International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Evaluation of Autoencoders: Training Using Original, Encoded and Decoded Images for Prediction

Ms. Nidhi B. Shah *[1], Dr. Amit P. Ganatra[2]

Submitted: 25/01/2024 **Revised**: 03/03/2024 **Accepted**: 11/03/2024

Abstract: An autoencoder (AE) is a neural network that seeks to make the same contribution to production and performance by directly reducing the inputs to the latent domain map and reconstructing the outputs from that map. Today data processing and reduction of data view size are considered to be the two most effective autoencoder systems. With the right size and sparsity limits, autoencoders can learn more interesting data predictions than PCA or other basic techniques. In this paper, we implement different types of autoencoders and evaluate them. Original image or raw image is taken and is trained. Using this training information testing is being performed on image prediction. This is being measured based on accuracy as well as time. The model is also applied by giving input as an encoded image which is an intermediate result of autoencoder and giving input as a decoded image a final result of autoencoder. All the above three models are applied and results are compared by implementing different types of autoencoders.

Keywords: *Autoencoder, encode, decode, dimensionality reduction, feature learning, neural network.*

1. Introduction

The purpose of autoencoder (AE) neural networks is to identify productivity efforts. An artificial neural network type called an autoencoder is utilized in unsupervised learning. The input data is intended to be encoded into a compressed representation, which is subsequently decoded to restore its original format. Autoencoders' main objective is to develop a condensed, effective representation of the input data that captures its most important attributes. An autoencoder's structure normally consists of three basic components: Encoder: The encoder processes the input data by mapping it to a representation in lower dimensions, which is referred to as the "latent space" or "encoding." The most significant and pertinent characteristics of the incoming data should ideally be captured by this encoding.

The structure of an autoencoder typically consists of three main parts:

Encoder: The encoder takes the input data and maps it to a lower-dimensional representation, also known as the "latent space" or "encoding." This encoding should ideally capture the most important and relevant features of the input data.

Bottleneck (latent space): The compressed version of the input data is represented by the bottleneck layer. Its dimensionality is lower than that of the input and serves as the bottleneck through which the information flows. This compressed representation is the crux of the autoencoder, as it retains the crucial information needed for reconstruction.

[1] *Ph.D. Research Scholar, Computer Engineering Department, CHARUSAT University, India*
ORCID ID : 0000-0003-4262-4244
[2] *Provost, Parul University, India*
ORCID ID : 0000-0002-9993-9901
** Corresponding Author Email: nbshah999@yahoo.com*

Decoder: The decoder reconstructs the original input data using the compressed representation obtained from the bottleneck layer. The goal of the decoder is to recreate the input data as accurately as possible from the reduced representation.

Training an autoencoder involves a two-step process:

Encoding: The input data is fed into the autoencoder, and the encoder produces a compressed representation.

Decoding: The compressed representation is then fed through the decoder to reconstruct the input data.

The autoencoder's performance is evaluated based on how closely the reconstructed output matches the original input.

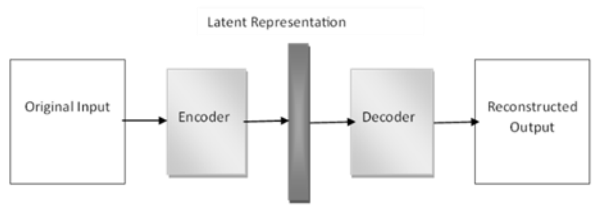

Fig. 1 Structure Of Autoencoder

If the only object of the autoencoder transformed into identical their participation to their productivity and it would be useless. So, we undertake that, by way of training the autoencoder to imitate their efforts, the illustration h will take on advantageous purposes[12].

This can be done with the help of increasing barriers to copying. Another way to learn practical skills with an autoencoder is to make h less than x. Here, the autoencoder is called inadequate. By training partial rendering, it forces the default scanner to investigate the most important aspects of the reality of training. If you have a lot of room in your autoencoder, you can learn how to run a copy job with almost no extraction of useful facts that are actually widespread[7].

2. Related Work

These days, information processing and reduction of information view length are considered to be the two simplest autoencoder structures. With the right length and sparsity limits, autoencoders can analyze more thrilling data predictions than PCA or different fundamental strategies[12].

