Efficient Discrete Cosine Transform (DCT) coefficients computation for Images Using Deep Learning

Author

Zirak Khan

FALL 2016 -MS-16 (CSE) 00000172451

MS-16 (CSE)

Supervisor

Dr. Farhan Hussain

DEPARTMENT OF COMPUTER& SOFTWARE ENGINEERING

COLLEGE OF ELECTRICAL & MECHANICAL ENGINEERING

NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY

ISLAMABAD

MAY, 2019

Efficient Discrete Cosine Transform (DCT) coefficients computation for Images Using Deep Learning

Author

Zirak Khan

FALL 2016 - MS-16 (CSE) 00000172451

A thesis submitted in partial fulfillment of the requirements for the degree of

MS Computer Software Engineering

Thesis Supervisor:

Dr. Farhan Hussain

Thesis Supervisor's Signature:-_____

DEPARTMENT OF COMPUTER& SOFTWARE ENGINEERING

COLLEGE OF ELECTRICAL & MECHANICAL ENGINEERING

NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY, ISLAMABAD

MAY, 2019

DECLARATION

'I hereby, certify that this thesis research work titled *"Efficient Discrete Cosine Transform (DCT) Coefficients Computation for Images Using Deep Learning"* is my own work under the supervision of Dr. Farhan Hussain and co-supervision by Dr. Aimal Khan. The work has not been presented elsewhere for assessment. The material used, from other sources in this research work, has been properly referred / acknowledged.'

_____

Signature of Student

Zirak Khan

FALL 2016 - MS-16 (CSE) 00000172451

LANGUAGE CORRECTNESS CERTIFICATE

This research document has been reviewed by a professional English writer and is free of any kind of spellings, semantic, syntax and grammatical mistakes. The thesis document is also in accordance with the format / template provided by the University for MS thesis work.

_____

Signature of Student

Zirak Khan

FALL 2016 - MS-16 (CSE) 00000172451

_____

Signature of Supervisor

Dr. Farhan Hussain

COPYRIGHT STATEMENT

This page is intentionally left blank

ACKNOWLEDGEMENTS

"I am extremely thankful to **ALLAH** Almighty for his bountiful blessings throughout this work. Indeed this would not have been possible without his substantial guidance through every step, and for putting me across people who could drive me though this work in a superlative manner. Indeed none be worthy of praise but the Almighty. In addition, my admirations be upon Prophet **Hazrat Muhammad (PBUH)** and his Holy Household for being source of guidance for people.

I would like to extend my special thanks of gratitude to my supervisor **Dr. Farhan Hussain** as well as my co-supervisor **Dr. Aimal Khan** for their precious assistance throughout my thesis, and for being available even for the pettiest of issues. I am very grateful for their meticulous evaluation of the thesis and guidance on how to improve it in the best way possible.

I am profusely thankful to **Dr. Muhammad Abbas** and **Dr. Arslan Shaukat** for an excellent guidance throughout this journey and for being part of my evaluation committee.

It is indeed a privilege to thank **Dr. Saif ul Islam** his constant encouragement throughout my research period. The sense of belief that he instilled in me has helped me sail through this journey. I would also like to extend gratitude to my family & friends, especially **Maaz Jamal**, **Mobeen** and **Shahid Ali Khan** who have rendered valuable assistance to my study.

Finally, I would like to appreciate and express heartfelt gratitude to all the people who have provided their valuable assistance during this period".

“Dedicated to my exceptional parents, excellent siblings and my best friends whose tremendous support and cooperation led me to this wonderful accomplishment”

ABSTRACT

Digital audio, videos and images corresponds to huge amount of data. The storage and transmission of this data requires significant amount of bandwidth and memory respectively. Compressing this digital data is a field which has been researched upon for decades. Many state of the art compression algorithms have been proposed to cater the storage and transmission requirements of digital data. Aiming at the digital image compression, Discrete Cosine Transform is a widely used transform to explore the frequencies present in a digital image. During the quantization step the less significant frequencies are discarded and only the more important frequencies are retained. This quantization results in the reduced representation of the image hence compression is achieved. The image reconstructed from this reduced frequency set is an approximation of the original image and hence it results in lossy image compression. Lately, a significant amount of research work has been conducted based on the use of neural networks for image compression. This thesis presents a detailed literature review to thoroughly analyze the existing literature and methods. In this thesis we target a deep neural network that can estimate the most important DCT coefficients for an image and then we utilize these most significant DCT Coefficients for the classification task. The estimation of DCT coefficients is targeted by a Multi-Layered Perceptron (MLP) model and a Deep Convolutional Neural Network (DCNN) model. The experimentation showed promising results and revealed that MLP models have relatively lower error rate between actual and predicted results, as compared to DCNN models. Later on, MNIST image dataset is applied to the proposed deep learning models for the prediction of its most significant DCT coefficients and the predicted results are then used for digits classification. The experimental results support the DCT based digits classification with an accuracy of 95%, which is quiet promising. In future, the proposed technique also leverages the use of compressed images for tackling different image classification and regression problems. Moreover, the proposed deep neural networks can be further generalized to support videos and color representations.

Keywords: Image Compression, lossy compression, DCT, Deep Learning, CNN, MLP.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

CHAPTER 1: INTRODUCTION

This chapter contains a brief introduction of the research performed. The problem statement along with some background study is elaborated in **Section 1.1**. Classification problem is specified in **Section 1.2**. **Section 1.3** includes the proposed methodology and **Section 1.4** provides a brief overview to our research contribution. Lastly, the thesis organization is stated in **Section 1.5**.

1.1. Problem Statement

In this thesis, we have targeted the lossy compression of digital images; by extracting the most important Discrete Cosine Transform (DCT) coefficients, with the help of deep neural networks. We also target, to take advantage of these reduced representations to classify images from the standard dataset i.e. MNIST data set with the help of deep neural network. Classification based on these reduced representation helps in decreasing the size and training time of our deep neural networks. Moreover, these deep neural networks show promising results on the classification task. Some background related to our work is given as follow:

1.1.1. **Image Compression**

With the advancement in communication technology the use of images and videos has increased which thus elevated the importance of image and video compression. Visual information collected from images needs to be stored and transmitted. But these visual data files are so enormous that their transmission and storage requires considerably large bandwidth and huge storage capacity. This is where image compression comes into play. The process of image compression is intended to provide a compact representation of an image. That is, to reduce the transmission and storage requirements by reducing the size of the data without losing any significant visual information. Almost all the digital images have some kind of redundancy in its data, it may be repeating pattern or repeating pixel values across the image. Reducing or eliminating one or more of these redundancies results in image compression. The three types of data redundancies exploited in image compression are: Inter Pixel Redundancy (Spatial redundancy), Coding Redundancy and Psycho Visual Redundancy. The **Inter pixel redundancy** also known as spatial redundancy assumes that the neighboring pixel values can predict the value of any given pixel and that the pixel values

are highly correlated. The image is said to have **coding redundancy**, if more codes are used than absolute necessary ones needed to represent each color. In **Psychovisual redundancy;** the limitation of human eye, to distinguish all the different colors in the image, is exploited. That is, every pixel value or luminance value in the digital image can't be comprehended by human perception.

An image compression system consists of a coder (compressor) and decoder (decompressor) as shown in **Figure 1**. Basically, image compression can be of two types: lossy and lossless compression. In lossy image compression, original image (I) is first compressed and during the compression phase some of the less important data is neglected. So, when the data is decompressed after transmission, we get an image (I') which is a closer approximation of the original image. Examples of lossy compression techniques include JPEG, Fractal compression, Transform coding and Vector quantization. While on the other hand, in lossless image compression no data or visual information is lost during the compression phase and an exact replica (I') of the original image (I) is retrieved after decompression. Examples of lossless compression techniques include: Run Length Encoding (RLE), Huffman Encoding and Delta Encoding.

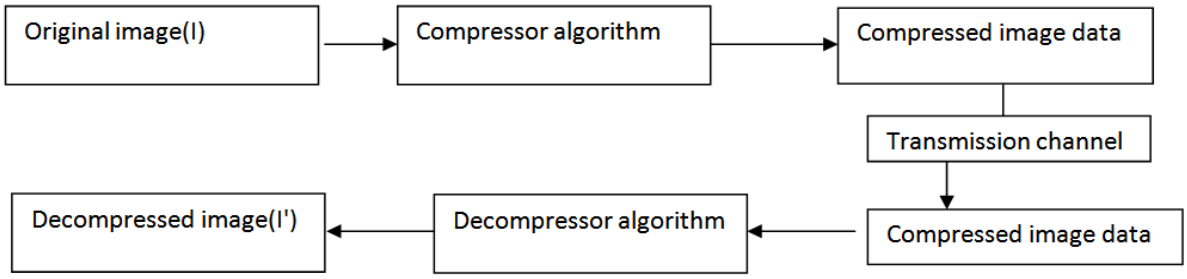


Figure 1: Image Compression Process

1.1.2. Discrete Cosine Transform (DCT)

Aiming at the digital image compression, Discrete Cosine Transform is a widely used transform to explore the frequencies present in a digital image. It was originally proposed by Ahmed et al [4]. DCT is mostly used either as a single algorithm or in combination with other compressors in multistage compressors like JPEG [3]. The DCT works by separating images into parts of differing frequencies. It is during the quantization step, where the compression occurs, the less significant frequencies (high frequency DCT Coefficients) are discarded and only the more important frequencies (low frequency DCT coefficients) are retained. The information about the pixels of the images is stored in these DCT coefficients. This quantization results in the reduced representation of the image hence compression is

achieved. The image reconstructed from this reduced frequency set is an approximation of the original image and hence it results in lossy image compression.

The DCT has the capability to represent most of visually substantial data of an image in just few coefficients. The DCT coefficients can be classified into "DC" and "AC" coefficients. The coefficient placed at the top left corner is the DC coefficient having zero frequency and gives the mean intensity of an image. All the remaining coefficients are known as AC coefficients and have non-zero frequencies. The coefficients organized in a zigzag manner (shown in **Figure 2**) carries information in the declining order. That is, the DC coefficient contains the most significant visual information about an image and is of high importance and the last AC coefficient traversed, following the zigzag pattern, have negligible visual information and is of least importance. Therefore, to accomplish compression; the DCT coefficients that are of no importance and lies at the end of the block are truncated.

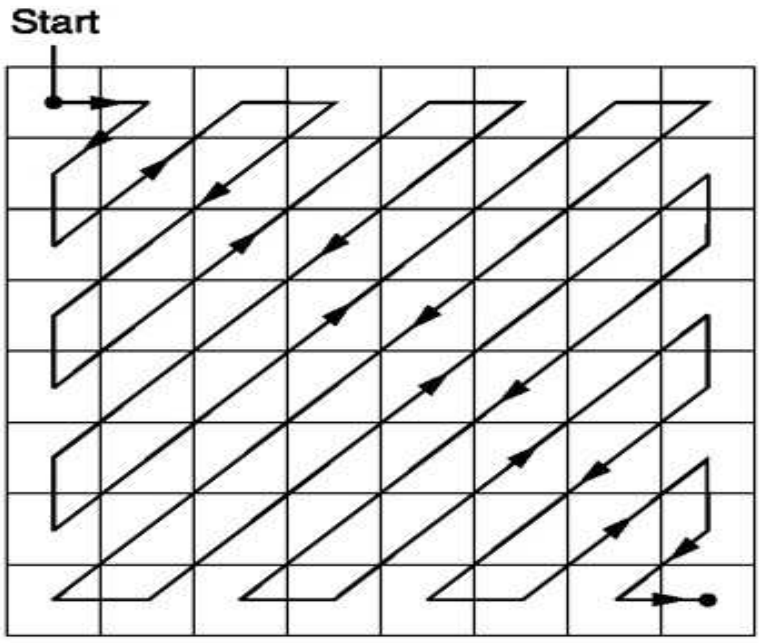


Figure 2: ZigZag Pattern

The DCT equation computing the u^{th} and v^{th} entry of the DCT of an image is shown in **Equation-1**. Since, cosine functions are used in the DCT, so the output matrix depends on the vertical, diagonal and horizontal frequencies. A sample image and its corresponding DCT image are shown in **Figure 3-4**.

$$f(u,v) = \left(\frac{2}{N}\right)^{-\frac{1}{2}} C(u)C(v) \sum_{u=0}^{N-1} \sum_{\mathrm{v}=0}^{\mathrm{N}-1} f(\mathrm{x},\mathrm{y}) \cos\left[\frac{(2\mathrm{x}+1)u\pi}{2\mathrm{N}}\right] \cos\left[\frac{(2\mathrm{y}+1)v\pi}{2\mathrm{N}}\right] \qquad (1)$$

Where C(u), C(v) = $\begin{cases} \frac{1}{\sqrt{2}} & if\ \ u,v = 0 \\ 1 & otherwise \end{cases}$ (2)

Figure 3: Pepper Image

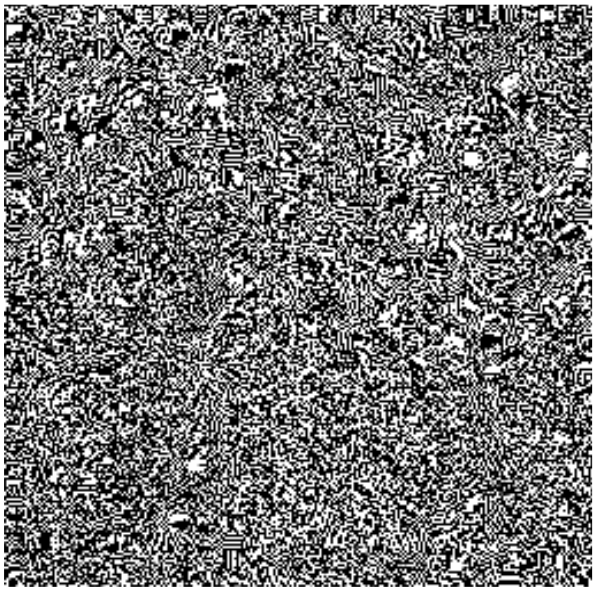


Figure 4: DCT of Pepper Image

When desired the reconstruction of the image through decompression is done using Inverse Discrete Cosine Transform (IDCT). The 2D IDCT of an image is given by **Equation-3**:

$$f(x,y) = \left(\frac{2}{N}\right)^{-\frac{1}{2}} C(x)C(y) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(u,v) \cos\left[\frac{(2u+1)x\pi}{2N}\right] \cos\left[\frac{(2v+1)y\pi}{2N}\right]$$

(3)

Where C(x), C(y) = $\begin{cases} \frac{1}{\sqrt{2}} & if\ \ x,y = 0 \\ 1 & otherwise \end{cases}$ (4)

1.1.3. **Neural Networks**

The inspiration for neural networks also known as artificial neural networks (ANNs) [2–4] comes from the way the calculations are performed and related decisions are being made by the human brain. We try to use artificial neurons to mimic the working of human nervous system. Biological neurons possess some computational ability and strength therefore; to imitate the actions of biological neurons each artificial neuron is also equipped with some computational strength. Many input signals can be taken in by an artificial neuron just like biological neuron and then a single output signal is produced based on an internal weighting system. An artificial neuron is depicted figuratively in **Figure 5**.

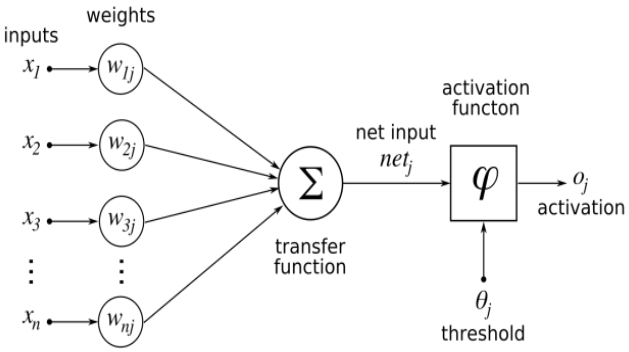


Figure 5: Structure of Artificial Neural Network

Here$X1$, $X2$, . . . ,Xn represents the n number of input signals which are multiplied with the respective connected weights shown in circles before reaching the transfer function represented by sigma. The weights are denoted by $W1$, $W2$, . . . , Wn. The output of the artificial neuron is generated by adding the product of all these inputs and weights and sending them to an activation function. ANN setup is created by connecting all these artificial neurons with each other similar to that of biological neurons which are part of our human nervous system.

With the advancement of neural network's structure and with further enhancement in the processing power, capabilities and strength of CPU; many researchers are attracted from numerous scientific fields of study to explore neural networks and their abilities to solve several difficulties that are encountered in the different fields of clustering, pattern recognition, function approx, prediction, categorization, optimization, and many more. The applications of neural networks include text classification and categorization, image recognition, name entity recognition, speech recognition and many more. Typical problems of using a NN include higher training time and higher computational complexity. Moreover, the black box nature is also a well known problem of neural networks. Another problem related to neural networks is the need for large and label dataset.

The neural networks tend to have good performance in non-linear capacity [2]. It has been proved that the multi-layered neural network can accurately approximate to any linear or nonlinear function. Neural network can be utilized to approximate to the continuous quantity, because any continuous quantity can be express by the combinations of linear and nonlinear function. Other than conventional algorithms [13] many ANNs are also being explored, developed and used to handle the problem we were facing during the compression of still images [5–12]. The goal was to design and build systems that utilize least possible

computational resources and attain high compression ratios without compromising the quality of reproduced images.

