Learning Autoencoders with Low-Rank Weights

Kavya Gupta
TCS
gupta.kavya@tcs.com

Angshul Majumdar
IIIT-Delhi
angshul@iiitd.ac.in

***Abstract*— In this work we propose to regularize the encoding and decoding weights of an autoencoder using low-rank penalty in the form of nuclear norm. Such a formulation models redundancy in the network. We show that our proposed method yields better classification accuracy (on an average) and denoising results than other stochastic and deterministic regularization techniques used in deep autoencoders. Our method is also considerably faster compared to these techniques. The experiments have been carried out on benchmark deep learning datasets.**

Keywords*— *autoencoder, deep learning, classification, denoising, nuclear norm

I. INTRODUCTION

An autoencoder is a unsupervised neural network (does not require class labels); it consists 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:

$$h = \phi(Wx) \tag{1}$$

Here the rows of W are the link weights from all the input nodes to the corresponding latent neurons. Usually a non-linear activation function (ϕ) like sigmoid / tanh is used at the output of the hidden nodes. The decoder portion (W') reverse maps the latent variables to the data space.

$$x = W'\phi(Wx) \tag{2}$$

Since the data space is assumed to be the space of real numbers, there is no activation 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:

$$\arg\min_{W,W'} \left\| X - W'\phi(WX) \right\|_F^2 \tag{3}$$

Here $X = [x_1 | ... | x_N]$ consists all the training sampled stacked as columns. The problem (3) is clearly non-convex, but it is smooth and hence can be solved by gradient descent techniques.

Stacked autoencoders have multiple hidden layers – one inside the other. The corresponding cost function is expressed as follows:

$$\arg\min_{W_1...W_L,W'_1...W'_L} \left\| X - g\left(f(X)\right) \right\|_F^2 \tag{4}$$

where $g = W_1'\phi\left(W_2'...\phi(W_L'\left(f(X)\right)\right)$ and $f = \phi\left(W_L\phi\left(W_{L-1}...\phi(W_1X)\right)\right)$.

Solving the complete problem (4) is computationally challenging. Moreover learning so many parameters (network weights) simultaneously leads to over-fitting. To address both these issues, the weights are usually learned in a greedy fashion – one layer at a time [1]. Once the stacked architecture is learnt, the decoder portions are removed and the targets / class labels are attached at the representation layer of the deepest autoencoder to form a deep neural network. This is fine-tuned by back-propagation to complete training the deep neural network.

There are several versions of regularized autoencoders. Stacked denoising autoencoder [2] use a stochastic regularization technique where noisy samples are input and clean samples are output, so that the autoencoder weights are learnt in a noise robust fashion.

There are several deterministically regularized autoencoders as well. Deterministic regularization takes the form of an additional penalty term on top of the autoencoder cost function,

$$\arg\min_{W,W'} \left\| X - W'\phi(WX) \right\|_F^2 + R(W,X) \tag{5}$$

The regularization can be a sparsity promoting term [3, 4] (l_0-norm penalty), [5] (KL divergence) or a weight decay term (Frobenius norm of the Jacobian) as used in the contractive autoencoder [6, 7].

All such regularization techniques are on the representation $\phi(WX)$ and not directly on the encoder / decoder weights. For example [3-5], yields sparse features. As mentioned in [8], sparsity can be in features and also in the neural network weights. Introducing sparsity in neural network weights dates back to LeCun's seminal work in 90's on optimal brain damage [9]; here sparsity was introduced in an intuitive fashion by looking at the saliency of the network. A recent work [10], introduces sparsity in the autoencoder network in a more optimal fashion – by adding l_1-norm penalties on the encoder / decoder weights.

Sparsity is one approach to handle redundancy in networks. It gets rid of irrelevant connections and keeps only the most significant ones. In this work, we want to explicitly

incorporate the redundancy. We propose to learn the encoder and decoder weights such that they are linearly dependent; in such a case the learnt weight matrix will be low-rank. This can be easily modeled by a nuclear norm penalty [11] on the cost function.

