International Journal of Scientific Research in Computer Science, Engineering and Information Technology
ISSN : 2456-3307 (www.ijsrcseit.com)
doi : https://doi.org/10.32628/IJSRCSEIT

Image Noise Reduction with Autoencoder

Aparna Mote, Aditya Bhamre, Atharv Waghmare, Diptesh Thakare, Tanoj Handal
Department of Computer Engineering, Zeal College of Engineering and Research, Pune, Maharashtra, India

Article Info
Volume 8, Issue 3
Page Number : 492-495

Publication Issue :
May-June-2022

Article History
Accepted: 10 May 2022
Published: 30 May 2022

ABSTRACT

Digital images play an important role in our daily lives, and they can be used for a variety of applications such as fingerprint recognition, video viewing, etc. Transmission through audio channels, faults in the memory, etc. may cause noise in the image and when it is processed afterwards it will lead to inaccurate output. Therefore, there is necessity to remove the noise before in order for the image to be processed. Here comes in the need of efficient denoising technique that helps to deal with the noisy image. Image denoising is a technique to restore or repair a damaged image back to its original quality. In line with that, we sometimes need to colour the decoloured image. Image data formatting requires a special take in neural network, called Convolutional Neural Network (CNN) or Convolutional Autoencoder. For reducing the dimensions and image noise, autoencoders are widely preferred. Therefore, by its virtue, training the model to perform denoising of the images.

Keywords: Deep-Learning, Machine-Learning, Tensor Flow, Autoencoder, Convolutional Neural Network, Image Denoising, Image Compression.

I. INTRODUCTION

Deep learning is a branch of machine learning based entirely on artificial neural networks, which are used to mimic the human brain.[5] It requires a special approach in the world of neural network. The best neural network for modelling image data is the Convolutional Neural Network (CNN, or Conv-Net) also known as Convolutional Autoencoder.[1] [7]

The autoencoder network architecture may vary between a simple Feed Forward network, an LSTM network or a Convolutional Neural Network depending on the operating environment.

Autoencoder is an unsupervised artificial neural artificial network that learns how to efficiently compress and encode data and learns how to reconstruct data from reduced encoded representation to as close to original input as possible. Autoencoder, by design, reduces data size by learning how to ignore noise present in the data.

Image denoising is one of the most important problems in the field of image processing as it requires many computer vision applications. Various methods have been used in image processing over the years from spatial filtering to model-based methods.

After going through all the traditional methods, the neural network-based discriminatory methods have become popular in recent years.

However, many of these methods are still struggling to gain flexibility against different levels of noise and its types. In this paper, we propose a deep convolutional autoencoder integrated with a variant of feature pyramid network for image denoising.

II. METHODS AND MATERIAL

Neural Networks autoencoders are very common is used to select and extract features. When the nodes enter the hidden layer rises above the nodes in the input layer and then the output is equivalent to the input of the Autoencoder useless. This problem can be solved by changing some of them at random input values are zero. Half input nodes are set to 0. Other sources suggest lower calculations, such as 30%. Data and input nodes affect it. When calculating the Loss function, it is important to compare output rates with initial input, not with damaged input. Thus, the risk of learning the copyrighted work instead of removing features is deleted. [2]

III. RESULTS AND DISCUSSION

A. Background Study

Autoencoder is an unsupervised neural network learning how to compress and compile data correctly, learns how to rebuild data after download coded representation in the introduction real input as much as possible. Out Autoencoder, by design, reduces data size by learning how to ignore noise data. [5]

B. Components

1) **Encoder:** Where the model learns how to reduce input size and compress input data into encoded representation.
2) **Bottleneck:** It is a layer that contains a compressed representation of input data.
3) **Decoder:** When the model learns to rebuild data from encoded representation is approximate of the actual input as possible.
4) **Loss of Reconstruction:** This is a measuring method how efficient and decoder the decoder is output to first input.

IV. ARCHITECTURE AND DESIGN

The capturing of feature data and encrypting it into a latent space is done by the network of encoders present inside autoencoders. This encoded data (i.e., code) is used by the decoder to convert it to feature data. In the encoder, what the model learns is how to encrypt data properly so that the code generator can convert it back to its original state. Therefore, an important part of autoencoder training is to produce an optimized latent space.