Deep Neural Networks are neural networks with several interconnected hidden layers between output and input layers as shown in **Figure** 6. Nonlinear relationships which are very complex can easily be modeled by these DNNs very efficiently. The high-level abstract depiction of the data can be easily learnt by DNNs and they can also extract highly variant features of inputs with the help of these extra layers of the DNNs. Convolutional Neural Networks (CNN) is a type of DNN that are specifically designed to deal with the visual data such as images and videos. CNNs are very similar to other DNN; they have neurons, weights, biases, activation functions just like any ordinary DNN except that the CNN makes this assumption that the inputs to these networks are images. This is the reason why CNN do better than traditional NN because the Convolutional layers in CNN utilizes the inherent features of images. The applications of CNNs include but are not limited to Facial Recognition, image recognition, video analysis, drug discovery and semantic parsing (NLP). The major drawback of using CNN is that they are computationally more expensive than many classic neural networks and needs a huge amount of image data for its training.

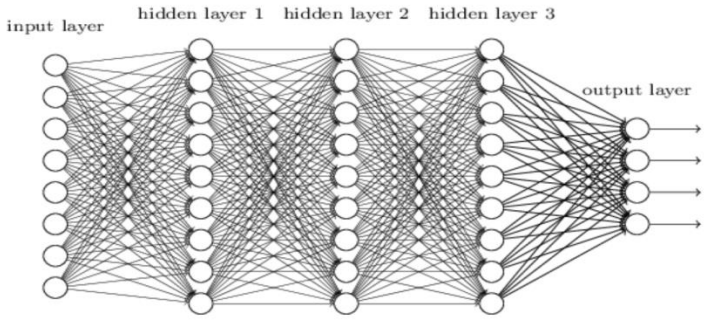


Figure 6: Structure of Deep Neural Network

1.2. Classification Problem

In Machine Learning (ML), classification is a process in which the machine learns to predict or identify the classes (labels) of unseen observations, based on the labeled input given to it, using some learning algorithm. The classification process is shown in **Figure 7**. Examples of classification include: handwriting recognition, Prediction of Chronic diseases in patients, Speech recognition, Stock Market Forecasting, Sentiment Analysis. A number of

learning algorithms has been introduced and used for the classification task. Different algorithms perform well in different scenarios based on the nature of data available and depending on the application. Classification algorithms commonly used in machine learning are Naïve Bayes classifier, Support Vector Machines (SVM), Random Forest, K- Nearest Neighbor, Decision Trees, Apriori Algorithm and last but not the least Artificial Neural Networks. Here we are going to discuss classification via neural networks and why it has performed impressively in real world applications. The three main reasons that set apart neural networks from the rest of the classification algorithms are: its ability to perform well on continuous valued inputs and outputs, its high tolerance to noisy data or outliers and its ability to deal with complex relations between variables. Moreover, its powerful tuning features helps in preventing over-fitting and under-fitting as well.

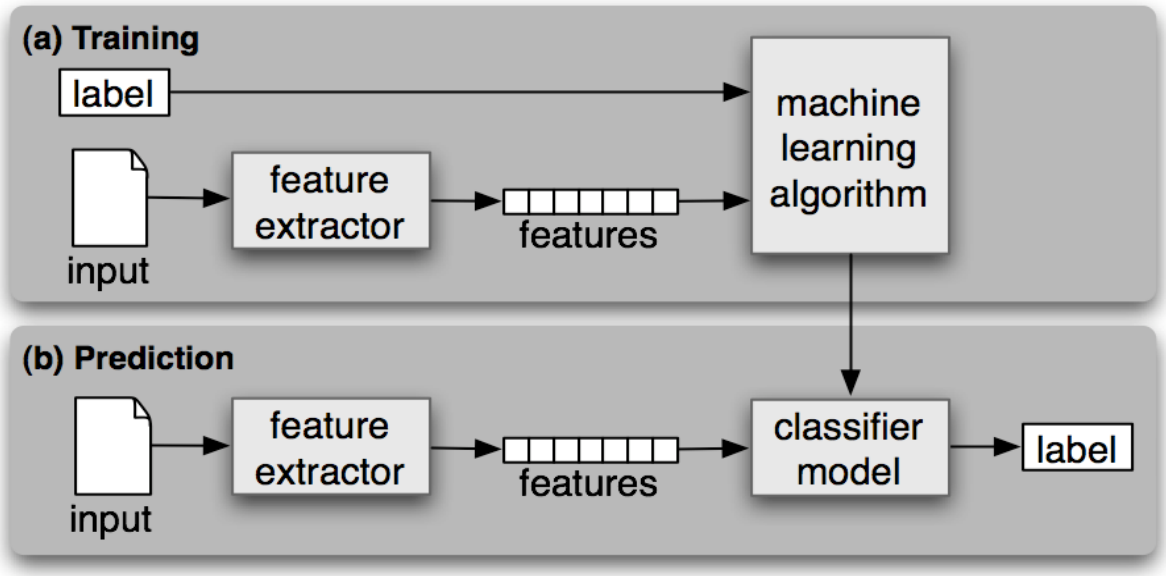


Figure 7: Classification Process

1.3. Proposed Methodology

In this thesis, we have targeted the problem of lossy image compression and proposed a deep neural network based solution for it. We have developed two types of deep neural networks; Multi layered Perceptron (MLP) and Convolutional Neural Network (MLP). These neural networks tend to achieve image compression by extracting most important DCT coefficients of images. The inputs to these networks are gray scale images whereas; the outputs from these models are important DCT coefficients. Different compression rates are achieved by varying the number of most significant DCT coefficients at the output layer. A

flow diagram which depicts the proposed methodology for the extraction of most important DCT coefficients from images is shown in **Figure 8**.

We have also tried to target the classification of images via neural networks using these reduced representations. We have explored the existing literature to highlight how neural networks are increasingly being examined and considered as possible solutions to problems and how the concept of neural network is shaping up the area of image compression.

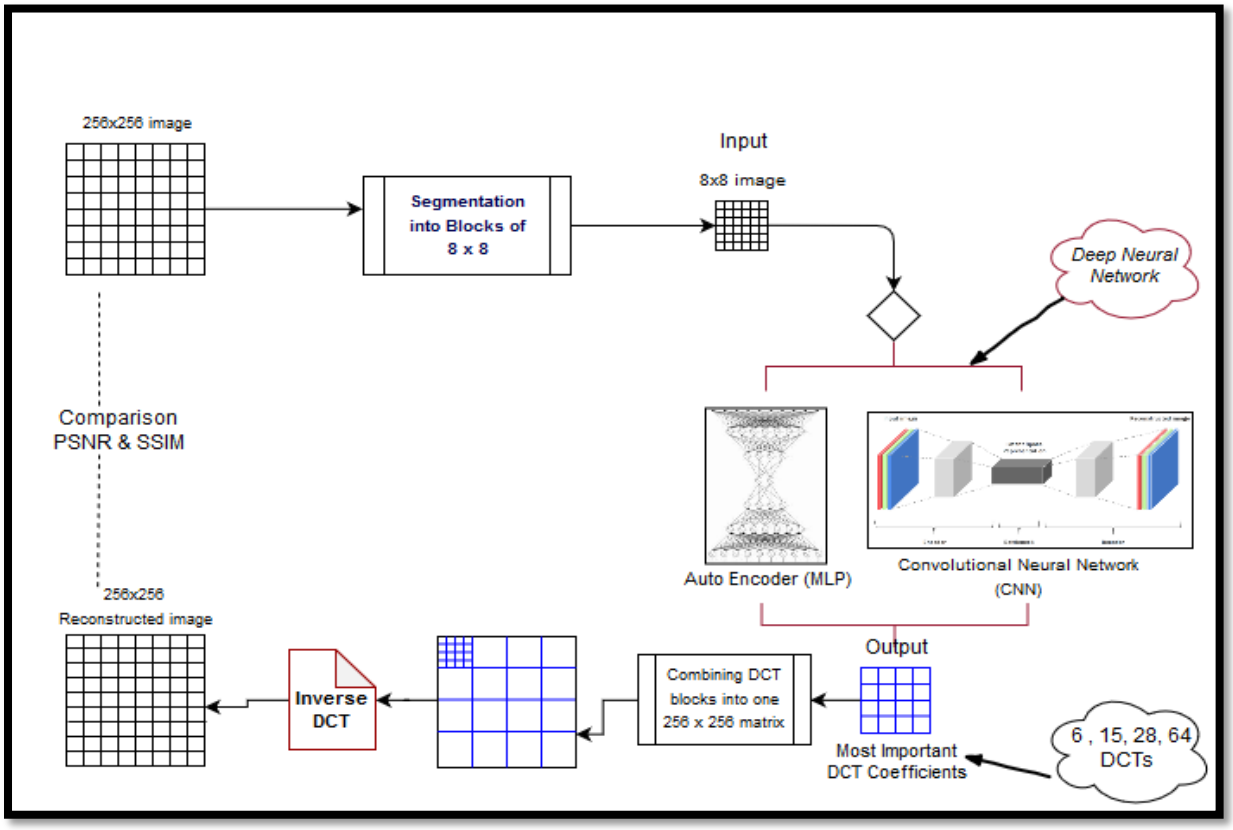


Figure 8: Proposed Methodology for the Extraction of Important DCT Coefficients

1.4. Research Contribution

The major contributions made by this research are as follow:

- Extracting the most important DCT coefficients via Deep Neural Networks
- Using these most important DCT coefficients for Digital Image compression
- Using the most important DCCT Coefficients for the classification of standard datasets
- We have developed Multi-Layered Perceptron and Deep Convolutional Neural Network that can predict the most important DCT coefficients of images.

We have assessed the performance of our proposed work mainly by using 3 state of the art image quality quantification methods; “Peak Signal to Noise Ratio (PSNR), Structural Similarity Index (SSIM) and Mean Squared Error (MSE)”, on 3 standard gray scale images; Tank, Baboon and Lena.

1.5. Thesis Organization

This Thesis document contains 5 chapters. Chapter 1 presents the problem statement, related background, Classification Problem, Proposed Methodology, Research Contribution and Thesis organization. In Chapter 2 provides detailed literature review highlighting the application of neural networks in the area of image compression. Chapter 3 includes the implementation details and experimental results of the proposed deep neural networks for the extraction of most important DCT Coefficients. Chapter 4 discusses image classification via reduced representation. In the final chapter the summary of our work is presented and a discussion on the future work is carried out.

Chapter 2

Literature Review

CHAPTER 2: LITERATURE REVIEW

This chapter presents research work conducted in the area of Image compression using neural networks. After a brief literature review of work conducted in this area we enlightened the research gaps that we found in previous works.

2.1. Related Work

Huge streams of images are expected to be produced daily in the emerging field of digital imaging. The size of these images can be an issue considering two particular scenarios: considering transmission time or required bandwidth capability and storage capacity required by these huge streams of visual data. Since, most of the social media platforms such as; Facebook, Instagram and Whatsapp etc mainly rely on video/image storage and transmission. As we know that the area of image compression has been researched for decades and significant advances are achieved in the form of reducing storage cost and transmission time. But the continuous improvement in the quality of digital images and the resulting rise in the sizes of images have led researchers to continuously consider the possibility of further reducing the size of digital images for better communication and storage. Even if an image can be compressed into half of its original size without compromising the image quality then, transmission time will be improved by 100% and the storage cost would be reduced by 50%. One thing to be kept in mind, that in order to successfully attain image compression the quality of the image should not be compromised.

Numerous researches have been carried out to discover and tackle the image compression problems using NN. Many image processing problems have been tackled using Neural Networks [1]. The first ever use of Neural Networks for targeting image compression problem has been recorded in the following papers [2, 3].Image compression techniques employed using neural networks are presented in [2, 4].

The basic and the most essential NN that is based on a single structure model have been elaborated in [5]. It contains a logistic transfer function and has 3-layers used to attain image compression. In [6-8] different parallel architectures have been suggested for image compression through neural networks. The basic goal is compression of different fragments of the images (i.e. as per complexity factor defined) using several neural networks which are placed in parallel manner to achieve the maximum compression ratio and quality of remodeled

image. In [9] to enhance the quality of image compression, authors have suggested an approach which works with single structured network and uses novel normalization function. In [10] an ANN has been explained that shed light on calculation of discrete cosine transform which is used in compression of images. A comparison of various neural network architectures has been provided for image compression for instance vector quantization that make up for a neural network in [11, 12].

Recently, competitive performances have been achieved using deep learning for image compression; lossless and lossy compression. A general framework was proposed by Toderici et al [13] for achieving lossy compression in images. The architecture was based on de-Convolutional Long Short Term Memory and Convolutional networks. Moreover, another neural network was also proposed by Toderici [14] that has competitive performance in compressing images of arbitrary sizes. Compressive auto-encoders were proposed by Theis et al. [15]. These auto-encoders for continuous relaxation, upper-bounds the discrete entropy rate and uses a smooth approximation of the rounding function. For replacing rounding quantization with additive uniform noise and for joint non-linearity, GDN was used by Ball et al [16]. A compression method based on content weight-age was proposed by Li et al. [17]. Moreover, state-of-the-art results were achieved by lossless compression methods proposed by van den Oord [19] and Theis [18].

Compressing images using Neural Networks can be categorized into three approaches. The first approach is to train an NN to achieve image compression by using relatively limited number of neurons in the hidden layers. The unprocessed image pixels are used as input to these NN and the output of these Neural Networks form the compressed image based on the output from hidden neurons and associated weights. The compression ratio (CR) is the ratio between the size of the original and compressed image. And the compression error is the difference between the decompressed and original image. Neural Networks using the first mechanism but with different alterations were presented in [20–28]. Second, self-organizing map (SOM) networks can be used to achieve image compression as reported in [4, 29–31]. “It is based on selecting a limited number of code words to approximate the distribution of the input vectors of the original image. The index of these code words produces the compression”. The third and final approach to achieve image compression via Neural Networks is by estimating the coefficients of image transformation such as discrete cosine transform (DCT) and wavelet transform [3, 32–34].

In this thesis, the final approach is followed but a new methodology is proposed. A lot of research has been performed for approximation of compression/decompression components of images in which internal data representations have been generated using ANN. Numerous training images are used as input (input/output sets) to train these networks, the goal is the approximation of conforming compression decompression algorithm efficiently, and structure it in a way so that it can be used for large dataset for testing. With the assistance of learning algorithm rigorous training has been performed to attain this goal. The progression of training is categorized by the comparison of the given output and the predicted output and adjusting all the weights contributing to this comparison. In this research work backward propagation has been used to train ANN. Back propagation can define as supervised learning mechanism which is best fit for networks having a feed-forward mechanism [35]. Feed-forward network mechanism represents a Perceptron network with multi-layers in which outputs generated from neurons passed to the following except for prior layer, due to which information flow is unidirectional and no responding feedback loops. Back propagation phenomenon refers to "back propagation for errors" which means that the error generated on output nodes are circulated backward toward inner nodes to attain the desired output. Hence error gradient with the assistance of adjustable weights is calculated through back propagation. Gradient descent mechanism uses this gradient to adjust weights and to reduce the error rate. Thus, back propagation is a procedure which is responsible of computation of gradient and their adjustment to gradient descent algorithm.

The aim of this thesis is to develop a deep neural network that can estimate the most important DCT coefficients of images and utilize these reduced representations of images for the classification of images. Moreover, we also tend to reduce the storage cost and attain better decompressed image quality.

Chapter 3

Implementation and Results

CHAPTER 3: IMPLEMENTATION & RESULTS

3.1. Introduction

In this chapter, we have presented two deep neural networks architectures that are employed for extracting the most important DCT Coefficients of gray scale images. The two types of DNN used are: Multi-Layered Perceptron (MLP) and Deep Convolutional Network (CNN). These most significant DCT coefficients are used as a reduced representation of the image and hence compression is achieved. We also propose two architectures respectively that are able to regenerate approximations of the original images from these reduced representations

3.2. MLP for Extracting the Most Important DCT Coefficients

Multi-Layered Perceptron (MLP) is one of the two types of deep neural networks used for the extraction of DCT coefficients. Four different MLP models are designed and trained to achieve different compression ratios of 1/2, 1/4, 1/8 and another to achieve non-compressed results. Each of the four MLPs consists of six hidden layers, the first three layers acts as an encoder, whereas, the later three layers acts as a decoder. The output layer consists of a varying number of neurons. The architecture of the four MLPs used is shown in **Figure 9**.

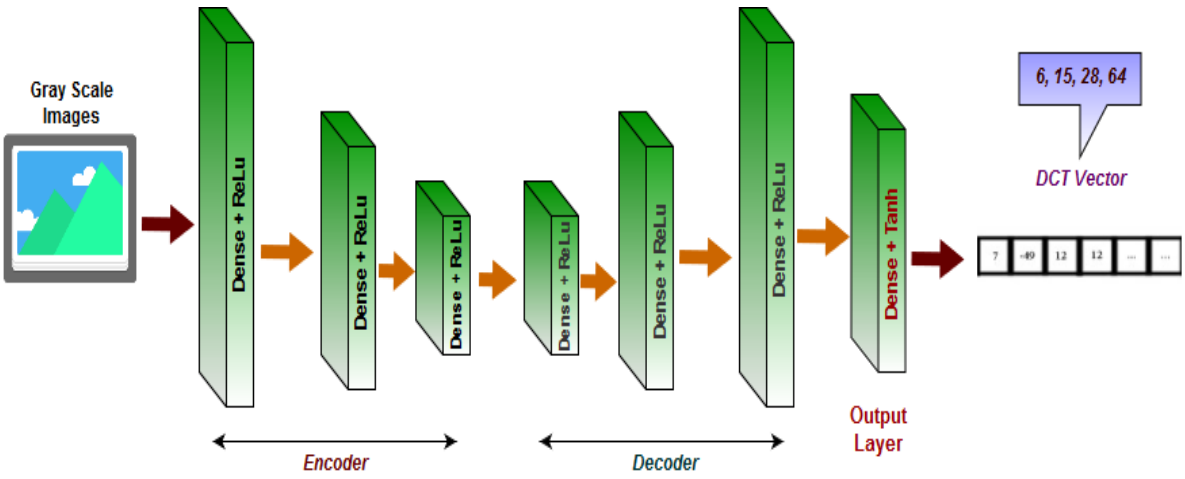


Figure 9: The architecture of the proposed MLPs

The dataset consisting of 30 gray scale images has been used for the training of MLP models. These images were divided into blocks of 8 x 8 and then the neural network was trained on these blocks of images. These blocks statistically co-related and hence can be

extended to large number of images. The pixel value of the images falls in the range between 0 and 255.In order to validate the proposed technique, the dataset has been divided into two parts i.e. training and testing, 80% of the data has been used for the former purpose and 20% for the latter. A normalizing function *f1* is applied to normalize the pixels values of the images between the range of 0 and 1. The pixel values for a grey scale image are usually between 0 and 255. Since, the desired output of the network is always known in case of supervised learning. So, in our case the most important DCT coefficients are used as labels. Moreover, a normalizing function *f2* is applied to normalize the values of DCT coefficients between the range of -1 and 1. DNN's input layer is fed with these normalized blocks of original images; each pixel is corresponded by a separate neuron in input layer of DNN; that is, the total number of neurons in the input layer is 8 x 8 (64). The output layer predicts the value of the most important DCT coefficients. Four different estimates of most significant DCT coefficients i.e. 6, 15, 28 and 64, were used as a label for the given training set of images. The stopping criterion for the neural networks is 500 epochs. With the help of inverse normalization function $f2^{-1}$ we can restore the dynamic and wide range of predicted DCT coefficients.