The autoencoder is read by the robot on the information instance. In this way, training a particular set of rules and conditions to work properly with a particular type of input is easy, does not require new techniques, and eliminates the simplest relevant instructions[12].

However, the autoencoder does perform a negative image compression function. Since the autoencoder is trained with a specific set of statistics, it gives reasonable compression results with the same statistics as the training set used, but it can also be a fully frame-based image compressor. Compression strategies like JPEG do much more[7].

Autoencoders are skilled to workshop as a good deal information as feasible whilst the input is implemented to the sender and then to the codec, however they're also skilled to make the brand-new presentation have a diffusion of beautiful homes. One-of-a-kind types of autoencoders goal to gain extraordinary sorts of systems[7].

3. Proposed Method

In these three different models are proposed for prediction of image. All these model servers a purpose of predicting image and are evaluated based on prediction accuracy and time. Working of individual model is discussed.

Method 1: Training using Original(raw) image

Method1: Here in this method an original image is taken and Autoencoder is applied to it. Consequently, we have a trained model. Further this model is used for testing by giving images for prediction. Here prediction accuracy and time are noted for the method1.

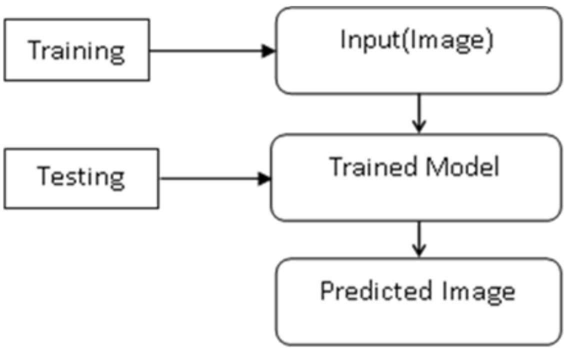

Fig. 2 System flow for Method 1

Method 2: Training using Compressed image /Encoded Image

Method 2: Here in this method an original image is taken and an autoencoder is applied to it. During this and intermediate result, an encoded layer is selected for training the image. Consequently, we have a trained model. Further this model is used for testing by giving images for prediction. Here prediction accuracy and time are noted for the method 2.

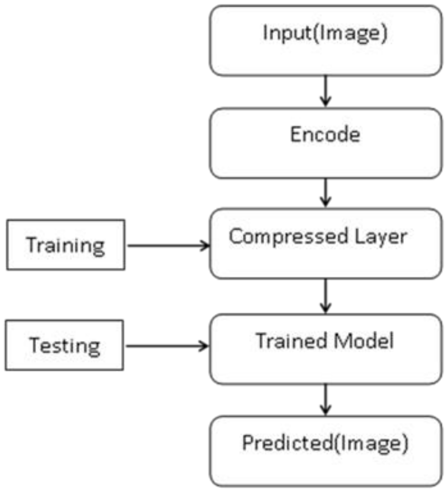

Fig. 3 System flow for Method2

Method 3: Training using Decoded Image

Method 3: Here in this method an original image is taken and an autoencoder is applied to it. During this as a last step after decoder is applied, result is the reconstructed image. This reconstructed image is selected for the training. Consequently, we have a trained model. Further, this model is used for testing by giving images for prediction. Here prediction accuracy and time are noted for the method3.

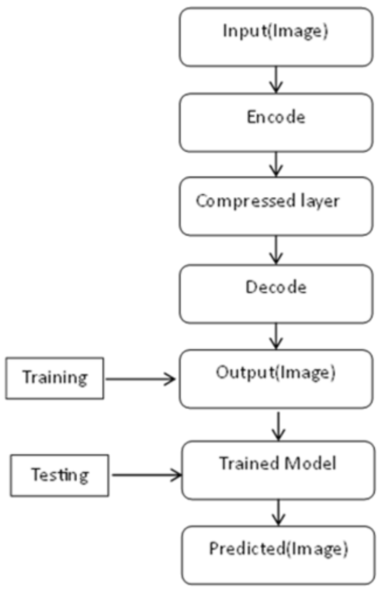

Fig. 4 System flow for Method3

Prediction accuracy and time for prediction is compared for all three models.

