<b>71.97</b>	71.12	70.49
GTZAN	<b>84.92</b>	81.50	83.87

We next compare the proposed algorithm with other deep learning approaches in terms of computational speed (train feature generation + test feature generation time). All the algorithms are run until convergence on a machine with Intel (R) Core(TM) i5 running at 3 GHz; 8 GB RAM, Windows 10 (64 bit) running Matlab 2014a. The run times for all the smaller MNIST variations are approximately the same. Therefore, we only report results for the larger MNIST dataset (60K) and the basic (10K) dataset. We do not include the training and testing time for classification here; since they will be almost the same for all the techniques as long as the dimensionality of the features remains the same. Further, Table V shows that for training the proposed algorithm is around two orders of magnitude faster than deep belief network and three orders of magnitude faster than stacked autoencoder. However, the testing times (Table VI) for the proposed algorithm is somewhat slower – since we need to solve an optimization problem for generating the test features whereas, the others simply need a few matrix vector products.



TABLE IV: COMPARING THE CLASSIFICATION ACCURACY OF DDL+SVM WITH EXISTING ALGORITHMS AND ARCHITECTURES

Dataset	DDL-SVM	Supervised DL	LC-KSVD	D-KSVD	DBN- Softmax	SDAE-Softmax	CNN
MNIST	98.64	<b>98.95</b>	93.30	93.6	98.76	98.72	<b>99.06</b>
basic	<b>97.28</b>	95.14	92.70	92.20	96.89	97.16	<b>98.56</b>
basic-rot	<b>90.34</b>	51.98	48.66	50.01	89.70	<b>90.47</b>	89.45
bg-rand	92.38	88.23	87.70	87.70	<b>93.27</b>	89.7	<b>93.23</b>
bg-img	<b>86.17</b>	80.92	80.65	81.20	83.69	83.32	<b>88.89</b>
bg-img-rot	63.85	74.31	<b>75.40</b>	<b>75.40</b>	52.61	56.24	57.97
20-newsgroup	<b>71.97</b>	69.22	68.90	69.45	<b>72.40</b>	70.93	-
GTZAN	<b>84.92</b>	79.86	76.85	76.78	81.62	<b>83.98</b>	69.91

TABLE V: TRAINING FEATURE GENERATION TIME (IN SECONDS)

Dataset	DDL	DBN	SAE
MNIST	107	30071	120408
basic	26	5974	24020

TABLE VI: TEST FEATURE GENERATION TIME (IN SECONDS)

Dataset	DDL	DBN	SAE
MNIST	79	50	61
basic	257	151	185

#### D. Electrical Appliance Classification

The previous sections demonstrate results on benchmarking datasets. In this section, we evaluate the effectiveness of deep dictionary learning for solving a real-world problem. Gupta et al. proposed the problem of classifying electrical appliances from their energy electro-magnetic interference (EMI) signatures [75]. They showed results on appliance classification from differential mode (DM) EMI signatures; but DM EMI has an inherent shortcoming. The power signal and its harmonics interfere with the DM EMI, hence analysis based on such signatures are not reliable [76]. In a technical report [77], it has been shown that, using rudimentary heuristics one is able to achieve decent results from CM EMI based appliance classification. The details regarding the development of the CM EMI sensor, data acquisition and details are given in [77].

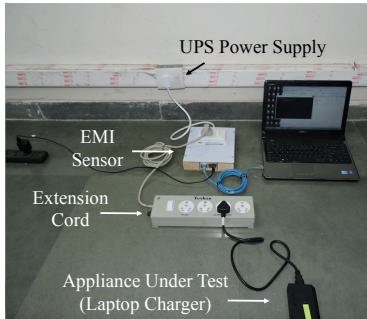


Fig. 10. Block view of EMI sensor

Fig. 10 shows the setup for data collection. The data is collected for 5 devices – CFL, CPU, LCD, Laptop Charger and Background Noise (when nothing is in use). There are 5 instances of every appliance. 1500 traces are collected for each instance of each appliance. Each trace is of 1 millisecond duration and constitutes a vector of length 16,384. The training set consists of samples from 1 instance for each appliance while the remaining 4 instances of each appliance constitutes the test set. Thus, for every problem, the training data for each class has 1500 traces and the test data has  $4 \times 1500 = 6000$  traces.