3.2.1. Activation Functions

In order to understand what activation functions do, we need to understand the functionality of an artificial neuron. Neurons calculate the weighted sum of their inputs adds a bias to it and make the decision of whether they should be fired or not. The value of Y can range from +inf to – inf, and the neuron cannot bound the value. Then how is the decision of whether to fire or not made? This is where activation functions come into play. There are numerous kinds of activation functions, namely: linear function, a sigmoid function, *tanh* function, and ReLU etc. However, the selection of these activation functions varies from case to case. In our case, we have used two different activation functions i.e. ReLU and tanh.

3.2.1.1 Rectified Linear Units or ReLU

ReLU makes sure that the output does not become a negative value. So, when *z* is greater than zero the output stays *z*, and if it does below zero the output stays zero. Moreover, it is also used when there are numerous output possibilities. In our case, ReLU is used for going from the input layer to the hidden layer.

$$(z) = \max(0, z)$$

What gives ReLU an edge is that it does not activate the entire set of neurons at once. E.g. when a negative input is received it will be converted into zero, and the neuron will not get activated. In other words, at a given point in time a few neurons will be active and hence ReLU helps in making the artificial neural network sparse and hence increases its efficiency.

3.2.1.2 Tanh Function

Tanh function is used to bind the output in a range (-1, 1). In our case, we use it in the output layer to predict the normalized values of DCT coefficients.

$$\mathbf{f(z) = tanh(z).}$$

We find the hyperbolic tangent of *z* and return it. The tanh function is used when we have regression problem at hand, i.e. just like our example where we to predict the normalized values of DCT coefficients. Since the values of DCT in frequency domain ranges from positive to negative that is why we opted for tanh at the output layer.

3.2.2. Optimization

Usually, forward propagation and backward propagation techniques are used as error functions or weight optimization. But in our case, we use the Adaptive moment estimation technique. However, we use the following technique for optimization.

3.2.2.1 Adam

Since 2014, a special optimization algorithm in the shape of Adam (Adaptive Moment Estimation) for deep neural networks is present. Adam is one of the best methods that are used to calculate adaptive learning rates for every parameter. It computes the adaptive learning rates for all the parameters. Apart from storing the exponentially decaying averages of previously squared gradients, for instance, RMSporp and Adadelta, it also keeps something similar to momentum. If momentum is thought of like a ball going down a slope, Adam can be termed as a heavy ball having friction and hence providing us with flat minima.

3.2.3. Loss Function

In machine learning, ‘loss functions’ are a group of objective functions that are minimized. A loss function determines how good the results of the prediction models are and how close the predicted output and actual output is. The target for the neural networks was to produce coefficients with little error.

3.2.3.1 Mean Absolute Error

"Mean Absolute Error (MAE) is a loss function used for regression models. MAE is the sum of absolute differences between our desired and actual variables. So it measures the average magnitude of errors in a set of predictions, without considering their directions". The range is also 0 to ∞. We have used MAE here instead of mean squared error, Since MAE is more robust to outliers and a single high value can not affect the results.

$$MAE = \frac{\sum_{i=1}^{n} |y_i - y^p{}_i|}{n}$$

3.2.4. Models' Accuracies

Since our proposed methodology is tackling a regression problem. MAE metrics is used for measuring the accuracy and loss of the models i.e. the lower the mean absolute error between the targeted and actual results the higher the accuracy and the lower the loss. The epoch vs mean absolute error graphs for the four models have been presented in the **Figures 10-13**. It is clear from the graphs, that MLP model with compression ratio of 1/2 has lower mean absolute error of 0.0014 and 0.00085 on both train and test data respectively, as compared to other three Models.

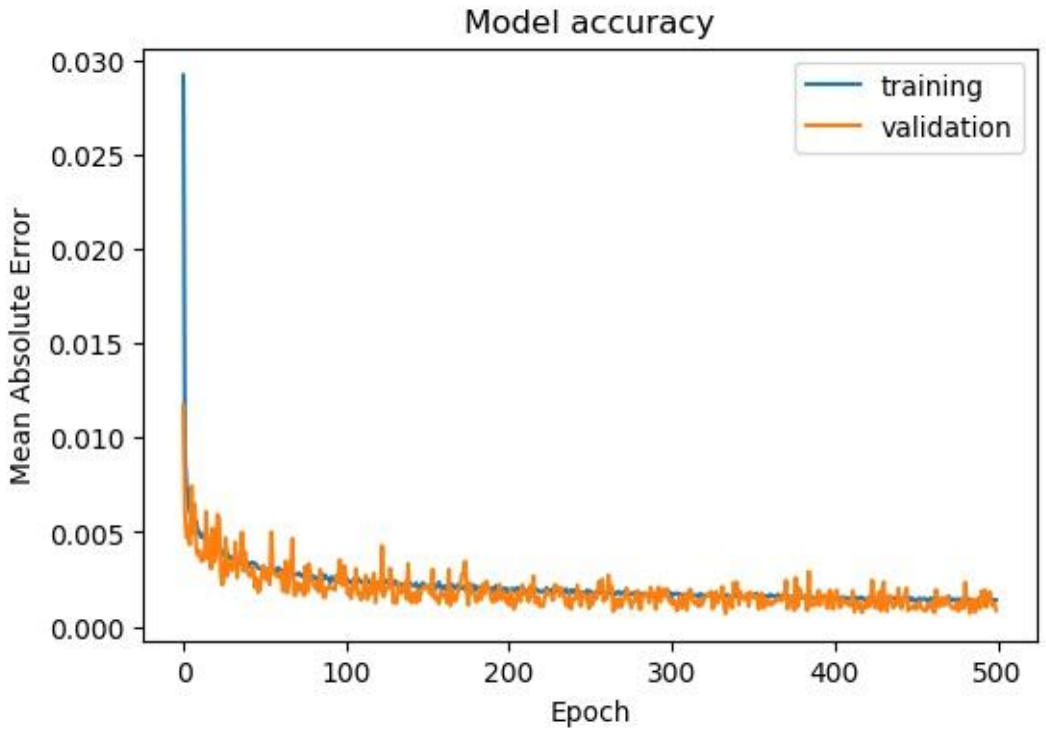


Figure-10: MAE vs Epoch for MLP model estimating 10% DCT Coefficients

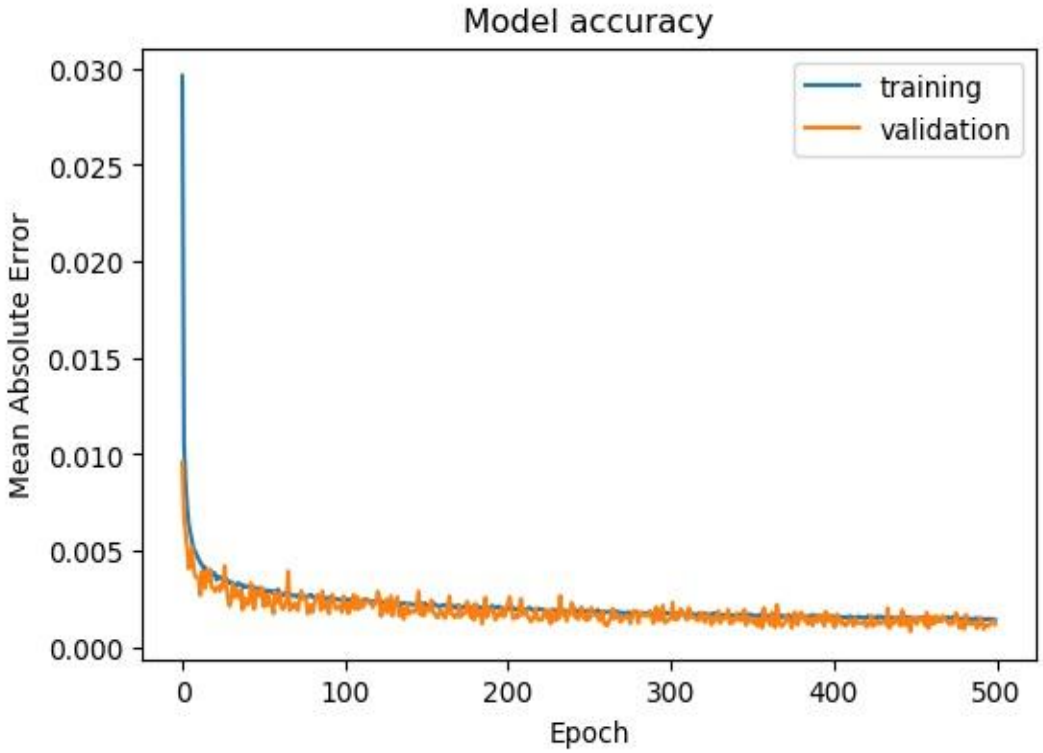


Figure-11: MAE vs Epoch for MLP model estimating 25% DCT Coefficients

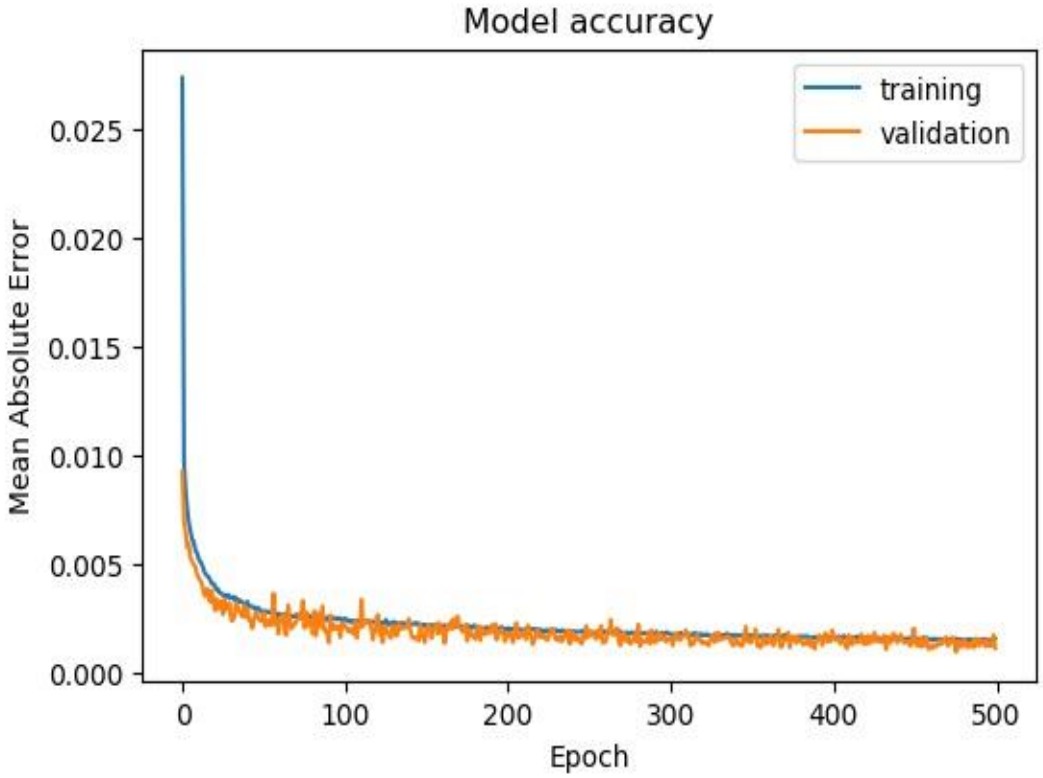


Figure-12: MAE vs Epoch for MLP model estimating 45% DCT Coefficients

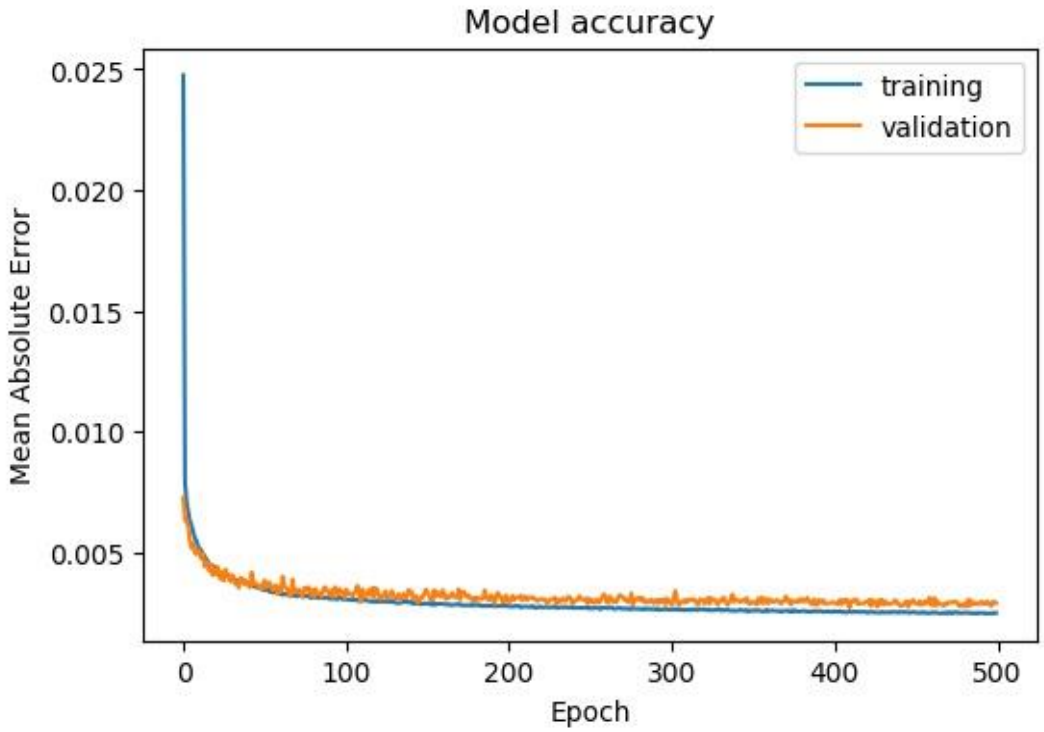


Figure-13: MAE vs Epoch for MLP model estimating 100% DCT Coefficients

3.2.5. Performance and Evaluation Criterion

The results from the above 4 MLP models were validated for image quality assessment. The results of the above DNN models i.e. Most important DCT coefficients, at the output layer were de-normalized first using an inverse normalization function $f2^{-1}$. Then the image data was transformed back from frequency domain to spatial domain using inverse DCT function (idct2).

The quality of the reconstructed image is then measured using three state of the art methods PSNR (Peak Signal to Noise Ratio), SSIM (Structural Similarity Index) and IMMSE (Mean Squared Error). "The PSNR is used here to measure the error between original and reconstructed image". PSNR is given by:

$$PSNR = 10\log_{10}\frac{255^2}{MSE}dB$$

"The SSIM is used to measure the structural similarity between reconstructed and original image". It is given by:

$$SSIM(x,y) = \frac{(2\mu_x\mu_y+C_1)(2\sigma_{xy}+C_2)}{(\mu_x^2+\mu_y^2)(\sigma_x^2+\sigma_y^2+C_2)}$$

The value of SSIM lies between 0 and 1, where 1 represents identical images. The IMMSE is used to calculate the mean squared error between reconstructed and original image and is given by:

$$\mathrm{MSE} = \frac{1}{N}\sum_{i=1}^{N}(\mathrm{x_{i-}}\ \mathrm{x'_i})^2$$

Table 1-3 depicts the SSIM, PSNR and IMMSE achieved by a standard gray scale images Tank, Baboon and Lena respectively, compressed at different ratios by MLP models. The decompressed images are evaluated for subjective quality. **Figures 14-17** show original images and their respective reconstructed images at different CRs.