APPLYING ALL THREE METHODS USING VARIOUS AUTOENCODERS

Here are four types of autoencoders that are applied on different forms of images using the Kera's framework and the MNIST dataset.

VANILLA AUTOENCODER

In its functional form, the autoencoder is a three-layer internet, a neural internet with unbound obfuscated layers. The inputs and outputs are the same, and Adam's putting and root-mean-squared loss distinctions are used to determine how to reinvent the input that occurs. Since the size of the hidden layer (64) is smaller than the input (784), we have developed a partial autoencoder. This edge drives the neural network to examine the compressed representation of the data. The values indicate the accuracy of the v / s periodic format for testing the model within the first frame, output frame, and encoded frame[11].

Architecture of Vanilla Autoencoder is shown in Fig. 3

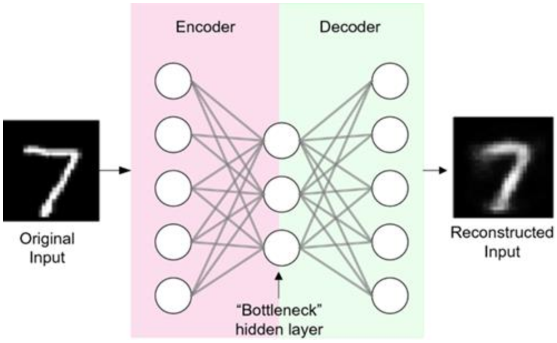

Fig. 5 Architecture of Vanilla Autoencoder[8]

Where Size of Input Layer=784, Size of Hidden Layer=64, Size of Output Layer=784

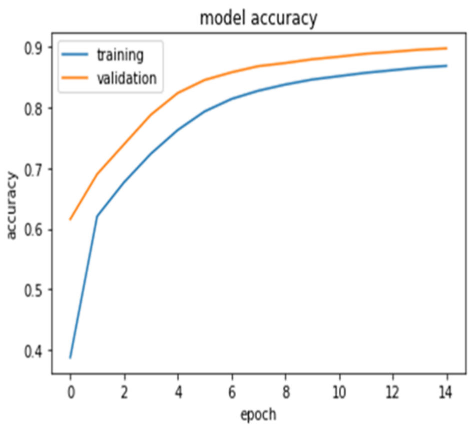

Fig. 6 Precision v/s epoch for original image

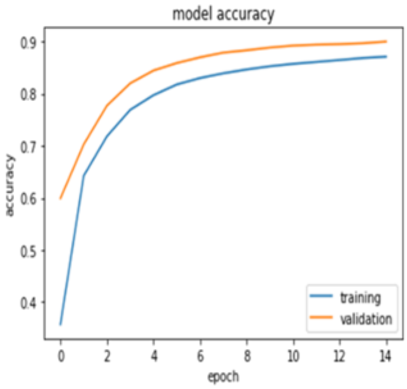

Fig. 7 Precision v/s epoch for Decoded image

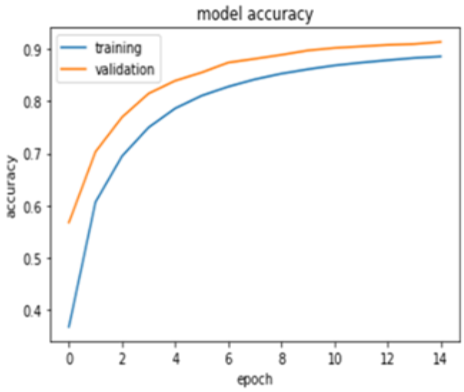

Fig. 8 Precision v/s epoch for Encoded image

MULTILAYER AUTOENCODER

If one concealed layer is not abundant, we will strikingly extend the autoencoder to extra hidden layers. Our utility makes use of 3 hidden layers in preference to impartial unique. In the least hidden layers can be nominated as a distinguishing illustration but will make the network superior uniform and use the center layer much extra. The values show the accuracy of the v / s length shape to check the version within the first photo, the output photo, and the coded image[11]. Architecture of Multilayer Autoencoder is shown in Fig. 7.

