ISSN 2712-0562
Sustainable Engineering and Innovation *Original Research*
Vol. 3, No. 2, June 2021, pp.102-111
https://doi.org/10.37868/sei.v3i2.id 142

Facial image noise classification and denoising using neural network

Milan Tripathi[1*]
[1] Computer Engineering, Tribhuvan University, Nepal

*Corresponding author: milan.tripathi@acem.edu.np

Abstract
Image denoising is an important aspect of image processing. Noisy images are produced as a result of technical and environmental flaws. As a result, it is reasonable to consider image denoising an important topic to research, as it also aids in the resolution of other image processing issues. The challenge, however, is that the traditional techniques used are time-consuming and inflexible. This article purposed a system of classifying and denoising noised images. A CNN and UNET based model architecture is designed, implement, and evaluated. The facial image dataset is processed and then it is used to train, valid and test the models. During preprocessing, the images are resized into 48*48, normalize, and various noises are added to the image. The preprocessing for each model is a bit different. The training and validation accuracy for the CNN model is 99.87% and 99.92% respectively. The UNET model is also able to get optimal PSNR and SSIM values for different noises.

Keywords: Classifying, CNN (convolutional neural network), Denoising, PSNR (peak signal-to-noise ratio), SSIM (structural similarity index measure), UNET

1. Introduction

Image Denoising is a crucial topic in image processing and a lot of work is currently being done on it, but there is very little attention towards automating the task of classifying the noised image. Few researchers have work in this field and many papers still only focus on the latter part of denoising. I want to design a convolutional neural network that classifies the noised images into different classes and a UNET based model for denoising the noised image. Since the manual selection of images consumes huge time, automatic classification and denoising save a lot of time and effort.

2. Literature review

Image denoising is a crucial task in image processing and deep learning. Different classical techniques and modern development are explained in his paper [1]. Different classical techniques like Spatial domain filtering, Transform Domain filtering, and modern techniques like CNN-based denoising methods are discussed is explained. Olaf Ronneberger [2] proposed a UNET structure for the first time for the segmentation of biomedical images. In this paper, a U-shaped model was introduced where unlike old models a skip connected between encoding and decoding layer was introduced which allows some data to flow and help in better image generation. Irfan Ali [3] purposed an AutoEncoder model for image denoising with Color Scheme. The work investigates performing denoising on the RGB dataset. Gaussian noise of 0.2 factor is added in all the images of the dataset and an autoencoder is used to remove the noise. Latha H N [4] purposed a local modified UNET Architecture for Image Denoising. Ali Awad [5] proposed a method to remove the

noise from an image corrupted from impulse, Gaussian, or a mixture of both. The method is based on divided into two, in which the first is removing the small noise component, and subsequent steps are based on principal component analysis. The process is assumed to remove the majority of noise in the first stage and smaller ones later. The work investigates the UNET model [2] in removing the noise and compares it with the local modified UNET Architecture. The model is trained in three types of noises Gaussian, Salt&Pepper, and Camera Shake. D. Sil [6] purpose a convolutional neural network for classification and denoising of images. VGG-16 and Inception-v3 were used for the classification of the noised image while a CNN-based denoising method FFDNet was used to denoise noise. J. Gurrola-Ramos [7] purposed a residual Dense U-Net Neural Network to the denoise image. The purposed model has many features like the denoising process does not need knowledge regarding noise before denoising. The model can gain an optimal PSNR and SSIM value. S. Ghose [8] purposed a CNN model to remove noise from an image and restore it to a high-quality image. The analysis is done only for Gaussian noise for different percentage Gaussian white noise and comparison traditional method is also done. O. Sheremet [9] proposed a CNN-based model for denoising images in Info communication systems. Image denoising is a crucial task in image processing and deep learning [11-15]. Hyun Park [16] presented a PCA reconstruction-based denoising approach for removing complicated color noise components on human faces that are difficult to remove with vectorial color filters. The projected methodology consists of the subsequent six steps: coaching of canonical eigenface area exploitation PCA, automatic extraction of countenance exploitation active look model and alignment of the input face to mean form, reconstruction of the associate initial noise-free face, relighting of reconstructed face employing a bilateral filter, extraction of noise regions exploitation the variances of the coloring of coaching information, and reconstruction exploitation partial info of input pictures and mixing of the reconstructed image with the first image.