The experimental protocol has been defined in [77]. For testing, all 1500 traces should be considered as a single signature; the task is to identify the appliance from this signature. Therefore the predicted class labels of the 1500 samples should be fused via majority voting to a single appliance. Cepstrum features [78] are used as feature set for classification; some examples of cepstrum features are shown in Fig. 11. It can be seen how the cepstrum features for the same appliance look similar and those from different appliances look different.

TABLE VII: CORRECT CLASSIFICATION ACCURACY FOR APPLIANCE CLASSIFICATION DATABASE.

	ElectriSense [72]	CLM+SVM [77]	LC-KSVD	Proposed
Fold 1	25.0	62.5	60.0	90.0
Fold 2	30.0	60.0	55.0	85.0
Fold 3	32.5	67.5	62.5	90.0
Fold 4	32.5	67.5	62.5	87.5
Fold 5	30.0	60.0	60.0	87.5
<b>Aggregate</b>	<b>30.0</b>	<b>63.5</b>	<b>60.0</b>	<b>88.0</b>

On this dataset, stacked autoencoders and deep belief networks, even after pre-processing and normalization, are not able to surpass the results with random assignment, and the classification accuracy is pegged at  $= 1/(\text{number of appliances})$ . The standard technique for classifying appliances from EMI signatures is [78]; therefore we compare our proposed technique against ElectriSense [78]

and features extracted using Conditional Likelihood Maximization (CLM) [79] followed by SVM classification. As a benchmark, the results are also compared using LC-KSVD and the results are shown for 5-fold cross validation. For the proposed method, two levels of dictionaries are learnt, in the first level the number of atoms are 500 and in the second level 100. The first level generates dense features whereas the second level generates sparse features. Table VII shows that the proposed algorithm significantly outperforms all three approaches. ElectriSense yields around 30% correct classification accuracy and CLM yields around 60-67% accuracy, whereas the proposed deep dictionary learning yields more than 85% for all the folds. This also suggests that the proposed algorithm can be extended to other interesting applications where deep learning techniques may not provide acceptable level of performance.

### E. Clustering

Most deep learning tools are applied for classification problems. A recent study [80] proposed GraphEncoder, which uses a stacked sparse autoencoder feature learning followed by K-means clustering. In this research, we compare the results of the proposed algorithm with GraphEncoder on a subset of the datasets (we chose only those datasets with ground-truth) used by them and follow the same experimental protocol. We extract features using deep dictionary learning and use K-means clustering on the learned features. The databases selected for this experiment are described below.

**Wine** [81]: This is a dataset from the UCI Machine Learning Repository, consisting of 178 instances with 13 attributes. Every instance corresponds to a certain wine with its chemical analysis information as the attributes. All instances are labeled with 3 wine categories. We built a cosine similarity graph using these instances and used the labels as the ground truth.

**20-newsgroup** [69]. The dataset has already been discussed before. Every document as a vector of tf-idf (term frequency – inverse document frequency) scores of each word; the cosine similarity graph was built based on the tf-idf scores. To demonstrate the robustness of our algorithms with different targeting cluster numbers, we constructed three graphs built from 3, 6, and 9 different newsgroups respectively. The newsgroup names in each graph are listed as the following, where the abbreviation NG used in graph names is short for Newsgroup.

- 3-NG: corp.graphics, rec.sport.baseball, talk.politics.guns.
- 6-NG: alt.atheism, comp.sys.mac.hardware, rec.motorcycles, rec.sport.hockey, soc.religion.christian, talk.religion.misc.
- 9-NG: talk.politics.mideast, talk.politics.misc, comp.os.mswindows.misc, comp.sys.ibm.pc.hardware, sci.electronics, sci.crypt, sci.med, sci.space, misc.forsale.

For each chosen group, 200 documents are randomly

selected and thus the three graphs contain 600, 1200, and 1800 nodes respectively. The document labels are used as the ground truth.

Table VIII shows the network architecture proposed in [80]. We use the same number of basis in our deep dictionary learning framework and the results are summarized in Table IX. Since the previous work (GraphEncoder) has already shown that the superiority of their technique over spectral clustering and K-means, here, we compare the results with the GraphEncoder formulation only. Along with that, we also use a DBN framework for feature extraction followed by K-means clustering. The DBN uses the same architecture as shown in the previous table. The metrics for evaluation is normalized mutual information (NMI).