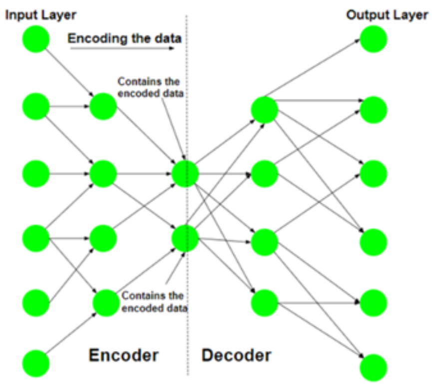

Fig. 9 Architecture of Multilayer Autoencoder[10].

Where Size of Input Layer=784, Size of Hidden Layer 1=128, Size of Code Layer=64, Size of Hidden Layer 2=128, Size of Output Layer=784

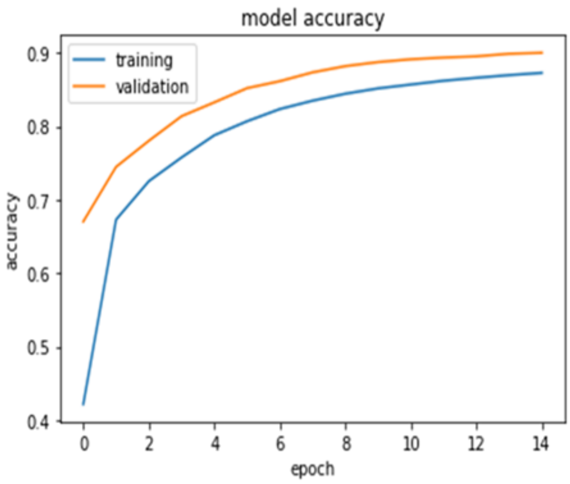

Fig. 10 Precision v/s epoch for original image

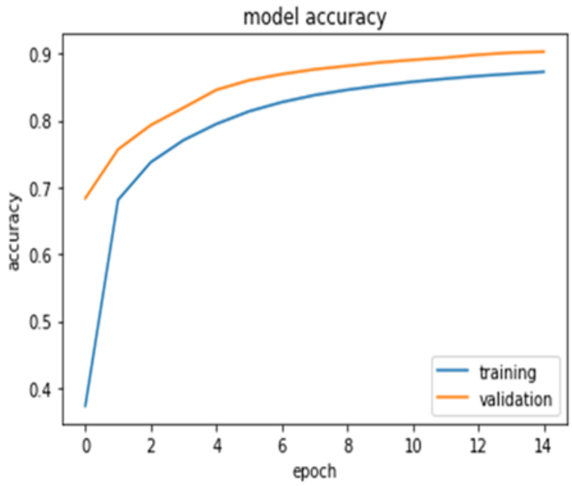

Fig. 11 Precision v/s epoch for Decoded image

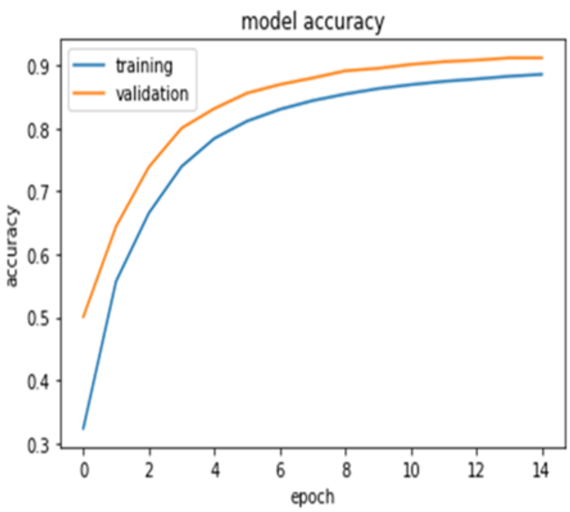

Fig. 12 Precision v/s epoch for Encoded image

CONVOLUTIONAL AUTOENCODER

Here the use of autoencoders with Convolutions as opposed to absolutely included layers. but you're the usage of pics (3-d vectors) as opposed to 1D flat vectors. The inserted image forces the default encoder to examine the compressed versions of the photos by providing a subtle representation of the smallest size through a pattern underneath. The values display the precision of the v/s duration structure to examine the model in the initial, output, and coded photos. [11].Architecture of Convolutional Autoencoder is shown in Table 1.