Table 1: Quality quantification of reconstructed image resulting from MLPs at different CR (Tank)

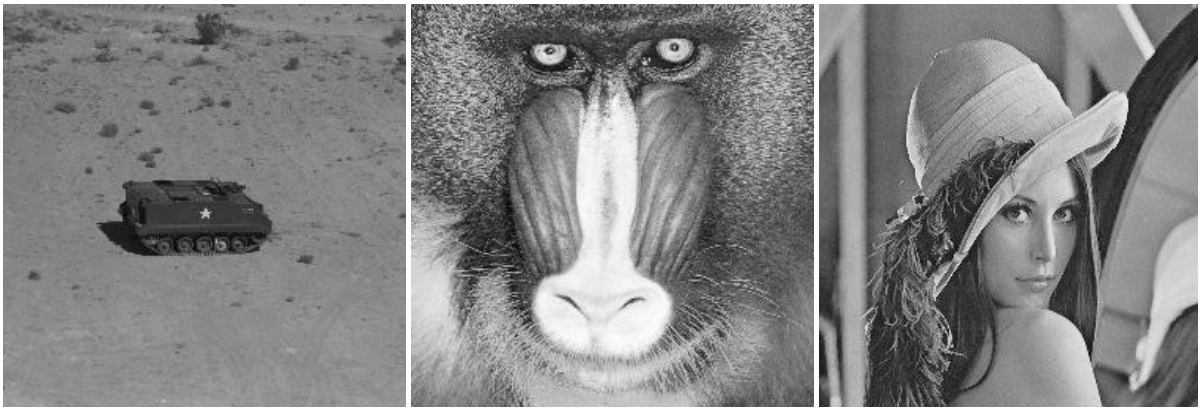
MLP Models			
DCT Coefficients	**PSNR**	**SSIM**	**IMMSE**
10%	30.039	0.713	64.44
25%	31.958	0.807	41.43
45%	34.033	0.876	25.69
100%	34.004	0.874	25.86

Table 2: Quality quantification of reconstructed image resulting from MLPs at different CR (Baboon)

MLP Models			
DCT Coefficients	**PSNR**	**SSIM**	**IMMSE**
10%	27.122	0.594	91.14
25%	28.402	0.720	83.94
45%	30.107	0.826	63.45
100%	31.328	0.835	60.30

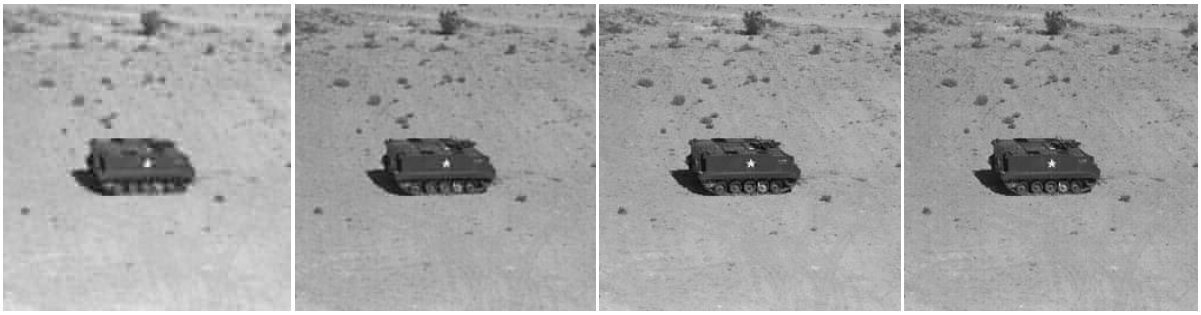
Table 3: Quality quantification of reconstructed image resulting from MLPs at different CR (Lena)

MLP Models			
DCT Coefficients	**PSNR**	**SSIM**	**IMMSE**
10%	29.085	0.771	66.92
25%	30.490	0.871	53.90
45%	32.776	0.916	34.48
100%	33.012	0.924	27.84



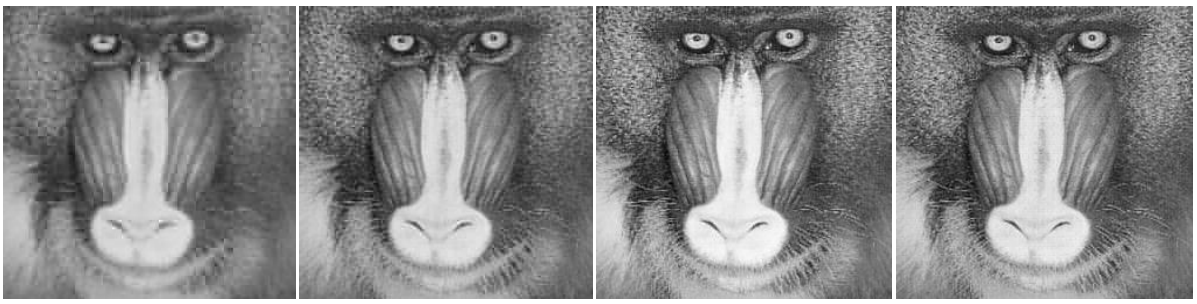
i ii iii

Figure 14: (i) Actual Image Tank (ii) Actual Image Baboon (iii) Actual Image Lena



i ii iii iv

Figure 15: Tank Image (i) Reconstructed Image From the 10% DCT coefficients estimated by MLP Model (ii) Reconstructed Image From the 25% DCT coefficients estimated by MLP (iii) Reconstructed Image From the 45% DCT coefficients estimated by MLP (iv) Reconstructed Image From the 100% DCT coefficients estimated by MLP



i ii iii iv

Figure 16: Baboon Image (i) Reconstructed Image From the 10% DCT coefficients estimated by MLP Model (ii) Reconstructed Image From the 25% DCT coefficients estimated by MLP (iii) Reconstructed Image From the 45% DCT coefficients estimated by MLP (iv) Reconstructed Image From the 100% DCT coefficients estimated by MLP

i ii iii iv

Figure 17: Lena Image (i) Reconstructed Image From the 10% DCT coefficients estimated by MLP Model (ii) Reconstructed Image From the 25% DCT coefficients estimated by MLP (iii) Reconstructed Image From the 45% DCT coefficients estimated by MLP (iv) Reconstructed Image From the 100% DCT coefficients estimated by MLP

3.3. CNN for Extracting the Most Important DCT Coefficients

The other type of deep neural network used for image compression in our proposed framework is Convolutional Neural Network (CNN). To get compression rates of 1/2, 1/4, 1/8 and a non-compressed output, four different deep CNN are developed and trained. Here also the CNNs developed, consisted of six hidden layers, working as auto-encoder neural networks. Moreover, the output layer neurons are adjusted accordingly to get higher and higher compression rates. The deep Convolutional Neural Networks' architecture, developed for image compression, is shown in **Figure 18**. The dataset consisting of 30 gray scale images has been used for the training of MLP models. These images were divided into blocks of 8 x 8 and then the neural network was trained on these blocks of images.

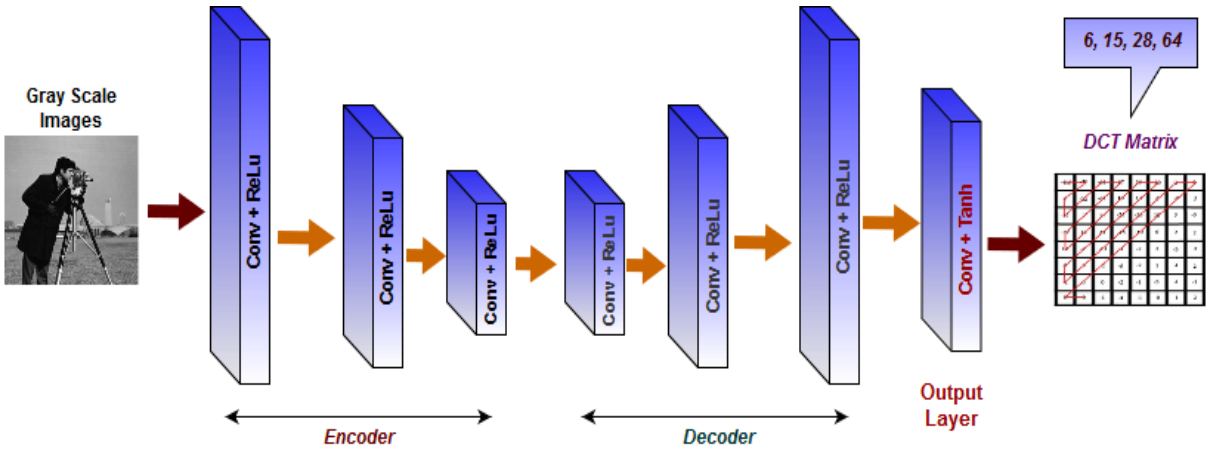


Figure 18: The architecture of the proposed DCNNs

These blocks are statistically co-related and hence can be extended to large number of images. The pixel value of the images falls in the range between 0 and 255.In order to validate the proposed technique, the dataset has been divided into two parts i.e. training and testing, 80% of the data has been used for the former purpose and 20% for the latter. A normalizing function *f1* is applied to normalize the pixels values of the images between the range of 0 and 1. The pixel values for a grey scale image are usually between 0 and 255. In our case the most important DCT coefficients are used as labels. Moreover, a normalizing function *f2* is applied to normalize the values of DCT coefficients between the range of -1 and 1. DNN's input layer is fed with these normalized blocks of original images; each pixel is corresponded by a separate neuron in input layer of DNN; that is, the total number of neurons in the input layer are 8 x 8 (64). The output layer predicts the value of the most important DCT coefficients. Four different estimates of most significant DCT coefficients i.e. 6 (3x3), 15 (5x5), 28 (7x7) and 64 (8x8), were used as a label for the given training set of images. The stopping criterion for the neural networks is 500 epochs. With the help of inverse normalization function $f2^{-1}$ we can restore the dynamic and wide range of predicted DCT coefficients.

3.3.1. Activation Functions

In our case, we have used two different activation functions i.e. ReLU and tanh.

3.3.1.1 Rectified Linear Units or ReLU

ReLU makes sure that the output does not become a negative value. So, when *z* is greater than zero the output stays *z*, and if it does below zero the output stays zero. Moreover, it is also used when there are numerous output possibilities.

In our case, ReLU is used for going from the input layer to the hidden layer.

$$(z) = \max(0, z)$$

What gives ReLU an edge is that it does not activate the entire set of neurons at once. E.g. when a negative input is received it will be converted into zero, and the neuron will not get activated. In other words, at a given point in time a few neurons will be active and hence ReLU helps in making the artificial neural network sparse and hence increases its efficiency.

3.3.1.2 Tanh Function

Tanh function is used to bind the output in a range (-1, 1). In our case, we use it in the output layer to predict the normalized values of DCT coefficients.

$$f(z) = tanh(z).$$

We find the hyperbolic tangent of *z* and return it. The tanh function is used when we have regression problem at hand, i.e. just like our example where we to predict the normalized values of DCT coefficients. Since the values of DCT in frequency domain ranges from positive to negative that is why we opted for tanh at the output layer.

3.3.2. Optimization

In our case, we use the Adaptive moment estimation technique.

3.3.2.1 Adam

Since 2014, a special optimization algorithm in the shape of Adam (Adaptive Moment Estimation) for deep neural networks is present. Adam is one of the best methods that are used to calculate adaptive learning rates for every parameter. It computes the adaptive learning rates for all the parameters. Apart from storing the exponentially decaying averages of previously squared gradients, for instance, RMSporp and Adadelta, it also keeps something similar to momentum. If momentum is thought of like a ball going down a slope, Adam can be termed as a heavy ball having friction and hence providing us with flat minima.

3.3.3. Loss Function

"A loss function is a measure of how good a prediction model does in terms of being able to predict the expected outcome". Since, the target for the neural networks was to produce DCT coefficients with little error. So in our case, MAE is used as loss function.

3.3.3.1 Mean Absolute Error

We have used MAE here instead of mean squared error, Since MAE is more robust to outliers and a single high value can not affect the results.

$$MAE = \frac{\sum_{i=1}^{n} |y_i - y^p{}_i|}{n}$$

3.3.4. Models' Accuracies

Four DCNN models were designed and tested to achieve different compression rates. The four different CNN models estimates different number of DCT coefficients at the output layer. The epoch vs mean absolute error graphs for the four models have been presented in the **Figures 19-22**. It is clear from the graphs, that the DCNN model with compression ratio of 1/2 has lower mean absolute error of 0.0050 and 0.0040 on both train and test data respectively, as compared to other three Models.

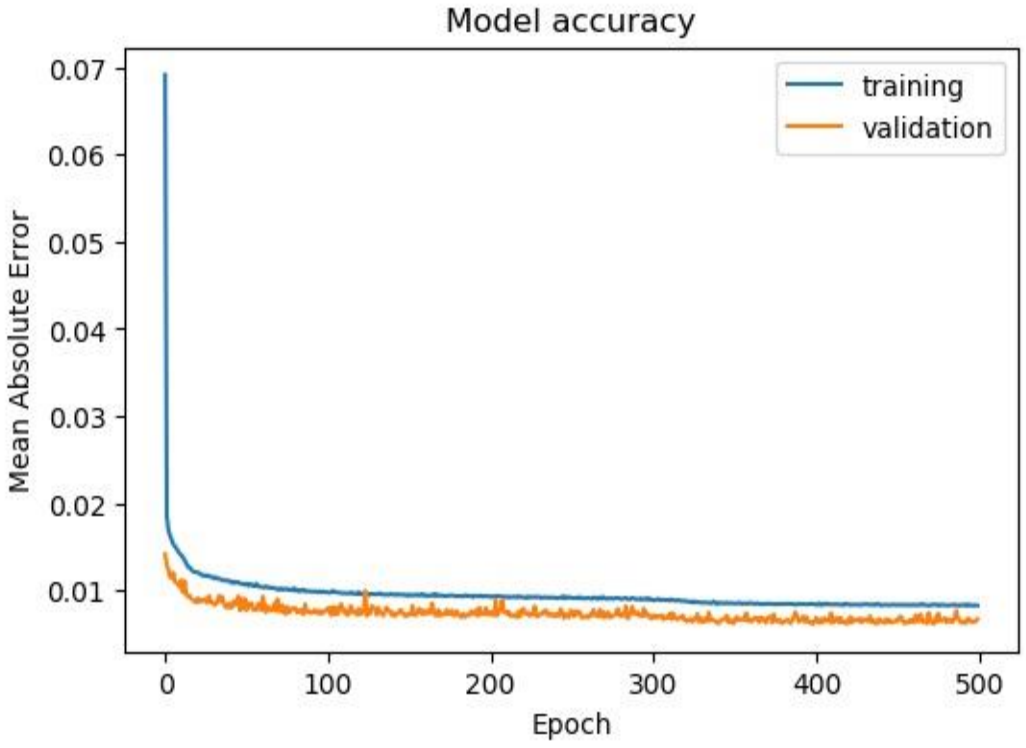


Figure 19: Mean Absolute Error vs Epoch for DCNN model estimating 10% DCT Coefficients

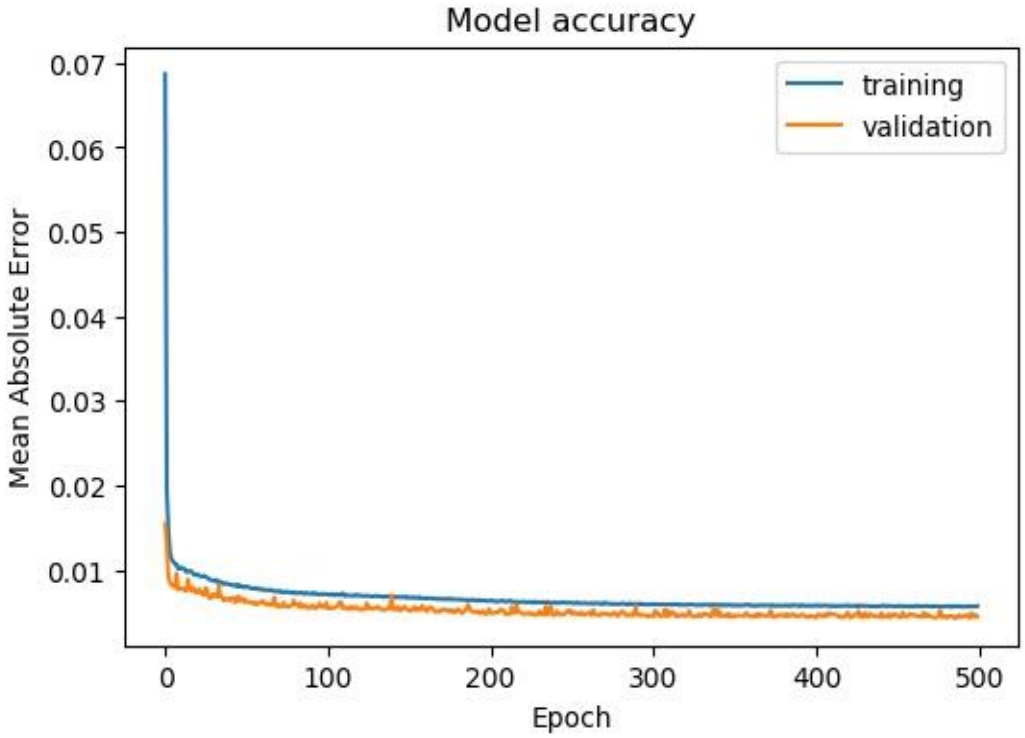


Figure 20: Mean Absolute Error vs Epoch for DCNN model estimating 25% DCT Coefficients