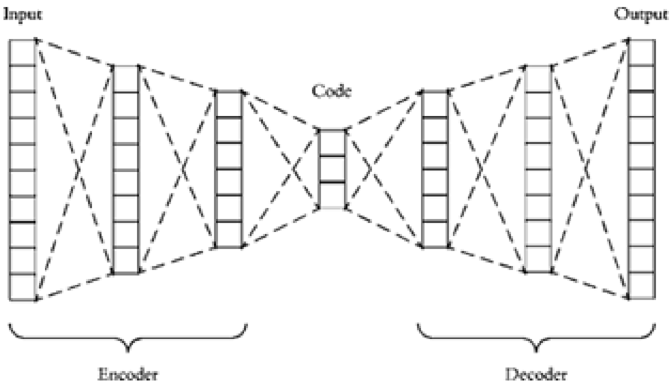

Figure 1: Autoencoder Architecture

V. IMPLEMENTATION DETAILS

A. Low-Resolution Images:

Firstly, all the images were assigned to the encoder and the data set is converted into low dimensional images data set.

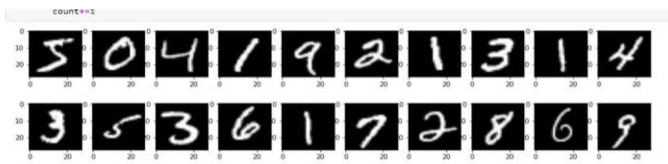

Figure 2: Original Dataset

B. Noised Images:

Then the noise was added to the low-resolution images:

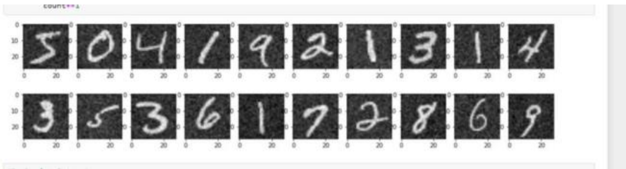

Figure 3: Noised Images

C. Epochs:

It defines the number of times that the learning algorithm will run through the dataset. Until the error or area of improvement within the model has been recognized or sufficiently optimized, it allows the learning algorithm to learn. Processing is done on MNIST image data set again and again (20 Epoch) to improve accuracy and achieving better results. Where first the image is encoded all the features are extracted from the image neglecting the noise and then reconstructed the image again.

```
Epoch 00014: val_loss did not improve from 0.06886
352/352 - 131s - loss: 0.0678 - val_loss: 0.0689 - 131s/epoch - 372ms/step
Epoch 15/20

Epoch 00015: val_loss improved from 0.06886 to 0.06837, saving model to denoising_model.h5
352/352 - 135s - loss: 0.0677 - val_loss: 0.0684 - 135s/epoch - 383ms/step
Epoch 16/20

Epoch 00016: val_loss improved from 0.06837 to 0.06793, saving model to denoising_model.h5
352/352 - 131s - loss: 0.0675 - val_loss: 0.0679 - 131s/epoch - 373ms/step
Epoch 17/20

Epoch 00017: val_loss did not improve from 0.06793
352/352 - 133s - loss: 0.0674 - val_loss: 0.0684 - 133s/epoch - 376ms/step
Epoch 18/20

Epoch 00018: val_loss did not improve from 0.06793
352/352 - 129s - loss: 0.0675 - val_loss: 0.0680 - 129s/epoch - 366ms/step
Epoch 19/20

Epoch 00019: val_loss improved from 0.06793 to 0.06767, saving model to denoising_model.h5
352/352 - 131s - loss: 0.0672 - val_loss: 0.0677 - 131s/epoch - 372ms/step
Epoch 20/20

Epoch 00020: val_loss improved from 0.06767 to 0.06764, saving model to denoising_model.h5
352/352 - 130s - loss: 0.0671 - val_loss: 0.0676 - 130s/epoch - 370ms/step
```