Table 1 Accuracy v / s duration

Network	***Layer***	***Dimension***	***Activation***
Input	Input	28x28x1	-
Encoder	Convolution	3x3@16	ReLu
	Max Pooling	2x2@16	-
	Convolution	3x3@8	ReLu
	Max Pooling	2x2@8	-
	Convolution	3x3@8	ReLu
	Max Pooling	2x2@8	-
Decoder	Convolution	3x3@8	ReLu
	Up Sampling	2x2@8	-
	Convolution	3x3@8	ReLu
	Up Sampling	2x2@8	-
	Convolution	3x3@16	ReLu
	Up Sampling	2x2@8	-
	Convolution	3x3@1	Sigmoid

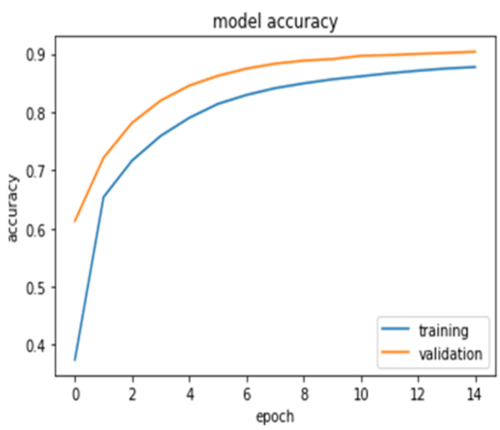

Fig. 13 Precision v/s epoch for original image

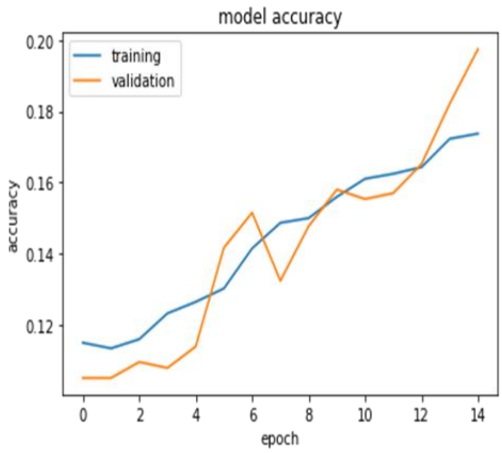

Fig. 14 Precision v/s epoch for Decoded image

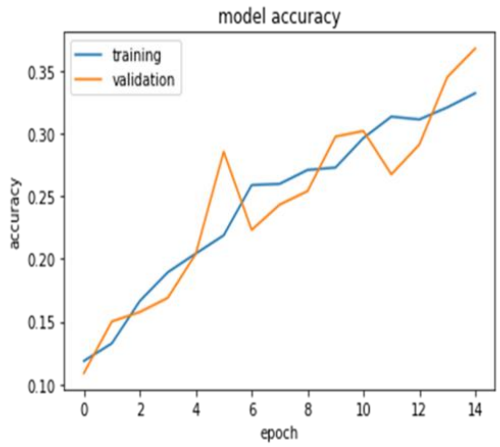

Fig. 15 Precision v/s epoch for Encoded image

REGULARIZED AUTOENCODER

Instead of placing a hidden layer that is smaller than the input, there are several ways you can prevent the autoencoder from rebuilding. A typical autoencoder uses an expiration function that guarantees a model that has a different function than the ability to mimic the input at the output, instead of limiting the version range by keeping the encoder and code dimensions minor. To do. Indeed, we usually find two types of dominant autoencoders: sparse autoencoders and vehicle audio output [4][11].

Sparse autoencoder: A small autoencoder is used regularly to look up the properties of all other functions like classes. Autoencoders that need to prove to be familiar with the minority need to address the unique mathematical features of the database being trained, rather than actually acting as proprietary. In this way, training to perform copy tasks at the expense of economy can create models that have learned useful functions, including by-products. The hidden layer provided a preferred l1 characteristic that applied excellent features to the loss of the entire extension section. The values show the accuracy of the v / s duration shape to validate the model of the first photo, output image, and encoded image[6]. Architecture of Sparse Autoencoder is shown in **Fig.** 14.

