Image Compression Using Convolutional Autoencoder

Yash. Raut[1], Tasmai. Tiwari[2], Ahmad Nabi. Ahmadi[3] ,Tejas. Bachhav[4] , Kashish. Topiwala[5] , Nidhi. Pai[6], Sanjana. Nambiar[7], Veerus D'Mello[8], Rhea Kolhapurkar[9], Rakshit. Kathawate[10], Ayush. Bhaisha[11], Darshil. Rijiya[12].

UG Students: (*ECE*) – NIT Bhopal[3], (*Computer science*) - PRMCEAM, Badnera[1,2], SAOE, Pune[4], SIT, Lonavala[5], NMAM Institute of Technology[6,7], St. Francis Institute of Technology, Mumbai[8,9], RCOEM, Nagpur[10], SCET, Surat[11,12] .

***Abstract* - Image compression is one of the advantageous techniques in several types of multimedia services. And recently deep learning is so developed that it is being used for image compression. In image compression consider we have images of various dimensions. One such dimension is 28 by 28. Images are formed by combining red, green and blue (RGB) in various proportions to obtain any colour in the visible spectrum. Image is made up of pixels and have some noise in them. We propose a Convolutional Auto encoder neural network for image compression by taking MNIST (Modern National Institute of Standards and Technology) dataset where we up sample and downs sample an image. We take an image 28 by 28 image with noise, which is a RGB image. By developing deep learning image should be compressed to 28 by 1 dimensional dense vector. After the compression final resulting image should have the original dimension of 28 by 28. Main objective is to compress a image without affecting the quality of image radically.**

Keywords: Image compression, Convolutional Auto encoder, Convolutional Neural Network (CNN), Down sample, MNIST dataset, noise, Up sample, optimizing loss.

I. INTRODUCTION

The most basic problem faced by everyone in the 21st century is time and space (memory) management. Digitising the image consumes more bandwidth which means more cost for the transmission of the data. So for the efficient use of bandwidth we use neural networks. Since Image compression is used for faster transmission in-order to provide better services to the user(society). This technology is designed to reduce the resolution of original image using Convolutional Auto encoder. The problems faced during compression are lossy compression and lack of efficiency during the process. And slower processing of autoencoder network. So to overcome these problems we are using this technology.

The original image of size (let's say 28*28) is reduced to 28*1 using the encoder and so the image is compressed after the input to the neural network data points. Since we are aware of forward propagation and backward propagation in machine learning we make use of any of them. It involves three layers via-input layer, output layer and multiple hidden layer between the input and output layer.

So after providing input to neural data points in compressed image makes entry to hidden layer. The hidden layer networks are logic functions that, based on input from one or more neurons in preceding layers will output information to one or more neurons in subsequent layers [1]. Afterwards the decoder comes into the picture, it takes the compressed (down sampled) image and un-compresses (up sampled) it giving the original image at output layer. In compression we get the code from image (encode) and then image from the code(decode). We had used tools like tensor flow: It is open source software library for high performance numerical computation, Open CV: a library of programming functions mainly aimed at real-time computer vision using the MNIST dataset and Convolutional neural network (CNN, or Conv Net): a class of deep neural networks, which is most commonly applied to analysing visual imagery.

The reasons why we choose to use this MNIST database are various. Firstly, as mentioned above it is a standard which is a relatively simple database for fast-testing theories and algorithms. And we want to test neural networks applied to the practical problems in the real world, the images in MNIST database have already been pre-processed including segmentation and normalization, so that it could be a good start for us spending minimal efforts on pre-processing and formatting. Besides, there are lots of researchers evaluating their theories and algorithms using MNIST, which means we can compare our results with the results from a rather comprehensive set of literature.

The purpose of this paper is to define how neural networks are used to resolve a problem of MNIST image compression. Our work is to design a neural network model and then implement it to solve the classification problem. Besides, some extra experiments have been done to test different methods that may consequently influence the performance of our model.