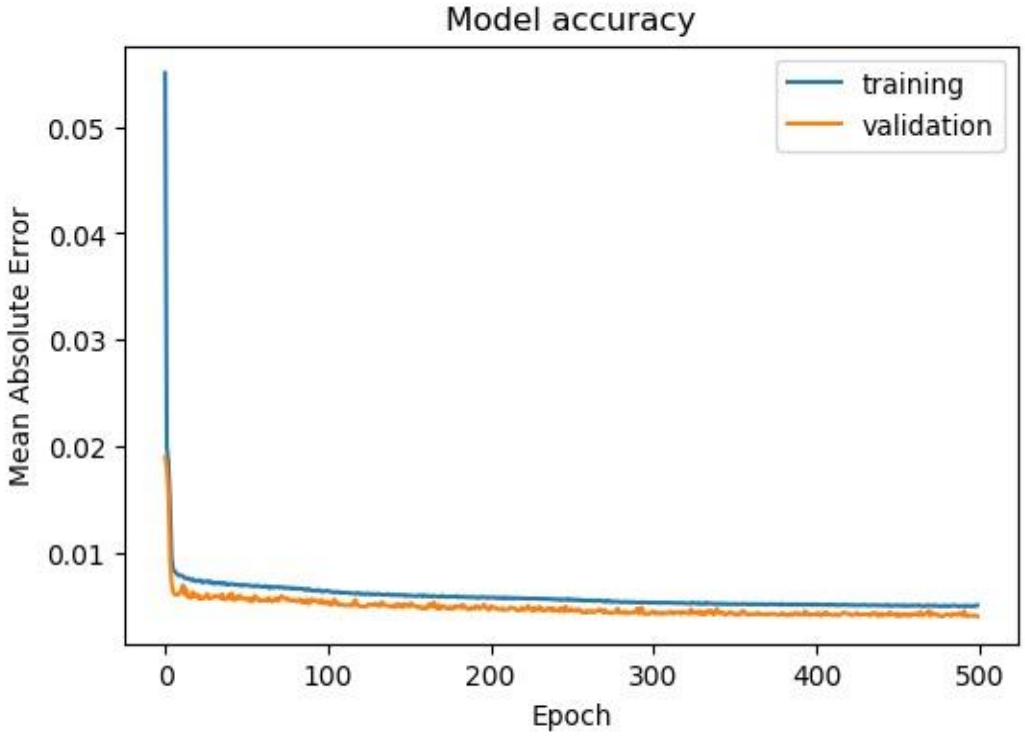


Figure 21: Mean Absolute Error vs Epoch for DCNN model estimating 45% DCT Coefficients

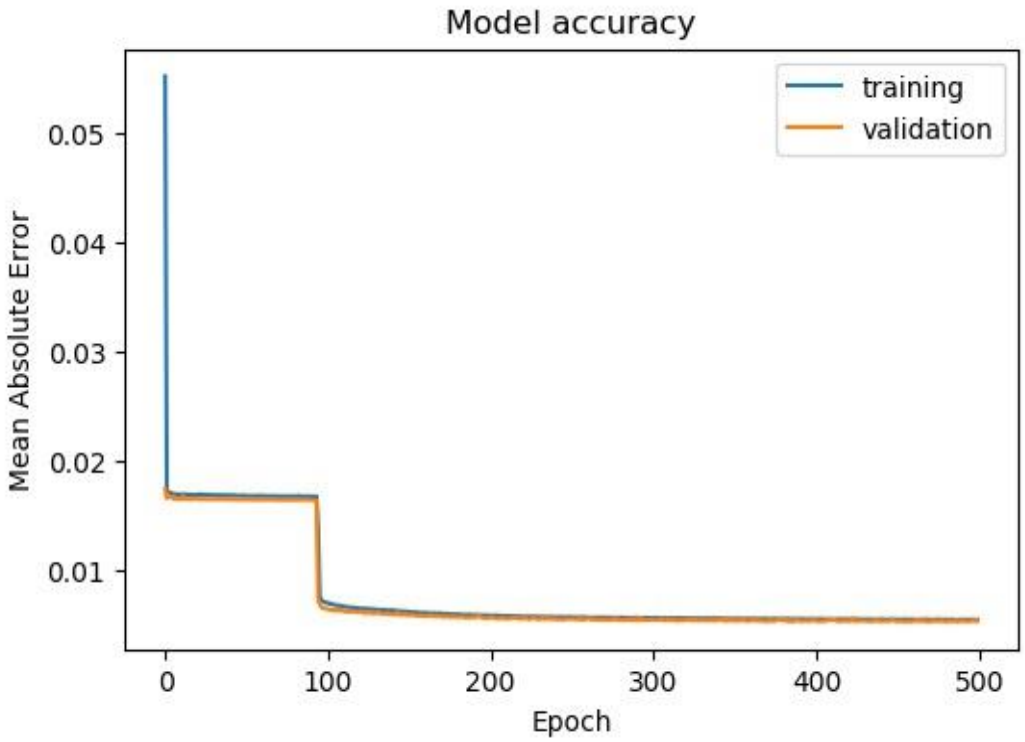


Figure 22: Mean Absolute Error vs Epoch for DCNN model estimating 100% DCT Coefficients

3.3.5. Performance and Evaluation Criterion

Since the main goal of the proposed methodology is to achieve high compression rates using deep neural networks without compromising the quality of the images. So, the results from the above 4 DCNN models were assessed for image quality. The results of the above DCNN models i.e. DCT coefficients, at the output layer were de-normalized first using an inverse normalization function $f2^{-1}$. Then the image data was then transformed back from frequency domain to spatial domain using inverse DCT function (idct2). The quality of the reconstructed image is then measured using three state of the art methods PSNR (Peak Signal to Noise Ratio), SSIM (Structural Similarity Index) and IMMSE (Mean Squared Error).

Table 4-6 depicts the SSIM, PSNR and IMMSE achieved by a standard gray scale images Tank, Baboon and Lena respectively, compressed at different ratios by MLP models. The decompressed images are evaluated for subjective quality. **Figures 23-26** show original images and their respective reconstructed images.

Table 4: Quality quantification of reconstructed image resulting from DCNNs at different CR (Tank)

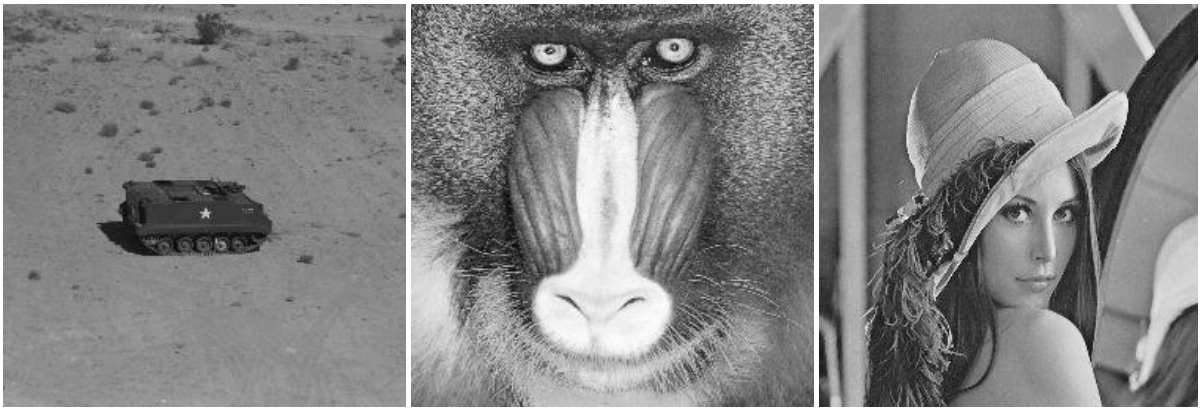
DCNN Models			
DCT Coefficients	**PSNR**	**SSIM**	**IMMSE**
10%	28.817	0.667	85.39
25%	29.755	0.718	68.79
45%	29.461	0.702	73.61
100%	29.338	0.692	75.73

Table 5: Quality quantification of reconstructed image resulting from DCNNs at different CR (Baboon)

DCNN Models			
DCT Coefficients	**PSNR**	**SSIM**	**IMMSE**
10%	26.362	0.540	150.29
25%	26.926	0.606	131.98
45%	26.853	0.591	134.19
100%	26.863	0.594	133.89

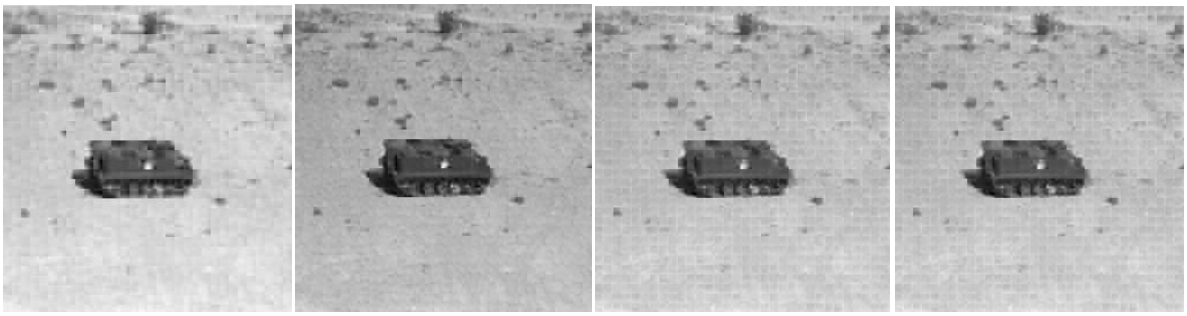
Table 6: Quality quantification of reconstructed image resulting from DCNNs at different CR (Lena)

DCNN Models			
DCT Coefficients	**PSNR**	**SSIM**	**IMMSE**
10%	25.485	0.709	94.44
25%	26.691	0.775	82.81
45%	28.105	0.749	76.70
100%	29.278	0.744	79.40



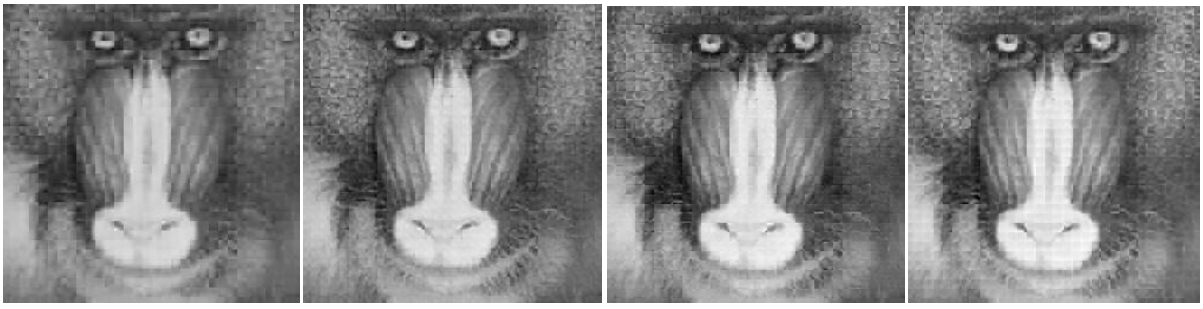
i ii iii

Figure 23: (i) Actual Image Tank (ii) Actual Image Baboon (iii) Actual Image Lena



i ii iii iv

Figure 24: Tank Image (i) Reconstructed Image From the 10% DCT coefficients estimated by DCNN Model (ii) Reconstructed Image From the 25% DCT coefficients estimated by DCNN (iii) Reconstructed Image From the 45% DCT coefficients estimated by DCNN (iv) Reconstructed Image From the 100% DCT coefficients estimated by DCNN



i ii iii iv

Figure 25: Baboon Image (i) Reconstructed Image From the 10% DCT coefficients estimated by DCNN Model (ii) Reconstructed Image From the 25% DCT coefficients estimated by DCNN (iii) Reconstructed Image From the 45% DCT coefficients estimated by DCNN (iv) Reconstructed Image From the 100% DCT coefficients estimated by DCNN

i ii iii iv

Figure 26: Lena Image (i) Reconstructed Image From the 10% DCT coefficients estimated by DCNN Model (ii) Reconstructed Image From the 25% DCT coefficients estimated by DCNN (iii) Reconstructed Image From the 45% DCT coefficients estimated by DCNN (iv) Reconstructed Image From the 100% DCT coefficients estimated by DCNN

3.4. Analysis

The results prove that the MLP models are relatively better than DCNN models for image compression. The lowest Test MAE achieved among MLP models is '0.00085'. Whereas, the lowest test MAE achieved among DCNN models is '0.0040'. Moreover, the MAE values of other MLP models are also lower than the DCNN. This means that the error between actual and predicted results is low in MLP models. On the other hand, the quality of the image tends to decrease as we increase the compression ratio in both types of DNN. But the quality of the images reconstructed from the output of the MLP models is also significantly better than the DCNN models. The highest PSNR and SSIM values observed in DNN models were '34.03' and '0.876' respectively, whereas, the highest PSNR and SSIM values recorded in DCNN models were '29.46' and '0.718'. This indicates that although both DNNs gives good results for image compression, but MLPs are more suited for image compression. It was also observed that the quality of images starts declining as the compression rate exceeds a limit of 2:1. The higher the compression ratio the lower will be the quality of the image.

3.5. Summary

This In this chapter, the architectures of the DNN models were presented and the results produced by the proposed DNNs have been presented in terms of accuracy and Mean absolute error. Different graphs have also been provided to assist the reader with a visual illustration of the performance of the different models. Moreover, the quality of the resulting output is quantified with the help of different methods and a comparative analysis of the quality of the different models is carried out.

Chapter 4

Classification via the Most Important DCT Coefficients

CHAPTER 4: CLASSIFICATION VIA MOST IMPORTANT DCT COEFFICIENTS

4.1. Introduction

In this chapter, the application of the proposed methodology has been discussed. The chapter also discusses the use of compressed images for classification purpose. The famous MNIST image dataset is first compressed using the proposed DNN models (Tailored to MNIST dataset). The compressed data is then used for digits recognition using classifier. The chapter also presents the architecture of the two classifiers being used for the classification of digits.

4.2. Dataset

The famous MNIST (Modified National Institute of Standards and Technology database) image dataset consisting 70,000 gray scale images of handwritten digits has been used. The dimension of the images was 28 x 28. The pixel value of the images falls in the range between 0 and 255.

4.3. Validation

In order to validate the classification models, the dataset has been divided into two parts i.e. training and testing, 60,000 of the images has been used for the former purpose and 10,000 for the latter.

4.4. Components of Classifier

4.4.1. Activation Functions

In our case, we have used two different activation functions i.e. tanh and softmax. Tanh is used because the input was in the range (-1, 1). And as we need multiple probabilities at the output layer of the classifier to know which class has the highest probability. Softmax is used to squash the matrix into output probabilities that sum to one.

4.4.2. Optimizer

We have used SGD optimizer. Stochastic gradient descent also known as SGD optimizer is an iterative method for optimizing a differentiable objective function. SGD does away with

redundant computations by performing one update at a time, as compared to batch gradient. That is why they are much faster than other optimizers. Frequent updates are performed by SGD with a high variance that causes the objective function to fluctuate heavily.

4.4.3. Loss Function

Categorical cross entropy is used in our case since we are dealing with multi class classification task. "Cross-entropy loss, or log loss, measures the performance of a classification model whose output is a probability value between 0 and 1. Cross-entropy loss increases as the predicted probability diverges from the actual label. A perfect model would have a log loss of 0".

$$-\sum_{c=1}^{M} y_{o,c} log(p_{o,c})$$

"M - number of classes (dog, cat, fish) ,log - the natural log ,y - binary indicator (0 or 1) if class label *c* is the correct classification for observation *o*,p - predicted probability observation *o* is of class *c".*

4.5. Classifiers

For the classification purpose two types of classifiers were developed; MLP classifiers and CNN Classifiers. The MLP classifiers deal with the compressed MNIST data resulting from proposed MLP models and perform the digits recognition task on compressed images. On the other hand, the CNN classifiers deal with the compressed MNIST image data resulting from proposed DCNN models for image compression and carries out the digits classification task on the given data. Overall, the CNN classifiers perform better than the MLP classifiers for the given digits classification task and has better accuracies.

4.5.1. MLP Classifiers

Since, the outputs from the proposed MLP models are used as input to our classifiers. Four different estimates of DCT coefficients i.e. 15, 28, 45 and 66 out of total 784, were fed as an input to the four different MLP classifiers. Since the DCT coefficients to be used at the input layer were predicted by a MLP model, therefore, the input will be in a one dimensional form. The value of the DCT coefficients lies in the range from -1 to 1. And as we know that

the desired output is always known in case of supervised learning. So, in this case the actual labels of the MNIST dataset are used as a label for the given training set of compressed images. The range of the labels is from 0 to 9. The stopping criterion for the classifiers is 50 epochs with a batch size of '32'. The four classification models were designed and trained independently to assess the affect of different compression ratios on digits recognition. Each of the four MLP classifiers consists of 3 layers; input, hidden and output layer. The number of neurons at the output layer are 10 i.e. 10 classes. The architecture of the four MLP classifiers used is shown in **Figure 27**.

