Group Sparse Autoencoder

Anush Sankaran, Mayank Vatsa, Richa Singh, Angshul Majumdar

Indraprastha Institute of Information Technology (IIIT) Delhi, India

Abstract

Unsupervised feature extraction is gaining a lot of research attention following its success to represent any kind of noisy data. Owing to the presence of a lot of training parameters, these feature learning models are prone to overfitting. Different regularization methods have been explored in the literature to avoid overfitting in deep learning models. In this research, we consider autoencoder as the feature learning architecture and propose $\ell_{2,1}$-norm based regularization to improve its learning capacity, called as Group Sparse AutoEncoder (GSAE). $\ell_{2,1}$-norm is based on the postulate that the features from the same class will have a common sparsity pattern in the feature space. We present the learning algorithm for group sparse encoding using majorization-minimization approach. The performance of the proposed algorithm is also studied on three baseline image datasets: MNIST, CIFAR-10, and SVHN. Further, using GSAE, we propose a novel deep learning based image descriptor for minutia detection from latent fingerprints. Latent fingerprints contain only a partial finger region, very noisy ridge patterns, and depending on the surface it is deposited, contain significant background noise. We formulate the problem of minutiae extraction as a two-class classification problem and learn the descriptor using the novel formulation of GSAE. Experimental results on two publicly available latent fingerprint datasets show that the proposed algorithm yields state-of-the-art results in automated minutia extraction.

Keywords: Supervised Autoencoder, Group Sparsity, Latent Fingerprint, Minutia Extraction

1. Introduction

Feature representation is an integral component of any object recognition task. A meaningful and representative feature can help in obtaining higher recognition/classification accuracies. However, there is no universal feature extraction algorithm which works best for all types of applications (i.e., no free lunch theorem applies on features as well). Therefore, researchers have proposed several feature representation algorithms. Broadly, existing algorithms can be classified into two categories: hand-crafted features and learnt features. The majority of the literature has focused on hand-crafted features such as Gabor features, local binary pattern and Scale invariant feature transform. In the last one decade, learning based representation algorithms have gained widespread attention [1], [2]. These algorithms utilize a large amount of training data to learn discriminatory feature representations that can tolerate noise and variations in data distribution. Popular examples of such algorithms include dictionary learning and deep learning (autoencoder, deep belief network, and convolutional neural network) [3]. Further, advancements in computing technology and GPU technology has instigated research in learning based representation approaches and almost every domain where abundant data is available, these approaches are providing state-of-the-art results.

One of the popular deep learning algorithms is the autoencoder which is a multi layer network that learns a non-linear mapping function between data and its feature space. Structurally, an autoencoder is similar to a feed-forward neural network; it consists of an input layer, a set of hidden layers, and an output layer which attempts to reconstruct the input layer data. It has been widely used for feature extraction and is unsupervised in nature, as it does not require labels for learning the features. Such networks are known to be universal approximators of any continuous function, with limited assumptions on the activation function [4], [5], [6]. As shown in Figure 1, the encoding and decoding layers are just inverse of each other, making the network symmetrical about the feature (innermost) layer. Bengio et al. [7] proposed a greedy layer-by-layer training approach to learn a multiple hidden layer stacked autoencoder. An autoencoder architecture has two kinds of parameters: nodes and weighted connections. Depending on the number of nodes in each layer and also by enforcing sparsity in learning the weights, different classes of mapping functions can be learnt by the autoencoder. However, as the number of nodes in the hidden layers increases, the number of weight parameters to be learnt exponentially increases, causing two important challenges (i) requiring large amount of data to train the network and (ii) the network parameters tend to overfit to the given training data.

Email address: `{anushs, mayank, rsingh, angshul}@iiitd.ac.in` (Anush Sankaran, Mayank Vatsa, Richa Singh, Angshul Majumdar)

Several regularization methods have been adopted to prevent overfitting in an autoencoder network, as shown in Figure 1. Generally, the overfitting resolution techniques employed with network-based-architectures aim to achieve one or both of these goals.

- avoid peaking of weights by adding penalty or normalizing the weights; for example ℓ_2-norm [8], max-norm [9], Contractive AutoEncoder [10] and
- introducing sparsity to the learned weights to avoid learning "noisy" patterns; for example ℓ_1-norm [8], KL-divergence [11], and maxout [12].

In several statistical regularization techniques, overfitting can be prevented by methods such as dropout [9] and dropconnect [13] approaches. In dropout, nodes are dropped (output set to 0) randomly with a probability p during the training phase and in dropconnect, the connections are dropped (weight set to 0) at random with a probability p. Often one or many of these regularizers are used together complementing their properties to achieve better learning; for example $\ell_2+\ell_1$ towards the end of training, Dropout+ℓ_2, and Dropout+max-norm. In a recent work, the dropall [14] framework proposes a combination of dropout and dropconnect, where the nodes, as well as the connections, are dropped. Hong et al. [15] have proposed a ℓ_1-norm based sparsity preserving feature representation algorithm at multiple stages to integrate multiview information. Further, Hong et al. [16] have adopted a $\ell_{2,1}$ regularization mechanism for handling variational noise in data. The $\ell_{2,1}$ is approximated as a l_2-norm of every column in the low-rank representation (LRR) of the original matrix.

It is well known in the machine learning community that supervised feature extraction usually leads to better classification [17]. For instance, incorporating Fisher criterion for determining projections in subspace methods reduces the intra-class variability and increases the inter-class variability, thereby increasing discrimination capabilities. Motivated by this observation, in this research, group sparse autoencoder is proposed which is the supervised version of autoencoders. The primary contributions of this research are as follows:

- Propose a group sparse autoencoder (GSAE) and derive a solution using majorization-minimization approach [18],
- Evaluate the performance of GSAE on baseline object classification datasets such as MNIST [19], CIFAR-10 [20], and SVHN [21],
- Utilize GSAE as a novel image descriptor for local latent fingerprint patches that can better distinguish the presence or absence of minutia, and
- Perform extensive testing and analysis of the proposed minutiae extractor on two public latent fingerprint databases, namely NIST SD-27 [22] and MOLF [23].

The remaining paper is organized as follows: Section 2 explains the proposed group sparse autoencoder algorithm and the solution for the optimization function is derived. Section 3 discusses the performance of the proposed algorithm on baseline object classification datasets including MNIST, CIFAR-10, and SVHN. Finally, Section 4 presents the proposed approach for minutiae extraction from latent fingerprints, the fingerprint databases used, and the results obtained.

2. GSAE: Group Sparse AutoEncoders

Autoencoders are generally unsupervised in nature and leverage the availability of large unlabeled data for feature representation. However, if a large amount of labelled data is available, the standard formulation of autoencoder needs to be updated to incorporate the labeled information. Sang et al. [24] recently proposed a supervised loss function in which they optimize the squared loss and the classification loss, simultaneously. Gao et al. [25] proposed a supervised deep autoencoder for face recognition by reducing the loss between a probe image and its corresponding gallery image. In this research, we propose a novel approach for modeling stacked sparse autoencoder that preserves group sparsity. The learning is performed such that the features of a single class will have the same sparsity signature. In other words, the non-zero values in the features occur at the same positions for a class. This is achieved by incorporating $\ell_{2,1}$-norm regularization [26], [27], [28].

