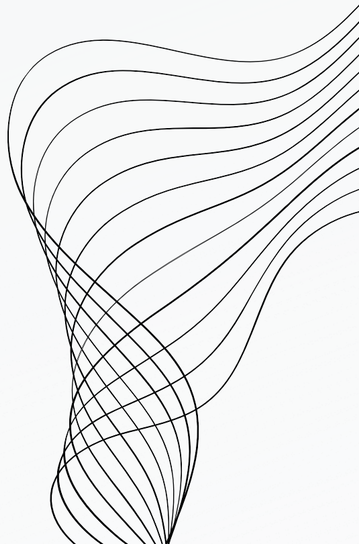
MASTERING DIGITAL IMAGE PROCESSING & DENOISING

NAME - ANURAG KAMBOJ
ROLL NO - 21124009
BRANCH - MATHS AND COMPUTING

PROJECT SUPERVISOR
DR R K PANDEY



CONTENT

01 BASICS OF DIGITAL IMAGE PROCESSING

02 IMAGE DENOISING

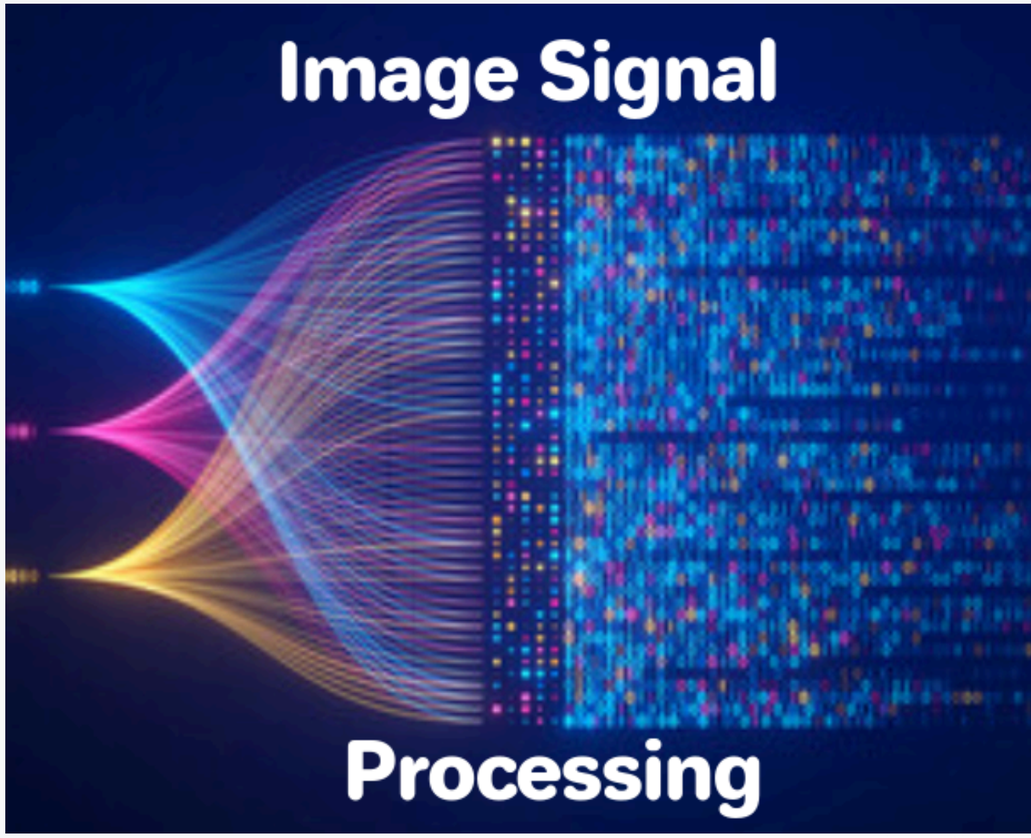
03 ML MODEL USING NEURAL NETWORKS FOR DENOISING

04 NEURAL NETWORKS

BASICS OF DIGITAL IMAGE PROCESSING

Digital image processing (DIP) is a field that focuses on the manipulation of digital images through computer algorithms

Applications in a wide range of fields:
Medical imaging,
satellite imaging,
surveillance,
digital photography, and more.



