# IMAGE DENOISING

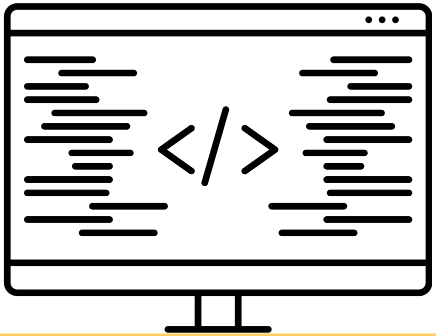
SUBMITTED BY :

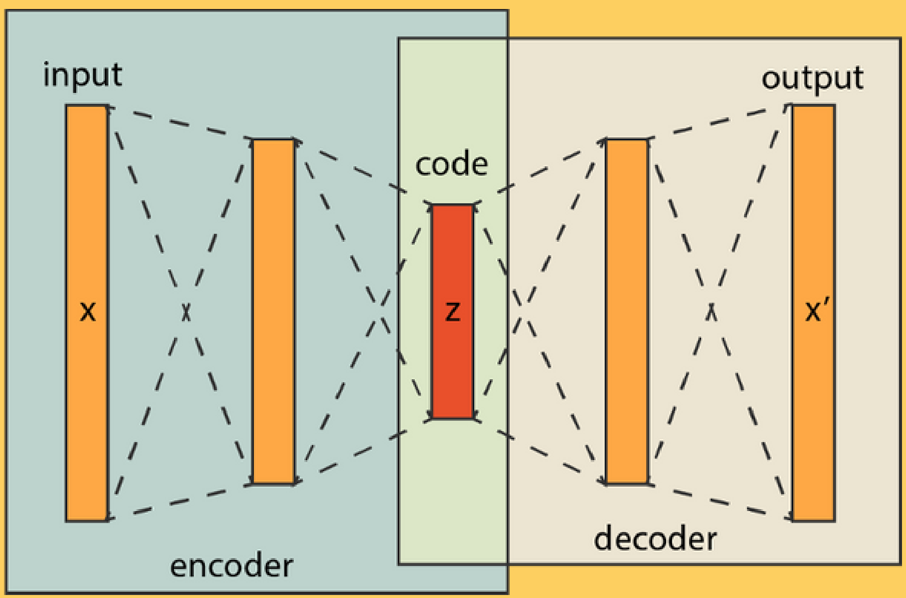

VIKRAM DEV 2019ME20939

ALIA KHAN 2019ME11071

NAWAZ ZAFAR 2019ME20903

# ALGORITHM IN A NUTSHELL






- We are using autoencoders which consist of 2 parts encoder and decoder.

- The encoder encodes the training data and a decoder which decodes the pattern

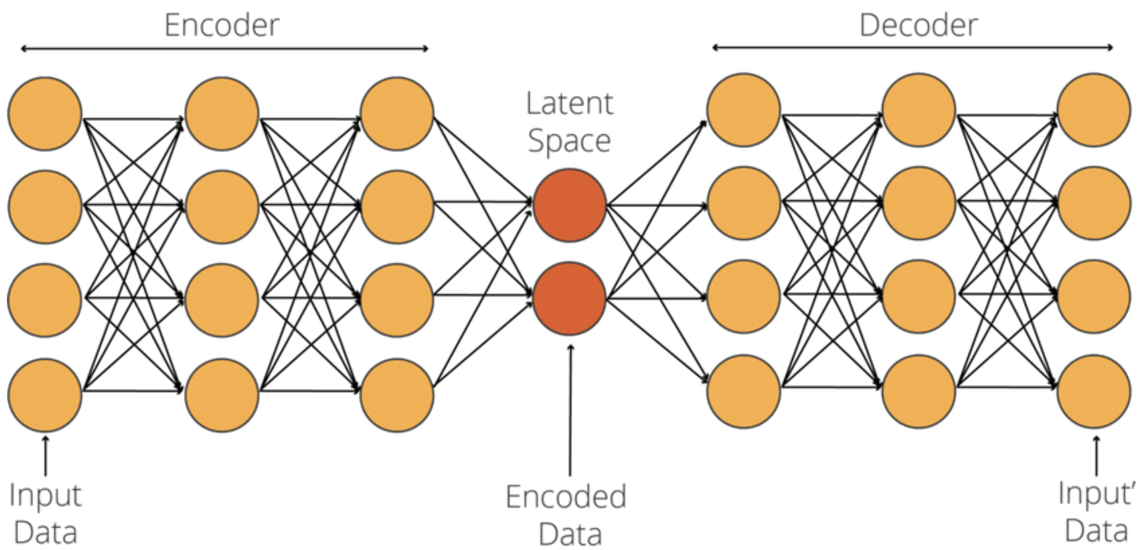
# WHY DO WE NEED TO ENCODE AND DECODE THE SAME THING?