TABLE VIII  
NETWORK ARCHITECTURE

Dataset	#nodes/#basis
Wine	178-128-64
3-NG	600-512-256
6-NG	1200-1024-512-256
9-NG	1800-1024-512-256

TABLE IX  
CLUSTERING PERFORMANCE

Dataset	GraphEncoder [79]	DBN	Proposed DDL
Wine	0.85	0.79	<b>0.86</b>
3-NG	<b>0.81</b>	0.75	<b>0.81</b>
6-NG	0.60	0.53	<b>0.63</b>
9-NG	0.40	0.35	<b>0.44</b>

The results clearly demonstrate that apart from 3-NG subset of the 20-newsgroup database on which both DDL and GraphEncoder yield same accuracy, the proposed algorithm yields better results than DBN as well as GraphEncoder.

## V. CONCLUSION

In this research, we propose the idea of deep dictionary learning, where, instead of learning one shallow dictionary, we learn multiple levels of dictionaries. Learning all the dictionaries simultaneously makes the problem highly non-convex. Also learning so many parameters (atoms of many dictionaries) is always fraught with the problem of over-fitting. To account for both these issues, we learn the dictionaries in a greedy fashion – one layer at a time. The representation / features from one level is used as the input to learn the following level. Thus, the basic unit of deep dictionary learning is a simple shallow dictionary learning algorithm; which is a well known and solved problem.

Experiments are carried out for both classification and clustering problems. We compare the proposed algorithm with existing methods such as stacked autoencoder, deep belief network, and convolutional neural network. We observe that the proposed method yields comparable or better results on benchmark datasets. Results on a practical problem of appliance classification show that our method

fairs where other deep learning techniques fail. Similar to the advancements made in “deep learning”, it is our assertion that the proposed formulation of *deep dictionary learning* provides the basis to develop more efficient dictionary learning algorithms and can help in advancing state-of-the-art.

In the future, we plan to test the robustness of dictionary learning in the presence of missing data, noise and limited number of training sample. We also plan to apply this technique for other practical problems such as biometrics, vision, and speech processing. Further, there has been a lot of work on supervised dictionary learning; we expect to improve the results even further by incorporating techniques from supervised learning paradigm.

#### ACKNOWLEDGEMENT

The authors would like to thank Dr. Shobha Sundar Ram and Mr. Manoj Gulati for providing the EMI sensing data. The authors also thank the associate editor and reviewers for their constructive feedback.

#### REFERENCES

- [1] B. Olshausen and D. Field, "Sparse coding with an overcomplete basis set: a strategy employed by V1?", *Vision Research*, Vol. 37 (23), pp. 3311-3325, 1997.
- [2] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization", *Nature* 401 (6755), pp. 788–791, 1999.
- [3] R. Rubinstein, A. M. Bruckstein and M. Elad, "Dictionaries for Sparse Representation Modeling", *Proceedings of the IEEE*, Vol. 98 (6), pp. 1045-1057, 2010.
- [4] M. Aharon, M. Elad and A. Bruckstein, "K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation", *IEEE Transactions on Signal Processing*, Vol. 54 (11), pp. 4311-4322, 2006.
- [5] J. Eggert and E. Körner, "Sparse coding and NMF", *IEEE International Joint Conference on Neural Networks*, pp. 2529-2533, 2004.
- [6] M. Elad and M. Aharon, "Image Denoising Via Sparse and Redundant Representations Over Learned Dictionaries," *IEEE Transactions on Image Processing*, Vol.15 (12), pp. 3736-3745, 2006.
- [7] M. Elad and M. Aharon, "Image Denoising Via Learned Dictionaries and Sparse representation," *IEEE Conference on Computer Vision and Pattern Recognition*, Vol.1, pp. 895-900, 2006.
- [8] M. Protter and M. Elad, "Image Sequence Denoising via Sparse and Redundant Representations," *IEEE Transactions on Image Processing*, Vol.18 (1), pp. 27-35, 2009.
- [9] K. Min-Sung and E. Rodriguez-Marek, "Turbo inpainting: Iterative K-SVD with a new dictionary," *IEEE International Workshop on Multimedia Signal Processing*, pp. 1-6, 2009.
- [10] C.-H. Son and H. Choo, "Local Learned Dictionaries Optimized to Edge Orientation for Inverse Halftoning," *IEEE Transactions on Image Processing*, Vol. 23 (6), pp. 2542-2556, 2014.
- [11] J. Caballero, A. N. Price, D. Rueckert and J. V. Hajnal, "Dictionary Learning and Time Sparsity for Dynamic MR Data Reconstruction," *IEEE Transactions on Medical Imaging*, Vol. 33 (4), pp. 979-994, 2014.
- [12] D. L. Donoho, "Compressed sensing", *IEEE Trans. Info. Theory*, Vol. 52 (4), pp. 1289–1306, 2006.