There are two other recent stochastic regularization techniques in neural network literature – DropOut [12] and DropConnect [13]. Both the techniques have been successful in improving the performance of neural networks; however there is no mathematical reason for such actions; the intuitive understanding is that such dropping of outputs / connections prevents co-adaptation and hence makes the learning more robust, thereby preventing over-fitting.

Our proposed work and the prior study [10] are deterministic regularization (on network weights) techniques; sparse and contractive autoencoders are deterministic as well (regularization on features). DropOut and DropConnect are stochastic in nature. This work compares the proposed approach with all the aforesaid studies.

II. Proposed Formulation

We are not experts in cognitive science, however the intuitive idea behind modelling redundancy in the autoencoder follows from past concepts in neural networks where they were believed to mimic the human brain. The human brain is apparently a redundant network [14, 15]; that seems to be plausible explanation why we are able to recover from trauma or seizures – when some parts of the brain get damaged. Another example of redundancy stems from the fact that after a certain age, neurons in our brain die; even then we are able to carry forth all our activities.

In our proposed formulation, we introduce linear dependency on the encoder and decoder weights; this in turn means that the connections to the hidden layer neurons (in the autoencoder) are linearly dependent; this in turn leads to redundancy in the neurons – something we expect from a brain-like network. The mathematical formulation leads to,

$$\arg\min_{W,W'} \left\| X - W'\phi(WX) \right\|_F^2 + \lambda\left(\left\| W' \right\|_{NN} + \left\| W \right\|_{NN} \right) \qquad (6)$$

The nuclear norm is the nearest convex surrogate of matrix rank and is defined as the sum of singular values of the matrix [16, 17]. The nuclear norm, although convex is not smooth. Hence the standard backpropagation / gradient descent based techniques cannot be used to solve (6). In this work we solve it using more modern optimization techniques. Our first step is to segregate the problem via alternating minimization.

$$\begin{aligned} W'_k &\leftarrow \arg\min_{W'} \left\| X - W'\phi(W_{k-1}X) \right\|_F^2 + \lambda \left\| W' \right\|_{NN} \\ &\equiv \arg\min_{W'} \left\| X^T - \left(\phi(WX)\right)^T W'^T \right\|_F^2 + \lambda \left\| W'^T \right\|_{NN} \end{aligned} \qquad (7)$$

$$W_k \leftarrow \arg\min_{W} \left\| X - W'_k \phi(WX) \right\|_F^2 + \lambda \left\| W \right\|_{NN} \qquad (8)$$

The first step in every iteration (7) is relatively straightforward to solve. It is a typical nuclear norm minimization problem, that can be efficiently solved by singular value shrinkage [11]. Note that since W' is low-rank, its transpose is low-rank as well.

Solving (8) is more involved. For this, we invoke the majorization minimization (MM) technique. In short MM algorithm proceeds as follows.

Let J(x) be the function to be minimized
1. Set *k=0* initialize x_0.
2. Choose $G_k(x)$ such that
   a. $G_k(x) \geq J(x)$ for all x.
   b. $G_k(x_k) \geq J(x_k)$.
3. Set x_{k+1} as the minimizer for $G_k(x)$.
4. Set k=k+1, go to step 2.

For our problem, $J(x) = \left\| X - W'_k \phi(WX) \right\|_F^2$. Its majorizer in iteration 'i' is given by,

$$\begin{aligned} G_i(x) &= \left\| X - W'_k \phi(WX) \right\|_F^2 \\ &+ \left(\phi(WX) - \phi(W_i X)\right)^T \left(aI - X^T X \right) \left(\phi(WX) - \phi(W_i X)\right) \end{aligned} \qquad (9)$$

where 'a' is the maximum eigenvalue of X^TX, thus guaranteeing the condition for MM.