The description of the proposed algorithm starts with the explanation of an autoencoder. Let X be the input data, where

$$X = \left[\underbrace{\{x_{1,1} \ldots x_{1,n_1}\}}_{X_1=\text{class 1}}, \ldots, \underbrace{\{x_{C,1} \ldots x_{C,n_c}\}}_{X_C=\text{class C}} \right] \quad (1)$$

Here, C is the number of classes, $\{n_1, n_2, \ldots, n_C\}$ are the number of data points in each of the C classes, and the data is organized such that all the data columns belonging to class 1 appear first, followed by data columns of class 2, and so on till data columns of class C. To learn a single layer generative autoencoder model, the loss function J is defined as:

$$J(W) = \arg\min_{W,U} \left[||X - U\phi(WX)||_2^2 + \lambda R(W) \right] \quad (2)$$

where, ϕ is a non-linear activation function such as sigmoid function, W and U are the encoding and decoding weights, respectively. Higher order representation can be learnt by stacking multiple layers together and training them in a greedy layer wise fashion. $R(W)$ can be any regularization

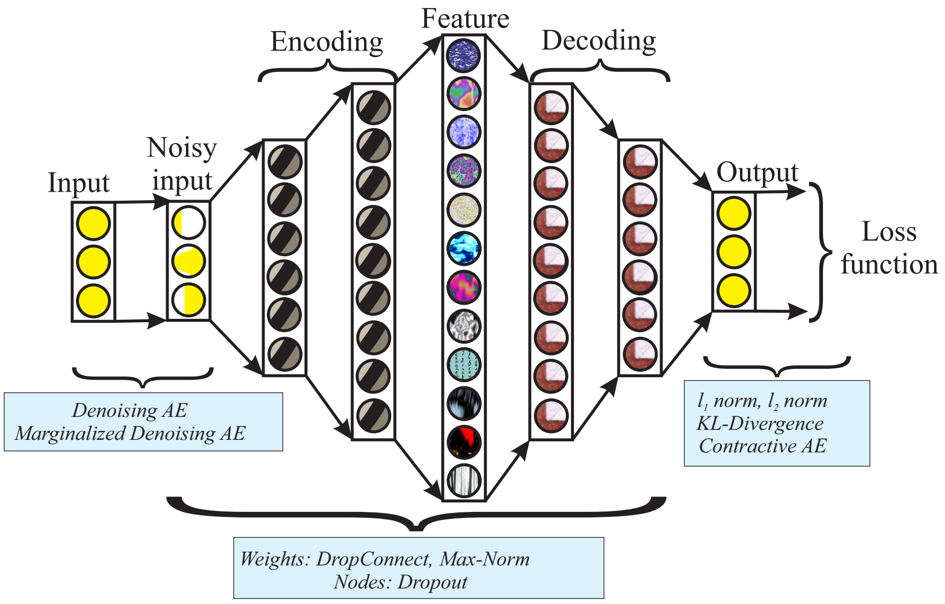

Figure 1: An overview of different kinds of regularization techniques used in the literature for an autoencoder based architecture. Note that the number of nodes in each hidden later is varying and may be dense or sparse depending on the application, data characteristics, and architecture.

function, controlled by the parameter λ, to avoid overfitting by introducing some additional constraints while learning the weight matrix. Some popular regularization functions are

- LASSO or the ℓ_1-norm enforces sparse learning of weights,
- Euclidean or the ℓ_2-norm adds higher penalty to the peak weights, thereby enforcing diffused learning of weights, and
- Elastic net or $(\ell_1+\ell_2)$-norm adds both the norms in the optimization function.

In the proposed group-sparse autoencoder framework, we introduce a $\ell_{2,1}$-norm based regularization as follows:

$$J(W) = \arg\min_{W,U} [||X - U\phi(WX)||_2^2 + \lambda \sum_{c=1}^{C} ||WX_c||_{2,1}] \quad (3)$$

where, $||\bullet||_{2,1} = \sum_j ||Z^{j\rightarrow}||_2$ is the sum of ℓ_2-norms of the rows (indicated by j). The inner ℓ_2-norm promotes a dense (non-zero) solution within the selected rows, however the outer ℓ_1-norm (sum) enforces sparsity in selecting the rows. In this proposed formulation, the regularizer enforces group sparsity within each class by adding the constraint that the features from the same group/class should have the similar sparsity signature. Note that, $\phi(\bullet)$ is only a clipping function applied term-by-term. Therefore, the second term of $\ell_{2,1}$-norm can be applied to both $\phi(WX)$ or just WX as both promote row-sparsity. This makes the optimization supervised as the information regarding the class labels is utilized during training. However, we are not enforcing any discriminative property to the features as we are not enforcing features from different groups to have different sparsity signatures.

2.1. Solution using Majorization-Minimization

The objective function in equation 3 is a non-convex optimization problem that can be solved using alternating minimization. At any k^{th} iteration, the solution for the non-convex problem can be split into two steps as follows:

$$\textbf{Step 1}: U = \arg\min_{U} ||X - U\phi(W_{(k-1)}X)||_2^2$$

$$\textbf{Step 2}: W = \arg\min_{W} \left[\| X - U_{(k)}\phi(WX) \|_2^2 + \lambda \sum_{c=1}^{C} ||WX_c||_{2,1} \right] \quad (4)$$

Step 1 is a linear least squares regression problem having a closed form solution. Step 2 is challenging; hence, we adopt the Majorization-Minimization (MM) [18] algorithm to solve it. In this approach, let $J(W)$ be the function to be minimized. For the initial point w_0, a smooth function $G_0(W)$ is constructed through w_0 which has a higher value than $J(W)$ for all values of w apart from w_0, at which the values are the same. This is the Majorization step where a smooth function $G_0(W)$ is constructed which is easy to minimize. Iteratively at each step, $G_k(W)$ is minimized to obtain the next iteration x_{k+1}. It can be understood that the solution at every iteration gets closer to the actual solution. For mathematical convenience, Step 2 can be rewritten as,

$$arg\min_{Z} \left[\| X - U_{(k)}\phi(Z) \|_2^2 + \lambda \sum_{c=1}^{C} ||Z_c||_{2,1} \right] \quad (5)$$

where, $Z_c = WX_c$ and Z is obtained by stacking the Z_c's in column. In this optimization problem, only the least square regression term has to be majorized and the penalty term is not affected. During the minimization step, the surrogate majorizer function, $G_k(Z)$ of the actual loss function $J(W)$ is chosen as follows,

$$\begin{aligned} G_k(Z) =& \| X - U_{(k)}\phi(Z) \|_2^2 + \lambda \sum_{c=1}^{C} ||Z_c||_{2,1} \\ &+ (\phi(Z) - \phi(Z)_{(k)})^T (aI - U_{(k)}^T U_{(k)})(\phi(Z) - \phi(Z)_{(k)}) \end{aligned} \tag{6}$$

Here, a is the maximum eigenvalue of the matrix $U_{(k)}^T U_{(k)}$ and I is the identity matrix. By simplifying $G_k(Z)$, we obtain,

$$\begin{aligned} G_k(Z) = & X^T X - 2X^T U_{(k)}\phi(Z) + \phi(Z)^T U_{(k)}^T U_{(k)}\phi(Z) \\ &+ (\phi(Z) - \phi(Z)_{(k)})^T (aI - U_{(k)}^T U_{(k)})(\phi(Z) - \phi(Z)_{(k)}) \\ &+ \lambda \sum_{c=1}^{C} ||Z_c||_{2,1} \end{aligned} \tag{7}$$

$$\begin{aligned} \implies G_k(Z) = & X^T X + \phi(Z)_{(k)}^T (aI - U_{(k)}^T U_{(k)})\phi(Z) \\ & - 2\left[X^T U_{(k)} + \phi(Z)_{(k)}^T (aI - U_{(k)}^T U_{(k)})\right]\phi(Z) \\ & + a\phi(Z)^T \phi(Z) + \lambda \sum_{c=1}^{C} ||Z_c||_{2,1} \end{aligned} \tag{8}$$