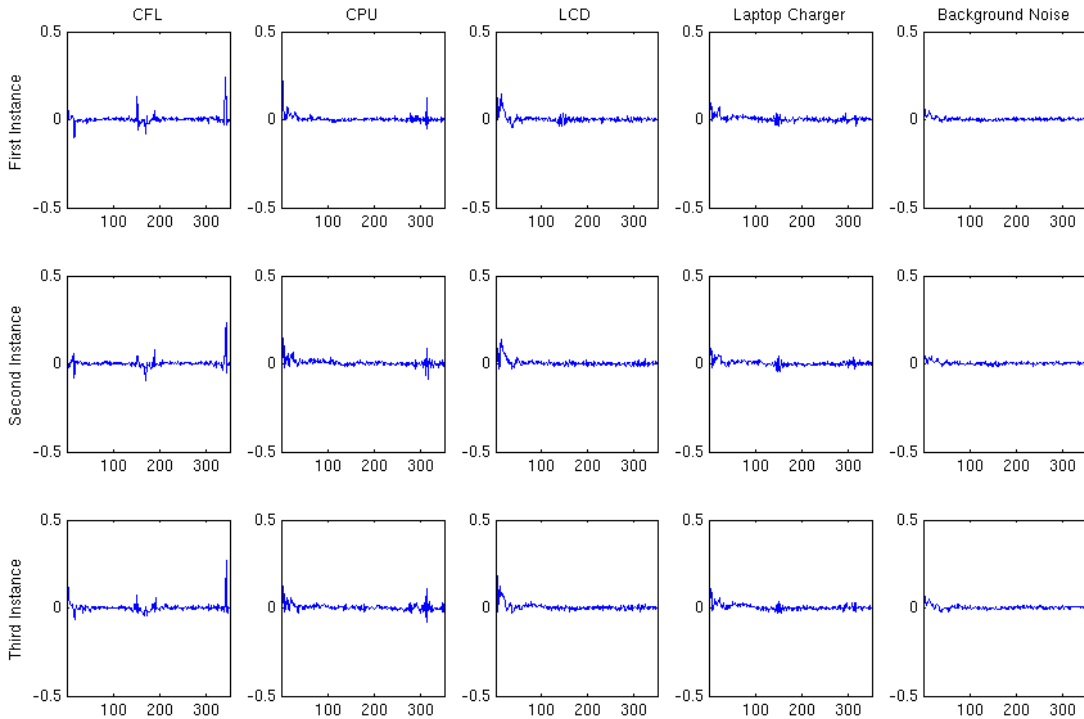


FIG. 11. CEPSTRUM FEATURES – HORIZONTAL AXIS – FREQUENCY IN KHZ AND VERTICAL AXIS – VOLT

- [13] E. J. Candes and T. Tao, "Near optimal signal recovery from random projections: Universal encoding strategies?", *IEEE Trans. Info. Theory*, Vol. 52 (12), pp. 5406–5425, 2006.
- [14] A. Majumdar and R. Ward, "Multiresolution methods in face recognition", in *Recent Advances in Face Recognition*, Eds. M. S. Bartlett, K. Delac and M. Grgic, I-Tech Education and Publishing, Vienna, Austria, pp. 79-96, 2009.
- [15] V. Dattatray, R. Jadhav and S. Holambe, "Feature extraction using Radon and wavelet transforms with application to face recognition", *Neurocomputing*, Vol. 72 (7-9), pp. 1951-1959, 2009.
- [16] S. Dabbaghchian, P. M. Ghaemmaghami, A. Aghagolzadeh, "Feature extraction using discrete cosine transform and discrimination power analysis with a face recognition technology", *Pattern Recognition*, Vol. 43 (4), pp. 1431-1440, 2010.
- [17] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. "Discriminative learned dictionaries for local image analysis". *IEEE Conference of Computer Vision and Pattern Recognition*, 2008.
- [18] L. Yang, R. Jin, R. Sukthankar, and F. Jurie. "Unifying discriminative visual codebook generation with classifier training for object category recognition". *IEEE Conference of Computer Vision and Pattern Recognition*, 2008.
- [19] W. Jin, L. Wang, X. Zeng, Z. Liu and R. Fu, "Classification of clouds in satellite imagery using over-complete dictionary via sparse representation", *Pattern Recognition Letters*, Vol. 49 (1), pp. 193-200, 2014.
- [20] Y. Boureau, F. Bach, Y. LeCun, and J. Ponce. "Learning mid-level features for recognition". *IEEE Conference of Computer Vision and Pattern Recognition*, 2010.
- [21] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. "Supervised dictionary learning". *Advances in Neural Information Processing Systems*, 2008.
- [22] N. Khan and M. F. Tappin, "Stable Discriminative Dictionary Learning via Discriminative Deviation", *International Conference on Pattern Recognition*, 2012.
- [23] K. Huang and S. Aviyente. "Sparse representation for signal classification". *Advances in Neural Information Processing Systems*, 2007.
- [24] D. Pham and S. Venkatesh. "Joint learning and dictionary construction for pattern recognition". *IEEE Conference of Computer Vision and Pattern Recognition*, 2008.
- [25] A. Fawzi, M. Davies, P. Frossard, "Dictionary learning for fast classification based on soft-thresholding", *arXiv:1402.1973*
- [26] J. Yang, K. Yu, and T. Huang. "Supervised translation-invariant sparse coding". *IEEE Conference of Computer Vision and Pattern Recognition*, 2010.
- [27] J. Mairal, M. Leordeanu, F. Bach, M. Hebert, and J. Ponce. "Discriminative sparse image models for class-specific edge detection and image interpretation". *European Conference on Computer Vision*, 2008.