Where size of Input Layer is 784, Size of Hidden Layer is 64 and Size of Output layer is 784

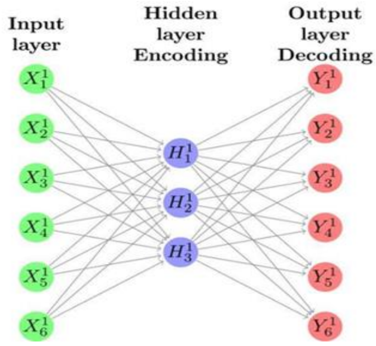

Fig. 16 Architecture of Sparse Autoencoder[9].

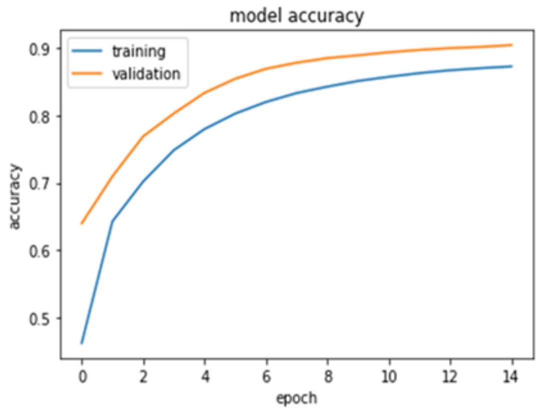

Fig. 17 Precision v/s epoch for original image

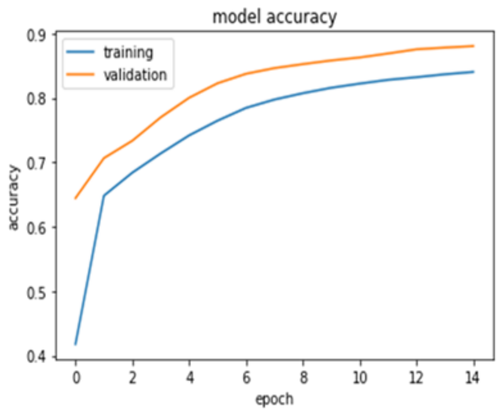

Fig. 18 Precision v/s epoch for Decoded image

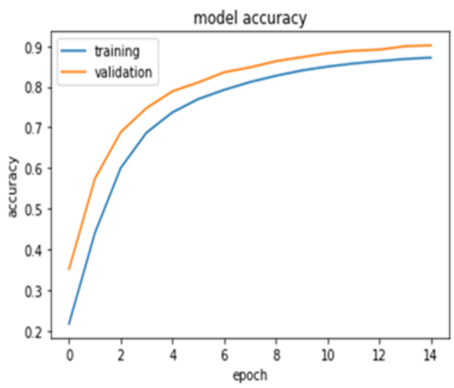

Fig. 19 Precision v/s epoch for Encoded image

Denoising autoencoder: Instead of including the result in the expiration function, you can see that the autoencoder is learning something useful by modifying the error call to the reconstruct function. This is done by accumulating another image in the input image and letting the autoencoder find a way to discard it. In this way, the encoder extracts the most important features and examines the powerful representation of the recording. The values indicate the accuracy of the v / s period structure for testing the model with the first image, output image, and code photo[6]. Architecture of Denoising Autoencoder is shown in Fig. 18.

Here noise factor is 0.5.

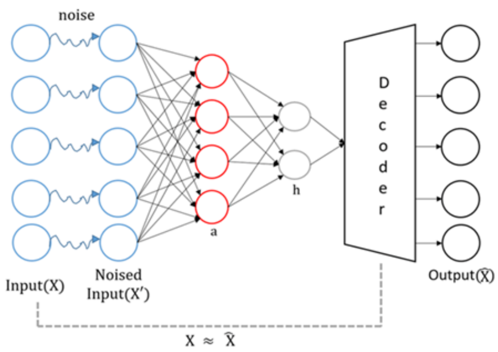

Fig. 20 Architecture of Denoising Autoencoder(Karagoz et al., 2020)