Let $B = \phi(Z)_{(k)}^T + \frac{1}{a} U_{(k)}^T (X^T - U_{(k)}\phi(Z)_{(k)}^T)$, equation 8 can be written as

$$G_k(Z) = a(-2B^T\phi(Z) + \phi(Z)^T\phi(Z)) + \lambda \sum_{c=1}^{C} ||Z_c||_{2,1} + \mathcal{E} \tag{9}$$

where, $\mathcal{E}$ consists of constant terms. Using the identity, $\| B - \phi(Z) \|_2^2 = B^T B - 2B^T\phi(Z) + \phi(Z)^T\phi(Z)$, equation 9 can be rewritten as

$$G_k(Z) = a\left(\| B - \phi(Z) \|_2^2 + \frac{\lambda}{a} \sum_{c=1}^{C} ||Z_c||_{2,1} \right) - aB^T B + \mathcal{E} \tag{10}$$

Removing the constant terms and re-writing in terms of W, the optimization function can be written as,

$$\arg\min_W \left(\| B^T - \phi(WX)^T \|_2^2 + \frac{\lambda}{a} \sum_{c=1}^{C} ||(WX_c)^T||_{2,1} \right) \tag{11}$$

All the matrices are written in terms of transpose, as the activation function is computed element-wise. Blumensath [29] has shown that it is possible to replace the above non-linear problem, into a simple linear problem using one step of gradient descent, as follows:

$$\arg\min_W \left(\| P - W^T \|_2^2 + \frac{\lambda}{a} \sum_{c=1}^{C} ||X_c^T W^T||_{2,1} \right) \tag{12}$$

where, $P = W_{(k)}^T - \sigma\nabla \| B^T - \phi(WX)^T \|_2^2 \Big|_{W_{(k)}}$, σ is the step size for gradient descent and can be found using Lipschitz bound. Summation can be removed by redefining equation 12 as follows,

$$\arg\min_W \left(\| P - W^T \|_2^2 + \frac{\lambda}{a} ||VW^T||_{2,1} \right) \tag{13}$$

where, V is defined as the block row concatenation of X_c^T's. Taking the derivative of equation 13 and setting it to zero, we obtain,

$$2P - 2W^T + \frac{\lambda}{a} V^T D V W^T = 0 \tag{14}$$

where, $D = diag(|VW^T|^{-1})$

$$\left(I + \frac{\lambda}{2a} V^T D V \right) W^T = P \tag{15}$$

Using matrix inversion lemma,

$$\begin{aligned} \left(I + \frac{\lambda}{2a} V^T D V \right)^{-1} &= I - V^T \left(\frac{2a}{\lambda} D^{-1} + V^T V \right)^{-1} V \\ \implies W^T &= P - V^T \left(\frac{2a}{\lambda} D^{-1} + V^T V \right)^{-1} VP \end{aligned} \tag{16}$$

If $T = \left(\frac{2a}{\lambda} D^{-1} + V^T V\right)^{-1} VP$, then equation 16 becomes $W^T = P - V^T T$. The solution for T is as follows,

$$\begin{aligned} &\left(\frac{2a}{\lambda} D^{-1} + V^T V \right) W^T = VP \\ \implies T = &\left(\frac{2a}{\lambda} D^{-1} + CI \right)^{-1} \left(cT_{(k-1)} + V(P - V^T T_{(k-1)}) \right) \end{aligned} \tag{17}$$

is obtained by adding cT on both sides of equation and subtracting with $V^T V W^T$. c is the maximum eigenvalue of $V^T V$. The complete algorithm is summarized as follows:

```
Initialize: W, V
For every iteration:
Step 1 : $U_{(k)} = \arg\min_U \| X - U\phi(WX) \|_2^2$
Step 2 : $B = \phi(Z)_{(k)}^T + \frac{1}{a} U_{(k)}^T (X^T - U_{(k)}\phi(Z)_{(k)}^T)$
Step 3 : $P = W_{(k)}^T - \sigma\nabla \| B^T - \phi(WX)^T \|_2^2 \Big|_{W_{(k)}}$
Step 4 : $T = \left(\frac{2a}{\lambda} D^{-1} + CI\right)^{-1} \left(cT_{(k-1)} + V(P - V^T T_{(k-1)})\right)$
Step 5 : $W = P^T - T^T V$
```

Another popular approach for solving the non-convex optimization problem is Alternating Direction Method of Multipliers (ADMM) [30]. However, this approach introduces a lot of hyper-parameters, that require fine-tuning. It can be understood that the proposed approach utilizes only the regularization constant λ as the parameter. The remaining parameters such as a, c, and σ can be computed and fixed.

Table 1: Overview of the standard image data sets used in the experiments.

Dataset	Image size	Train set	Test set
MNIST	28×28 gray	$60,000$	$10,000$
CIFAR-10	32×32 color	$60,000$	$10,000$
SVHN	32×32 color	$604,388$	$26,032$

2.2. Classification

In this research, we have used 2ν Support Vector Machine (2ν-SVM) [31] with radial basis function kernel for classification. 2ν-SVM is a "cost-sensitive" version of SVM that penalizes the training errors of one class more than the other by assigning class specific weights to both the classes. This explicit penalty minimizes the false negatives while restricting the false positives below a certain significance level. Hence, in the case of imbalanced class data or different cost of error, different importance can be given to the two types of errors, making sure that the majority class is not creating a bias. Further, in case of c-class classification problems, c different binary classifiers are created using the "one-vs-all" approach to train binary 2ν-SVMs. The primal form of 2ν-SVM optimization function [31] is given as

$$\min_{w,b,\psi,\rho} \frac{1}{2}||w||^2 - \nu\rho + \frac{\gamma}{n}\sum_{i\in I_+}\psi_i + \frac{1-\gamma}{n}\sum_{i\in I_-}\psi_i \quad (18)$$

such that, (i) $y_i(k(w,x_i)+b) \geq \rho-\psi_i$, (ii) $\psi_i \geq 0$, and (iii) $\rho \geq 0$. Here, w is the decision boundary, x are the support vectors, y are the corresponding class labels, $k(w,x_i)$ is the kernel function, ψ_i are the slack variables, $\gamma \in \{0,1\}$ is a parameter controlling the trade-off between false positives and false negatives, and $i = \{1,2,\ldots,n\}$ for n support vectors.

3. Experimental Results on Standard Image Datasets

To showcase the effectiveness and compare with existing (similar) approaches, we demonstrate the results on three standard databases namely, MNIST [19], CIFAR-10 [20], and SVHN [21].

3.1. Standard Image Datasets

The properties of the datasets are summarized in Table 1 and a short description of the standard image datasets are provided below:

- **MNIST:** It is a handwritten digit classification problem from gray-scale images of size 28×28. The dataset has ten classes of digits varying from $\{0-9\}$, with 6000 images per class. As per the defined protocol, 5000 images per class are used for training while the remaining 1000 of them are used for testing, making it a total of $50,000$ images for training and $10,000$ for testing.
- **CIFAR-10:** It is a labelled subset of the 80 million tiny images dataset [32]. It contains colored images corresponding to ten different object classes, and the size of the images are 32×32. The ten classes are airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. 6000 images are available per class, with $50,000$ images for training and $10,000$ images for testing. The dataset is split into five training sets and one testing set, each containing $10,000$ images. The testing set consists of exactly 1000 images from each class.
- **SVHN:** It is a real world image dataset consisting of house numbers (digits, similar to MNIST) obtained from Google Street View images. The dataset consists of 10 classes with digits from (0 to 9). Each image is RGB and of size 32×32 centered around a single character. There are $73,257$ training images, $26,032$ testing images, and an additional set of $531,131$ images is used for training.

All the standard protocols provided with the databases are followed for evaluation, thus making the results comparable with the literature.