- **Because, on the sample data, we are creating random noises (almost), then the neural network tries to find the correlation between noised images (input) and the noise-free image (output).**

- **The output of the ‘code’ is based on the training images, which means that the model learns from the training data and apply it to test data.**

- **This also implies that as we change training data the output is going to be different as well.**

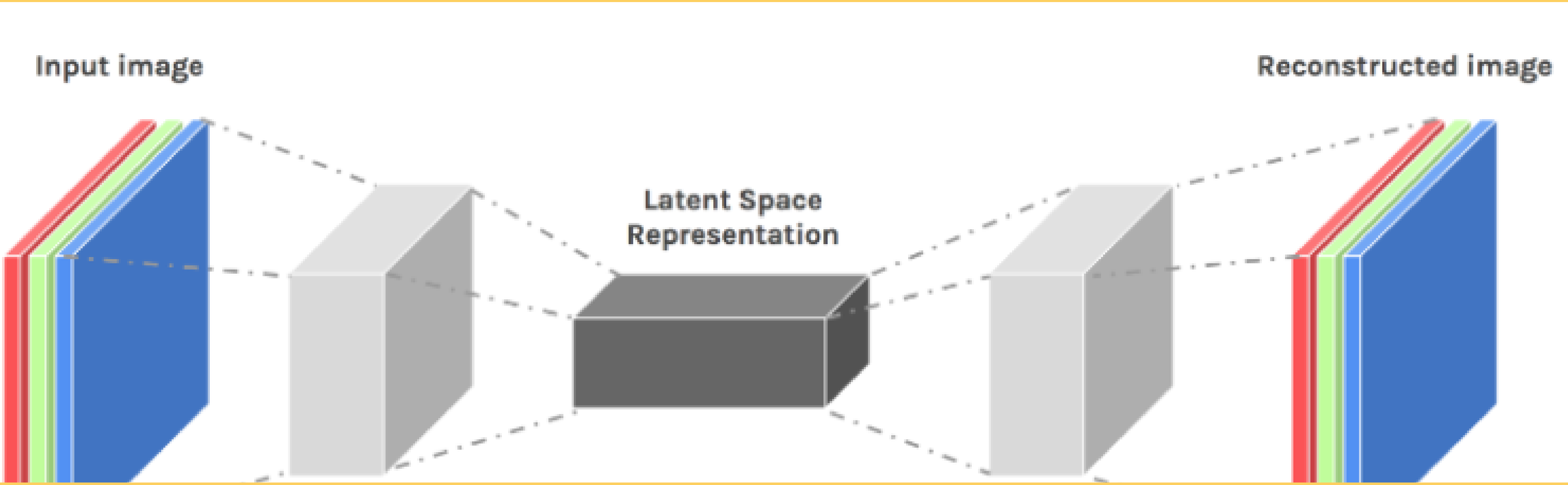
# AUTOENCODERS




- **Autoencoders consists of an encoder network, which takes the feature data and encodes it to fit into the latent space. In an encoder, what the model learns is how to encode the data efficiently so that the decoder can convert it back to the original.**

- **Therefore, the essential part of autoencoder training is to generate an optimized latent space.**