All the papers discuss the possible solutions of image denoising, but a complete solution to the problem is not provided. This paper aims to bridge that gap by providing a complete automated system of image classification based on noise and denoising. The models are deployed in a web application to provide users an interactive and easy tool to perform image denoising.

3. System overview

Our work aim is to classify and denoise images. A general overview of the system is presented in Figure.1. During the denoising, the noise type determines which UNET to activate.

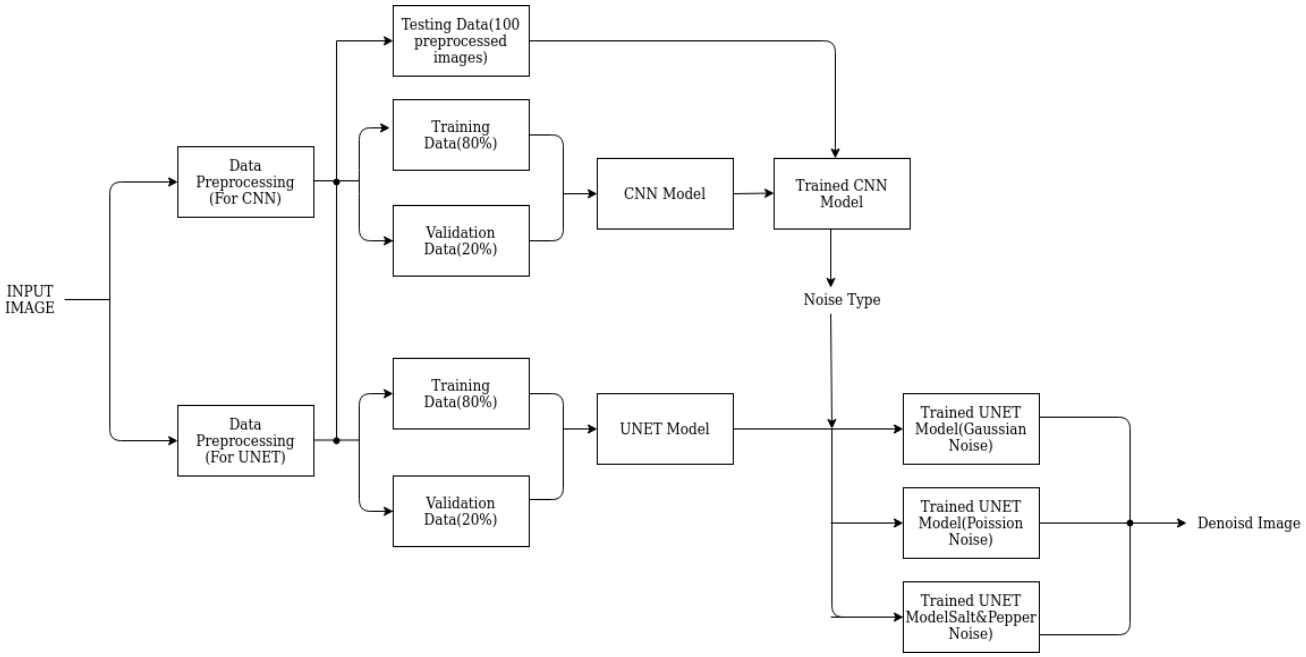

Figure 1. System architecture

4. Deep learning model

Two models are used for classification and denoising respectively. A CNN custom model is designed and implemented for the classification of the image and a UNET based model is used for denoising.

4.1. CNN

A custom CNN model is designed and implemented. To avoid overfitting, a bottom-to-top approach for model building is used. The model which can give optimal results is used. The final gained optimal model is shown in Figure.2.

Layer	Output Shape
conv2d	(None, 48, 48, 6)
maxpooling2d	(None, 24, 24, 6)
conv2d	(None, 24, 24, 16)
activation	(None, 24, 24, 16)
maxpooling2d	(None, 12, 12, 16)
conv2d	(None, 10, 10, 64)
maxpooling2d	(None, 5, 5, 64)
Flatten	(None, 1600)
Dense	(None, 128)
Dropout	(None, 128)
dense	(None, 3)

Figure 2. Model architecture

4.2. UNET

Autoencoder is commonly used for image manipulation functions such as deblurring, denoising, encoding, and so on. The dimensionality of the image can be preserved using an autoencoder model, but the linear comparison of the input results in a bottleneck that does not relay all of the features. The UNET, on the other hand, overcomes this constraint by including a skip relation that enables feature representations to move through. UNET was developed for Biomedical Image Segmentation [2], but it can also be used for image denoising and other image processing activities. Figure.3 depicts the architecture of the UNET model used in this experiment. Certain changes in the original architecture [2] are done as per the requirement while experimenting.