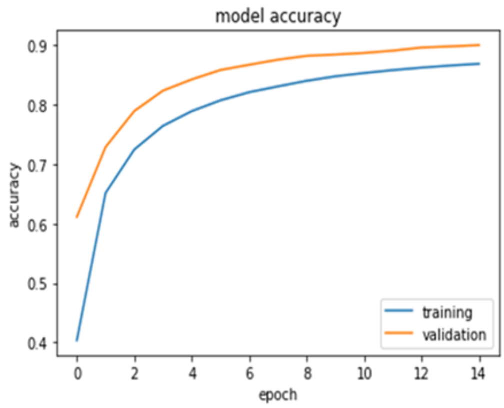

Fig. 21 Precision v/s epoch for original image

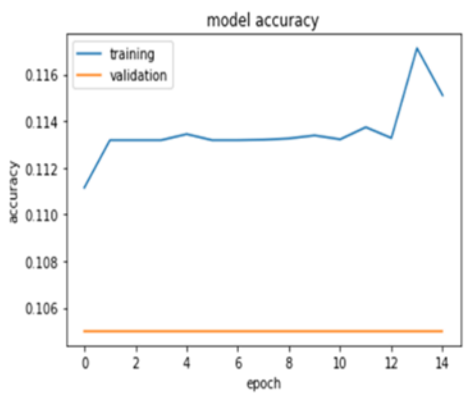

Fig. 22 Precision v/s epoch for Decoded image

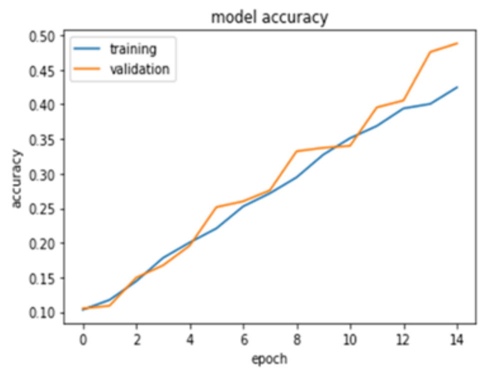

Fig. 23 Precision v/s epoch for Encoded image

4. EXPERIMENTAL RESULT

Table 2 show different Autoencoder and their evaluation accuracy on original image, encoded image and decoded image used for training.

Table 2 Accuracy comparison of various autoencoder on Original, Encoded and Decoded Images

Autoencoder	***Accuracy***		
	Original	***Encoded***	***Decoded***
Vanilla Autoencoder	88.3	90.3	88.2
Multilayer Autoencoder	88.3	89.5	88.4
Convolutional Autoencoder	89.1	42.3	20.0
Regularized Sparse autoencoder	88.3	88.5	85.1
Regularized Denoising Autoencoder	88.5	47.8	11.3

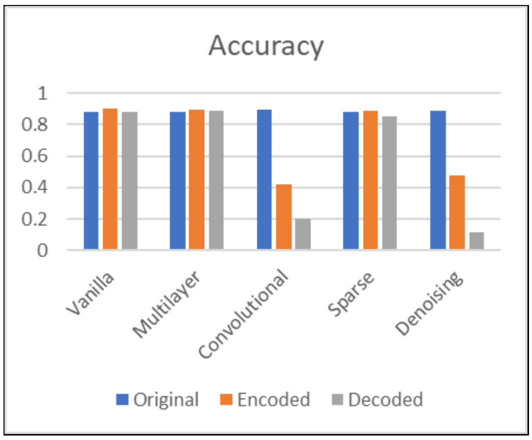

Fig. 24 Graph for Accuracy comparison of various autoencoder on Original, Encoded and Decoded Images

Following Table show different Autoencoder and their execution time on Original Image, encoded image and decoded image used for training.

Table 3 Execution time comparison of various autoencoder on Original, Encoded and Decoded Images

Autoencoder	*Execution Time in Seconds*		
	Original	*Encoded*	*Decoded*
Vanilla Autoencoder	15.29	10	15.06
Multilayer Autoencoder	21.27	10.71	14.14
Convolutional Autoencoder	13.81	11.3	13.73
Regularized Sparse Autoencoder	14.62	10.01	14.43
Regularized Denoising Autoencoder	21.64	19.06	21.64