3.2. Experimental Setup

In the experimental setup, each of the hidden layers is pre-trained in a greedy layer-wise fashion, and the overall network is further fine-tuned using the training set[1]. The other hyper-parameters of our model are as follows, constant learning rate = 0.01, regularization parameter = 0.45, initial momentum = 0.5, final momentum = 0.95, and KL-divergence (KLD) sparsity constant = 0.05. The effectiveness of the proposed algorithm is demonstrated with the following regularizers:

1. **KLD**: In the standard architecture of autoencoder, only KL-divergence based regularization is used during both pre-training and fine-tuning.
2. **GSAE**: This utilizes only the proposed $\ell_{2,1}$ norm to introduce group-sparsity into the training loss function. In this architecture, $\ell_{2,1}$-norm based supervision is used during both greedy layer-wise pre-training, as well as, the overall architecture fine-tuning.
3. **KLD + GSAE**: This architecture shows that the proposed $\ell_{2,1}$-norm can be used to complement other existing regularization methods. In this method, greedy layer-wise pre-training is performed using KL-divergence based sparsity regularizer (without class labels). $\ell_{2,1}$-norm based supervision is performed only during fine-tuning. This hybrid architecture suggests that the existing pre-trained architectures can be fine-tuned using the proposed regularizer to obtain an improvement in performance.

[1]The network design choices are inspired from the elaborate research from Hinton and Salakhutdinov [33].

Table 2: The performance of the proposed algorithm using different regularizers on the standard image data sets. The architecture of the autoencoder used for each dataset is also provided.

Dataset	**Architecture**	**Performance Metric**	**KLD**	**GSAE**	**KLD + GSAE**
MNIST	[784 500 500]	Error Rate (%)	1.71	1.19	1.10
CIFAR-10	[3072 2000 2000]	Accuracy (%)	74.3	76.8	77.4
SVHN	[1024 1000 1500]	Accuracy (%)	89.9	92.1	92.4

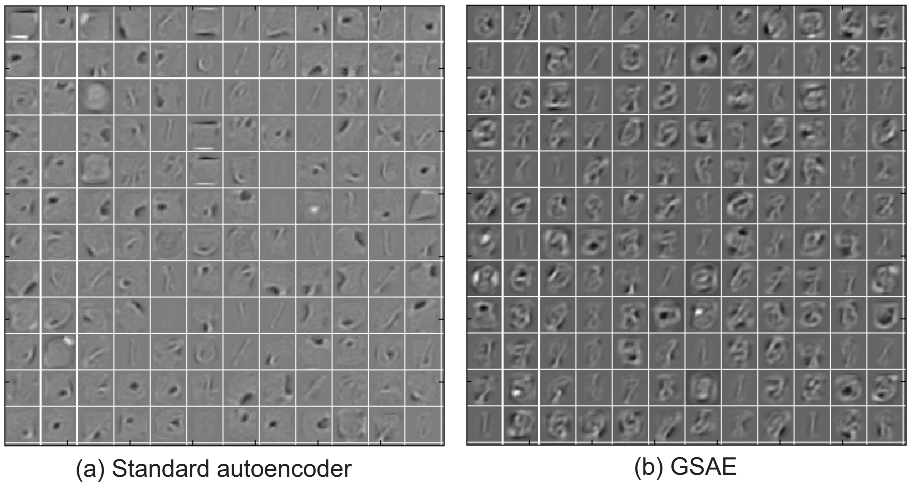

Figure 2: Visualization of the features learnt in the first hidden layer of the autoencoder on MNIST dataset with (a) standard autoencoder using only KL-divergence based sparsity, (b) proposed GSAE learning algorithm.

3.3. Evaluation Metrics

The three standard image datasets use different metrics to evaluate and report the performance of algorithms. On MNIST dataset, error rate (%) is used which measures the percentage of misclassifications. For instance, an error rate of 0.68 means 68 images out of $10,000$ images are misclassified. For the CIFAR-10 and SVHN datasets, the accuracy (%) of correct classification is used as a metric to evaluate the performance of algorithms.

3.4. Image Classification Performance

Table 2 shows the results of the proposed GSAE algorithm on the three standard image datasets along with the performance of some existing algorithms. On the MNIST dataset, apart from the aforementioned experimental setup, we compare our best reported result along with the best reported results of variants of autoencoder proposed in the literature. The autoencoder variants that are compared are:

- Marginalized Denoising AutoEncoder (MDAE) [34],
- Stacked AutoEncoder (SAE) [35],
- Stacked Denoising AutoEncoder (SDAE) [35],
- Contractive AutoEncoder (CAE) [10], and
- Autoencoder Scoring [36].

The following are some key observations from the set of experiments.

- In all three datasets, a similar trend can be observed across the architectures and it can be found that KLD + GSAE provides the best performance. This confirms that the group sparsity constraint assists in learning improved features for the classification tasks.
- Figure 2 shows the obtained hidden layer visualizations of the autoencoder trained using KL-divergence and the proposed GSAE algorithm. It can be visually observed that GSAE algorithm learns better descriptive features that improve the classification performance.
- On the MNIST dataset, Table 3 shows the comparative performance of the proposed algorithm along with existing variants of autoencoder, as reported in the literature. It can be observed that the proposed GSAE provides comparable performance with the existing autoencoder variates. However, it is worthwhile noting that the autoencoder architectures across these variates are different. For every autoencoder variate, the architecture providing the best performance is reported.
- In the proposed formulation (i.e. Equation 3), λ controls the dominance of $\ell_{2,1}$ regularization dur-

Table 3: Summarizing the results of the proposed GSAE with best reported results of different variants of autoencoder based algorithms and the proposed algorithm, GSAE, on the MNIST dataset [19].

Algorithm	**Error Rate %**
MDAE [34]	1.29
SAE [35]	1.40
SDAE [35]	1.28
CAE [10]	1.14
Autoencoder Scoring [36]	1.27
Proposed	**1.10**

ing learning. In the literature, Vincent et. al. [35] have shown that the regularization constant could strongly influence the learnt features. Therefore, we have performed experiments by varying λ in the range of 0 to 1 with a varying step sizes, i.e. $\lambda = \{(0.00001 : 0.00001 : 0.0001), (0.0001 : 0.0001 : 0.001), (0.001 : 0.001 : 0.01), (0.01 : 0.01 : 0.1), (0.1 : 0.1 : 1)\}$. As shown in Figure 3, on the MNIST dataset, the lowest error rate is obtained with $\lambda = 0.08$. Similarly, for other two databases, the best performing results are obtained with $\lambda = 0.09$ and $\lambda = 0.08$ respectively. With group sparsity, we observe that smaller values of λ yields better results compared to higher values and the optimal performance is obtained for the values in the range of 0.05 - 0.1.

- The aim of any supervised classifier is to learn a function that maps a learnt feature representation to a set of appropriate classes. In machine learning paradigm, Wolpert [37] formulated the "No free lunch" theorem stating that, under noise-free environment there is no prior for the distinction in supervised learning algorithms based on training-set error. According to the understanding of this theorem, it is very challenging to hypothesize that a particular supervised classifier is going to perform better than other classifiers, without performing experimental evaluation. Hence, in comparison with 2ν-SVM, we evaluate the performance of other popular classifiers, the softmax classifier, multilayer neural network (with 2 hidden layers), and classic SVM classifier. On the MNIST dataset, the proposed algorithm with 2ν-SVM classifier provides the lowest error rate of 1.10 which is at least 0.75% better than other three classifiers.

3.5. Comparison with State-of-the-art Algorithms

It is to be noted that state-of-the-art performance for the datasets used in this research are: (i) MNIST is 0.21%, obtained using ConvNet architecture and dropconnect regularization [13], (ii) CIFAR-10 is 91.78%, obtained using a strictly ConvNet architecture [38], and (iii) SVHN is 98.3%, obtained by improving pooling layers in ConvNet architecture [39]. We would like to mention that the main motivation is to show that adding group sparse regularizer can improve the performance of stacked autoencoder based feature representation. The results presented in this section showcase that $\ell_{2,1}$ norm, solved via majorization-minimization approach, helps in improving the classifier performance.