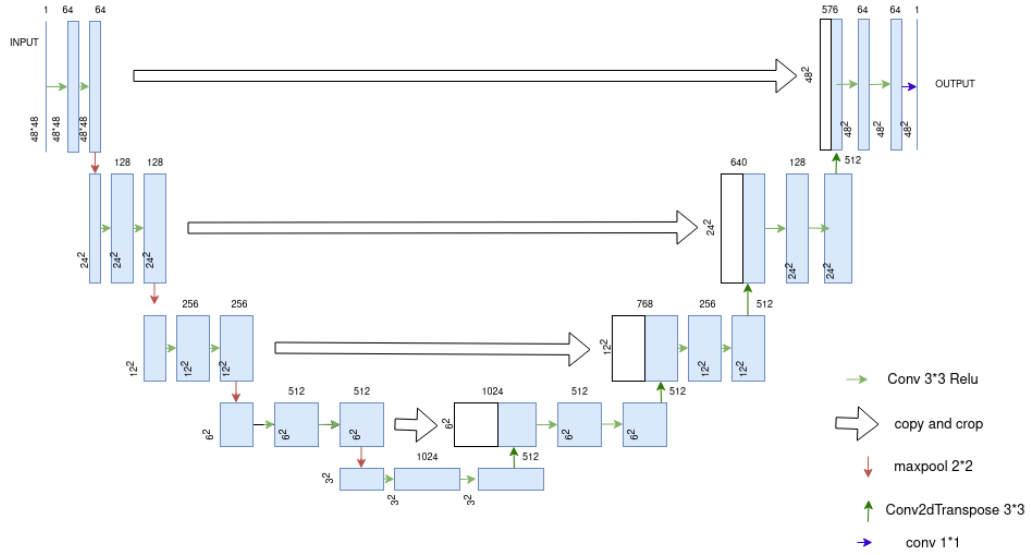

Figure 3. Architecture UNET (modified)

4.3. Basic components of CNN and UNET

The basic components which are required to build the model are described below.

Convolution

A convolution is a combined integration of two functions that demonstrates how one modifies the other. Equation (1) and (2) is the mathematical representation of the operation.

$$(f * g)(t) = \int_{-\infty} f(\tau)g(t-\tau)d\tau \quad (1)$$

$$= \int_{-\infty}^{\infty} f(t-\tau)g(\tau)d\tau \quad (2)$$

There are three major items of this operation: input image, feature detector, and feature map. The matrix representation of the input image is multiplied element-wise with the feature detector to gain a feature map. Another thing is stride which is the shift of the number of pixels over the input image. Figure.4 shows an example of the working of convolution.