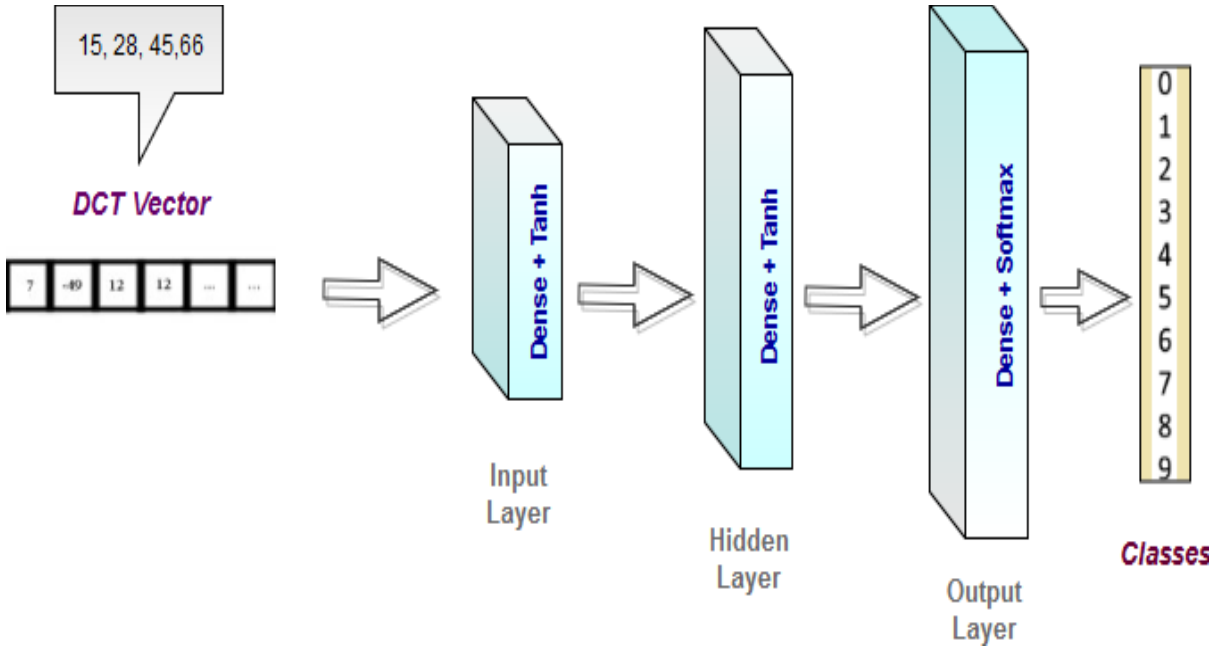


Figure 27: The architecture of the MLP Classifiers

4.5.1.1 Model Accuracies

The accuracy of the four MLP classifier models is shown in **Figures 28-31**. It is clear from the graphs, that the lower the compression of the input images the higher the accuracy of the classification model. The model with 66 DCT coefficients i.e. CR=1/12, at the input layer has a better classification rate of 88% among the four classification models.

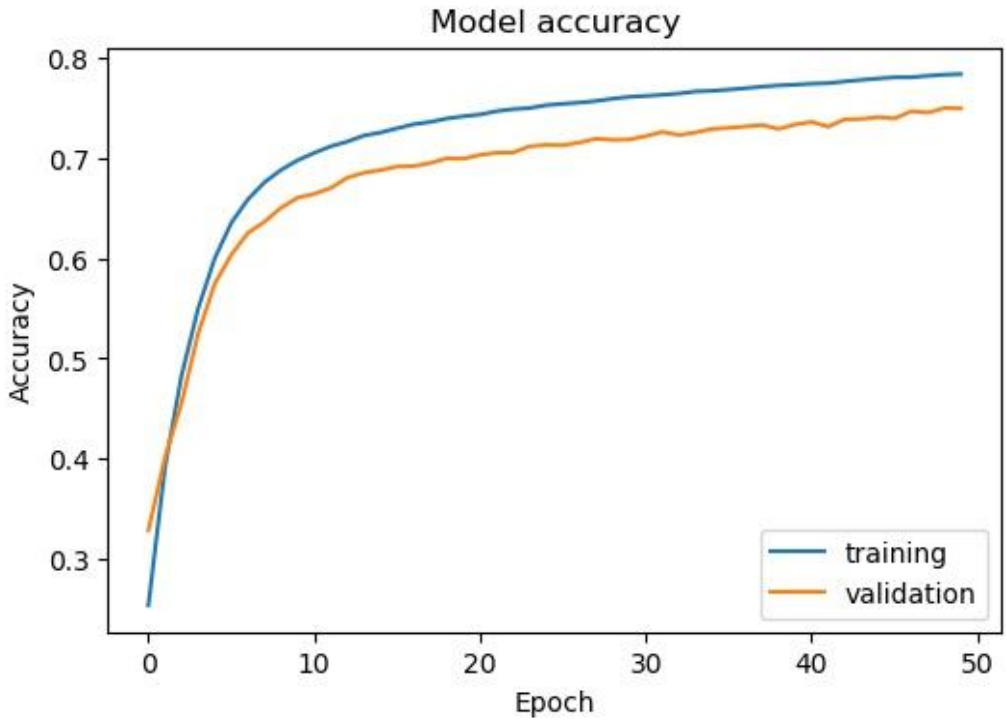


Figure 28: Accuracy vs Epoch for training and Validation data on MLP Classifier with 15 DCT Coefficients at the input Layer

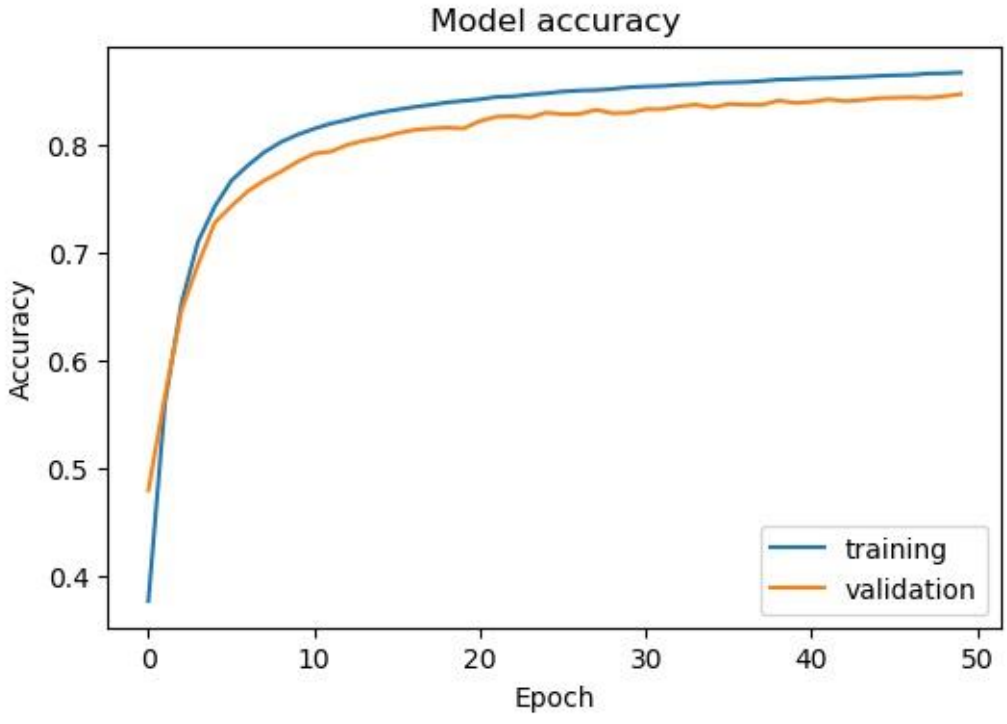


Figure 29: Accuracy vs Epoch for training and Validation data on MLP Classifier with 28 DCT Coefficients at the input Layer

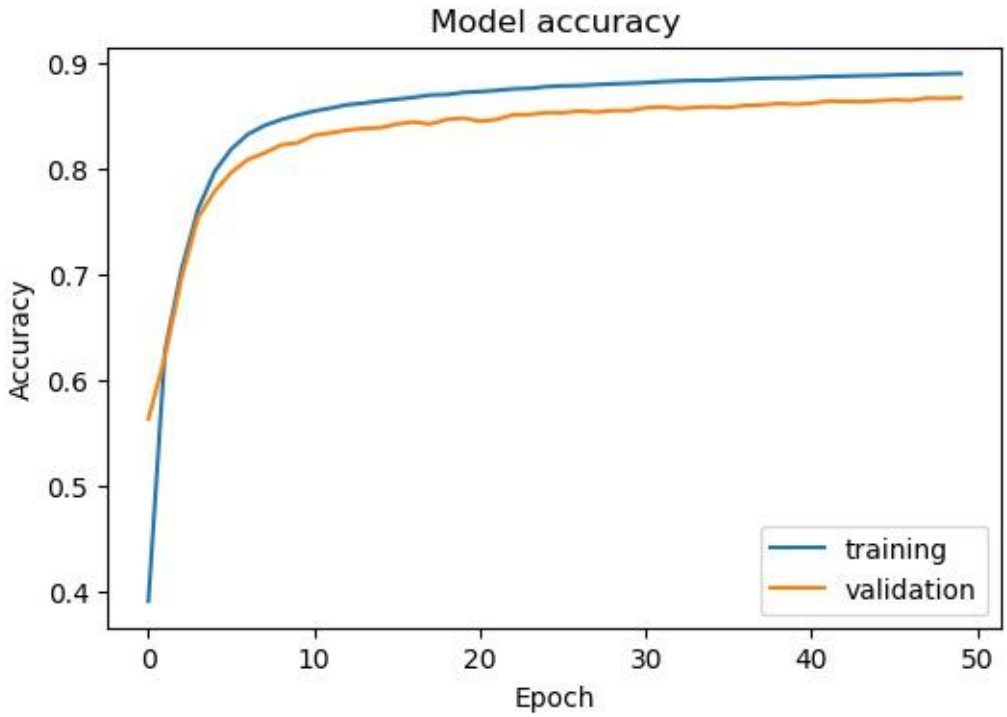


Figure 30: Accuracy vs Epoch for training and Validation data on MLP Classifier with 45 DCT Coefficients at the input Layer

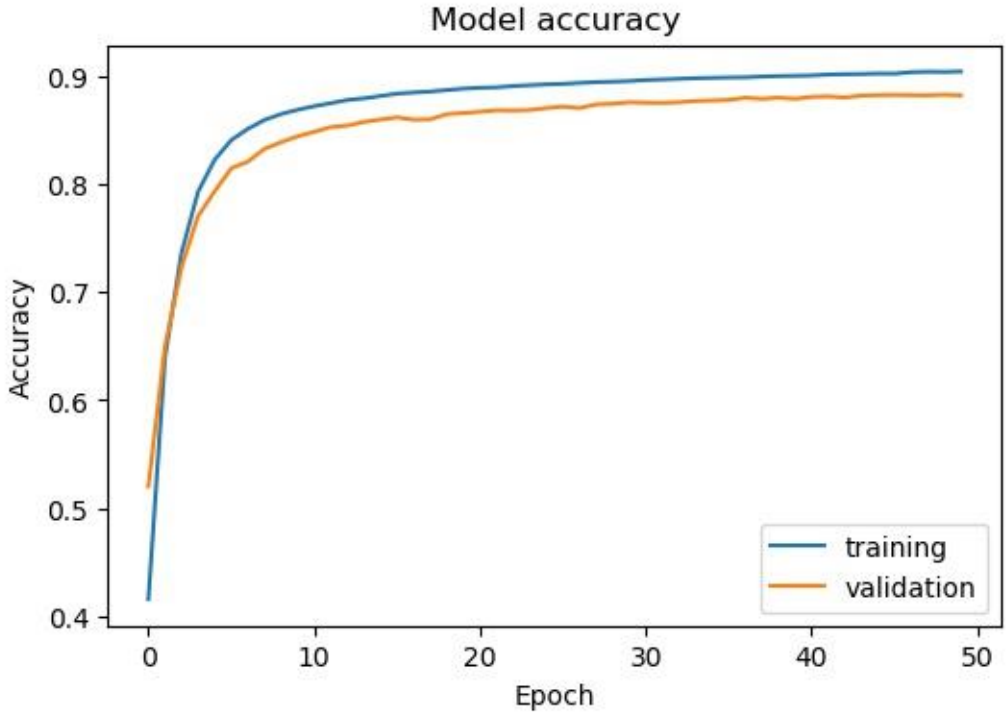


Figure 31: Accuracy vs Epoch for training and Validation data on MLP Classifier with 66 DCT Coefficients at the input Layer

4.5.2. CNN Classifier

Since, the outputs from the proposed DCNN models are used as input to our classifiers. Four different estimates of DCT coefficients i.e. 15, 28, 45 and 66 out of total 784, were fed as an input to the four different CNN classifiers. Since the DCT coefficients to be used at the input layer were predicted by a DCNN models, therefore, the input will be in a two dimensional form. The value of the DCT coefficients lies in the range from -1 to 1. And as we know that the desired output is always known in case of supervised learning. So, in this case the actual labels of the MNIST dataset are used as a label for the given training set of compressed images. The range of the labels is from 0 to 9. The stopping criterion for the classifiers is 50 epochs with a batch size of '32'. The four classification models were designed and trained independently to assess the affect of different compression ratios on digits recognition. Each of the four CNN classifiers consists of 5 layers; input layer, three hidden layers and output layer. The number of neurons at the output layer are 10 i.e. 10 classes. The architecture of the four CNN classifiers used is shown in **Figure 32**.

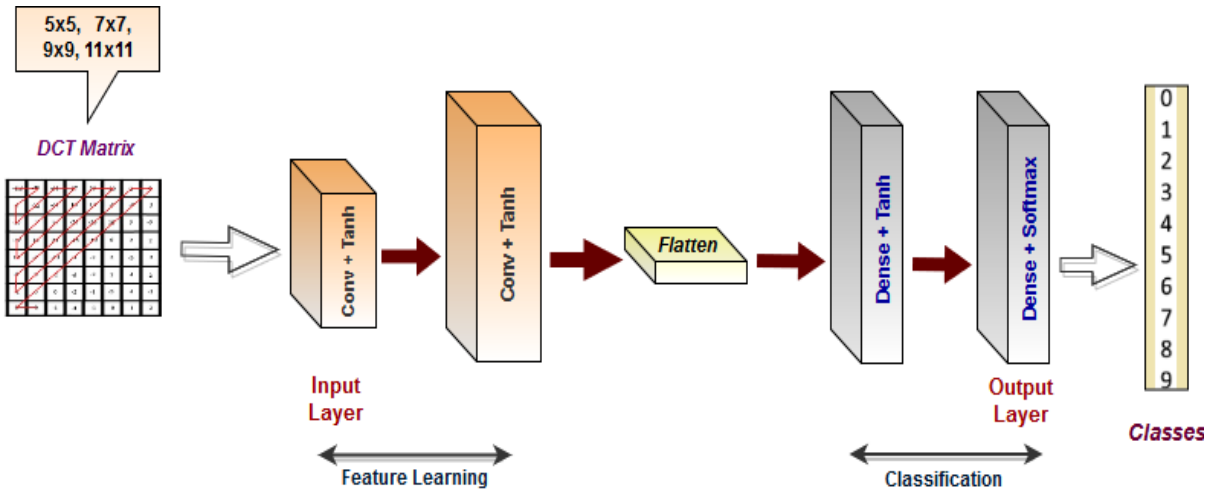


Figure 32: The architecture of the CNN Classifiers

4.5.2.1 Model Accuracies

The accuracy of the four CNN classification models is shown in **Figures 33-36**. The model with 45 DCT coefficients i.e. CR=1/17, at the input layer has a better classification rate of 95% among the four classification models.

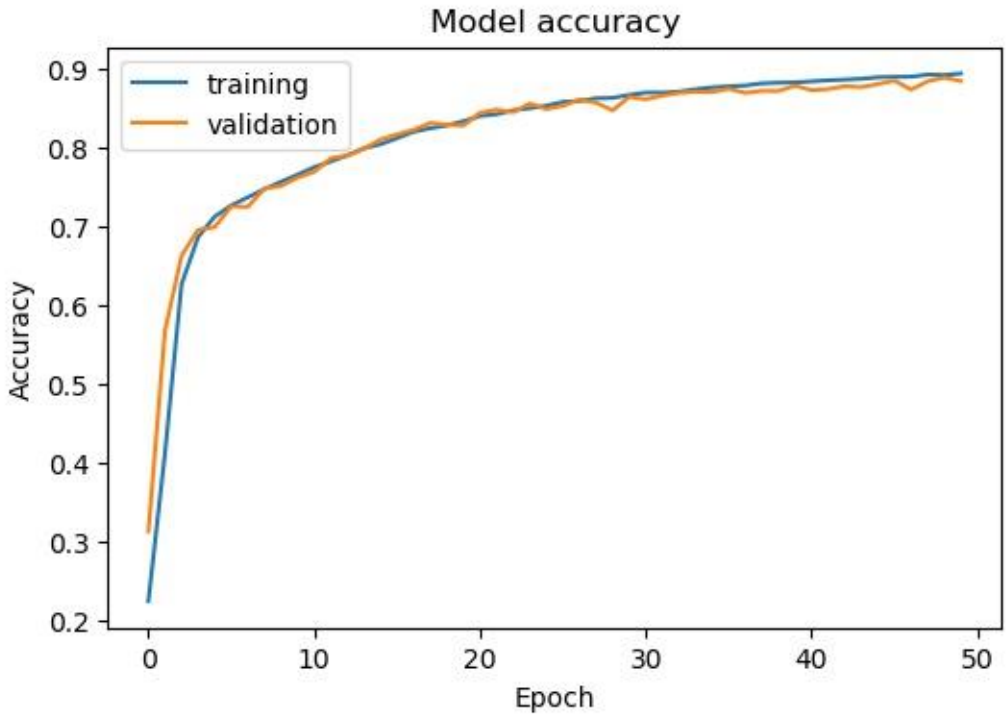


Figure 33: Accuracy vs Epoch for training and Validation data on CNN Classifier with 15 DCT Coefficients at the input Layer

