Department of Information Technology

National Institute of Technology Karnataka, Surathkal

Soft Computing (IT355)

Mini Project Report

Title- ***Denoising high resolution multispectral images using deep learning approach***

Submitted by

Supriya Sahoo 15IT145
Akriti Kumari 15IT107
Ankita Bhalavi 15IT108
Vandana Baidya 15IT250

Submitted to

Dr. Nagamma Patil
Assistant Professor
Department of Information Technology
NITK Surathkal

November 10, 2017

CERTIFICATE

This is to certify that the project entitled **"Denoising high resolution multispectral images using deep learning approach"** has been presented by Supriya Sahoo, Akriti Kumari, Ankita Bhalavi, Vandana Baidya, students of third year (V sem), B.Tech (IT), Department of Information Technology, National Institute of Technology Karnataka, Surathkal, on November 10, 2017, during the odd semester of the academic year 2017- 2018, in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Information Technology at NITK, Surathkal.

Place: NITK, Surathkal

Date: November 10, 2017 (Signature of the Examiner)

ACKNOWLEDGEMENT

This project would not have been possible without the kind support and help of many individuals and organizations. We would like to extend our sincere thanks to all of them. We are highly indebted to the Department of Information Technology, NITK for their guidance and constant supervision as well as for providing necessary information and resources for the project and also for their support in completing the project.

We would like to express our gratitude towards our teacher Dr.Nagamma Patil for her kind co-operation and encouragement which helped us in completion of this project. Our thanks and appreciations also go to our group members in developing the project and people who have willingly helped us out with their abilities.

ABSTRACT

In this project, we have addressed the image denoising problem specifically for high resolution multispectral images. We explored the possibility to device a method that approximates the denoising function using the ideology of learning from data. The overall objective was to build a model that replicates the denoised results of Non-local means algorithm using far less computational resources for each of the four spectral bands (blue, green, red and near infrared). We have used deep neural networks and in particular stacked auto encoders to learn the function that maps a noisy image to its denoised version. We showed that after training the model on a large set noisy and denoised images, excellent results that are comparable to the results of Non-local means algorithm can be obtained in much less computational time. The scope of the model can further be extended to denoise any other natural image by training it on the appropriate data set.

INDEX

Contents **Page No.**

INTRODUCTION

Introduction of noise in a digital image is the result of image acquisition which corrupts the intensity level of certain pixels resulting in some kind of visual distortion. The observed image signals have an invariant tendency to get corrupted by some degree of noise. Image denoising problem falls under the category of image restoration problems in which the goal is to restore the original image content from the corrupted version of it. The corruption can be in the form of motion blur, noise, camera misfocus. The ideal result for an image denoising procedure is to completely remove the noise content after being fed a noisy image. But in practice, no matter how well engineered the denoising algorithm is, reducing the noise content level to absolute zero is not achievable.

The most important application of the image denoising procedure is in industries and organizations which continuously have to deal with satellite imagery. This is because unlike other natural images captured by digital camera, the image signals do travel a long distance before being received by the satellites. This results in considerable corruption of the data. Denoising high resolution images becomes computationally expensive and time consuming process.

Some of the standard denoising techniques like Non-local means algorithm takes around 10-15 minutes to denoise a single high resolution image of dimension around 4000 x 80000. BM3D (Block matching and 3-d filtering) is another such algorithm producing state of the art results but can take upto 2-3 hours to denoise a similar high dimensional image. This time consuming factor serves as a motivation to come up with a model that can provide comparable results, if not better, in much less time. Also, if we have access to a large dataset of noisy and denoised images, we can think of constructing a model that intuitively learns the patterns in the denoising procedure, evolving itself continuously while adapting to the learned features.

The project hence revolves around the idea to automatically learn the denoising function purely based on training on a set of noisy images and denoised versions. The model can then be used to perform denoising operation on unseen data. Here, we have narrowed down our focus to denoising multispectral images corrupted by Poisson-Gaussian noise.

LITERATURE SURVEY

Since the paper is focused on a learning based approach, an important decision has to be made regarding the choice of the model. Literature shows that this choice heavily depends on the complexity of the function to be learned, which in turn is determined by the dimensionality of the problem. Since this paper deals with high resolution multi-spectral images, the model chosen has to have sufficiently large architecture to accommodate all the essential image features. Moreover, the task of image processing requires powerful techniques capable of learning and extracting deeper insights from the data. That's why we've chosen deep neural networks over other machine learning algorithms because traditional machine learning techniques are much shallower and have lesser capabilities to model high-level abstractions in data.

