International Journal of Science and Research (IJSR)
ISSN: 2319-7064
SJIF (2019): 7.583

Image Noise Reduction with Autoencoder using Tensor Flow

Jai Sehgal[1], Dr Yojna Arora[2]

[1]Department of CSE, Amity University, Haryana, India
sehgaljai81[at]gmail.com

[2]Department of CSE, Amity University, Haryana, India
Yojana183[at]gmail.com

Abstract: ***The shaping of image data requires a special approach in the neural network world. The well known neural network for shaping image data is the Convolutional Neural Network (CNN) or called Convolutional Autoencoder. Autoencoders have widely applied in dimension reduction and image noise reduction. In this project, Noise Reduction on images using the fashion-mnist dataset is performed. Convolutional Autoencoders are used to remove the noise of the noisy fashion-mnist images. The model was then checked for the training loss and the validation loss.***

Keywords: Deep-Learning, Machine-Learning, Tensor Flow, Autoencoder, Convolutional Neural Network

1. Introduction

Deep learning is a branch of machine learning which is completely based on artificial neural networks, which is used to mimic of human brain [1].

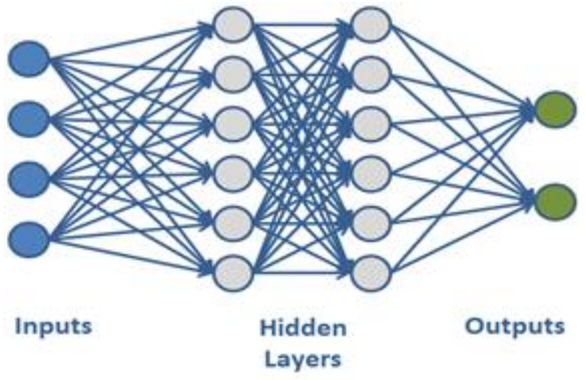

Figure 1: Basic Neural Network [1]

Modelling image data requires a special approach in the neural network world. The best known neural network for modelling image data is the **Convolutional Neural Network (CNN, or Conv-Net)** or called **Convolutional Autoencoder**.The network architecture for autoencoders can vary between a simple Feed Forward network, LSTM network or Convolutional Neural Network depending on the use case [4] [5]

2. Background of Study

Autoencoder is an unsupervised artificial neural network that learns how to efficiently compress and encode data then learns how to reconstruct the data back from the reduced encoded representation **to** a representation that is as close to the original input as possible.Autoencoder, by design, reduces data dimensions by learning how to ignore the noise in the data.[1]

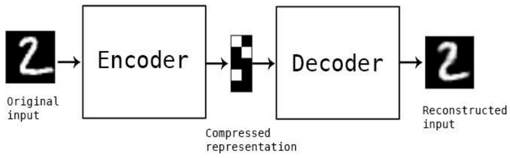

Figure 2: Flow Chart of Model

Noise reduction is the process of removing noise from a signal. This can be an image, audio or a document.

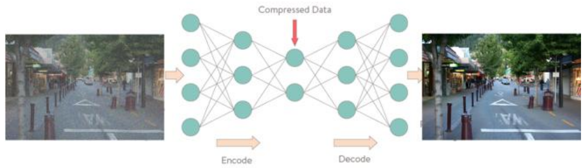

Figure 3: Working of Model

3. Methodology

Autoencoders are Neural Networks which are commonly used for feature selection and extraction. When the nodes in hidden layer increases than nodes in input layer then the output equals the inputmarking the Autoencoder useless [2]. This problem can be solved by randomly turning some of the input values to zero. Half of the input nodes are set to 0. Other sources suggest a lower count, such as 30%. It depends on the amount of data and input nodes available.
When calculating the Loss function, it is important to compare the output values with the original input, not with the corrupted input. That way, the risk of learning the identity function instead of extracting features is eliminated [3]

3.1 Components of Model

1) **Encoder:** In which the model learns how to reduce the input dimensions and compress the input data into an encoded representation.
2) **Bottleneck:** It is the layer that contains the compressed representation of the input data.

3) **Decoder:** In which the model learns how to reconstruct the data from the encoded representation to be as close to the original input as possible.
4) **Reconstruction Loss:** This is the method that measures how well the decoder is performing and how close the output is to the original input.