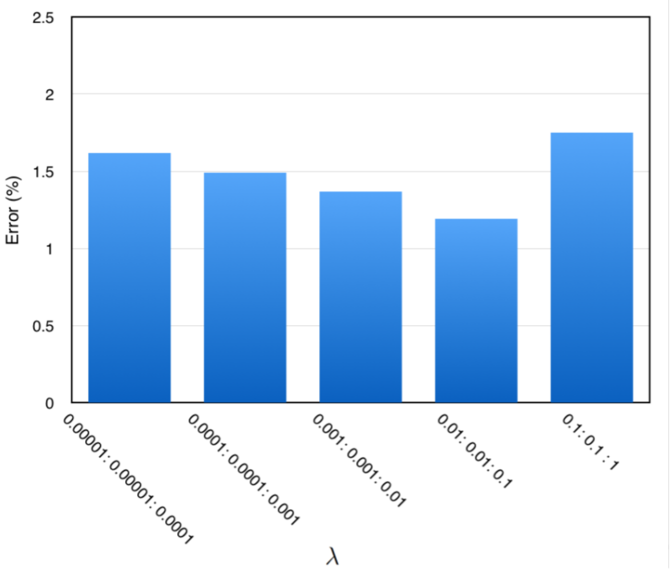

Figure 3: Effect of λ on the MNIST database. For each of the five intervals, lowest error is reported.

4. Latent Fingerprint Minutia Extraction with GSAE

Latent fingerprint recognition is one of the important forensic and law enforcement applications that can benefit from the advances in machine learning. FBIs *Next Generation Identification* program requires "lights-out mode in latent fingerprint identification where no (or very limited) human intervention is allowed. In designing such a system, there are four major steps involved: (1) latent fingerprint segmentation from complex/noisy background, (2) enhancement of region of interest, (3) feature extraction including level-1 (e.g. singular points) and level-2 (e.g. minutia) features, and (4) matching against a gallery database and return top-k probable matches (typically k = 50).

In this section, we show a case study of automatic latent fingerprint feature extraction using the proposed deep learning approach. The main idea is to use group sparse constraint in autoencoders so as to better distinguish between the minutia and non-minutia patches from latent fingerprints. Though minutia extraction from inked and live-scan fingerprints are well-addressed problems [40], extracting minutia from latent fingerprint images is still an open research problem [41]. Due to the challenging nature

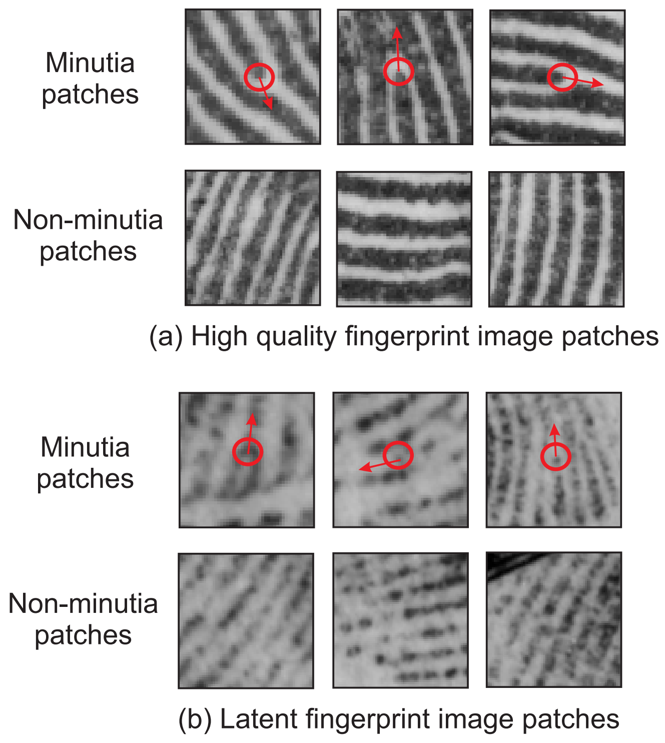
Minutia patches
Non-minutia patches
(a) High quality fingerprint image patches
Minutia patches
Non-minutia patches
(b) Latent fingerprint image patches

Figure 4: (a) High quality fingerprint patches illustrating the difference in the ridge structure between minutia and non-minutia patches, (b) Local patches from latent fingerprints illustrating the lack of well defined structures and noisy ridge patterns.

of the problem, not many algorithms exist for automated latent fingerprint minutia extraction. In some of the earlier research, existing tenprint matchers are utilized to extract minutia information from latent fingerprints [42], [43], [44]. However, with the poor performance of these algorithms, researchers have realized the need for latent specific minutia extractor which can handle poor quality information in a more robust way. Paulino et al. [43], [45] proposed a Minutiae Cylinder Code (MCC) [46] based descriptor for manually annotated minutia features. Recently, Sankaran et al. [47] proposed one of the first automated algorithms for latent fingerprint minutiae extraction using deep learning. They used an unsupervised feature learning algorithm using Stacked Denoising Sparse AutoEncoder (SDAE) [35] to learn latent fingerprint local patch description. They formulated minutia extraction as a binary classification problem, with every local patch classified as a minutia or a non-minutia containing patch.

It can be observed from Figure 4(a) that the local region around a minutia has a different ridge structure than a non-minutia patch. However, as shown in Figure 4(b), latent fingerprint minutia patches lack a definite structure, making it challenging to learn meaningful information. Due to the non-uniform and uncertain variations in latent fingerprints, it has been challenging for researchers to define a model for extracting minutiae. Human engineered features such as gradient information and frequency based information, provide limited performance due to the presence of background noise. Therefore, in this research, we design an automated minutiae extraction algorithm using the proposed GSAE for latent fingerprint. Figure 5 illustrates the three main stages of the formulation.

1. *Pre-training*: In the first stage, lots of high quality fingerprint image patches are used to pre-train a group sparse autoencoder by preserving group sparsity, as follows,

$$J(W) = \arg\min_{W,U} [||X - U\phi(WX)||_2^2 + \lambda(||WX_{hqm}||_{2,1} + ||WX_{hqnm}||_{2,1})] \quad (19)$$

   where, X_{hqm} and X_{hqnm} represent the high quality fingerprint minutia and non-minutia patches. The regularization term ensures that the group sparsity is preserved within the minutia and non-minutia patches.

2. *Supervised Fine-tuning*: In the second stage, labeled latent fingerprint image patches are used to fine-tune the GSAE. Further, a binary classifier (2ν-SVM) is trained using the extracted patches to differentiate between minutia patches and non-minutia patches.

3. *Testing*: In the third stage, the learnt feature descriptor and the trained classifier are tested. An unknown latent fingerprint is divided into overlapping patches, feature descriptor for each patch is extracted using the proposed fine-tuned GSAE algorithm and then classified using the trained 2ν-SVM classifier.

5. Experimental Results of GSAE for Fingerprint Minutia Extraction

We next describe the fingerprint datasets used for evaluation and the experimental results obtained.