After some mathematical manipulations (which we skip for the sake of brevity), (9) can be expressed as

$$G_i(x) = \left\| B - \phi(WX) \right\|_F^2 + K \qquad (10)$$

where $B = \phi(W_i X) + \frac{1}{a} W'^T_k \left(X - W'_k \phi(W_i X) \right)$ and K consists of terms devoid of the variable.

Following this derivation, (8) can be alternately expressed by MM as follows,

$$\begin{aligned} W_k &\leftarrow \arg\min_{W} \left\| B - \phi(WX) \right\|_F^2 + \lambda \left\| W \right\|_{NN} \\ &\equiv \arg\min_{W} \left\| \phi^{-1}(B) - WX \right\|_F^2 + \lambda \left\| W \right\|_{NN} \end{aligned} \qquad (11)$$

As the activation function is element-wise, its inversion is trivial. As we did for updating W', we express (11) in terms of transposes,

$$W_k \leftarrow \arg\min_{W} \left\| \phi^{-1}(B^T) - X^T W^T \right\|_F^2 + \lambda \left\| W^T \right\|_{NN} \qquad (12)$$

As before, this turns out to be a solved problem; this is just a nuclear norm minimization which can be solved efficiently using singular value shrinkage [11].

There are two exit criteria. The first one is specified maximum number of iterations. The second one is the convergence of the objective function; iterations stop upon reaching a local minimum.

III. EXPERIMENTAL RESULTS

A. Classification

We carried our experiments on several benchmarks datasets. The first one is the MNIST dataset which 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. No preprocessing has been done on this dataset.

We also tested on variations of MNIST, which are more challenging primarily because they have fewer training samples (10,000 + 2,000 validation) and larger number of test samples (50,000). This one was created specifically to benchmark deep learning algorithms [17].

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)

The USPS dataset refers to numeric data obtained from the scanning of handwritten digits from envelopes by the U.S. Postal Service. The original scanned digits are binary and of different sizes and orientations; the images here have been deslanted and size normalized, resulting in 16 x 16 grayscale images. There are 7291 training observations and 2007 test observations.

Devnagari descended from the Brahmi script sometime around the 11th century AD. Its original form was developed to write Sanksrit but was later adapted to write many other languages such as Hindi, Marathi and Nepali. This database [18, 19] of isolated handwritten Devnagari numerals consists of 22556 samples from 1049 persons..

Bangla is also derived from the Brahmi script and it started to diverge from the Devnagari script during the 11th Century AD. This script is also used to write a few other languages such as Assamese, Manipuri etc. The present database [18, 19] of handwritten isolated Bangla numerals consists of 23392 samples written by 1106 persons.

The techniques compared against can be categorized into two: 1. Deterministic Regularization – contractive autoencoder (CAE) [6], K-sparse autoencoder (KAE) [3], sparsely connected autoencoder (SparseConnect) [10]; 2. Stochastic Regularization – DropOut stacked autoencoder, DropConnect stacked autoencoder and stacked denoising autoencoder (SDAE) [2]. As a benchmark Deep Belief Network (DBN) [20] has been used. For all the tools, the optimal configuration have been mentioned in the corresponding papers.

For our proposed low-rank autoencoder, we use a three layer architecture. In the first layer the number of neurons are twice the size of input dimensionality. In the second layer, the number of neurons is the same as the input dimensionality and in the third layer the number of neurons are half the size of the input dimensionality. The values of lambda used are 0.1, 0.1 and 0.01 in the three successive layers. These sizes and values were determined using the validation set.

In this work our goal is to test the representation capability of the different learning tools. Therefore the training is fully unsupervised (no class label is used). The generated features from the deepest level are used to train two non-parametric – nearest neighbor (NN) (Table 1) and sparse representation based classification (SRC) [21] (Table 2); and one parametric – support vector machine (SVM) classifier with rbf kernel (Table 3) (tuned with grid search).