THE BASIC STEPS INVOLVED IN DIGITAL IMAGE PROCESSING ARE:

Image acquisition

This involves capturing an image using a digital camera or scanner, or importing an existing image into a computer.

Image enhancement

This involves improving the visual quality of an image, such as increasing contrast, reducing noise, and removing artifacts.

Image restoration

This involves removing degradation from an image, such as blurring, noise, and distortion.

Image segmentation

This involves dividing an image into regions or segments, each of which corresponds to a specific object or feature in the image.

Image analysis

This involves using algorithms and mathematical models to extract information from an image, such as recognizing objects, detecting patterns, and quantifying features.

Image synthesis

This involves generating new images or compressing existing images to reduce storage and transmission requirements.

DIGITAL IMAGE

Definition

An image is defined as a two-dimensional function,F(x,y), where x and y are spatial coordinates, and the amplitude of F at any pair of coordinates (x,y) is called the intensity of that image at that point. When x,y, and amplitude values of F are finite, we call it a digital image.

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & f(0,2) & \dots & f(0,N-1) \\ f(1,0) & f(1,1) & f(1,2) & \dots & f(1,N-1) \\ \vdots & \vdots & \vdots & & \vdots \\ f(M-1,0) & f(M-1,1) & f(M-1,2) & \dots & f(M-1,N-1) \end{bmatrix}$$

Every element of this matrix is called image element , picture element , or pixel.

KEY PROCESSES IN DIP

IMAGE ACQUISITON

The Image is captured by a sensor(e.g. Camera) and digitalized using analog to digital convertor

IMAGE ENHANCEMENT

The process of manipulation an image so that the result is more suitable than the original image

IMAGE RESTORATION

The process of improving the apperance of image.

IMAGE SEGMENTATION

Paritioning an image into its constituent parts or object. The more successful the segmentation, the more likely the recognition is to succeed

IMAGE COMPRESSION

Includes techniques for reducing the storage required to save an image

OBJECT REGONITION

This process assigns a label to an object based on the information provided by its description

SAMPLING AND QUANTIZATION

In Order to become suitable for digital processing, an image function is digitalized both spatially and in amplitude.

SAMPLING

Digitizing the coordinate value in the sampling

QUANTIZATION

Digitizing the amplitude value in Quantization

CONT.