The project hence revolves around the idea to automatically learn the denoising function purely based on training on a set of noisy images and denoised versions. The model can then be used to perform denoising operation on unseen data. In this paper, we have narrowed down our focus to denoising multispectral images corrupted by Poisson- Gaussian noise.

METHODOLOGY AND DESIGN

We have used deep neural networks and in particular stacked autoencoders to learn the function that maps a noisy image to its denoised version. We show that after training the model on a large set noisy and denoised images, excellent results that are comparable to the results of Non-local means algorithm can be obtained in much less computational time.

Stacked Autoencoders :

An autoencoder is a three-layer neural network which is trained to reproduce the input it receives with the help of a hidden layer. The functioning of an autoencoder involves two steps. When the data flows from the input layer to the hidden layer, the process is called encoding i.e. the process of mapping of the input to its representative features. And when this encoded data flows from the hidden layer to the output layer, decoding process takes place with an aim to reproduce the input i.e. the process in which the learned features aim to reconstruct the input.

Dataset used:

The scope of the model can further be extended to denoise any other natural image by training it on the appropriate data set.

We've worked on the project using 17 multispectral images each of the dimension around 4000 x 80,000 pixels. Each multispectral image consists of 4 bands i.e. blue, green, red and near infrarred band. The noise present in these images is a combination of Gaussian and Poisson noise. For the training purpose we've used 12 of the 17 noisy images and their corresponding denoised images. Feeding the whole image of dimension 4000 x 80000 as the input is computationally highly expensive. Therefore, from each of the 12 training images, we've chosen 100000 random patches of size 26 x26. So our training set consists of 1200000 patches of 26 x 26 and their corresponding outputs. While training the model, it is always desirable that the training set consist of patches that are sufficiently informative so that the model is able to learn high level features quickly.

To make sure that this problem doesn't arise, we divided the whole image into 8 parts horizontally, and selected 12500 patches randomly from each of these 8 parts to ensure that random patches don't represent the same area in the image. Similar data set was prepared for the remaining 3 bands.

Model description :

We've used stacked autoencoders with following conguration:

- First layer (input layer): 26 x 26 = 676 neurons
- Second layer (first hidden layer): 2620 neurons (for blue, green and red band), 2480 neurons (for infrared band)
- Third layer (second hidden layer): 2620 neurons
- Last layer (output layer): 676 neurons

The models were selected after repeated experimentations with its configuration and the above structure of the model provided the best results. The activation function for the first and second hidden layer is sigmoid function

$$\sigma(x) = 1/1 + e^{-x} \quad (1)$$

while the output layer's activation function is simply : $y = x$ (2)

Initialization of model's parameters and data preprocessing :

1) Data normalization: We've scaled our input data between 0 and 1. Two standard techniques for scaling the data are standardization and normalization. We've used normalization method which results in the following transformation:

$$Inew = (I - Imin)/(Imax - Imin) \quad (3)$$

where Imax and Imin are the highest and lowest intensities in the respective patches [11,16]. One major necessity of this data scaling is in the pre-training step of the model where the cost function requires its input to be in the range (0,1].

2) Weight initialization: We use the normalized initialization in which the weights are sampled from a uniform distribution. This is done to break the symmetry between hidden units of the same layer. This uniform distribution varies from layer to layer depending on the number of neurons in the input and output layer and is given by the formula:

$$\left[\frac{-\sqrt{6}}{\sqrt{ni+n0}}, \frac{-\sqrt{6}}{\sqrt{ni+n0}}\right] \quad (4)$$

where ni and no are the number of neurons in the input and output side of the weights. The weights of the output layer are initialized to zero because different output units receive different gradient signals and therefore the issue of breaking the symmetry does not concern the weights.

3) Bias initialization: All the biases of neurons in the hidden layers as well as the output layer have been initialized to zero.

Training the model :

Training of stacked autoencoders is done in two phases:

1) Unsupervised pre-training: The cost function chosen as a measure of error is cross-entropy cost function. No external regularization term has been used because the unsuperwised pre-training itself produces a regularization effect .

$$C_1(x,y) = -1/N\sum[y ln(h\theta(x)) + (1-y)\ln(1-h\theta(x))] \quad (5)$$

where hθ(x) represents the hypothesis, parameterized by θ (weights and biases). The learning rate for the pre-training purpose has been kept constant and equal to 0.1.