- [28] Q. Zhang and B. Li, "Discriminative K-SVD for dictionary learning in face recognition". *IEEE Conference of Computer Vision and Pattern Recognition*, 2010.
- [29] Z. Jiang, Z. Lin and L. S. Davis, "Learning A Discriminative Dictionary for Sparse Coding via Label Consistent K-SVD", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 35, pp. 2651-2664, 2013
- [30] G. E. Hinton and R. R. Salakhutdinov, "Reducing the Dimensionality of Data with Neural Networks", *Science*, Vol. 313 (5786), pp. 504–507, 2006.
- [31] H. Bourlard and Y. Kamp, "Auto-association by multilayer perceptrons and singular value decomposition". *Biological Cybernetics*, Vol. 59 (4–5), pp. 291–294, 1989.
- [32] Y. Bengio, P. Lamblin, P. Popovici and H. Larochelle, "Greedy Layer-Wise Training of Deep Neural Networks", *Advances in Neural Information Processing Systems*, 2007.
- [33] G. E. Hinton, S. Osindero and Y. W. Teh, "A fast learning algorithm for deep belief nets", *Neural Computation*, Vol. 18, pp. 1527-1554, 2006.
- [34] Y. Bengio, "Learning deep architectures for AI", *Foundations and Trends in Machine Learning*, Vol. 1(2), pp. 1-127, 2009.
- [35] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio and P. A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion", *Journal of Machine Learning Research*, Vol. 11, pp. 3371-3408, 2010.
- [36] K. Engan, S. Aase, and J. Hakon-Husoy, "Method of optimal directions for frame design," *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1999.
- [37] B. K. Natarajan, "Sparse approximate solutions to linear systems", *SIAM Journal on computing*, Vol. 24, pp. 227-234, 1995.
- [38] Y. Pati, R. Rezaifar, P. Krishnaprasad, "Orthogonal Matching Pursuit: recursive function approximation with application to wavelet decomposition", *Asilomar Conference on Signals, Systems and Computers*, 1993.
- [39] M. Yaghoobi, T. Blumensath and M. E. Davies, "Dictionary Learning for Sparse Approximations With the Majorization Method," *IEEE Transactions on Signal Processing*, Vol. 57 (6), pp. 2178-2191, 2009.
- [40] A. Rakotomamonjy, "Applying alternating direction method of multipliers for constrained dictionary learning", *Neurocomputing*, Vol. 106, pp. 126-136, 2013.
- [41] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma. "Robust face recognition via sparse representation". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 31, 2, pp. 210–227, 2009.
- [42] A. Majumdar and R. K. Ward, "Robust Classifiers for Data Reduced via Random Projections", *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, Vol. 40 (5), pp. 1359 - 1371.
- [43] A. Majumdar and R. K. Ward, "Fast Group Sparse Classification", *IEEE Canadian Journal of Electrical and Computer Engineering*, Vol. 34 (4), pp. 136-144, 2009