5.1. Fingerprint Datasets

Similar to autoencoder, GSAE also requires a large database for learning a robust feature representation. Since the collection of latent fingerprints is a time consuming and challenging task, there are only a few latent fingerprint datasets available in the public domain. Therefore, we first prepare the heterogeneous fingerprint database by combining four publicly available live-scan fingerprint databases and use it as the pre-training data set. The four databases are: NIST SD-14 v2 [48], CASIA-FingerprintV5 [49], MCYT [50], and FingerPass [51]. The description and properties of these datasets are summarized in Table 4. To make the feature learning supervised, minutiae are extracted from all the fingerprints using an open source minutia extractor *mindtct* of the NIST Biometric Imaging Software (NBIS) [52]. An image patch of size 64×64 ($w = 64$) is extracted with minutia at the center, thereby creating $10,088,218$ number of minutia patches extracted from all the images. To compute the results on the datasets,

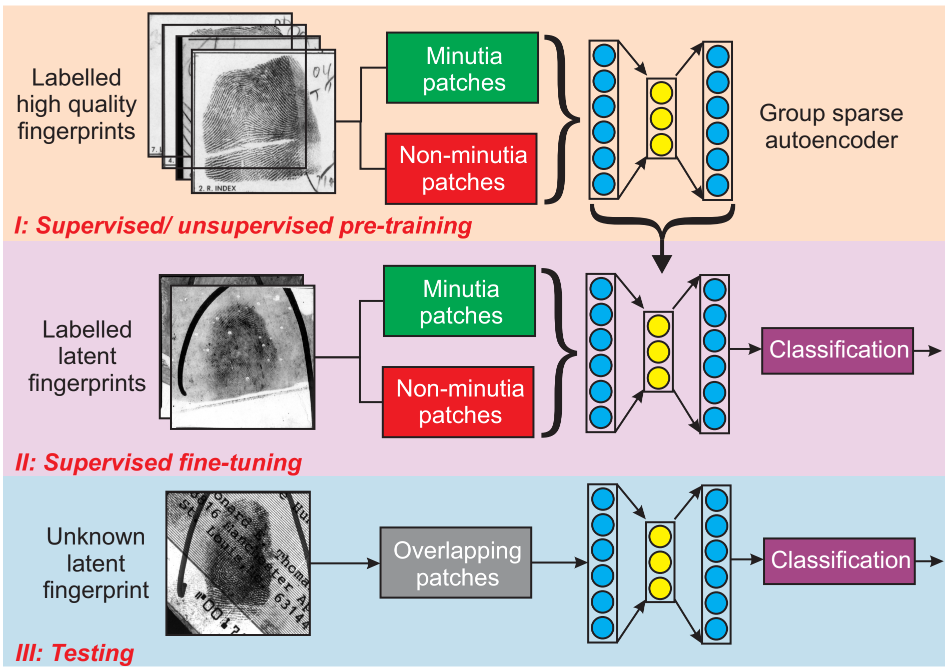
Labelled high quality fingerprints
Minutia patches
Non-minutia patches
Group sparse autoencoder
I: Supervised/ unsupervised pre-training
Labelled latent fingerprints
Minutia patches
Non-minutia patches
Classification
II: Supervised fine-tuning
Unknown latent fingerprint
Overlapping patches
Classification
III: Testing

Figure 5: Block diagram to explain the various stages in the proposed algorithm. I: Pre-training stage where the group sparse deep autoencoder is learnt from labelled high quality fingerprints, II: Fine-tuning stage where the feature learner and classifier are trained with labelled latent fingerprints, and III: Testing stage in which local patches from unknown latent fingerprint are classified as minutia and non-minutia patches.

a stacked group sparse autoencoder is designed with the network layer sizes as $\{4096, 3000, 2000, 1000, 500\}$. With a mini-batch size of $10,000$ image patches, we pre-trained each layer for $40,000$ epochs and then performed $10,000$ epochs for fine-tuning. From every fingerprint, the same number of non-minutia patches and minutia patches are extracted to ensure same number of samples from both the classes. The proposed algorithm is trained with raw image intensities of these image patches (vector size 1×4096) as input. For evaluation, the following two publicly available latent fingerprint databases are selected. A summary of the latent fingerprint datasets is shown in Table 5.

- **NIST SD-27 latent dataset** [22]: The database has 258 latent fingerprints, pre-classified as good, bad, and ugly, based on their biometric quality along with minutiae points, manually annotated by forensic experts. Since the minutia patches of high quality latent fingerprints (as a part of heterogeneous database) are different from field quality latent fingerprints, 50% randomly chosen images from the NIST SD-27 database are used to fine-tune the GSAE model learned using the heterogeneous fingerprints. The remaining 50% images (129 fingerprints) are used for testing the classification performance. For fine-tuning the proposed GSAE and classification model, the same number of minutia and non-minutia patches of size 64×64 are extracted from each training image. Three times random cross-validation is performed to remove any training bias.
- **MOLF dataset** [23]: It consists of $4,400$ latent fingerprints from 100 different subjects (all 10 fingers). All the latent fingerprints are lifted using black powder from a tile background. The manually annotated minutiae are also available along with this database. Since the pre-defined protocol of MOLF does not provide any training subset, the best trained model obtained from the NIST SD-27 database is used for performance evaluation and the entire MOLF dataset. The test set comprises $422,000$ minutia and non-minutia patches.

5.2. Latent Fingerprint Minutia Extraction Performance

The primary objective of this algorithm in fingerprint recognition is correctly extracting minutiae from latent fingerprint images. Therefore, the performance metric used in all these experiments is Correct Classification Accuracy (CCA), which denotes the ratio of correctly classified patches with the total number of patches. The overall accuracy is further split into class-specific classification accuracy: Minutia Detection Accuracy (MDA) and Non-Minutia Detection Accuracy (NMDA). In terms of MDA and NMDA, although both the accuracies should be high, it is important to detect all the minutia patches accurately along with minimizing the occurrence of spurious minutia patches.

$$\text{MDA} = \frac{\text{No. of correctly classified minutia patches}}{\text{Total no. of minutia patches}} \times 100 \tag{20}$$

Table 4: Summarizing the composition and characteristics of the heterogeneous fingerprint database. This heterogeneous database is used as the pre-training dataset for the proposed deep learning approach.

Database	**Capture Type**	**No. of Images**	**No. of Minutiae**
NIST SD-14 v2 [48]	Card print	54,000	8,188,221
CASIA-FingerprintV5 [49]	Optical	20,000	515,641
MCYT [50]	Optical, capacitive	24,000	571,713
FingerPass [51]	Optical, capacitive	34,560	812,643
Total		**132,560**	**10,088,218**

Table 5: Summarizing the characteristics of the latent fingerprint databases used in this research, including the number of train patches and test patches used in each of the three cross validation experiments.

Database	**No. of Images**	**No. of Train Patches**	**No. of Test Patches**
NIST SD-27 [22]	258	9757	65,274
	Fold 1	5,503	65,274
	Fold 2	5,439	65,274
	Fold 3	5,441	65,274
MOLF [23]	4,400	-	422,400

$$\text{NMDA} = \frac{\text{No. of correctly classified non-minutia patches}}{\text{Total no. of non-minutia patches}} \times 100 \tag{21}$$

The performance of the proposed approach is evaluated on two different datasets, NIST SD-27 and MOLF, under four different experimental scenarios:

- using VeriFinger, a popular commercial tool for fingerprints,
- using the proposed architecture with only KLD,
- using the proposed architecture with only GSAE, and
- using the proposed architecture with KLD + GSAE.

We also compared the results with current state-of-the-art algorithm proposed by Sankaran et al. [47]. The results on NIST SD-27 and MOLF are summarized in Table 6 respectively.

As shown in Table 6, on the NIST SD-27 database, the correct patch classification accuracy of the proposed algorithm is as high as 95% when using KLD + GSAE, compared to VeriFinger providing around 90% accuracy. The standard deviation of cross-validation experiments is in the range of ±0.2, denoting very small training bias. However, the MDA of VeriFinger is around 20% showing that it rejects a lot of genuine minutia patches. The architecture of Sankaran et al. [47] yield the MDA of 65%. In comparison to that, the proposed KLD + GSAE yields an improvement of more than 30%.

While detecting non-minutia patches, we have observed that the algorithm of Sankaran et al. [47] yields the lowest accuracy of 41.21% followed by VeriFinger which yields 96.20%. This results shows that VeriFinger can efficiently detect the background patches. The proposed GSAE algorithm yields 100% NMDA on the same experimental protocol. Such a high accuracy can be attributed to 2ν-SVM classification, which supports in making the false positive error almost zero. As shown in Figure 6 and Figure 7, on the NIST SD-27 database, the minutia detection accuracy is very high and very small number of spurious minutia patches are extracted.