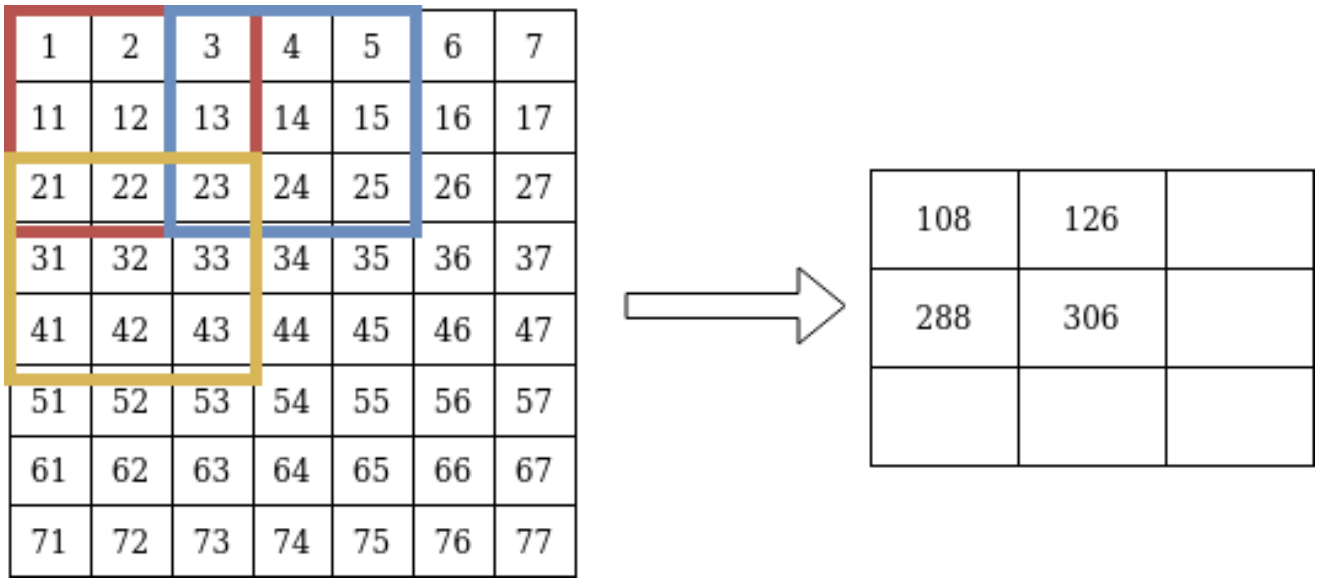

Figure 4. Convolution operation with filter 3*3 with stride 2

Max pooling

It is one of the types of pooling in which a matrix of a certain size is placed on a feature map and the highest value among it is gained. Like convolution in this operation, the stride is used. It enables a CNN model to detect features irrespective of the difference in lighting and angle. Figure.5. shows an example of the working of max pooling.

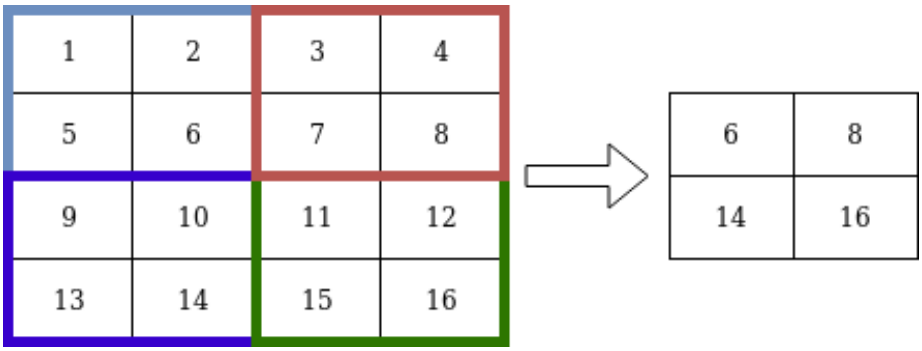

Figure 5. Max pooling with filter 2*2 and stride 2

Dropout

By randomly dropping out nodes during training, a single model can be used to simulate having a large number of different network architectures. A dropout is a regularization approach that reduces over fitting and improves generalization error in deep neural networks of all kinds. It is computationally cheap and surprisingly efficient.

Activation function

It is a critical component of the neural network that introduces non-linear properties. This enables a neural network to learn complex, non-linear mappings between inputs and outputs. There are many types of Activation Function. The ones which are used in the network are Sigmoid, relu, and softmax.

Relu

Relu is the abbreviation for Rectified Linear Unit. If x is positive, it outputs x; otherwise, it outputs zero. It can be mathematically summarized as in equation (3).

$$A(x) = \max(0, x) \tag{3}$$

Softmax

It returns a vector containing the probability distributions of a set of possible outcomes. The mathematical representation of the softmax is given in equation (4).

$$S(y_i) = \frac{e^{y_i}}{\sum_j e^{y_j}} \tag{4}$$

Sigmoid

It compresses a vector in the range (0,1). The mathematical representation of the softmax is given in equation (5).

$$A = \frac{1}{1+e^{-x}} \tag{5}$$

Optimizer

Optimizers are algorithms or methods for changing the characteristics of neural networks, such as weights and learning rate, to minimize losses. There are different types of optimizers such as Gradient Descent, Momentum, Adagrad, RMSProp, etc. In this paper, an Adam optimizer is used.

5. Results

Both models require different ways of processing data and training. So, the explanation is divided into two parts.