TABLE I. COMPARISON ON KNN

Dataset	SDAE	DropOut	DropConnect	KAE	CAE	DBN	SparseConnect	Proposed
MNIST	97.33	**97.36**	**97.36**	96.90	92.83	97.05	95.91	97.12
basic	95.25	94.94	95.02	91.64	90.92	**95.37**	92.49	**95.37**
basic-rot	**84.83**	84.53	84.31	80.24	78.56	84.71	81.01	**84.83**
bg-rand	86.42	85.99	85.87	85.89	85.61	86.36	84.87	**86.77**
bg-img	77.16	76.46	77.31	76.84	78.51	77.16	**79.84**	77.39
bg-img-rot	52.21	52.06	**52.40**	50.27	47.10	50.47	48.91	**52.40**
USPS	94.91	95.32	94.62	93.85	91.96	94.36	94.62	**95.42**
Bangla	95.22	94.97	95.77	93.27	92.49	94.08	95.77	**96.20**
Devanagari	88.79	87.18	**90.10**	87.26	87.16	88.23	**90.10**	89.68

TABLE II. COMPARISON ON SRC

Dataset	SDAE	DropOut	DropConnect	KAE	CAE	DBN	SparseConnect	Proposed
MNIST	**98.33**	98.33	98.29	97.91	87.19	88.43	97.16	98.20
basic	96.91	96.93	**96.97**	95.07	95.03	87.49	95.43	**96.97**
basic-rot	90.04	89.97	90.16	88.85	88.63	79.47	87.76	**90.23**
bg-rand	91.03	91.00	91.09	83.59	82.25	79.67	86.17	**91.61**
bg-img	84.14	85.19	85.77	84.12	85.68	75.09	**85.84**	84.67
bg-img-rot	62.46	60.09	62.73	58.06	54.01	49.68	57.75	**63.27**
USPS	95.49	95.57	96.06	94.19	94.07	85.73	96.01	**96.11**
Bangla	92.11	96.51	93.70	93.57	93.20	95.11	96.78	**97.24**
Devanagari	90.06	93.00	91.05	91.19	91.69	59.50	**93.16**	93.05

TABLE III. COMPARISON ON SVM

Dataset	SDAE	DropOut	DropConnect	KAE	CAE	DBN	SparseConnect	Proposed
MNIST	98.50	98.39	98.51	98.46	97.74	**98.53**	97.97	98.51

basic	96.96	96.97	96.96	97.02	96.61	97.07	96.23	**97.13**
basic-rot	89.43	89.16	89.31	88.75	72.54	89.05	89.21	**89.55**
bg-rand	91.28	91.17	**91.52**	90.07	85.20	89.59	90.70	**91.52**
bg-img	84.86	85.03	85.12	80.17	78.76	85.46	84.52	**85.37**
bg-img-rot	60.53	60.70	60.98	60.01	60.97	58.25	**61.76**	**61.76**
USPS	96.61	95.41	95.07	93.59	92.87	94.91	95.62	**97.58**
Bangla	95.09	95.43	93.33	92.06	92.69	94.51	96.81	93.50
Devanagari	94.11	89.83	88.98	90.52	90.99	93.64	95.18	**96.07**

TABLE IV. TRAINING TIME IN SECONDS

Dataset	SDAE	DropOut	DropConnect	KAE	CAE	DBN	SparseConnect	Proposed
MNIST	251	209	201	111	97	78	4	14
basic	53	42	42	31	24	21	1	5

The results show that our proposed method yields the best results on an average. It is significantly better than other deterministic regularization techniques like CAE and KAE; we are also slightly better than SparseConnect. The stochastic regularization techniques do better than CAE and KAE and are comparable with SparseConnect. They only fall slightly short of our proposed low rank regularization method.

We have also shown the training times from different methods. Experiments have been carried out on an i7 Windows Desktop with 16 GB of RAM running Matlab 2012. Results show that our proposed method is very fast. This is because it converges in only 20 iterations. It is only slower than [11] (this is because they do not have to compute SVD). Other techniques, with backpropagation based training are more than an order of magnitude slower.