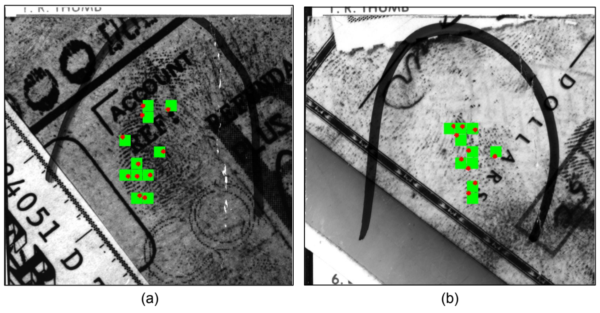

Figure 6: Sample example latent fingerprints from NIST SD-27 database showing correct results of the proposed algorithm. Red dots denote manually annotated minutiae and green patches represent the minutia patches predicted using the GSAE algorithm.

The second database used for performance evaluation is the MOLF database. This is a very large database; however, there is no defined training database. Therefore, the results on the MOLF database are obtained by training the model on the NIST SD-27 dataset and testing the best learned model with the MOLF database. Since Sankaran et al. yields lower accuracies on the NIST SD-27 database, on the MOLF database, we have only compared with VeriFinger, KLD and GSAE. As shown in Table 6,

Table 6: Classification results (%) obtained on the NIST SD-27 and MOLF latent fingerprint datasets. **CCA**: Correct Classification Accuracy, **MDA**: Minutia Detection Accuracy, **NMDA**: Non-Minutia Detection Accuracy.

Database	Algorithm	Classifier	CCA	MDA	NMDA
NIST SD-27 [22]	VeriFinger	VeriFinger	90.33	20.41	96.80
	Sankaran et. al. [47]	Softmax	46.80	65.18	41.21
	KLD	2ν-SVM	91.90	91.90	100
	GSAE	2ν-SVM	94.48	94.48	100
	KLD + GSAE	2ν-SVM	95.37	95.37	100
NIST MOLF [23]	VeriFinger	VeriFinger	78.52	21.33	92.92
	KLD	2ν-SVM	59.25	84.17	52.97
	GSAE	2ν-SVM	90.14	90.44	90.07
	KLD + GSAE	2ν-SVM	90.74	90.63	90.37

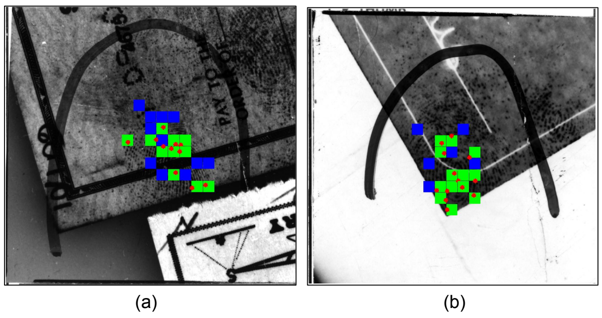

Figure 7: Sample latent fingerprints from NIST SD-27 database showing some incorrect predictions obtained using the proposed algorithm. Red dots represent the annotated minutiae, green patches represent the correct minutiae patches predicted using the GSAE algorithm and blue patches show the incorrect ones (false positive). It can be observed that no genuine minutiae is rejected by the proposed algorithm.

the proposed architecture yields classification accuracies of over 90% with the standard deviation in the range of ± 0.15 compared to VeriFinger which provides around 78.5% accuracy. Comparing with different regularizations reveals that KLD + GSAE provides the best results on both the datasets and the performance of GSAE is better than the traditional KLD regularization. On 3349 images out of the total 4400 images present in the MOLF database, VeriFinger failed to extract any minutia. This shows that VeriFinger yields poor results in extracting genuine minutiae whereas using only KLD regularization extracts lots of spurious minutiae - the non-minutiae detection accuracy is only 52.97%.

The results on both the databases show that the performance of the proposed minutiae extraction algorithm is better than the existing algorithms. It is our assertion that the performance of the proposed algorithm on the MOLF database is not as good as on the NIST SD-27 because (i) the number of testing data points on the MOLF database is very large compared to the NIST SD-27 dataset and (ii) there are significant variations in the characteristics of the two databases. Using the model trained with NIST SD-27 also contributes to lower accuracies on the MOLF database.

We further evaluate the efficacy of the proposed GSAE based minutiae extraction approach using a popular latent fingerprint identification system[2]. The identification system is modular in nature which provides the flexibility of keeping the entire pipeline constant while changing only one component. This facilitates evaluating the performance of the proposed approach with minutiae extracted from other approaches as well. The system has over 2 million pre-enrolled identities in the database that can be used as the large gallery in the experiments. Since the NIST SD27 database may have been used to train the system, we have performed matching experiments only with the MOLF database. Gallery images from the MOLF database are enrolled[3] and probe images are used for testing. From a probe fingerprint image, first the minutiae points are extracted using the proposed algorithm by first finding the minutiae patches and then taking its center as the minutia point. The minutiae template/feature is given to latent fingerprint matching system which compares against the large gallery (which also includes gallery from the MOLF) and output the top matches. We also compare the performance with the inbuilt approach of latent fingerprint system in which minutiae are extracted by the system itself and matched against the gallery. We obtain rank-50 accuracies for both these experiments. For the proposed approach, rank-50 accuracy is 69.83% whereas the inbuilt feature extraction approach in latent fingerprint system yields 69.21%. Though this seems slight improvement by the proposed algorithm, this is still noteworthy because of the large scale matching (using more than 2 million gallery identities) with over 4,000 probe latent fingerprint images. This experiment demonstrates that the proposed algorithm is highly promising for automatic latent fingerprint feature extraction.

[2]The license agreement does not allow us to name the commercial system in any comparisons.

[3]After the experiments, these enrollments are deleted from the system.

6. Conclusion

This paper presents a novel supervised regularization method for autoenocders using $\ell_{2,1}$-norm which utilizes class labels to learn supervised features for the specific task at hand. The optimization function is solved using a majorization-minimization approach and classification is performed using 2ν-SVM. The effectiveness of the proposed GSAE based representation learning approach is evaluated on three standard databases: MNIST, CIFAR-10, and SVHN. Further, using GSAE, an automatic latent fingerprint minutia extraction algorithm is formulated as a binary classification algorithm. The minutiae extraction algorithm is evaluated on two publicly available latent fingerprint databases, NIST SD27 and MOLF. The results show that the proposed algorithm improves the performance of automated latent fingerprint feature extraction. It is our assertion that the effectiveness of GSAE can be further utilized to improve the classification performance with challenging and noisy databases.

References

[1] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436–444.

[2] C. P. MarcAurelio Ranzato, S. Chopra, Y. LeCun, Efficient learning of sparse representations with an energy-based model, in: Advances in neural information processing systems, 2007.

[3] Y. Bengio, Learning deep architectures for AI, Foundations and Trends® in Machine Learning 2 (1) (2009) 1–127.

[4] G. Cybenko, Approximation by superpositions of a sigmoidal function, Mathematics of control, signals and systems 2 (4) (1989) 303–314.

[5] K. Hornik, M. Stinchcombe, H. White, Multilayer feedforward networks are universal approximators, Neural networks 2 (5) (1989) 359–366.

[6] I. Sutskever, G. E. Hinton, Deep, narrow sigmoid belief networks are universal approximators, Neural Computation 20 (11) (2008) 2629–2636.

[7] Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle, et al., Greedy layer-wise training of deep networks, Advances in neural information processing systems 19 (2007) 153.

[8] S. J. Nowlan, G. E. Hinton, Simplifying neural networks by soft weight-sharing, Neural Computation 4 (4) (1992) 473–493.

[9] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: A simple way to prevent neural networks from overfitting, Journal of Machine Leanring Research 15 (1) (2014) 1929–1958.

[10] S. Rifai, P. Vincent, X. Muller, X. Glorot, Y. Bengio, Contractive auto-encoders: Explicit invariance during feature extraction, in: International Conference on Machine Learning, 2011, pp. 833–840.