2) Supervised finetuning: In the fine-tuning step we compare the output of the model with the desired output and the performance is measured in terms of another cost function.
We've used quadratic cost function to evaluate the results of the finetuning step. In this step, the cost function has been regularized with L2 regularization.

$$C_2(x,y) = 1/2N\sum||y - h\theta(x)||^2 + \lambda/2N\sum(w\hat{}2) \quad (6)$$

where W represents the weights of the model and λ is the regularization parameter. The parameters of the model are then tuned with the help of stochastic gradient descent algorith. But unlike the pre-training step, we've not taken a constant learning rate. Instead we've chosen an adaptive learning rate α whose functioning is as follows:

- if validation error decreases:
  $\alpha = \alpha + (0.05 \times \alpha)$
- if validation error increases: undo the updates,
  $\alpha = \alpha - (0.5 \times \alpha)$

This algorithm is called bold driver method. Also, one of the most common problems while working with stochastic gradient descent is the ability to escape local minima. We have tackled the problem by using a momentum term while updating the parameters. Not only does it help in overcoming local minima, it also smoothens out the steps by preventing the oscillations in areas with high variations. The combination of learning rate and momentum helps in converging at the global minimum of the cost function in far less epochs.

The following equation shows using the momentum term while updating the parameters during gradient descent:

$$\_w(t) = \beta\_w(t-1) + \alpha\nabla wC(w) \quad (7)$$

$$w(t) = w(t) - \_w(t) \quad (8)$$

Here α is the learning rate, β determines what fraction of the previous update is being added as the momentum.

RESULT

1) Effect of configuration of the model on the training of the model:
The number of hidden layers required to produce the best results may vary from model to model. Deeper layers become increasingly difficult to train and addition of more layers increases the complexity of the model. Increased complexity means there is a high probability that stochastic gradient descent algorithm may get stuck at a local optimum of the complex error landscape thus providing poor results.

Validation error for different number of the hidden layers

Hidden layers configuration	Best validation error achieved
[2620]	0.003866
[2620,2620]	0.003412
[2620,2620,2620]	0.004348
[2620,2620,2620,2620]	0.004823

Table No. 1

2) Variation with the size of the hidden layers:
Larger architecture generally provides better results. Increasing the size beyond a certain limit also degrades the results. Again, this may be due to the increased dimensionality and a more complex error landscape.

Validation error for different sizes of the hidden layers

Hidden layer configuration	Best validation error achieved
[520,520]	0.003654
[1420,1420]	0.002678
[2620,2620]	0.004432
[4420,4420]	0.004687

Table No. 2

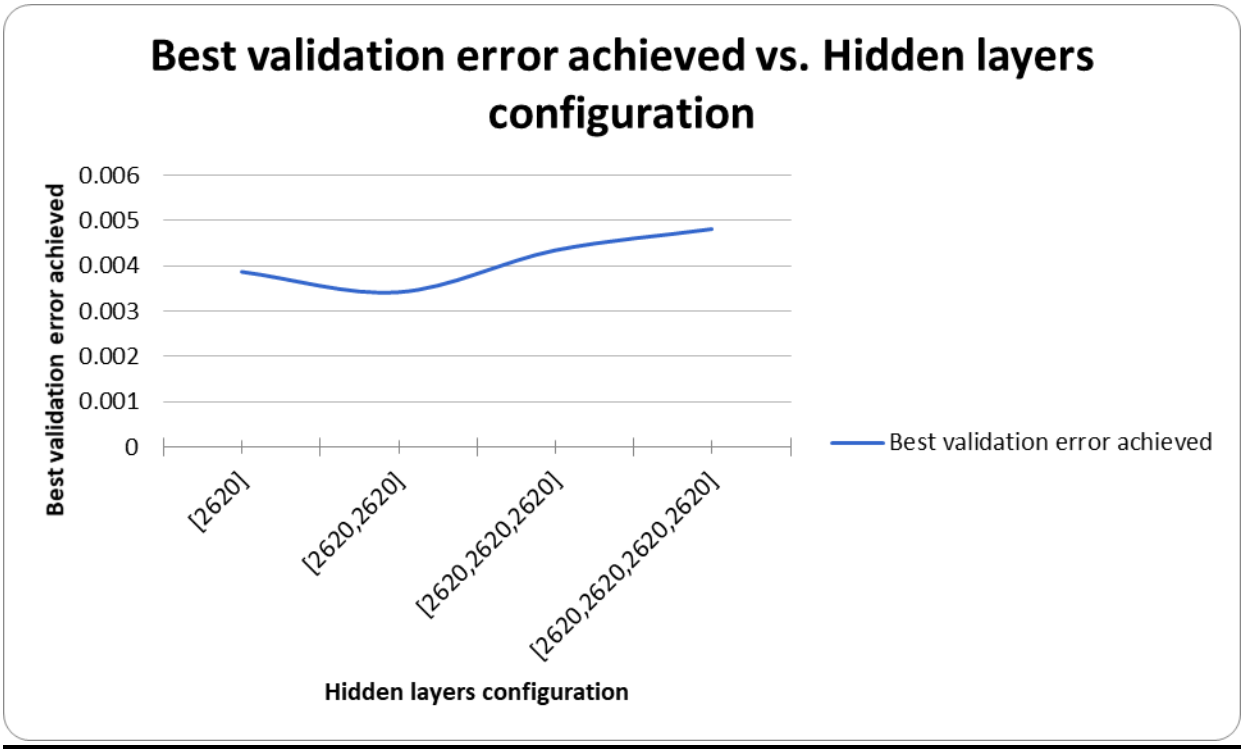
3) Variation with the block size:

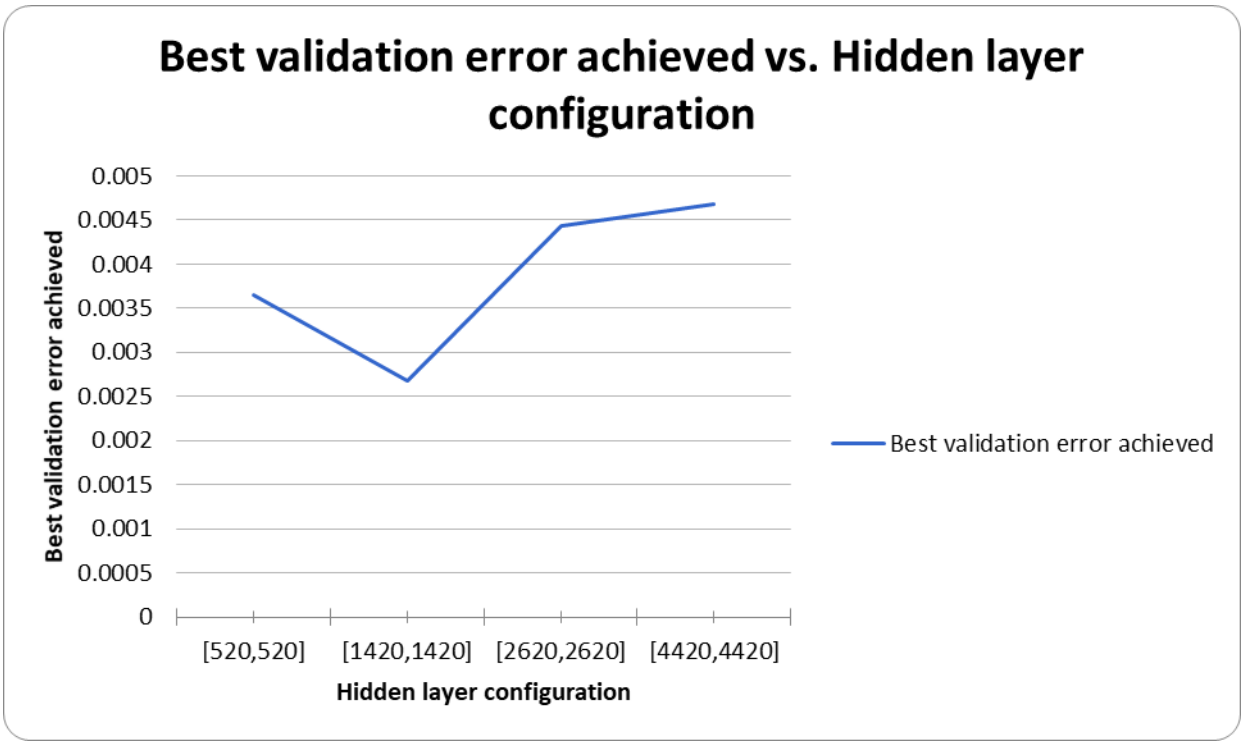
If the patch is too small, it may not be able to give sufficient information about the features in the region and hence resulting in poor performance. On the other hand if the patch size is too large then the stochastic gradient descent may not work properly due to the increased dimensionality and complexity of the model.