B. Denoising

We show the denoising performance on the CIFAR-10 dataset. The 50,000 training images were used to learn the autoencoder and the remaining 10,000 test images were used for testing denoising performance. The color images were converted to greyscale. The input images were corrupted by Gaussian noise and the output is the clean image while training. For testing, the input is corrupted and passed through the autoencoder. The output image is compared with the clean groundtruth to measure performance.

We compare the results with the Denoising Autoencoder (DAE) [2], the sparse DAE [4] and SparseConnect [10]. All the autoencoders are single layer and have 392 neurons in the hidden layer.

As a benchmark we also denoise using the gold standard BM3D [22]. Denoising performance is measured in terms of Peak Signal to Noise Ratio (PSNR) and Structural Similarity Index (SSIM). The results are shown in Table 5. Obviously simple autoencoder based techniques cannot beat engineered tools like BM3D; within the autoencoder based techniques, both in terms of PSNR and SSIM our method yields the best results. It is interesting to observe that while the performance of BM3D decreases significantly with increasing noise level, the autoencoder based methods degrade more slowly.

TABLE V. COMPARISON OF DENOISING PERFORMANCE

Original PSNR and SSIM	DAE	Sparse DAE	SparseConnect	BM3D	Proposed
Variance=0.01 PSNR=19.97 SSIM=0.57	PSNR=21.96 SSIM=0.63	PSNR=22.94 SSIM=0.68	PSNR=23.90 SSIM=0.72	PSNR=33.90 SSIM=0.94	PSNR=**26.41** SSIM=**0.78**
Variance=0.05 PSNR= 14.02 SSIM=0.33	PSNR=21.67 SSIM=0.62	PSNR=22.64 SSIM=0.67	PSNR=23.53 SSIM=0.70	PSNR=31.53 SSIM=0.86	PSNR=**26.26** SSIM=**0.73**
Variance=0.1 PSNR= 10.49 SSIM=0.20	PSNR=21.31 SSIM=0.60	PSNR=22.25 SSIM=0.65	PSNR=23.01 SSIM=0.67	PSNR=27.53 SSIM=0.80	PSNR=**25.86** SSIM=**0.70**

IV. CONCLUSION AND FUTURE WORK

In this work we propose a new low-rank regularization penalty for autoencoders. This stems from the requirement of modelling redundancy in autoencoders. We evaluate the performance of our proposed low-rank autoencoder on two tasks. The first task is classification. Our method performs better than all other deterministic and stochastic regularization techniques.

The second task, image denoising. Our proposed method is compared with all known denoising autoencoders; as gold standard we also denoise using the BM3D algorithm. Our proposed low-rank autoencoder always yields better denoising compared to other autoencoders.

In future we would like to adopt the proposed regularization for supervised autoencoding tasks, for example label consistent autoencoder [23, 24] and class-sparse autoencoder [25, 26]. Also adopting such regularization techniques for robust autoencoders [27, 28] may help improve solution of inverse problems.

ACKNOWLEDGEMENT

The first author was at IIITD while pursuing this work.

REFERENCES

[1] Y. Bengio, P. Lamblin, D. Popovici and H. Larochelle, "Greedy layer-wise training of deep networks", NIPS, 2006.

[2] 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.

[3] A. Makhzani and B. Frey, "k-Sparse Autoencoders", ICLR, 2013.

[4] K. Cho, "Simple sparsification improves sparse denoising autoencoders in denoising highly noisy images", ACM ICML, 2013.

[5] S. Z. Su, Z. H. Liu, S. P. Xu, S. Z. Li and R. Ji, "Sparse auto-encoder based feature learning for human body detection in depth image", Signal Processing, Vol. 112, pp. 43-52, 2015.