II. PROPOSED METHODOLOGY

The block diagram of proposed image compression based on CAE is illustrated in

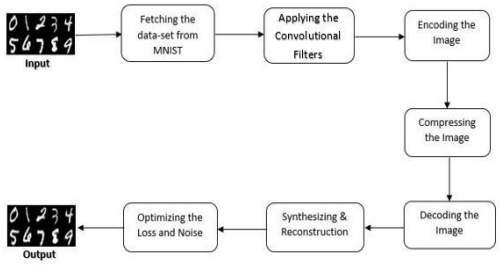


Fig.1: Block diagram of the proposed CAE based image compression.

A. Fetching the Dataset:

The standard available dataset for the image processing is MNIST database which is built by NIST. The dataset consists of various types of images including digits, fashion and etc. We would be dealing with grayscale images available in the dataset. Considering the Fashion dataset available at the MNIST, we would be proceeding with our architecture, we fetch the MNIST data set which consists of 60,000 images which are stored in the form of 728x1 ND array. Reshape those ND arrays into 28x28. We would encode the label provided in the image by making all the black pixels as 0 and all other pixels as 1.

We provide this data to the next level in the form of 28x28 matrices, 28x28 grayscale image, associated with a label from 10 classes. Each image is 28 pixels tall and 28 pixels broad, which gives a total of 784 pixels. Each pixel features a single pixel-value associated with it, indicating the lightness or darkness of that pixel, with higher numbers meaning darker. The pixel-value is an integer which lies between 0 and 255. The training and test data sets have a total of 785 columns. The first column consists of the class, pre-processing is applied to Dataset in order to make it suitable to feed it to the Tensorflow neural network. We add a dimension for the labels and as well as images in order to make it 2-Dimensional and 4-Dimensional vectors to work with, we use Tensorflow library to load the dataset and feed it to our algorithm.

B. Appling the Convolutional Filters:

Feature extraction is the dimensionality reduction process, where an initial set of raw variables (data) is reduced to the manageable group (or called as features) for processing, while still accurately and completely describing the original data set. We use Convolutional Autoencoder Neural Network to perform the dimensionality reduction in our project and we try to find out the filters which we could use to generate the image maps required during the compression process, we apply Convolutional filters over the image and subsample it to smaller and smaller images so that relative information about the image is preserved, Convolution is often defined as the "masking over the image", If A(x, y) is our image and h(x, y) is our filter then the convolution function g(x, y) is defined as

$$\mathbf{g(x, y)=A(x, y)*[Elementwise]h(x, y)}$$

We choose the 32 filters each of the size [2,2] with stride [2,2]. We mask it over the image which results in the formation of an image map of size 14x14x32. During each masking it performs the element wise multiplication between the image matrix and filter matrix and stores the result in a resultant matrix, then we move the filter by skipping the two columns as specified in the stride and continues till we reach the end column, after that we slide down the filter two rows down and continue the same. The dimension of image reduces from 28x28 to 14x14 because we are taking 2 rows and columns at a time and hence total no of iterations, we have to go through will be 14 and hence the dimension of the resultant image is 14x14. An ideal filter of size [1,1] should in order to avoid the loss of information of image but it wouldn't help us in reducing the size of the image. Hence, we used the [2,2] filter at each phase on order to reduce the size of the image.

The conventionalized image might introduce the negative values in the image and also the linearity in the way pixels are arranged so we use the activation function to introduce the non-linearity and make the image more robust and effective. We will be using the Leaky RELU (type of RELU) activation function as it is observed that it behaves better during the processing of the image. RELU is the abbreviation of rectified linear unit, which applies the non-saturating activation function $f(x) = \max(0, x)$. It effectively removes negative values from an activation map by setting them to zero. It increases the nonlinear properties of the decision function and of the overall network without affecting the receptive fields of the convolution layer, The MNIST data-sets which is already given to us is as follow:

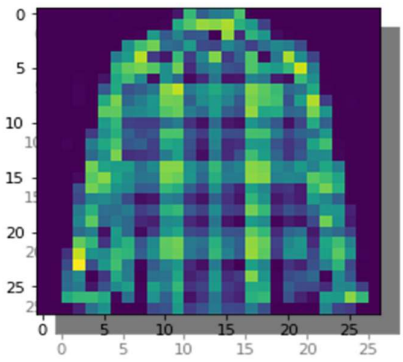
No Training dataset images = 60,000

No of testing dataset images =10,000

No of classes/labels= 10 (0 to 9)

Dimension of each image = 28x28

The data set example is as shown in fig (2) below:



C. Encoding and Decoding the image:

The image compression system is composed of two distinct functional component: an encoder and decoder, Encoder performs compression while Decoder performs decompression, both operations are performed in Software particularly in case of Web browser and many commercial editing programs. Input image f(x,…..) is given as input to the encoder, which creates a compressed representation of input It is stored for later use or transmitted for storage and used at a remote location. When the compressed image is given to decoder, a reconstructed output image f'(x,…..) is `generated as shown in fig 1., In still image applications, the input provided to the encoder and output from the decoder are f(x, y) & f'(x, y) respectively, If both functions are equal then the system is called lossless, error free. If not then its referred to as lossy, Encoding and Decoding process generates a fixed or variable length code to represent the quantized output and maps the output in accordance with the code. n most cases, a variable-length code is used to represent the mapped and quantized data set. The shortest code words here are assigned to the most frequently occurring output values and thus reduces coding redundancy It is reversible Upon its completion, the input image has been processed for the remove of all 3 redundancies.

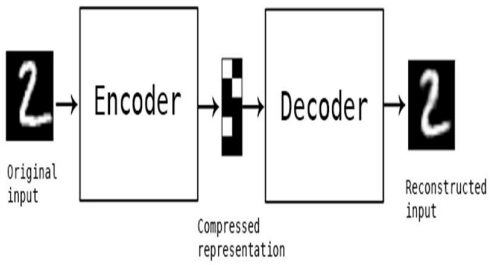


Fig 3. Encoding and decoding of input data.

D. Compressing the image:

We will use convolutional layer and polling layer of CNN in order to make our image dimension to 128-dimensional vector, we then flatten that image, and feed it to the **MLP** (Multi-layered Perceptron). We would have 128 inputs for our input layer, we would specify the number of nodes in the hidden layer and keep one output node in the output node. Initialization of the weight matrix is done randomly and we keep the bias in order to avoid zero weights initially as it would produce trivial outputs. The equation which is evaluated at each neuron in hidden layer is:

Error! Not a valid embedded object.

Where W[x, y] is weight matrix , A[x, y] is the input matrix and B[x, y] is the bias. The resultant image output of the hidden layer is the compressed image which we would be using during our project this is our final encoded image. The general structure of compressing the image is shown in fig (4)

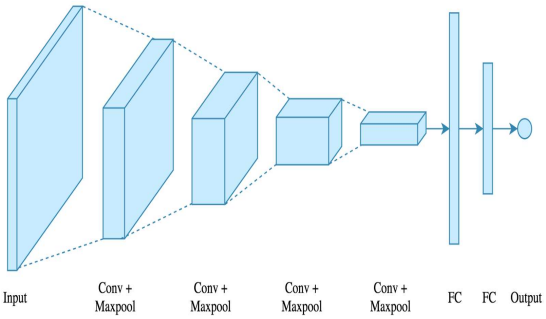


Fig 4. Compressing the image

E. Synthesizing and Reconstruction of Image.

Deep neural networks have been designed as a compelling alternative to traditional iterative solvers for reconstruction problems. Tuning parameters for conventional solvers, such as, regularisation parameters and step sizes are learned during training of these networks which increase the robustness of the final image reconstruction algorithm. Here, we aim at developing an approach for image reconstruction with deep neural networks applied to all patches of data in the frequency domain. It is capable of hiding useful contexts and adding arbitrary information that makes the output ultimately larger. A good transform should minimize these effects.