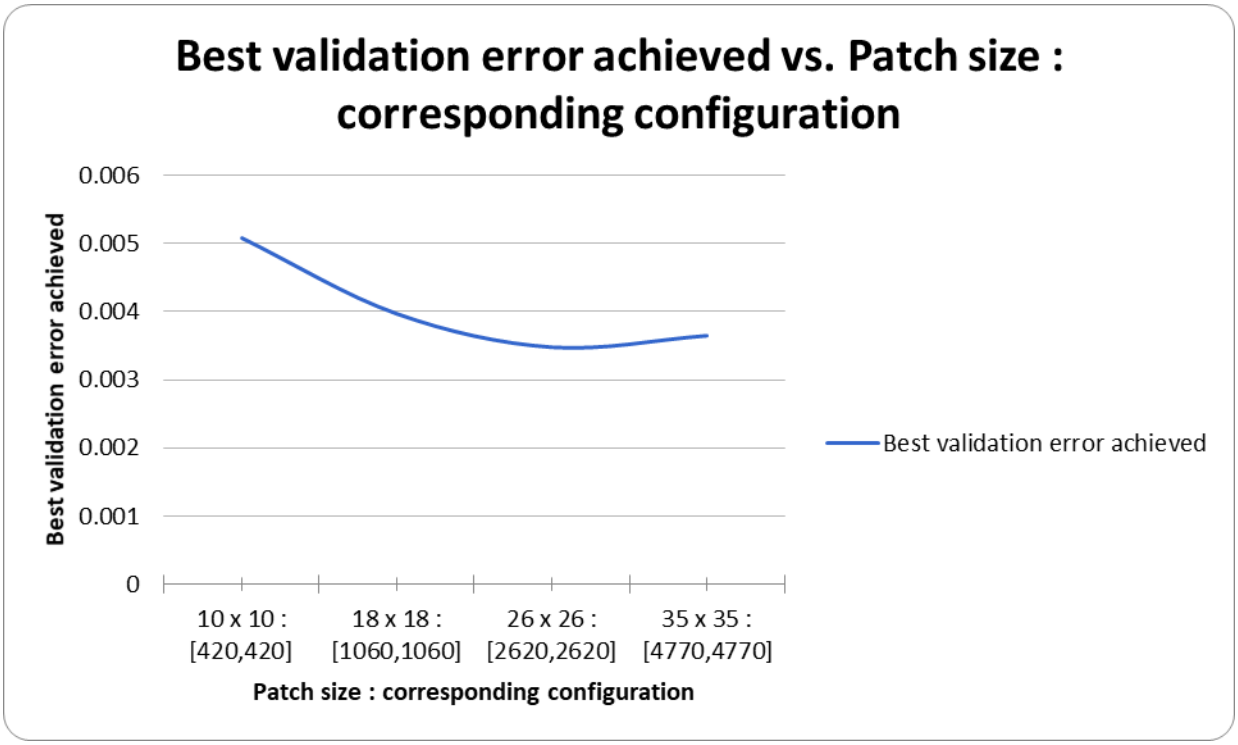
Validation error for different patch sizes

Patch size : corresponding configuration	Best validation error achieved
10 x 10 : [420,420]	0.005078
18 x 18 : [1060,1060]	0.003965
26 x 26 : [2620,2620]	0.003474
35 x 35 : [4770,4770]	0.003644

Table No. 3




Best validation error achieved vs. Hidden layer configuration
Best validation error achieved
0.005
0.0045
0.004
0.0035
0.003
0.0025
0.002
0.0015
0.001
0.0005
0
[520,520]
[1420,1420]
[2620,2620]
[4420,4420]
Hidden layer configuration
Best validation error achieved


Best validation error achieved vs. Patch size : corresponding configuration
Best validation error achieved
0.006
0.005
0.004
0.003
0.002
0.001
0
10 x 10 : [420,420]
18 x 18 : [1060,1060]
26 x 26 : [2620,2620]
35 x 35 : [4770,4770]
Patch size : corresponding configuration
Best validation error achieved

CONCLUSION

We presented a novel approach to tackle the image denoising problem for high resolution multispectral images. Our model achieves excellent performance and produces results comparable to the results of Nonlocal means algorithm and outperforms Non-local means and BM3D in time consuming factor. The limitation of this model however lies in the fact that its results can always get closer to the standard denoising results but can never outperform them. So in our future work, we would like to explore the possibility of creating an architecture that serves as a generalized model to denoise an image in any of the spectral band.

REFERENCES

- 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. The Journal of Machine Learning Research , 11:33713408, 2010.
- D. Erhan, Y. Bengio, A. Courville, P. A. Manzagol, P. Vincent. Why Does Unsupervised Pre-training Help Deep Learning?, Journal of Machine Learning Research 11 (2010) 625-660.
- Y. LeCun, L. Bottou, G. Orr, and K. Muller. Ecient backprop. In Neural Networks, Tricks of the Trade , Lecture Notes in Computer Science LNCS 1524. Springer Verlag, 1998b.
- Yoshua Bengio and Xavier Glorot. Understanding the diculty of trainingdeep feedforward neural networks. In Proceedings of AISTATS , volume 9, pages 249-256, 2010.