Figure 4: Epochs

D. Encoder Summary

Autoencoder encoder summary is calculated where the reduce dimension that is 7, 7,32 is observed

```
#Building encoder

encoder_input = Input(shape = X_train.shape[1:])

x = Conv2D(32, (3,3), activation = 'relu', padding = 'same')(encoder_input)
x = BatchNormalization()(x)
x = MaxPool2D(pool_size = (2,2), padding = 'same')(x)
x = Conv2D(32, (3,3), activation = 'relu', padding = 'same')(x)
x = BatchNormalization()(x)

encoded = MaxPool2D(pool_size = (2,2), padding = 'same')(x)

encoded

<KerasTensor: shape=(None, 7, 7, 32) dtype=float32 (created by layer 'max_pooling2d_1')>
```

Figure 5: Encoder Summary

E. Decoder Summary

Autoencoder decoder summary is calculated where the reconstructed image dimension that is 28, 28, 1 is obtained.

```
# decoder

x = Conv2D(32, (3,3), activation = 'relu', padding = 'same')(encoded)
x = BatchNormalization()(x)
x = UpSampling2D()(x)
x = Conv2D(32, (3,3), activation = 'relu', padding = 'same')(x)
x = BatchNormalization()(x)
x = UpSampling2D()(x)

decoded = Conv2D(1, (3,3), activation = 'sigmoid', padding = 'same')(x)

decoded

<KerasTensor: shape=(None, 28, 28, 1) dtype=float32 (created by layer 'conv2d_4')>
```

Figure 6: Decoder Summary

F. Final Output

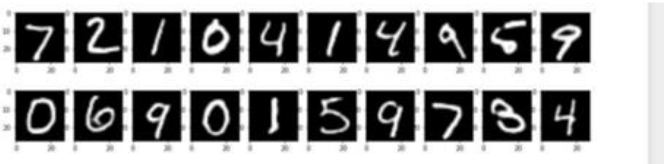

Figure 7: Final Output

VI. CONCLUSION

This network is trained employing a sizable amount of natural images, and it learns to finish or correct the pictures to seem locally just like the training image patches. The model described relies on the Convolutional Neural Network which integrates image into a compact representation, followed by a decoded image to get rid of noise from the image by neglecting it. It worked alright when tested on a on several images from data-set. The image created after the method was very accurate. But the source of input image also played a vital role in feature extraction and also within the removal of noise. Certain images aren't well recognized and it's found that there's, still some scope of improvement.

In conclusion, images with less level of the noise may be easily denoised and without a doubt this Neural Network is used not just for denoising but also as a transformer for other images. However, we are able to get fine results only if we use noisy images.

VII. REFERENCES

[1]. Michael Nielsen, "Neural Networks and Deep Learning", Springer International Publishing, 2019.

[2]. Debjani Bhowmick, Deepak K Gupta, Saumen Maiti, Uma Shankar, "Stacked autoencoders based machine learning for noise reduction and signal reconstruction in geophysical data", arXiv, 2019

[3]. Martin Abadi, "Tensor Flow: Learning Functions at scale", International Conference on Functional Programming, ACM, 2016

[4]. Donghoon Leea Sunghoon Choib Hee-JoungKim," Performance Evaluation of Image denoising developed using convolutional denoising autoencoders in chest radiography", Vol 88, pp 97-104, 2018

[5]. J. Kim, S. Song and S. Yu, "Denoising auto-encoder based image enhancement for high resolution sonar image," 2017 IEEE Underwater Technology (UT), Busan, 2017

[6]. Kopczyk D., Denoising Autoencoder: Part I – Introduction to Autoencoders, https://dkopczyk.quantee.co.uk/dae-part1 .

[7]. Jonathan Masci, Ueli Meier, Dan Ciresen, Jurgen Schmidhuber, "Stacked Convolutional AutoEncoders for Hierarchical Feature Extraction", International Conference on Artificial Neural Networks, 2011

[8]. R. Nagy, A. Dicker and K. Meyer Wegener, "NEOCR: A Configurable Dataset for Natural Image Text Recognition". In CBDAR Workshop 2011 at ICDAR 2011. pp. 5358, September 2011. 2019 8th European Workshop on Visual Information Processing (EUVIP)

[9]. Guo, S., Yan, Z., Zhang, K., Zuo, W., & Zhang, L. Toward convolutional blind denoising of real photographs. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 1712-1722.

[10]. Bharambe, Yogesh, "Assessing employability of students using data mining techniques", (ICACCI), 2017.