- [44] A. Majumdar and R. K. Ward, "Improved Group Sparse Classifier", *Pattern Recognition Letters*, Vol. 31 (13), pp. 1959-1964, 2010
- [45] J. Yin, Z. Liu, Z. Jin and W. Yang, "Kernel sparse representation based classification", *Neurocomputing*, Vol. 77 (1), pp. 120-128, 2012.
- [46] Y. Chen, N. M. Nasrabadi, T. D. Tran, "Hyperspectral Image Classification via Kernel Sparse Representation," *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 51 (1), pp. 217-231, 2013.
- [47] L. Zhang, W. D. Zhou, P. C. Chang, J. Liu, Z. Yan, T. Wang and F. Z. Li, "Kernel Sparse Representation-Based Classifier," *IEEE Transactions on Signal Processing*, Vol. 60 (4), pp. 1684-1695, 2012.
- [48] M. Yang, L. Zhang, J. Yang, and D. Zhang. metaface learning for sparse representation based face recognition. *IEEE International Conference on Image Processing*, 2010.
- [49] I. Ramirez, P. Sprechmann, and G. Sapiro. Classification and clustering via dictionary learning with structured incoherence and shared features. *IEEE Conference of Computer Vision and Pattern Recognition*, 2010.
- [50] M. Yang, L. Zhang, X. Feng, and D. Zhang. "Fisher discrimination dictionary learning for sparse representation", *IEEE International Conference on Computer Vision*, 2011.
- [51] H. Larochelle and Y. Bengio, "Classification using Discriminative Restricted Boltzmann Machines", *International Conference on Machine Learning*, 2008.
- [52] Z. Cui, S. S. Ge, Z. Cao, J. Yang and H. Ren, "Analysis of Different Sparsity Methods in Constrained RBM for Sparse Representation in Cognitive Robotic Perception", *Journal of Intelligent Robot and Systems*, pp. 1-12, 2015.
- [53] H. Luo, R. Shen and C. Niu, "Sparse Group Restricted Boltzmann Machines", *arXiv:1008.4988v1*
- [54] R. Salakhutdinov and G. Hinton, "Deep Boltzmann Machines", *International Conference on Artificial Intelligence and Statistics*, 2009.
- [55] K.H. Cho, T. Raiko and A. Ilin, "Gaussian-Bernoulli deep Boltzmann machine," *IEEE International Joint Conference on Neural Networks*, 2013, pp. 1-7, 2013.
- [56] A. Makhzani and B. Frey, "k-Sparse Autoencoders", *arXiv:1312.5663*, 2013.
- [57] K. Cho, "Simple sparsification improves sparse denoising autoencoders in denoising highly noisy images", *International Conference on Machine Learning*, 2013.
- [58] S. Rifai, P. Vincent, X. Muller, X. Glorot, Y. Bengio: Contractive Auto-Encoders: Explicit Invariance During Feature Extraction, *International Conference on Machine Learning*, 2011
- [59] P. Jain, P. Netrapalli and S. Sanghavi, "Low-rank Matrix Completion using Alternating Minimization", *Symposium on Theory Of Computing*, 2013.
- [60] A. Agarwal, A. Anandkumar, P. Jain and P. Netrapalli, "Learning Sparsely Used Overcomplete Dictionaries via Alternating Minimization", *International Conference On Learning Theory*, 2014.