5.1. For CNN

Generation of noisy images

Our dataset consists of about 34034 images collected from a website [10]. The images consist of facial images with different types of facial reactions. To perform the intended operation, these images need to be preprocessed. Figure.6 shows the sample of the dataset.

The preprocessing step is shown in Figure.7 and Figure.8 is the visualization of the dataset after preprocessing. A noise factor of 0.1 is added to each image which the images very unclear which is good for training the images as the model will be able to distinguish images with low noise factor efficiently.

	emotion	pixels	usage
0	emotion	pixels	Usage
1	0	70 80 82 72 58 58 60 63 54 58 60 48 89 115 121...	Training
2	0	151 150 147 155 148 133 111 140 170 174 182 15...	Training
3	2	231 212 156 164 174 138 161 173 182 200 106 38...	Training
4	4	24 32 36 30 32 23 19 20 30 41 21 22 32 34 21 1...	Training
5	6	4 0 0 0 0 0 0 0 0 0 0 0 3 15 23 28 48 50 58 84...	Training
6	2	55 55 55 55 55 54 60 68 54 85 151 163 170 179 ...	Training
7	4	20 17 19 21 25 38 42 42 46 54 56 62 63 66 82 1...	Training
8	3	77 78 79 79 78 75 60 55 47 48 58 73 77 79 57 5...	Training
9	3	85 84 90 121 101 102 133 153 153 169 177 189 1...	Training

Figure 6. Dataset Sample

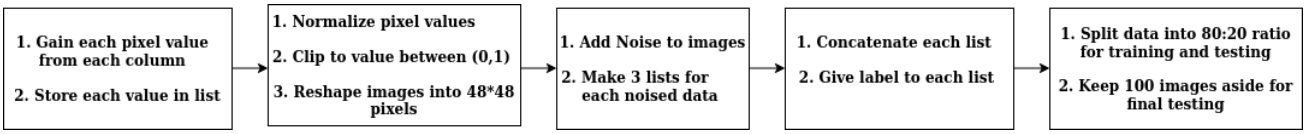

Figure 7. Preprocessing dataset

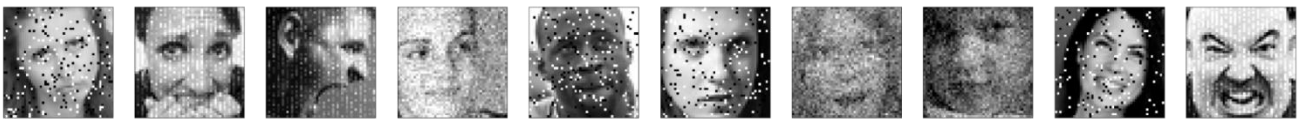

Figure 8. Dataset visualization

Training and validation

Secondly, the model is trained using the training and validation data. The hyperparameters and their values being used while training is listed in Table 1. To avoid overfitting, early stopping and validation data are used while training the model. Even though an epoch of 20 is defined, the model stops training at 8 epochs. The validation data test model at each epoch helps to evaluate model training more precisely.

Table 1. Model hyperparameters

Parameter	**Values**
Optimizer	Adam
Epochs	20
Batch Size	64
Learning Rate	0.001
Loss	categorical cross entropy
Metrics	accuracy

Evaluation of the trained model

Thirdly, the model is used to classify the noised images into different classes based on the noise they are overlapped at. To evaluate the model training and validation are used. The two Figures Figure.9 and Figure.10

clearly show the model training validation accuracy and loss respectively. The testing accuracy of the model is shown in the form of a confusion matrix Figure 11 as it can convey more detailed information. The model gives the training and validation accuracy of 99.87% and 99.92% respectively.