Image restoration is the neural network-based methods. The most significant difference between neural network methods and other methods is that they typically learn parameters for image restoration

directly from training data (e.g., pairs of clean and corrupted images) rather than relying on predefined image priors. We have proposed a very deep network architecture for image restoration. The network consists of a chain of symmetric convolutional layers and deconvolution layers. The convolutional layer act as the feature extractor which encodes the primary components of image contents while eliminating the corruptions.

The deconvolutional layers then decode the image abstraction to get back the image content details. To the best of our knowledge, the proposed framework is the first attempt to used both convolution and deconvolution for low-level image restoration. We propose to add, skip connections to improve the deep network between corresponding convolutional and deconvolutional layers. These shortcuts divide the network into several blocks. These skip connections help to back-propagate the gradients to bottom layers and pass image details to the top layers. The above mentioned characteristics make training of the end-to-end mapping from corrupted image to the clean one efficient and more effective and thus achieve the performance improvement while the network goes deeper, our network can be trained to obtain good restoration performance on different levels of corruption even using a single model.

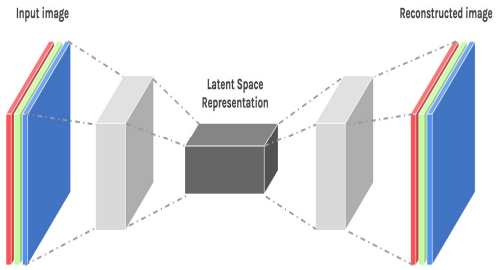


Fig 5. Synthesizing and Reconstruction of Image.

F. Optimizing the loss and noise in the reconstructed image:

We present a theoretically grounded approach to train deep neural networks, including recurrent networks, subject to class-dependent label noise. We propose two procedures for loss correction that are agnostic to both application do- main and network architecture. Large datasets used in training modern machine learning models, such as deep neural networks, are often affected by label of different noise. Primarily developed in Computer Vision. The second strand is recent Machine Learning research on the theoretically grounded means of combating label noise.

In particular, we are interested in the design of correct losses that are robust to label noise. For optimizing the loos and noise from our image we have to use ADAM (Adaptive Moment Estimation) optimizer function, and this function will optimize the weights during CNN hidden layer steps, Adam optimizer Function will optimize the differentiable objective function used for weights and for its input we only have to provide the learning rate, learning rate is used to prevent the big jumps so that we could end up with small and normalize weights. All the noisy images require the network to gather enough features from the image so that a noise-free version can be computed from them. The proposed architecture is shown in Fig(1). It includes three main components (i) A set of filters simultaneously extracts feature at multiple scales from the image. We call these filters collectively as multiscale feature extraction, layer(ii)a combination of filters which allow dampening the features contaminated by noise and (iii) reconstruction layers with filters that do not have any spatial resolution.

Finally, we train the model with all the images in the dataset using above settings of the CNN and try to encode all the images of the data-set by minimizing the noise in image.

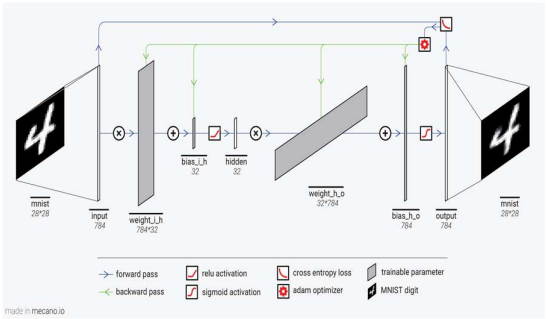


Fig 6. Optimizing the loss and noise in the reconstructed image.

III. EXPERIMENTAL ANALYSIS:

Images from handwritten images dataset available on MNIST is used as the test data set for our product. The results expected after training model for 1000 epochs were closed to loss of 2000. The yielded loss and error were in range of 10000. Results of the several different epochs were shown in below table.