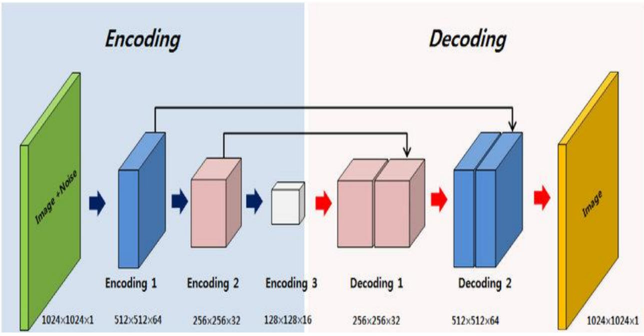

Figure 4: Schematic diagram of Noise Reduction

3.2 Tensor Flow

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks [6]. It is a symbolic math library, and is also used for machine learning applications such as neural networks.

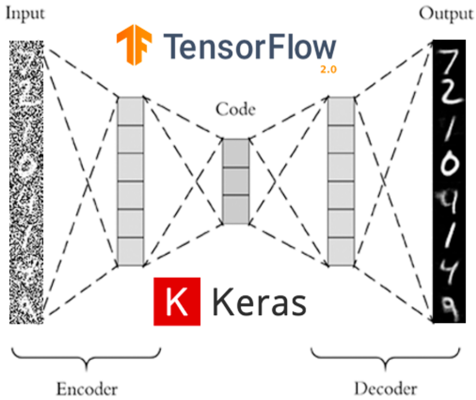

Figure 5: Working Model

3.3 Flow Chart of Model

The convolution autoencoder allows filtering an input signal in order to extract some part of its content. Autoencoders in their traditional formulation do not take into account the fact that a signal can be seen as a sum of other signals. Convolutional Autoencoders, instead, use the convolution operator to exploit this observation.

- Add random noises to the MINST image data and use them as input for training.
- Train a new autoencoder with the noisy data as input and the original data as expected output.
- During the training, the autoencoder learns to extract important features from input images and ignores the image noises because the labels have no noises.

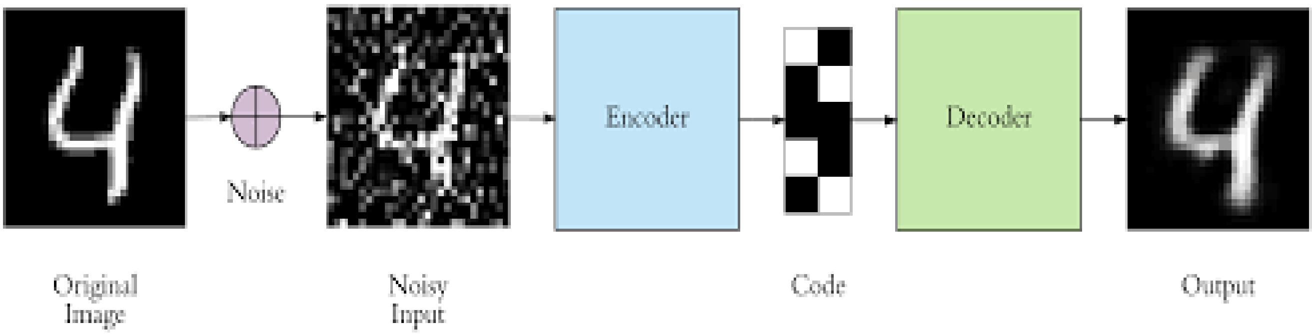

Figure 6: Flow Chart of Model

4. Implementation & Results

The image Denoising model was implemented and we were able to generate moderately comparable output with compared to original input.