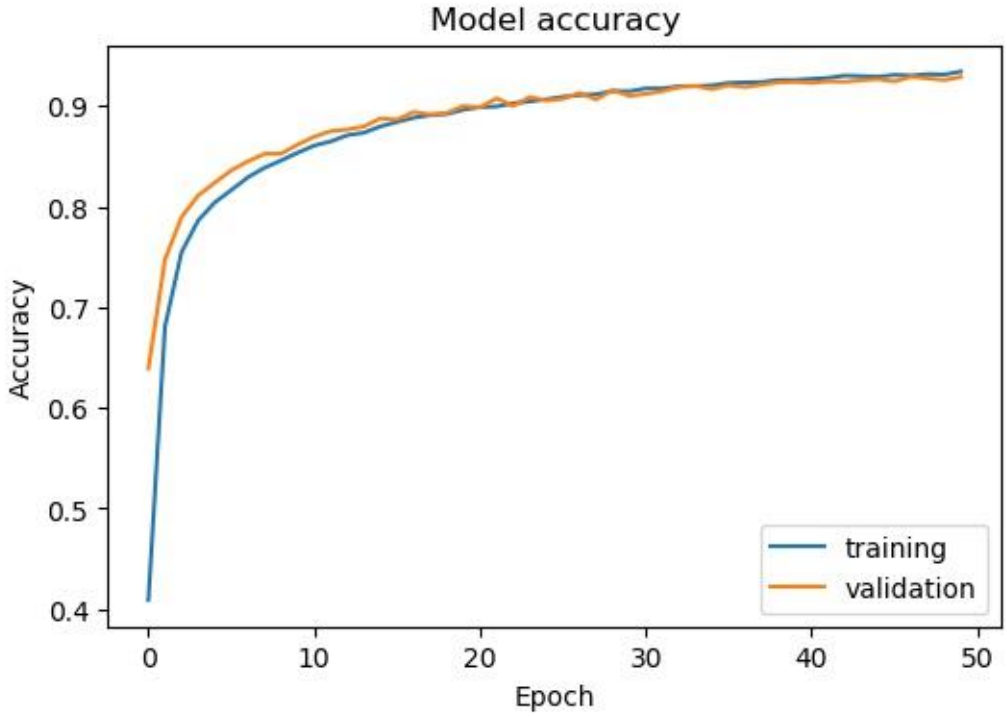


Figure 34: Accuracy vs Epoch for training and Validation data on CNN Classifier with 28 DCT Coefficients at the input Layer

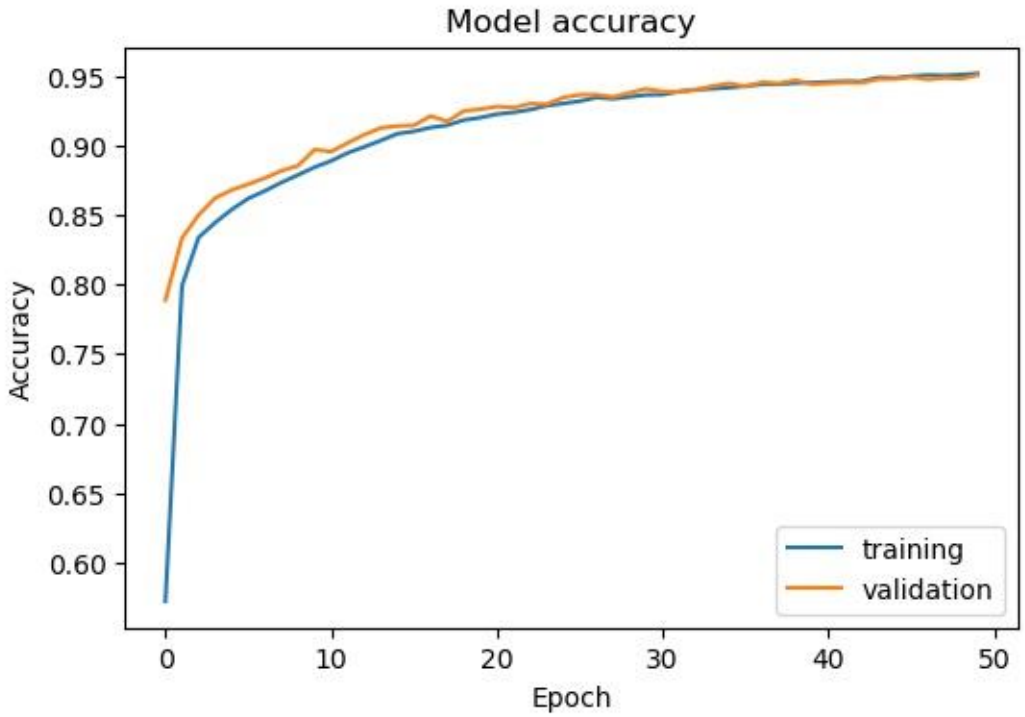


Figure 35: Accuracy vs Epoch for training and Validation data on CNN Classifier with 45 DCT Coefficients at the input Layer

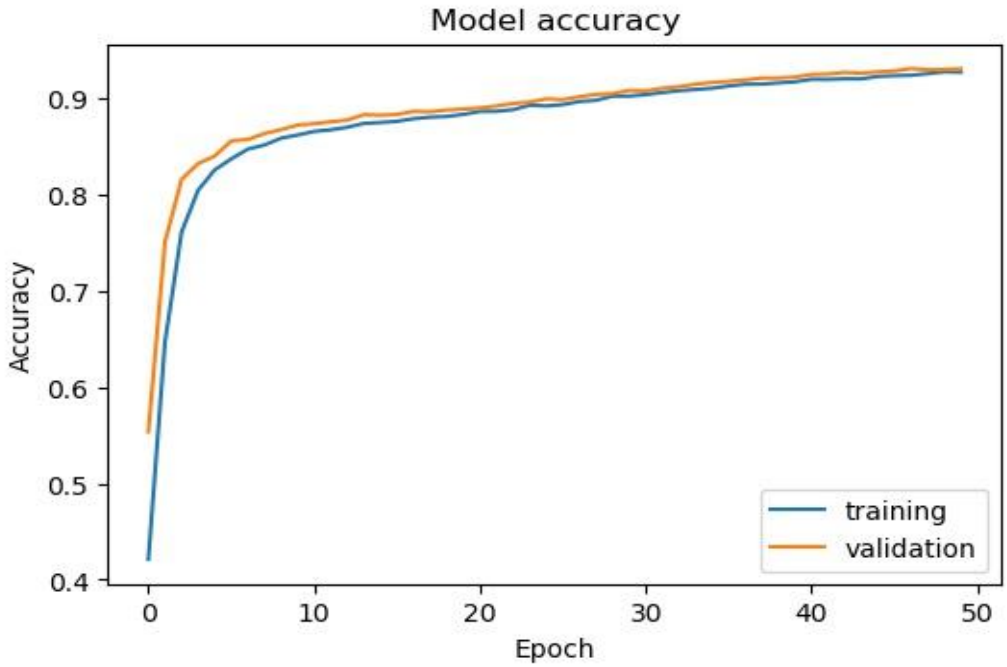


Figure 36: Accuracy vs Epoch for training and Validation data on CNN Classifier with 66 DCT Coefficients at the input Layer

4.6. Confusion Matrix

Once the classification Models were trained and validated. The classifiers were then tested on a test data of 5000 unseen compressed images, to deeply analyze the classification results of different models using confusion matrix. "A confusion matrix describes the performance of a classification model on a set of test data, for which the true values are known, in the form of a table". The structure of a confusion matrix for multi-class classification problem is shown in **Figure 37**.

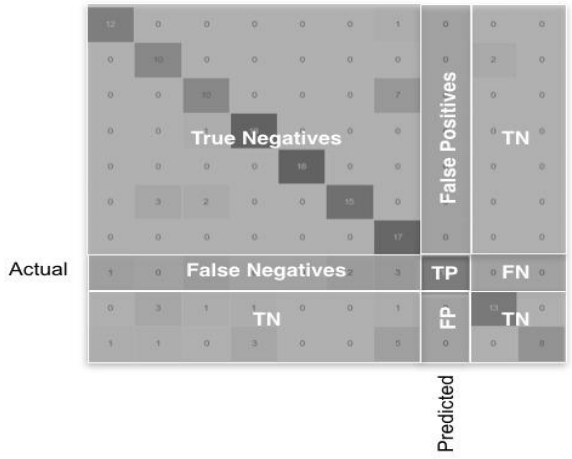


Figure 37: Structure of Confusion Matrix (CM) for Multi Class Classification

The confusion matrix reports the results in the form of True positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN). The True positives and True Negatives are the classes that are correctly classified whereas; the False Positives and False Negatives are the misclassified classes. "In general, in a confusion matrix, the predicted classes are compared with the actual classes. Each column of the matrix represents the results of prediction for the corresponding class at that column, while each row represents the actual class". The diagonal cells show the number of correct classifications i.e. 'TP', by the trained classifier; while the off diagonal cells in that given row and column represent the misclassified predictions i.e. 'FN' and 'FP'. And all the other cells show the number of correct misclassification i.e. 'TN' for that particular class.

4.6.1. MLP Classifiers Confusion Matrix

Comparison of the confusion matrix (CM) for the given MLP classifiers on the unseen test data of 5000 compressed images of handwritten digits is shown in **Figures 38-41**.

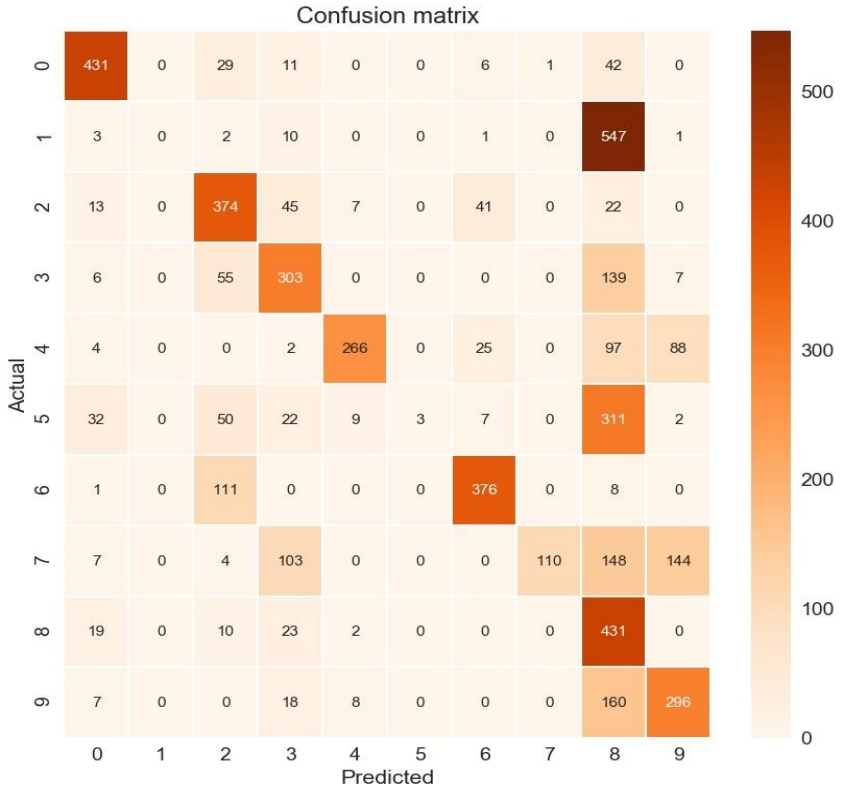


Figure 38: CM for MLP-Classifier with 15 DCT Coefficients at the input Layer

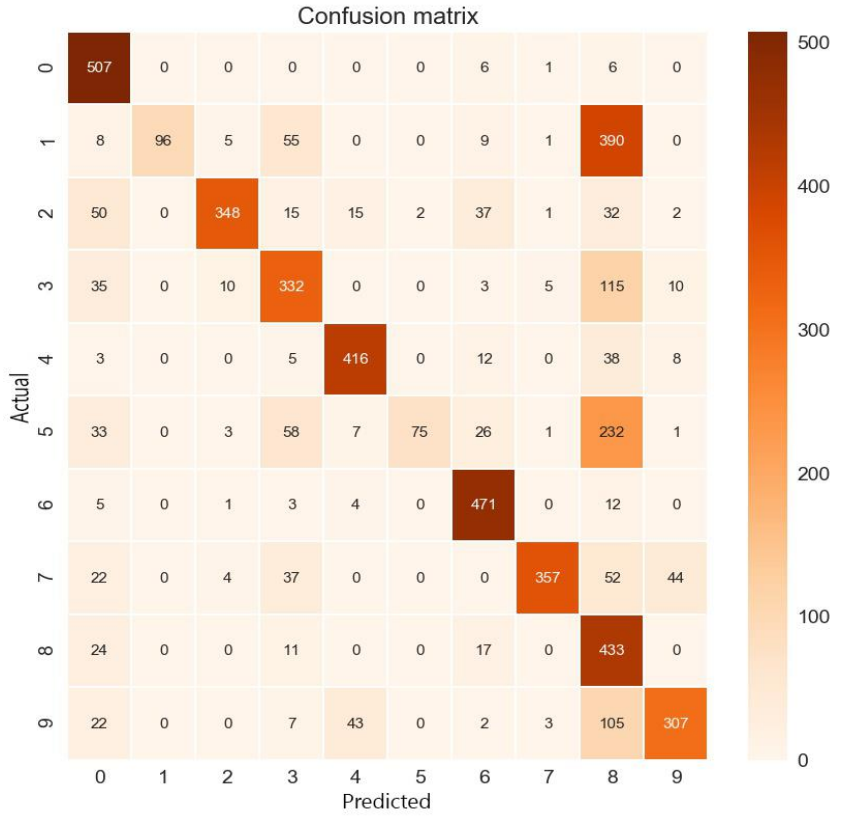


Figure 39: CM for MLP-Classifier with 28 DCT Coefficients at the input Layer

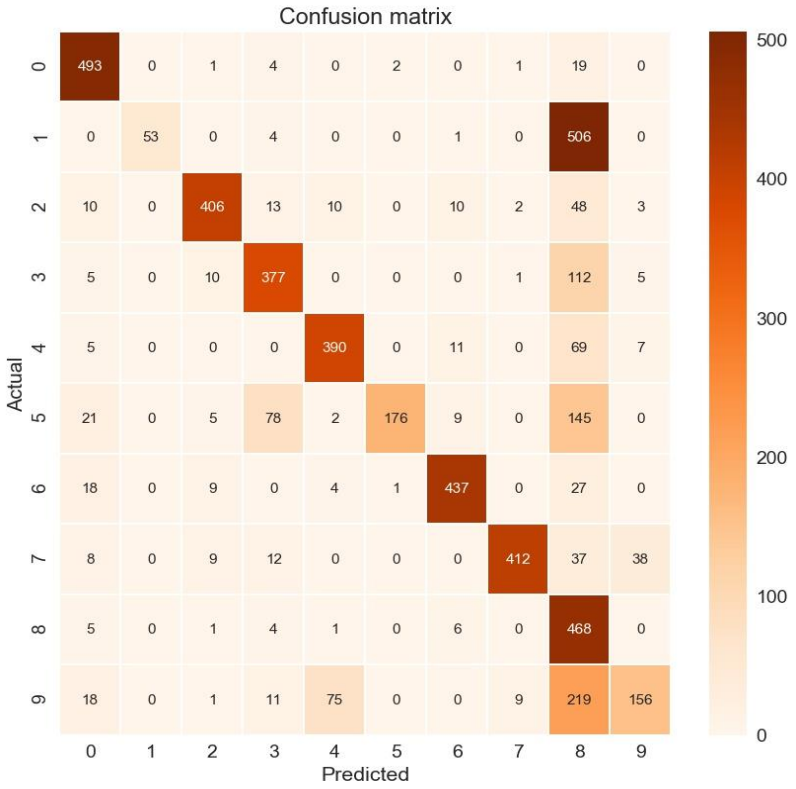


Figure 40: CM for MLP-Classifier with 45 DCT Coefficients at the input Layer

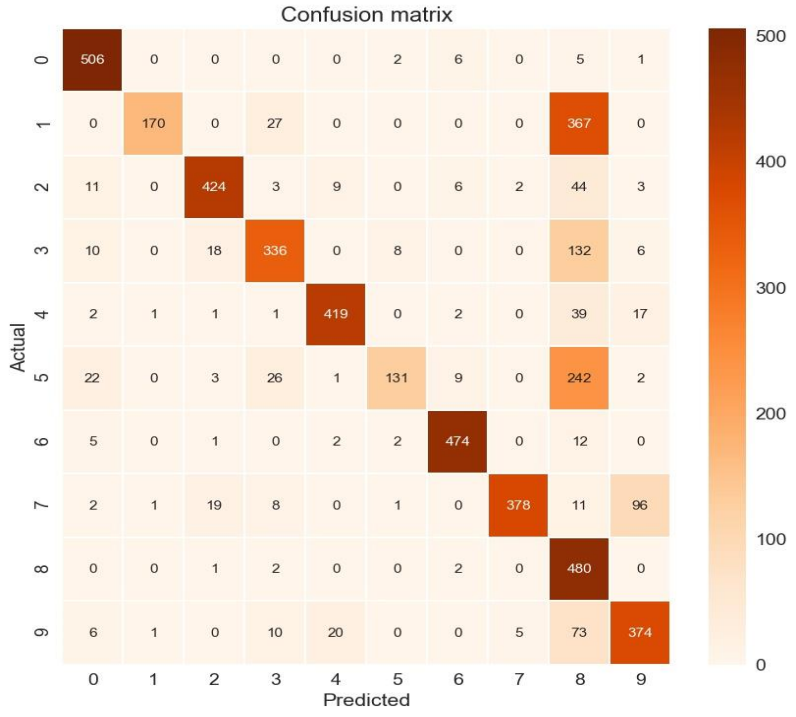


Figure 41: CM for MLP-Classifier with 66 DCT Coefficients at the input Layer

4.6.2. CNN Classifiers Confusion Matrix

Comparison of the confusion matrices (CM) for the CNN classifiers on the unseen test data of 5000 compressed images of handwritten digits is shown in **Figures 42-45**.