Epoch	50	500	1000	2000
Loss	4989.5	789.1	246.3	108.2

Addition of ADAM optimizer to the output of the decoded image, decreased the loss from range of 3990 to 2389, to the range of 1200. The squared mean error for training set was found to be performing better than CNN model without optimizer and it had a significant improvement over the no of epochs also. The following graphs illustrate it:

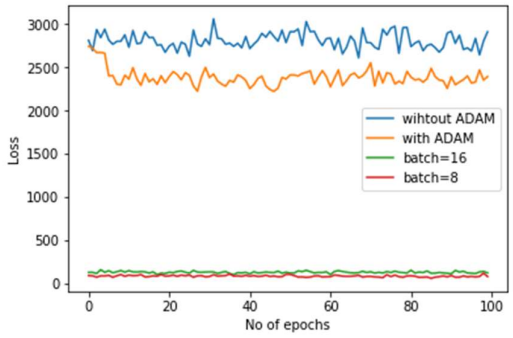
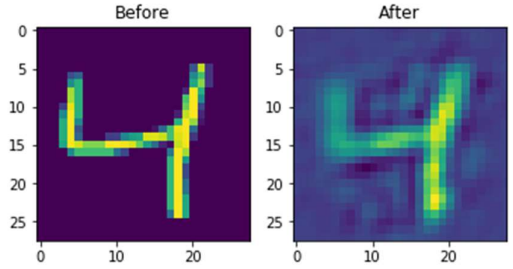


Fig. Experimental Analysis

The loss obtained was very high and hence we tried adding more convolutional layers with minimum stride size to get the more accurate results. The results obtained were outperforming our expectations. Average loss was reduced to 230s and with epoch 2000 it was below 100. This are the results obtained after 2000 epochs with batch size of 8



The average loss of information after reconstruction was 112.341. Error rate was below 23% after training model over dataset with large images.

IV. CONCLUSION:

In this paper, we have proposed Convolutional Auto Encoder approach to compression as well as decompress the image. The architecture of Auto Encoder consists of 5 convolutional layers and 4 de-convolutional layers. The images were passed to the various range of filters through the Convolutional layer. Output image is then evaluated with the Leaky RELU function in order to remove the negative values and introduce the non-linearity in the model. Decoder was feed the encoded image with 1x128 dimension and the loss and error were optimized with ADAM optimizer.

Experimental results show that CAE networks behave better than traditional encoders which uses DCT and quantization. This paper is theoretical approach to the image compression using CAE and proposes the optimal, efficient and accurate way to achieve this. Results with CNN were not to the point of excellence; hence we would be using the dense NET [2] for creating more complex architecture which could compress and reconstruct the image with loss of less than 500. We will be training our model on more data sets to make it robust. Besides, the generative adversarial network (GAN) shows more promising performance than using autoencoder only; therefore, we will utilize GAN to improve the coding efficiency further.

REFERENCES

[1] Li Deng "The MNIST Database of Handwritten Digit Images for Machine Learning Research "in Proc.Digital Object Identifier 10.1109/MSP.2012.2211477

[2] Gao Huan," Densely Connected Convolutional Networks" Dec-2016

[3] Hang Zhao, Orazio Gallo, uri Frosio and Jan Kautz "Loss Functions for Image Restoration with Neural Networks"in Proc.NVIDIA, MIT Media Lab.

[4] Tingxi Wen , Zhongnan Zhang "Deep Convolution Neural Network and Autoencoders-Based Unsupervised Feature Learning of EEG Signals"

[5] Wan Zhu"Classification of MNIST Handwritten Digit Database using Neural Network" in proc Research School of Computer Science, Australian National University , Acton, ACT 2601, Australia u6148896@anu.edu.au.

[6] Mohamed El Zorkany"A Hybrid Image Compression Technique Using Neural Network and Vector Quantization With DCT"
El Zorkany M. (2014) A Hybrid Image Compression Technique Using Neural Network and Vector Quantization With DCT. In: S. Choras R. (eds) Image Processing and Communications Challenges 5. Advances in Intelligent Systems and Computing, vol 233. Springer, Heidelberg.

[7] J. Jiang "Image compression with neural networks A survey" in proc School of Computing, University of Glamorgan, Pontypridd CF37 1DL, UK Received 24 April 19.