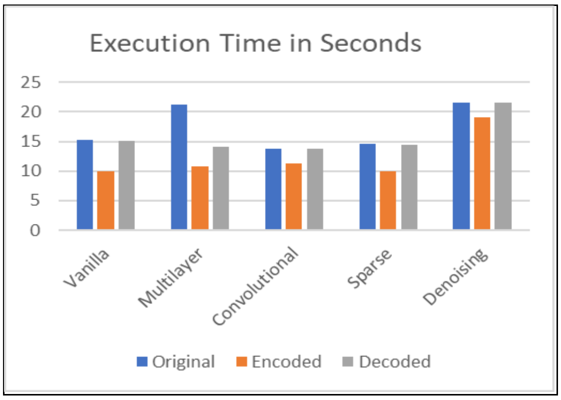

Fig. 25 Graph for Execution time comparison of various autoencoder on Original, Encoded and Decoded Images

5. Conclusion

In this paper, we also observed many unparalleled problems with autoencoders, vanillas, multi-layers, convolutions, and regularization. Each has a unique home, depending on either the limits imposed, the reduced dimensions of the hidden layers or other types of consequences. From these experiments, we conclude that the prediction accuracy of encoded snapshots, with the exception of the convolution and noise reduction auto-encoder, yields higher results with the model than with encoded or unique photographs. For execution time results of the encoded image have better results with comparison to original and decoded images for all model.

References

[1] G. N. Karagoz, A. Yazici, T. Dokeroglu, and A. Cosar, "Analysis of Multiobjective Algorithms for the Classification of Multi-Label Video Datasets," IEEE Access, vol. 8, pp. 163937–163952, 2020, doi: 10.1109/access.2020.3022317.

[2] J. Zhai, S. Zhang, J. Chen, and Q. He, "Autoencoder and Its Various Variants," 2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Oct. 2018, Published, doi: 10.1109/smc.2018.00080.

[3] C.-Y. Chen, J.-S. Leu, and S. W. Prakosa, "Using autoencoder to facilitate information retention for data dimension reduction," 2018 3rd International Conference on Intelligent Green Building and Smart Grid (IGBSG), Apr. 2018, Published, doi: 10.1109/igbsg.2018.8393545.

[4] D. Tomar, Y. Prasad, M. K. Thakur, and K. K. Biswas, "Feature Selection Using Autoencoders," 2017 International Conference on Machine Learning and Data Science (MLDS), Dec. 2017, Published, doi: 10.1109/mlds.2017.20.

[5] F. Zhuang, X. Cheng, P. Luo, S. J. Pan, and Q. He, "Supervised Representation Learning with Double Encoding-Layer Autoencoder for Transfer Learning," ACM Transactions on Intelligent Systems and Technology, vol. 9, no. 2, pp. 1–17, Oct. 2017, doi: 10.1145/3108257.

[6] A. Caliskan, H. Badem, A. Basturk, and M. E. Yuksel, "The effect of autoencoders over reducing the dimensionality of a dermatology data set," 2016 Medical Technologies National Congress (TIPTEKNO), Oct. 2016, Published, doi: 10.1109/tiptekno.2016.7863101.

[7] "Autoencoders in Deep Learning: Tutorial & Use Cases [2023]," Autoencoders in Deep Learning: Tutorial & Use Cases [2023]. https://www.v7labs.com/blog/autoencoders-guide

[8] K. Sangwan, "Journey through Sequence-to-Sequence models, Attention and Transformer," Medium, Jul. 14, 2021. https://kuldeepsangwan.medium.com/journey-through-sequence-to-sequence-models-attention-and-transformer-56365c32e99e

[9] "Explain about Sparse Autoencoder? | i2tutorials," i2tutorials, Sep. 13, 2019. https://www.i2tutorials.com/explain-about-sparse-autoencoder/

[10] "ML | Auto-Encoders - GeeksforGeeks," GeeksforGeeks, Jun. 21, 2019. https://www.geeksforgeeks.org/ml-auto-encoders/

[11] R. Khandelwal, "Deep Learning—Different Types of Autoencoders," Medium, Jan. 25, 2019. https://medium.datadriveninvestor.com/deep-learning-different-types-of-autoencoders-41d4fa5f7570

[12] N. Hubens, "Deep inside: Autoencoders," Medium, Apr. 10, 2018. https://towardsdatascience.com/deep-inside-autoencoders-7e41f319999f