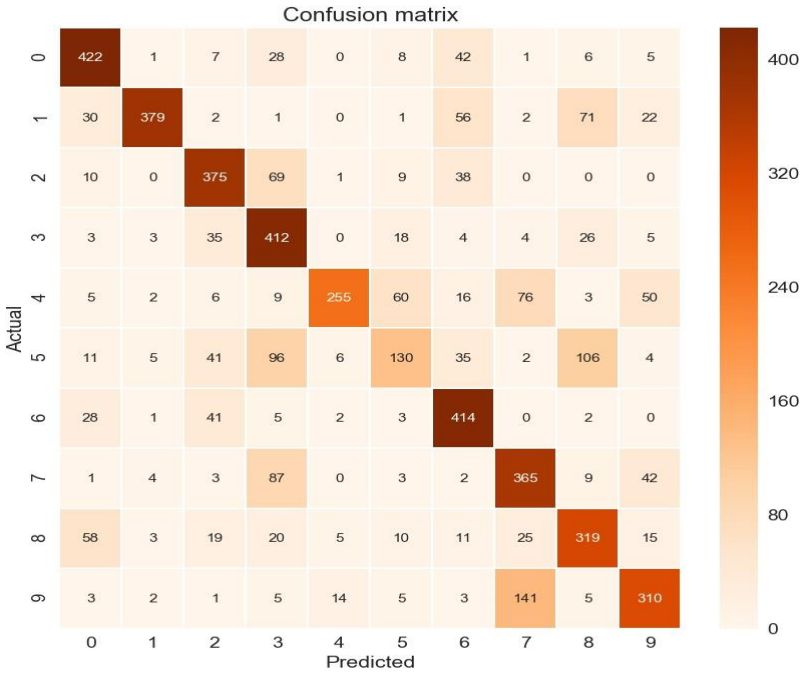


Figure 42: CM of DCNN Classifier with 15 DCT Coefficients at the input Layer

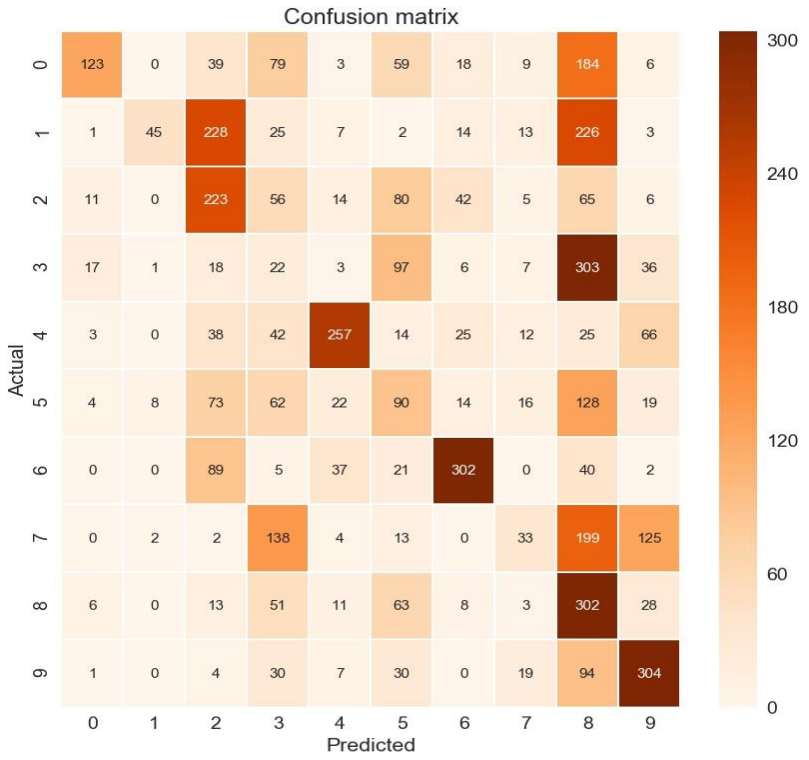


Figure 43: CM of DCNN Classifier with 28 DCT Coefficients at the input Layer

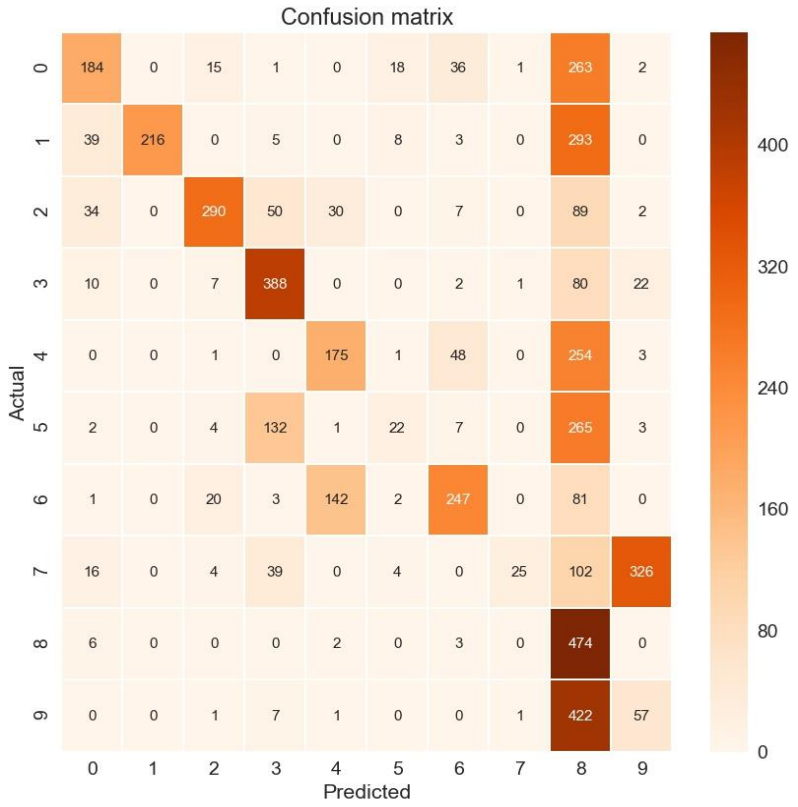


Figure 44: CM of DCNN Classifier with 45 DCT Coefficients at the input Layer

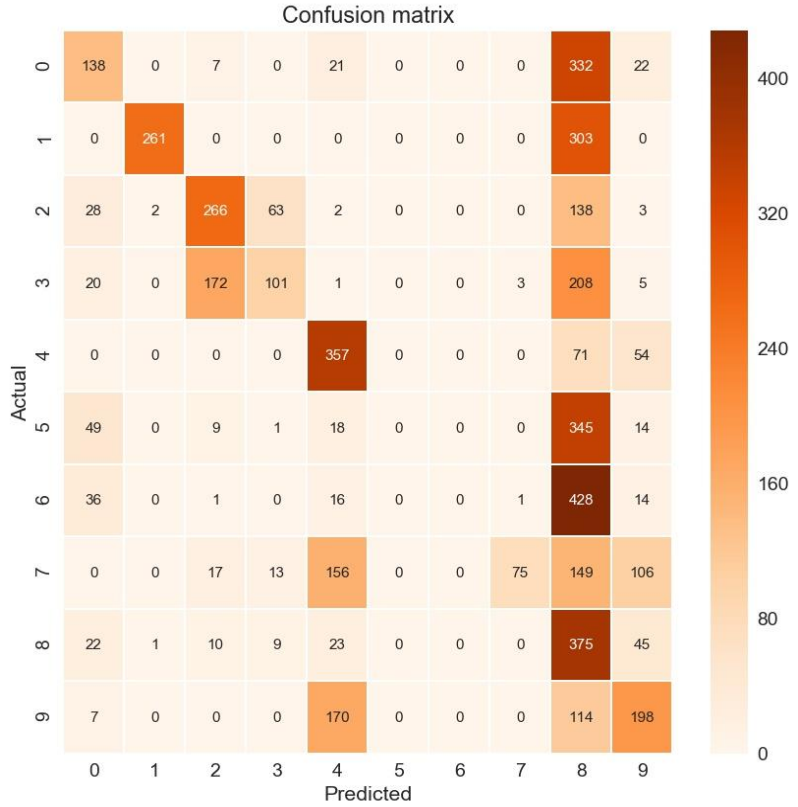


Figure 45: CM of DCNN Classifier with 66 DCT Coefficients at the input Layer

4.7. Analysis

The results prove that the proposed methodology for image compression gives pretty good results when used for digits classification. The highest accuracy recorded among the CNN classifiers is of 95% by a classifier, where the compression ratio of the test images was 17:1. Whereas, the highest accuracy achieved among MLP classifiers is 88%. Both the CNN and MLP classifiers performed well for the classification task.

When we talk about the accuracy of a multi-class classification model, we mean the ratio of the sum of TP and TN to the sum of all the possible outcomes for that class and for all classes. As we analyze the above confusion matrices, we observe that the classification models perform well in recognizing the instances of all the ten classes accurately, except the test instances of class 8.

4.8. Summary

In this chapter, it is discussed how the MNIST image dataset can be compressed using the proposed methodology and how the resulting compressed images can be used for solving the classification problem. Moreover, to assist the reader with the visual illustration of the performance of different classifiers, multiple graphs have been provided. And the detailed analysis of the results is carried out using confusion matrices.

Chapter 5

Conclusion and Future Work

CHAPTER 5: CONCLUSION AND FUTURE WORK

This chapter deals with an overview of research conclusion covered in **Section 5.1** whereas future work is mentioned in section **5.2**

5.1. Conclusion

This research presents a Novel image compression methodology for gray scale images using deep neural architecture. The research was executed in three phases as follow:

During the first phase, a comprehensive study of literature was performed to identify the different image compression techniques and the use of neural networks in the area of image compression. Over 34 studies were comprehensively reviewed during this process. Finally, it was concluded that very limited work is carried out for image compression in the DCT domain. And tackling this area using deep neural networks was a promising research issue. Moreover, another issues identified during the literature review was that the existing image compression techniques are computationally expensive.

In the second phase, a methodology is proposed to execute the job of image compression using deep neural networks. Two types of deep neural networks were designed and tested for this purpose; MLP models and DCNN models. The use of RELUs and Tangent Sigmoid was advocated in these DNNs, since very simple functions can be used to realize these units. Then the performance of the proposed methodology was evaluated. Results showed that the performances of MLPs are relatively better than the CNNs. Moreover, the limited use of DCT coefficients for image compression has accelerated the training time of the network, the networks have become computationally inexpensive and have better learning characteristic. It has also been shown experimentally that the DNNs with a compression ratio of 4:1 were the most efficient and provided better results on unseen data. Hence, we can argue that the first 25 percent DCT coefficients of an image contain the highest information and are the most important ones.

In the final phase, a use case of the proposed methodology was identified and executed. The MNIST dataset was first compressed using proposed methodology and then that compressed data was used for classification task. The evaluation of the results showed that an accuracy of 95% is achieved on the classification problem. Further analysis was performed using confusion matrix. It was concluded that the proposed methodology achieves high

compression without compromising the quality of the images, is computationally inexpensive and reduces storage cost.

5.2. Future Work

Future work includes the further generalization of these deep neural networks in order to support images of high dimensions and colorful structure. Moreover, Future work on this research also leverages the use of compressed images from these DNNs to tackle different image classification and regression problems. In addition to that, compression of video data can also be achieved by extending these DNNs to video processing domain.

References

[1] Egmont-Petersena, M., de Ridder, D., Handels, H.: 'Image processing with neural networks – a review', Pattern Recognit., 2002, 35, pp. 2279–2301

[2] Dony, R.D., Haykin, S.: 'Neural network approaches to image compression', Proc. IEEE, 1995, 83, (2), pp. 288–303

[3] Dangman, J.: 'Complete discrete 2-D Gabor transforms by neural networks for image analysis and compression', IEEE Trans. ASSP, 1988, 36, pp. 1169–1179

[4] Vaddella, V., Rama, K.: 'Artificial neural networks for compression of digital images: a review', Int. J. Rev. Comput., 2010, 3, pp. 75–82

[5] G. L. Sicuranza, G. Ramponi, and S. Marsi, "Artificial neural network for image compression," Electronics Letters, vol. 26, no.3, pp. 477–479, 1990.

[6] S. Carrato and S. Marsi, "Parallel structure based on neural networks for image compression," Electronics Letters, vol. 28,no.12, pp. 1152–1153, 1992.

[7] G. Qiu, M. R. Varley, and T. J. Terrell, "Image compression by edge pattern learning using multilayer perceptions," Electronics Letters, vol. 29, no. 7, pp. 601–603, 1993.

[8] A.Namphol, S.H.Chin, andM.Arozullah, "Image compression with a hierarchical neural network," IEEE Transactions on Aerospace and Electronic Systems, vol. 32, no. 1, pp. 326–338,1996.

[9] Y. Benbenisti, D. Kornreich, H. B. Mitchell, and P. A. Schaefer, "A high performance single-structure image compression neural network," IEEE Transactions on Aerospace and Electronic Systems, vol. 33, no. 3, pp. 1060–1063, 1997.

[10] K. S.Ng and L. M. Cheng, "Artificial neural network for discrete cosine transform and image compression," in Proceedings of the 4th International Conference on Documents Analysis and Recognition, vol. 2, pp. 675–678, August 1997.

[11] C. Cramer, "Neural networks for image and video compression: a review," European Journal ofOperational Research, vol. 108, no. 2, pp. 266–282, 1998.

[12] J. Jiang, "Image compression with neural networks—a survey," Signal Processing: Image Communication, vol. 14, no. 9, pp. 737– 760, 1999.

[13] G. Toderici, S. M. O'Malley, S. J. Hwang, D. Vincent, D. Minnen, S. Baluja, M. Covell, and R. Sukthankar, "Variable rate image compression with recurrent neural networks," arXiv preprint arXiv:1511.06085, 2015.

[14] G. Toderici, D. Vincent, N. Johnston, S. J. Hwang, D. Minnen, J. Shor, and M. Covell, "Full resolution image compression with recurrent neural networks," arXiv preprint arXiv:1608.05148, 2016.

[15] L. Theis, W. Shi, A. Cunningham, and F. Husz´ar, "Lossy image compression with compressive autoencoders," arXiv preprint arXiv:1703.00395, 2017.

[16] J. Ball´e, V. Laparra, and E. P. Simoncelli, "End-to-end optimized image compression," arXiv preprint arXiv:1611.01704, 2016.

[17] M. Li, W. Zuo, S. Gu, D. Zhao, and D. Zhang, "Learning convolutional networks for content-weighted image compression," arXiv preprint arXiv:1703.10553, 2017.

[18] L. Theis and M. Bethge, "Generative image modeling using spatial lstms," in Advances in Neural Information Processing Systems, 2015, pp. 1927–1935.

[19] A. v. d. Oord, N. Kalchbrenner, and K. Kavukcuoglu, "Pixel recurrent neural networks," arXiv preprint arXiv:1601.06759, 2016.

[20] 6 Dimililer, K.: 'Backpropagation neural network implementation for medical image compression', J. Appl. Math., 2013, doi:10.1155/2013/453098, pp. 1–8

[21] 7 Alexa, F., Gui, V., Caleanu, C., et al.: 'Lossless data compression using neural networks'. Proc. of the Seventh WSEAS Int. Conf. on Circuits, Systems, Electronics, Control and Signal Processing, 2008, pp. 128–132

[22] 8 Patel, B., Agrawal, S.: 'Image compression techniques using artificial neural network', Int. J. Adv. Res. Comput. Eng. Technol. (IJARCET), 2013, 2, (10), pp. 2725–2729

[23] 9 Mehare, V., Shibu, S.: 'A neural network approach to improve the lossless image compression ratio', People J. Sci. Technol., 2012, 2, (1), pp. 53–58

[24] 10 Rao, P., Madhusudana, S., Nachiketh, S., et al.: 'Image compression using artificial neural networks'. Second Int. Conf. on Machine Learning and Computing, 2010, pp. 121–124

[25] 11 Huifang, L., Mo, L.: 'A new method of image compression based on quantum neural network'. Int. Conf. of Information Science and Management Engineering, 2010, pp. 567–570

[26] 12 Veisi, H., Jamzad, M.: 'A complexity-based approach in image compression using neural networks', Int. J. Signal Process., 2009, 5, pp. 82–92

[27] 13 Karthikeyan, P., Sreekumar, N.: 'A study on image compression with neural networks using modified Levenberg Maruardt method', Global J. Comput. Sci. Technol., 2011, 11, (1), pp. 1–5

[28] 14 Dutta, D., Choudhury, S., Hussain, M., et al.: 'Digital image compression using neural networks'. Int. Conf. on Advances in Computing, Control, and Telecommunication Technologies, 2009, pp. 116–120

[29] 15 Savant, Y., Admuthe, L.: 'Compression of grayscale image using KSOFM neural network', Int. J. Sci. Eng. Res., 2013, 4, (1), pp. 1–4

[30] 16 Amerijckx, C., Verleyse, M., Thissen, P., et al.: 'Image compression by self-organized Kohonen maps', IEEE Trans. Neural Netw., 1998, 9, (3), pp. 503–507

[31] 17 Barbalho, J., Dória Neto, A., Costa, J., et al.: 'Hierarchical SOM applied to image compression'. Proc. of the Int. Joint Conf. on Neural Networks (IEEE), 2001, pp. 442–447

[32] 18 Ben Amar, C., Jemai, O.: 'Wavelet networks approach for image compression', GVIP Spec. Issue Image Compression, 2007, pp. 15–23

[33] 19 Denk, T., Parhi, V., Cherkasky, V.: 'Combining neural networks and the wavelet transform for image compression', IEEE Proc. ASSP, 1993, 1, pp. 637–640

[34] 20 Lo, S., Li, H., Wang, Y., et al.: 'On optimization of orthonormal wavelet decomposition: implication of data accuracy, feature preservation and compression effects', SPIE Proc., 1996, 27, (7), pp. 201–214

[35] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," in Neurocomputing: Foundations of Research, MIT Press, Cambridge, Mass, USA, 1988.