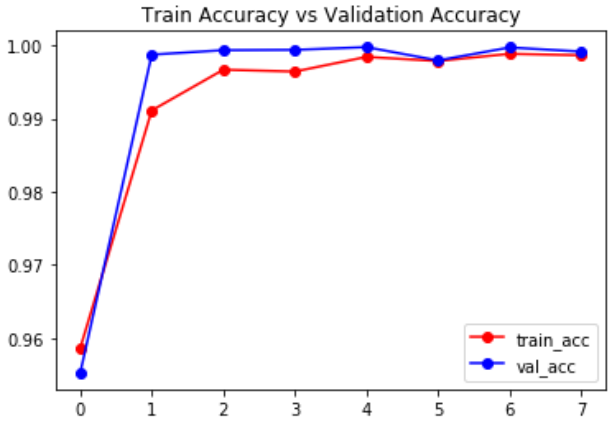

Figure 9. Training accuracy vs. validation accuracy

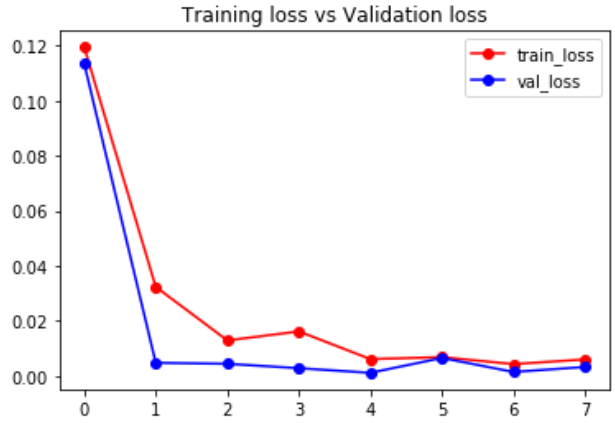

Figure 10. Training accuracy vs. validation loss

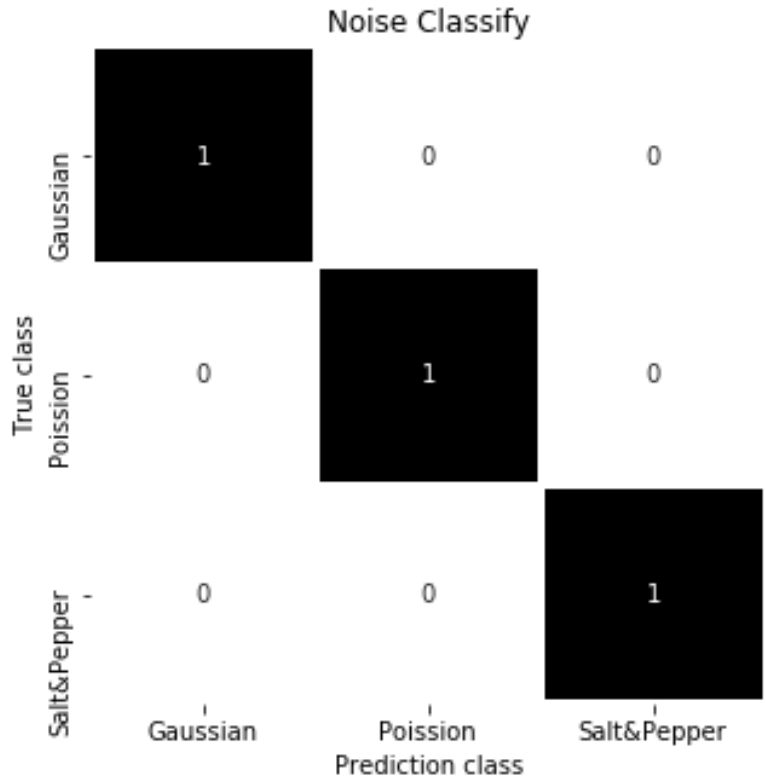

Figure 11. Confusion matrix for test data

Visualizing output

Finally, the actual and predicted class of the test image is shown in Figure.12.

Figure 12. Test Image actual and predicted class

5.2. For UNET

Generation of noisy images

Initially, images are gained as pixel value as they are represented in terms of this form. To perform the intended operation, these images need to be preprocessed. Figure.6 shows the sample of the dataset. The preprocessing step is shown in Figure.13.

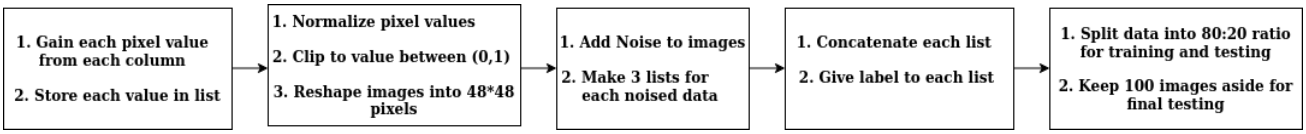

Figure 13. Preprocessing dataset

Training and validation

Secondly, the model is trained in training data to generate a clear noise-free image. The hyperparameters and their values being used while training is listed in Table 2.