# WORKING OF AUTOENCODERS



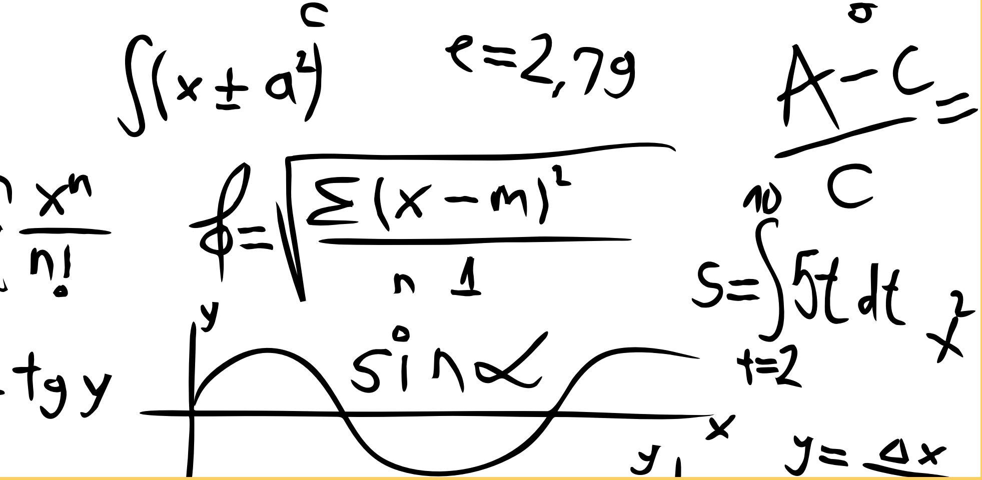

- **The input is the same as the output. They work by compressing the input into a latent-space representation and then reconstructing the output from this representation .**
- **The mechanism which we are applying is that it will replace fully connected layers with convolutional layers.**

# WORKING OF AUTOENCODERS

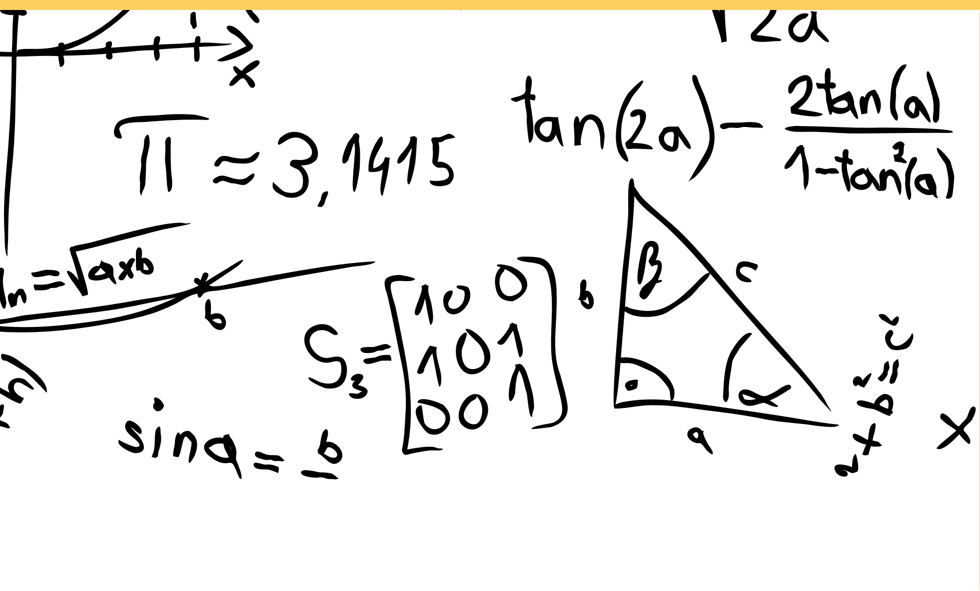
- These, along with pooling layers, convert the input from wide and thin (let’s say 100 x 100 px with 3 channels — RGB) to narrow and thick.

- This helps the network extract visual features from the images, and therefore obtain a much more accurate latent space representation. The reconstruction process uses upsampling and convolutions.

- The resulting network is called a Convolutional Autoencoder (CAE).



# MATHEMATICS INVOLVED



**Minmax normalization**

**Adam optimizer**

**Mean Squared error**

# MIN-MAX NORMALIZATION

**Variables that are measured at different scales do not contribute equally to the model fitting & model learned function and might end up creating a bias.**

**Thus, to deal with this potential problem feature-wise normalization such as MinMax Scaling is usually used prior to model fitting.**

**The formula is as follows:**

$$\frac{(\text{Value - Min})}{(\text{Max - Min})}$$

# ADAM OPTIMIZER

## Efficient Stochastic Optimization

Optimizing an objective function in the presence of randomness.

