**CS 351 PROJECT REPORT**

**IMAGE DENOISING AND UPSCALING**

**GROUP MEMBERS:**

**MUTEEB-UR-REHMAN 2019419**

**TAIMOOR AHMAD 2019511**

**AHMAD HASSAN 2019041**

Table of Contents

[Introduction 2](#_Toc104753927)

[Auto-Encoders - A brief Description 2](#_Toc104753928)

[Denoising autoencoders 2](#_Toc104753929)

[Our Models 3](#_Toc104753930)

[Convolutional Autoencoder 3](#_Toc104753931)

[Dual Autoencoder 3](#_Toc104753932)

[Image Upscaling 3](#_Toc104753933)

[Description of Dataset Used 3](#_Toc104753934)

[Technical contribution 3](#_Toc104753935)

[Results in form of graph and images 4](#_Toc104753936)

[Dual auto-encoder results without upscaling on regular noise: 4](#_Toc104753937)

[Convolutional auto-encoder results without upscaling on regular noise: 4](#_Toc104753938)

[Upscaled images after denoising: 5](#_Toc104753939)

[Convolutional auto-encoder results without upscaling on synthetic noise: 6](#_Toc104753940)

[Dual auto-encoder results without upscaling on synthetic noise: 6](#_Toc104753941)

[Graph of loss vs epochs for convolutional auto-encoder: 7](#_Toc104753942)

[Graph of loss vs epochs for dual auto-encoder: 7](#_Toc104753943)

[Summary of convolutional auto-encoder model: 8](#_Toc104753944)

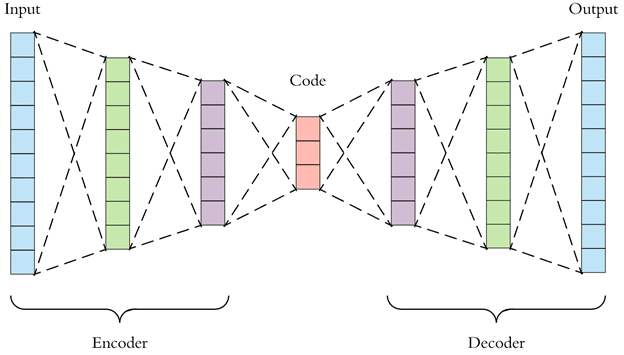
[Summary of Dual auto-encoder model: 8](#_Toc104753945)

[Conclusion 9](#_Toc104753946)

# Introduction

## Auto-Encoders - A brief Description

At a high level, an autoencoder contains an encoder and decoder. These two parts function automatically and give rise to the name “autoencoder”. Encoder transforms high-dimensional input into lower-dimension (latent state, where the input is more compressed), while a decoder does the reverse encoder job on the encoded outcome and reconstructs the original image.



## Denoising autoencoders

In denoising, data is corrupted in some manner through the addition of random noise, and the model is trained to predict the original uncorrupted data. Another variation of this is about omitting parts of the input in contrast to adding noise to input so that model can learn to predict the original image. In this case, the idea is storing the output generated by the encoder as a feature vector, which can be used in a supervised model train-prediction approach.

## Our Models

### Convolutional Autoencoder

We have developed a convolutional autoencoder model which is categorized as an unsupervised learning model to denoise, synthetic noise from sample images. Our sequential model consists of three convolutional 2D layers, a single max pooling layer and a single up-sampling layer from TensorFlow Keras library. The input size of the first convolutional layer is 150 X 150. The image is bottlenecked to a size of 75 X 75 i.e., half of the original size of the image. The up-sampling layer doubles the size of the image provided as an input to it. The last convolutional layer hence generates the image of size 150 X 150 same as our original input image.

### Dual Autoencoder

We have developed a Simple autoencoder model which consists of dense layers and is also categorized as an unsupervised learning model to denoise, synthetic noise from sample images. Our sequential model consists of two convolution 2D layers, five fully connected layers, a single max pooling layer and a single up-sampling layer from TensorFlow Keras library. The input size of the first convolutional layer is 150 X 150. The image is bottlenecked to a size of 75 X 75 i.e., half of the original size of the image. The up-sampling layer doubles the size of the image provided as an input to it. The last convolutional layer hence generates the image of size 150 X 150 same as our original input image.

### Image Upscaling

For this module in our project, we are using a pre-trained Enhanced Deep Super Resolution network (EDSR) model from dnn\_superres library of OpenCV. It upscales our denoised image by a factor of four, from 150 X 150 to 450 X 450.

# Description of Dataset Used

Our dataset consists of 4000 RGB images of PNG format which is divided into 8 batches of 500 images to prevent program from exceeding available memory resources. Each image is resized to 150 by 150 pixels before use and converted to a numpy array. Synthetic noise is added randomly to the array. The resultant dataset is used as an input to our model. Our dataset is obtained from following site: <https://www.cs.toronto.edu/~kriz/index.html>

# Technical contribution

Main goal of our project is to compare our convolutional auto-encoder model with our simple auto-encoder model which uses a mix of dense and convolutional layers and come up with a conclusion that which method is better in terms of performance. Our models are trained on the same dataset. We have analyzed our models in terms of loss in both models for 180 epochs over each batch of images. We have also displayed a side-by-side comparison of resulting images of both the models after every 20 epochs.

# Results in form of graph and images

## 

## Dual auto-encoder results without upscaling on regular noise:

A picture containing text, different

Description automatically generated

## 

## Convolutional auto-encoder results without upscaling on regular noise:

A picture containing calendar

Description automatically generated

## Upscaled images after denoising:

A close up of a purple flower

Description automatically generated with medium confidence

A close up of a purple flower

Description automatically generated 🡪

A picture containing indoor

Description automatically generatedA picture containing indoor

Description automatically generated 🡪

## Convolutional auto-encoder results without upscaling on synthetic noise:

A picture containing graphical user interface

Description automatically generated

## Dual auto-encoder results without upscaling on synthetic noise:

A picture containing graphical user interface

Description automatically generated

## Graph of loss vs epochs for convolutional auto-encoder:

Chart

Description automatically generated

## Graph of loss vs epochs for dual auto-encoder:

Chart

Description automatically generated

## Summary of convolutional auto-encoder model:

A black screen with white text

Description automatically generated with low confidence

## Summary of Dual auto-encoder model:

Text

Description automatically generated

# Conclusion

From our findings, we conclude that convolutional auto encoders perform better that our dual auto encoder in terms of amount of loss, quality, and color of our regenerated images. The convolutional auto encoder was also faster that than our dual auto encoder model.