[11] G. Hinton, S. Osindero, Y.-W. Teh, A fast learning algorithm for deep belief nets, Neural Computation 18 (7) (2006) 1527–1554.

[12] I. J. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville, Y. Bengio, Maxout networks, in: International Conference on Machine Learning, 2013.

[13] L. Wan, M. Zeiler, S. Zhang, Y. L. Cun, R. Fergus, Regularization of neural networks using dropconnect, in: Internation Conference on Machine Learning, 2013, pp. 1058–1066.

[14] X. Frazão, L. A. Alexandre, Dropall: Generalization of two convolutional neural network regularization methods, in: Image Analysis and Recognition, Springer, 2014, pp. 282–289.

[15] C. Hong, J. Yu, D. Tao, M. Wang, Image-based three-dimensional human pose recovery by multiview locality-sensitive sparse retrieval, IEEE Transactions on Industrial Electronics 62 (6) (2015) 3742–3751.

[16] C. Hong, J. Yu, J. Wan, D. Tao, M. Wang, Multimodal deep autoencoder for human pose recovery, IEEE Transactions on Image Processing 24 (12) (2015) 5659–5670.

[17] Z. Zheng, J. Yang, Supervised locality pursuit embedding for pattern classification, Image and Vision Computing 24 (8) (2006) 819–826.

[18] J. Mairal, Stochastic majorization-minimization algorithms for large-scale optimization, in: Advances in Neural Information Processing Systems, 2013, pp. 2283–2291.

[19] Y. LeCun, C. Cortes, MNIST handwritten digit database, AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist.

[20] A. Krizhevsky, G. Hinton, Learning multiple layers of features from tiny images (2009).

[21] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, A. Y. Ng, Reading digits in natural images with unsupervised feature learning, in: NIPS Workshop on Deep Learning and Unsupervised Feature Learning, no. 2, 2011, p. 5.

[22] Fingerprint minutiae from latent and matching tenprint images, NIST Special Database 27. Available: http://www.nist.gov/srd/nistsd27.htm.

[23] A. Sankaran, M. Vatsa, R. Singh, Multisensor optical and latent fingerprint database, IEEE Access 3 (2015) 653–665.

[24] R. Sang, P. Jin, S. Wan, Discriminative feature learning for action recognition using a stacked denoising autoencoder, in: Intelligent Data analysis and its Applications, Volume I, Springer, 2014, pp. 521–531.

[25] S. Gao, Y. Zhang, K. Jia, J. Lu, Y. Zhang, Single sample face recognition via learning deep supervised autoencoders, IEEE Transactions on Information Forensics and Security 10 (10) (2015) 2108–2118.

[26] A. Majumdar, R. K. Ward, Synthesis and analysis prior algorithms for joint-sparse recovery, in: IEEE International Conference on Acoustics, Speech and Signal Processing, 2012, pp. 3421–3424.

[27] F. Nie, H. Huang, X. Cai, C. H. Ding, Efficient and robust feature selection via joint l2, 1-norms minimization, in: Advances in neural information processing systems, 2010, pp. 1813–1821.

[28] Y. Yang, H. T. Shen, Z. Ma, Z. Huang, X. Zhou, l2, 1-norm regularized discriminative feature selection for unsupervised learning, in: International Joint Conference on Artificial Intelligence, Vol. 22, 2011, p. 1589.

[29] T. Blumensath, Compressed sensing with nonlinear observations and related nonlinear optimization problems, IEEE Transactions on Information Theory 59 (6) (2013) 3466–3474.

[30] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, Distributed optimization and statistical learning via the alternating direction method of multipliers, Foundations and Trends® in Machine Learning 3 (1) (2011) 1–122.

[31] M. Davenport, R. G. Baraniuk, C. D. Scott, et al., Controlling false alarms with support vector machines, in: International Conference on Acoustics, Speech and Signal Processing, 2006, Vol. 5, 2006.

[32] A. Torralba, R. Fergus, W. T. Freeman, 80 million tiny images: A large data set for nonparametric object and scene recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence 30 (11) (2008) 1958–1970.

[33] G. E. Hinton, R. R. Salakhutdinov, Reducing the dimensionality of data with neural networks, Science 313 (5786) (2006) 504–507.

[34] M. Chen, K. Q. Weinberger, F. Sha, Y. Bengio, Marginalized denoising auto-encoders for nonlinear representations, in: International Conference on Machine Learning, 2014, pp. 1476–1484.

[35] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.-A. Manzagol, Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion, The Journal of Machine Learning Research 11 (2010) 3371–3408.

[36] K. Swersky, M. Ranzato, D. Buchman, B. Marlin, N. Freitas, On autoencoders and score matching for energy based models, in: International Conference on Machine Learning, 2011, pp. 1201–1208.

[37] D. H. Wolpert, The lack of a priori distinctions between learning algorithms, Neural computation 8 (7) (1996) 1341–1390.

[38] J. T. Springenberg, A. Dosovitskiy, T. Brox, M. Riedmiller, Striving for simplicity: The all convolutional net, arXiv preprint arXiv:1412.6806.

[39] C.-Y. Lee, P. W. Gallagher, Z. Tu, Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree, in: International Conference on Artificial Intelligence and Statistics, 2016.

[40] D. Maltoni, D. Maio, A. K. Jain, S. Prabhakar, Handbook of fingerprint recognition, 2nd edition, Springer-Verlag, 2009.

[41] A. Sankaran, M. Vatsa, R. Singh, Latent fingerprint matching: A survey, IEEE Access 2 (2014) 982–1004.

[42] J. Feng, J. Zhou, A. Jain, Orientation field estimation for latent fingerprint enhancement, IEEE Transactions on Pattern Analysis and Machine Intelligence 35 (4) (2013) 925–940.

[43] A. A. Paulino, J. Feng, A. K. Jain, Latent fingerprint matching using descriptor-based hough transform, IEEE Transactions on Information Forensics and Security 8 (1) (2013) 31–45.

[44] A. Sankaran, T. I. Dhamecha, M. Vatsa, R. Singh, On matching latent to latent fingerprints, in: Proceedings of International Joint Conference on Biometrics, 2011, pp. 1–6.

[45] A. A. Paulino, J. Feng, A. K. Jain, Latent fingerprint matching using descriptor-based hough transform, IEEE Transactions on Information Forensics and Security 8 (1) (2013) 31–45.

[46] R. Cappelli, M. Ferrara, D. Maltoni, Minutia cylinder-code: A new representation and matching technique for fingerprint recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence 32 (12) (2010) 2128–2141.

[47] A. Sankaran, P. Pandey, M. Vatsa, R. Singh, On latent fingerprint minutiae extraction using stacked denoising sparse autoencoders, in: IEEE International Joint Conference on Biometrics, 2014, pp. 1–7.

[48] NIST 8-bit gray scale images of Fingerprint Image Groups (FIGS), NIST Special Database 14.

[49] CASIA-Fingerprint V5, Chinese Academy of Sciences Institute of Automation (CASIA) Fingerprint Image Database Version 5.0.

[50] J. Ortega-Garcia, J. Fierrez-Aguilar, D. Simon, J. Gonzalez, M. Faundez-Zanuy, V. Espinosa, A. Satue, I. Hernaez, J.-J. Igarza, C. Vivaracho, D. Escudero, Q.-I. Moro, MCYT baseline corpus: a bimodal biometric database, IEE Proceedings on Vision, Image and Signal Processing 150 (6) (2003) 395–401.

[51] X. Jia, X. Yang, Y. Zang, N. Zhang, J. Tian, A cross-device matching fingerprint database from multi-type sensors, in: International Conference on Pattern Recognition, 2012, pp. 3001–3004.

[52] NBIS (NIST Biometric Image Software), Developed by National Institute of Standards and Technology (NIST), http://www.nist.gov/itl/iad/ig/nbis.cfm.