## Requires First-Order Gradients

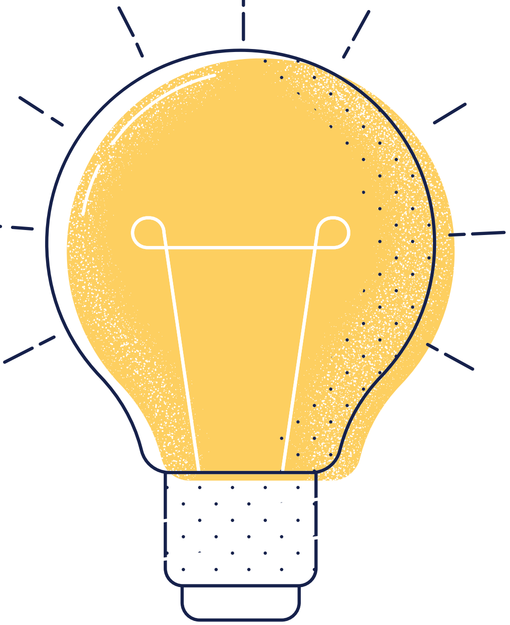
Adam only requires the first derivative of the parameters.

# MEAN SQUARED ERROR

**The mean squared error (MSE) of an estimator is, the average squared difference between the estimated values and what is estimated.**

$$\mathrm{MSE} = \underbrace{\frac{1}{n}\sum_{i=1}^{n}}_{\text{test set}} (\underbrace{y_i}_{\text{predicted vaue}} - \underbrace{\hat{y}_i}_{\text{actual value}})^2$$

**MSE is a risk function, corresponding to the expected value of the squared error loss.**



- It is to be noted that technically MSE is not a random variable, because it is an expectation. It is subjected to the estimation error for a certainly given estimator of θ with respect to the unknown true value. Therefore, the estimation of the mean squared error of an estimated parameter is actually a random variable.

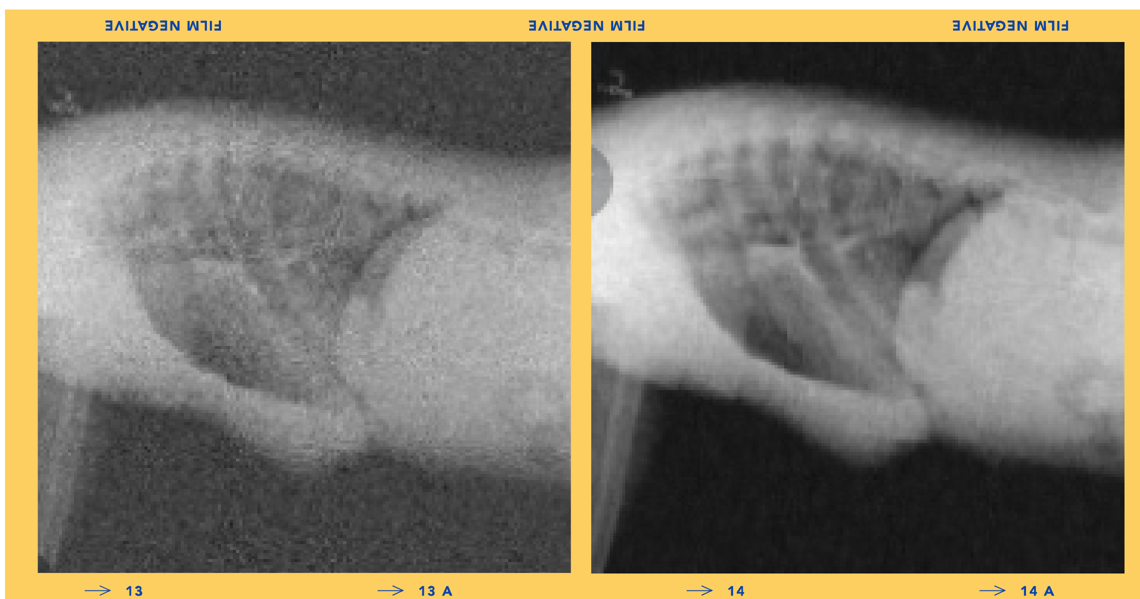
# APPLICATIONS OF IMAGE DENIOSING

**Image denoising plays an important role in a wide range of applications such as**

- **Image Restoration**
- **Visual Tracking**
- **Image Registration**
- **Image Segmentation**
- **Image Classification**

# APPLICATIONS OF IMAGE DENIOSING

**This includes applications in the health industry like detecting cancer, where images of tumours are not clear.**




**After Denoising**

# THANK YOU