[6] S. Rifai, P. Vincent, X. Muller, X. Glorot, Y. Bengio, "Contractive Auto-Encoders: Explicit Invariance During Feature Extraction", ACM ICML, 2011

[7] Y. Liu, X. Feng and Z. Zhou, "Multimodal video classification with stacked contractive autoencoders", Signal Processing, Vol. 120, pp. 761-766, 2016.

[8] M. Thom and G. Palm, "Sparse Activity and Sparse Connectivity in Supervised Learning", Journal of Machine Learning Research, Vol. 14, pp. 1091-1143, 2013.

[9] Y. LeCun, "Optimal Brain Damage", NIPS, 1990.

[10] K. Gupta and A. Majumdar, "Sparsely Connected Autoencoder", IEEE IJCNN, 2016.

[11] A. Majumdar and R. K. Ward, "Some Empirical Advances in Matrix Completion", Signal Processing, Vol. 91 , pp. 1334-1338, 2011.

[12] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting", Journal of Machine Learning Research, Vol. 15 (1), pp. 1929-1958, 2014.

[13] L. Wan, M. Zeiler, S. Zhang, Y. LeCun and R. Fergus, Regularization of neural networks using dropconnect. ACM ICML, 2013.

[14] R. B. Glassman, "An hypothesis about redundancy and reliability in the brains of higher species: Analogies with genes, internal organs, and engineering systems", Neuroscience & Biobehavioral Reviews, Vol. 11 (3), pp. 275-285, 1987.

[15] S. Finger, "Recovery of function: Redundancy and vicariation theories." Handbook of clinical neurology, Vol. 95, pp. 833-841, 2009.

[16] X. Lin and G. Wei, "Accelerated reweighted nuclear norm minimization algorithm for low rank matrix recovery", Signal Processing, Vol. 114, pp. 24-33, 2015.

[17] H. Larochelle, D. Erhan, A. Courville, J. Bergstra and Y. Bengio, "An Empirical Evaluation of Deep Architectures on Problems with Many Factors of Variation", ACM ICML, 2011.

[18] U. Bhattacharya and B. B. Chaudhuri, "Handwritten numeral databases of Indian scripts and multistage recognition of mixed numerals", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 31 (3), pp. 444-457, 2009.

[19] U. Bhattacharya and B. B. Chaudhuri, "Databases for research on recognition of handwritten characters of Indian scripts", IEEE ICDAR, 2005.

[20] G. E. Hinton, S. Osindero, S. and Y. Teh, "A fast learning algorithm for deep belief nets", Neural Computation, Vol. 18 (7), pp. 1527-54, 2006.

[21] J. Wright, A. Y. Yang, A. Ganesh, S. 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.

[22] A. Danielyan, V. Katkovnik, and K. Egiazarian, "BM3D Frames and Variational Image Deblurring", IEEE Transactions on Image Processing, Vol. 21 (4), pp. 1715-1728, 2012.

[23] A. Majumdar, A. Gogna and R. K. Ward, "Semi-supervised Stacked Label Consistent Autoencoder for Reconstruction and Analysis of Biomedical Signals", IEEE Transactions on Biomedical Engineering.

[24] A. Gogna and A. Majumdar, "Semi Supervised Autoencoder", ICONIP, pp. 82-89, 2016.

[25] A. Majumdar, M. Vatsa and R. Singh, "Face Recognition via Class Sparsity based Supervised Encoding", IEEE Transactions on Pattern Analysis and Machine Intelligence, 2016

[26] A. Sankaran, G. Sharma, R. Singh, M. Vatsa and A. Majumdar, "Class Sparsity Signature based Restricted Boltzmann Machines", Pattern Recognition, 2016.

[27] J. Mehta, K. Gupta, A. Gogna and A. Majumdar, "Stacked Robust Autoencoder for Classification", ICONIP, pp. 600-607, 2016.

[28] J. Mehta and A. Majumdar, "RODEO: Robust DE-aliasing autoencOder for Real-time Medical Image Reconstruction", Pattern Recognition, Vol. 63, pp. 499-510, 2017.