Table 2. Model hyperparameters

Parameter	Values
Optimizer	Adam
Epochs	10
Batch Size	64
Learning Rate	0.001
Loss	MSE

Metrics

Two metrics are mostly used to evaluate image denoising tasks, PSNR and SSIM. PSNR stands for Peak signal-to-noise ratio. The equation of the PSNR is shown in (6).

$$PSNR = 20 - log_{10}(MAX_I) - 10 * log_{10}(MSE) \quad (6)$$

$$MSE = \frac{1}{m*n}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[I(i,j) - K(i,j)]^2 \quad (7)$$

MSE stands for mean square error. Its mathematical representation is shown in (7). The m*n represents noise-free monochrome image 'I' having 'K' as noise approximation. MAX_I is the maximum pixel values per pixel. SSIM stands for structural Similarity. The PSNR is not highly indicative of the perceived similarity of the image. So, SSIM is used to address the shortcoming by taking texture into account. Equation (8) is the mathematical representation of SSIM.

$$Q = \frac{4\sigma_{xy}\bar{x}\,\bar{y}}{(\sigma_x^2+\sigma_y^2)[(\bar{x})^2+(\bar{y})^2]} \quad (8)$$

$$Q = \frac{\sigma_{xy}}{\sigma_x\sigma_y}.\frac{2\bar{x}\bar{y}}{(\bar{x})^2+(\bar{y})^2}.\frac{2\sigma_x\sigma_y}{\sigma_x^2+\sigma_y^2} \quad (9)$$

SSIM consists of three parts. These parts are represented in (9). The first part represents the loss of correlation, the second part represents luminance distortion and the last part represents contrast distortion.

Evaluation of the trained model

Thirdly, the model is used to generate a clear image using noised test image, and PSNR and SSIM between the original image and generated image are calculated.

The PSNR is not highly indicative of the perceived similarity of the image. So, SSIM is used to address the shortcoming by taking texture into account. The PSNR and SSIM values of the UNET model are shown in Table 3. It clearly shows optimal values. The model can generate noise-free images with great efficiency in the case of Poisson and Salt & Pepper noise. The image generated in the case of Gaussian has also gained optimal value but less compared to other noises.

Table 3. PSNR and SSIM value dealing with noise factor of 0.05, 0.07 and 0.1

Noise Factor				
		Gaussian	Poisson	Salt & Pepper
0.05	PSNR	29.82395	32.93522	35.49066
	SSIM	0.96428	0.983057	0.99451
0.07	PSNR	28.41099	32.79867	33.25161
	SSIM	0.95555	0.98696	0.99305
0.1	PSNR	26.31419	31.45639	34.41611
	SSIM	0.92677	0.980316	0.992365

Training model deployment and visualizing output

Finally, the models are deployed in a web application. This application is made using HTML, CSS, and Flask. Flask is a web-based framework for the backend and HTML, CSS is used for the frontend. The output gained after passing the image in the web application is shown in Figure. 13.

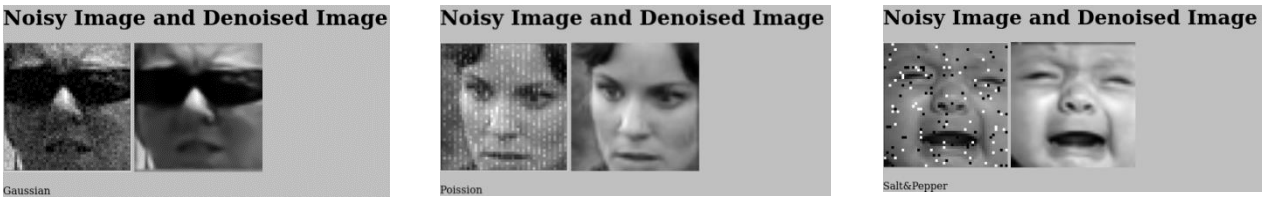

Figure 13. Web application

6. Conclusion

The experiment shows that the proposed CNN model can classify the images based on the noise they are overlapped with optimal training and validation accuracy. It also gives an optimal result while testing. Also, the proposed UNET model can denoise images with optimal PSNR and SSIM values. Thus, the proposed system provides a complete solution for denoising images and also can be used for other image processing tasks.