- [61] D. A. Spielman, H. Wang and J. Wright, "Exact Recovery of Sparsely-Used Dictionaries", International Conference On Learning Theory, 2012
- [62] S. Arora, A. Bhaskara, R. Ge and T. Ma, "More Algorithms for Provable Dictionary Learning", arXiv:1401.0579v1
- [63] C. Hillar and F. T. Sommer, "When can dictionary learning uniquely recover sparse data from subsamples?", arXiv:1106.3616v3.
- [64] I. Daubechies, M. Defrise, C. De Mol, "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint", Communications on Pure and Applied Mathematics, Vol. 57: 1413-1457, 2004.
- [65] R. Rubinstein, T. Peleg and M. Elad, Analysis K-SVD: A Dictionary-Learning Algorithm for the Analysis Sparse Model, IEEE Transactions on Signal Processing, Vol. 61 (3), pp. 661-677, 2013.
- [66] Y. Suo, M. Dao, U. Srinivas, V. Monga, T. D. Tran, "Structured Dictionary Learning for Classification", arXiv:1406.1943
- [67] L. Bar and G. Sapiro, "Hierarchical dictionary learning for invariant classification", IEEE ICASSP, pp. 3578 – 3581, 2010.
- [68] R. Rubinstein, M. Zibulevsky and M. Elad, "Double Sparsity: Learning Sparse Dictionaries for Sparse Signal Approximation," in IEEE Transactions on Signal Processing, Vol. 58, no. 3, pp. 1553-1564, March 2010.
- [69] A. Courville, J. Bergstra and Y. Bengio, "An Empirical Evaluation of Deep Architectures on Problems with Many Factors of Variation", ICML 2007.
- [70] <http://people.csail.mit.edu/jrennie/20Newsgroups/20news-bydate-matlab.tgz>
- [71] G. Tzanetakis and P. Cook, "Musical genre classification of audio signals", IEEE Transactions on Audio and Speech Processing 2002.
- [72] <http://www.cs.toronto.edu/~hinton/MatlabForSciencePaper.html>
- [73] <http://ceit.aut.ac.ir/~keyvanrad/DeeBNet%20Toolbox.html>
- [74] T.-H. Chan, K. Jia, S. Gao, J. Lu, Z. Zeng and Y. Ma, "PCANet: A Simple Deep Learning Baseline for Image Classification?" IEEE Transactions on Image Processing, Vol.24 (12), pp.5017-5032, 2015
- [75] S. Gupta, M. S. Reynolds, and S.N. Patel. "ElectriSense: single-point sensing using EMI for electrical event detection and classification in the home." ACM UBICOMP. 2010.
- [76] M. Gulati, S. S. Ram, and A. Singh. "An in depth study into using EMI signatures for appliance identification." ACM BuildSys. 2014.
- [77] M. Gulati, S. S. Ram, A. Majumdar and A. Singh, "An In Depth Study into Using EMI Signatures for Appliance Identification", [http://www.academia.edu/10808246/An\\_In\\_Depth\\_Study\\_into\\_Using\\_EMI\\_Signatures\\_for\\_Appliance\\_Identification](http://www.academia.edu/10808246/An_In_Depth_Study_into_Using_EMI_Signatures_for_Appliance_Identification).
- [78] S. Kong, Y. Kim; S. K. Joo and J. H. Kim, "Home appliance load disaggregation using cepstrum-smoothing-based method", IEEE Transactions on Consumer Electronics, Vol. 61 (1), pp. 24-30, 2015.
- [79] G. Brown, A. Pocock, M. J. Zhao, M. Luján, "Conditional Likelihood Maximisation: A Unifying Framework for Information Theoretic Feature Selection", Journal of Machine Learning Research, Vol. 13, pp. 27–66, 2012.
- [80] F. Tian, B. Gao, Q. Cui, E. Chen, T. Y. Liu, "Learning Deep Representations for Graph Clustering", AAAI, 2014.
- [81] <https://archive.ics.uci.edu/ml/datasets/Wine>