4.1 Low-Resolution Images

First of all the image was assigned to the encoder and the data is reconstructed into low dimension image data[2] fashion_mnist(data set):

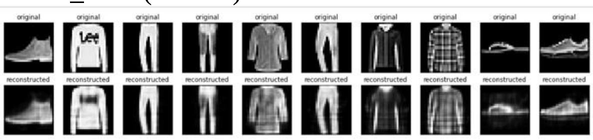

Figure 7: Autoencoder Output

4.2 Noised image

Then the noise is added to low-resolution images:

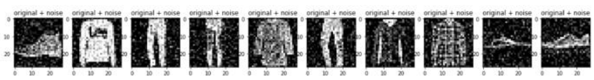

Figure 8: Noised Output

4.3 Epochs

Processing is done on image data again and again (10 Epoch).Where first the image is encoded all the features are extracted from the image ignoring the noise and then reconstructed the image again

```
Epoch 1/10
1875/1875 [==============================] - 72s 38ms/step - loss: 0.0167 - val_loss: 0.0096
Epoch 2/10
1875/1875 [==============================] - 57s 30ms/step - loss: 0.0089 - val_loss: 0.0085
Epoch 3/10
1875/1875 [==============================] - 59s 32ms/step - loss: 0.0081 - val_loss: 0.0079
Epoch 4/10
1875/1875 [==============================] - 77s 41ms/step - loss: 0.0077 - val_loss: 0.0076
Epoch 5/10
1875/1875 [==============================] - 81s 43ms/step - loss: 0.0075 - val_loss: 0.0074
Epoch 6/10
1875/1875 [==============================] - 57s 30ms/step - loss: 0.0073 - val_loss: 0.0073
Epoch 7/10
1875/1875 [==============================] - 85s 45ms/step - loss: 0.0072 - val_loss: 0.0072
Epoch 8/10
1875/1875 [==============================] - 79s 42ms/step - loss: 0.0071 - val_loss: 0.0071
Epoch 9/10
1875/1875 [==============================] - 69s 37ms/step - loss: 0.0071 - val_loss: 0.0071
Epoch 10/10
1875/1875 [==============================] - 61s 33ms/step - loss: 0.0070 - val_loss: 0.0070
```

Figure 9: Epoch Output

4.4 Encoder Summary

Autoencoder encoder summary is calculated where the reduce dimension that is 7, 7, 8 is observed

```
In [67]: autoencoder.encoder.summary()

Model: "sequential_15"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_20 (Conv2D)           (None, 14, 14, 16)        160
_________________________________________________________________
conv2d_21 (Conv2D)           (None, 7, 7, 8)           1160
=================================================================
Total params: 1,320
Trainable params: 1,320
Non-trainable params: 0
_________________________________________________________________
```

Figure 10: Encoder Summary Output

4.5 Decoder Summary

Autoencoder decoder summary is calculated where the reconstructed image dimension that is 28, 28, 16 is obtained.

```
In [68]: autoencoder.decoder.summary()

Model: "sequential_16"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_transpose_12 (Conv2DT (None, 14, 14, 8)         584
_________________________________________________________________
conv2d_transpose_13 (Conv2DT (None, 28, 28, 16)        1168
_________________________________________________________________
conv2d_22 (Conv2D)           (None, 28, 28, 1)         145
=================================================================
Total params: 1,897
Trainable params: 1,897
Non-trainable params: 0
_________________________________________________________________
```

Figure 11: Decoder Output

4.6 Final output

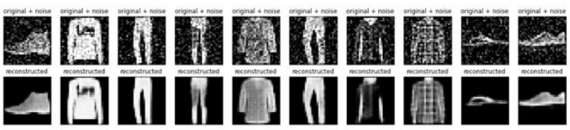

Figure 12: Decoder Output

5. Conclusion

The described model is based on a CNN that encodes an image into a compact representation, followed by the decoded image to remove the noise from the image. It worked quite well when tested on several images. The image generated after the process was quite accurate. But the source of input image also played an important role in feature extraction and hence the removal of noise. Certain images are not well recognized and it is found that there is, still some scope of improvement. There are certain points which can be incorporated into this model to make it even better like larger dataset

References

[1] 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, pp. 1-5, doi: 10.1109/UT.2017.7890316.

[2] Author links open overlay panel Donghoon Lee[a] Sunghoon Choi[b] Hee-JoungKim, "Performace Evaluation of Image denoising developed using convolutional denoising autoencoders in chest radiography", Nuclear Instruments and methods in Physics Research Section Accelerators, Spectrometers, Detectors and Associated Equipments, Vol 88, pp 97-104, 2018

[3] DebjaniBhowmick, Deepak K Gupta, SaumenMaiti, Uma Shankar, "Stacked autoencoders based machine learning for noise reduction and signal reconstruction in geophysical data", arXiv, 2019

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

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

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