References

[1] L. Fan, F. Zhang, H. Fan, and C. Zhang, "Brief review of image denoising techniques," Visual Computing for Industry, Biomedicine, and Art, vol. 2, no. 1. 2019, doi: 10.1186/s42492-019-0016-7.

[2] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 9351, pp. 234–241, 2015, doi: 10.1007/978-3-319-24574-4_28.

[3] I. Ali et al., "Image Denoising with Color Scheme by Using Autoencoders," Ijcsns, vol. 18, no. 12, pp. 158–161, 2018.

[4] Latha H N and Rajiv R Sahay, "A Local Modified U-net Architecture for Image Denoising," Int. J. Mod. Trends Sci. Technol., vol. 6, no. 8S, pp. 140–144, Sep. 2020, doi: 10.46501/IJMTSTCIET27.

[5] A. Awad, "Denoising images corrupted with impulse, Gaussian, or a mixture of impulse and Gaussian noise," Eng. Sci. Technol. an Int. J., vol. 22, no. 3, pp. 746–753, Jun. 2019, doi: 10.1016/J.JESTCH.2019.01.012.

[6] D. Sil, A. Dutta, and A. Chandra, "Convolutional Neural Networks for Noise Classification and Denoising of Images," IEEE Reg. 10 Annu. Int. Conf. Proceedings/TENCON, vol. 2019-October, pp. 447–451, Oct. 2019, doi: 10.1109/TENCON.2019.8929277.

[7] J. Gurrola-Ramos, O. Dalmau, and T. E. Alarcon, "A Residual Dense U-net Neural Network for Image Denoising," IEEE Access, 2021, doi: 10.1109/ACCESS.2021.3061062.

[8] S. Ghose, N. Singh, and P. Singh, "Image denoising using deep learning: Convolutional neural network," Proc. Conflu. 2020 - 10th Int. Conf. Cloud Comput. Data Sci. Eng., pp. 511–517, Jan. 2020, doi: 10.1109/CONFLUENCE47617.2020.9057895.

[9] O. Sheremet, K. Sheremet, O. Sadovoi, and Y. Sokhina, "Convolutional Neural Networks for Image Denoising in Infocommunication Systems," 2018 Int. Sci. Conf. Probl. Infocommunications Sci. Technol. PIC S T 2018 - Proc., pp. 429–432, Jan. 2019, doi: 10.1109/INFOCOMMST.2018.8632109.

[10] "fer2013 | Kaggle." https://www.kaggle.com/deadskull7/fer2013 (accessed Jul. 12, 2021).

[11] H. Tao, L. Zhao, J. Xi, L. Yu, and T. Wang, "Fruits and vegetables recognition based on color and texture features," Nongye Gongcheng Xuebao/Transactions Chinese Soc. Agric. Eng., vol. 30, no. 16, pp. 305–311, Aug. 2014, doi: 10.3969/J.ISSN.1002-6819.2014.16.039.

[12] D. H. Hubel and T. N. Wiesel, "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex," J. Physiol., vol. 160, no. 1, 1962, doi: 10.1113/jphysiol.1962.sp006837.

[13] S. H. Lee, C. S. Chan, S. J. Mayo, and P. Remagnino, "How deep learning extracts and learns leaf features for plant classification," Pattern Recognit., vol. 71, pp. 1–13, Nov. 2017, doi: 10.1016/J.PATCOG.2017.05.015.

[14] P. Wang, W. Li, S. Liu, Z. Gao, C. Tang, and P. Ogunbona, "Large-scale Isolated Gesture Recognition using Convolutional Neural Networks," Proc. - Int. Conf. Pattern Recognit., vol. 0, pp. 7–12, Jan. 2016, doi: 10.1109/ICPR.2016.7899599.

[15] X. W. Gao, R. Hui, and Z. Tian, "Classification of CT brain images based on deep learning networks," Comput. Methods Programs Biomed., vol. 138, pp. 49–56, Jan. 2017, doi: 10.1016/J.CMPB.2016.10.007.

[16] H. Park and Y. S. Moon, "Automatic denoising of 2D color face images using recursive PCA reconstruction," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 4179 LNCS, pp. 799–809, 2006, doi: 10.1007/11864349_73.