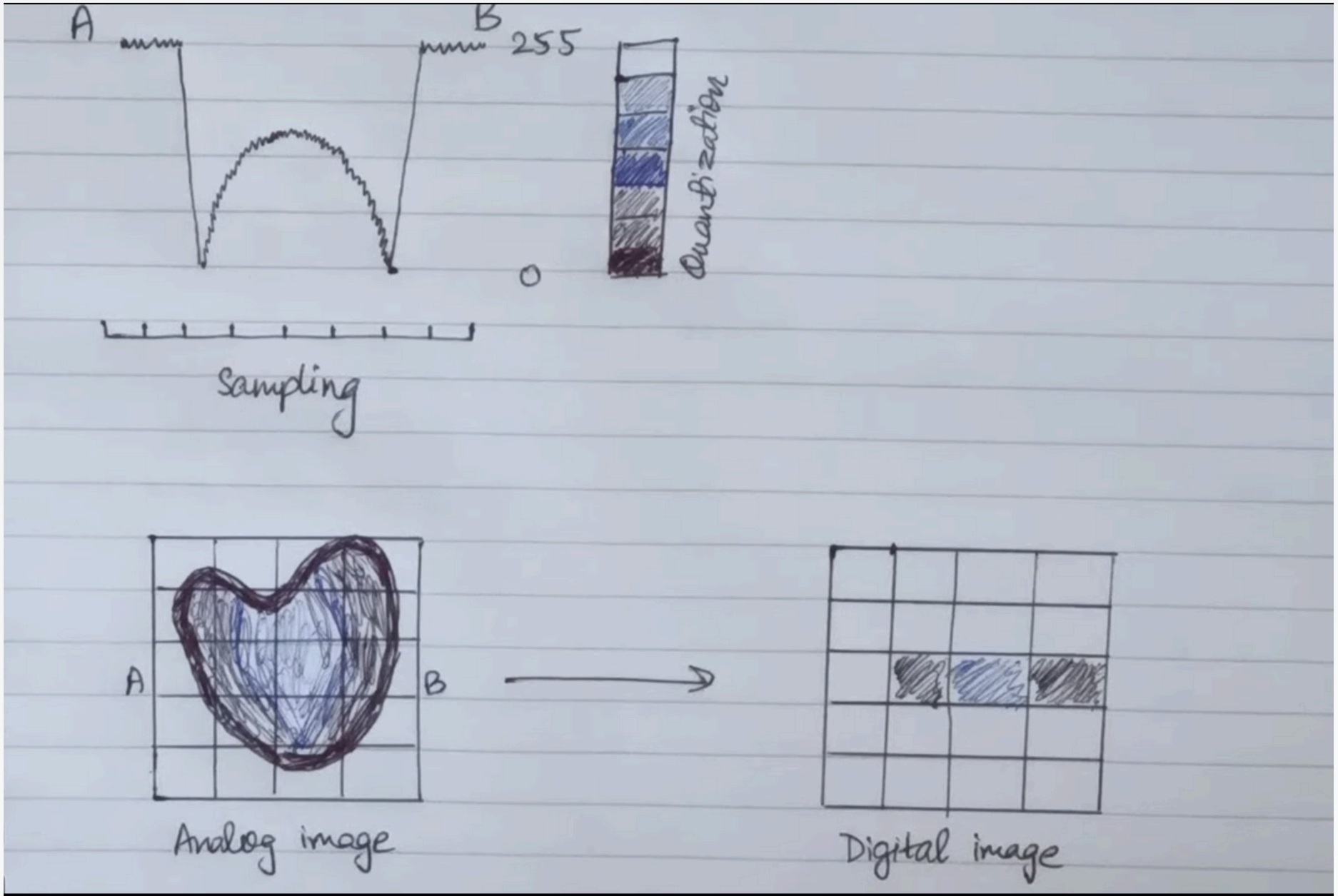

A
B
255
0
Quantization
Sampling
A
B
Analog image
Digital image

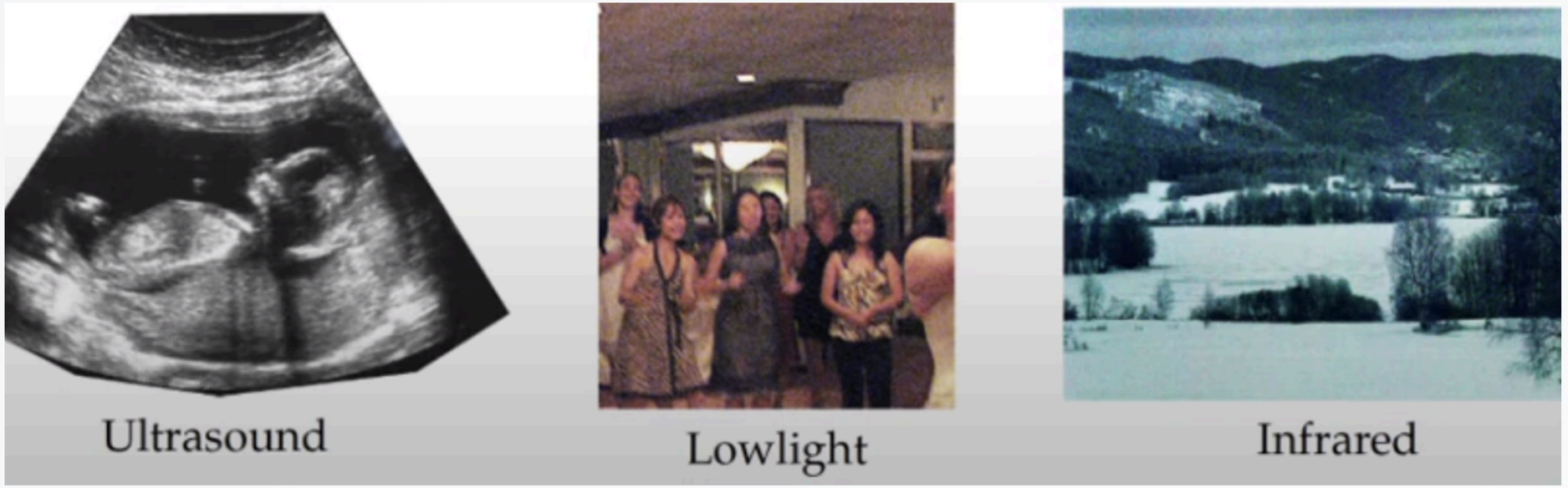
IMAGE DENOISING

NOISY IMAGE DOMAIN INCLUDES:

- Low Light
- Medical Imagery
- Remote sensing
- Infrared(Wavelet thresholding)

SOURCES OF NOISE:

- Sensor Noise
- Quantization Noise
- Low Light



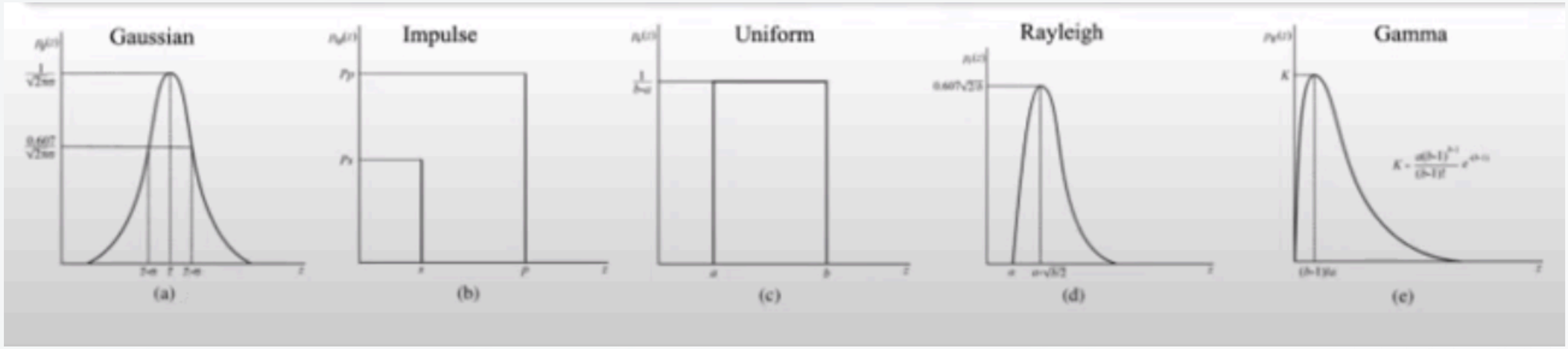
CATEGORIZATION OF NOISE (BASED ON DISTRIBUTIONS):

- Gaussian, Laplacian, Generalized Gaussian
- Uniform, Impulse
- Rayleigh, Poisson, Gamma, Exponential
- Apha-Stable

$$p(z) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(z-\mu)^2/2\sigma^2}$$

Where z represents gray level
μ is the **mean of average** values of z
σ is the standard deviation.
The standard deviation squared, σ^2, is known as **variance** of

Remark: Understanding type and distribution of noise is Important Effective Denoising Models.



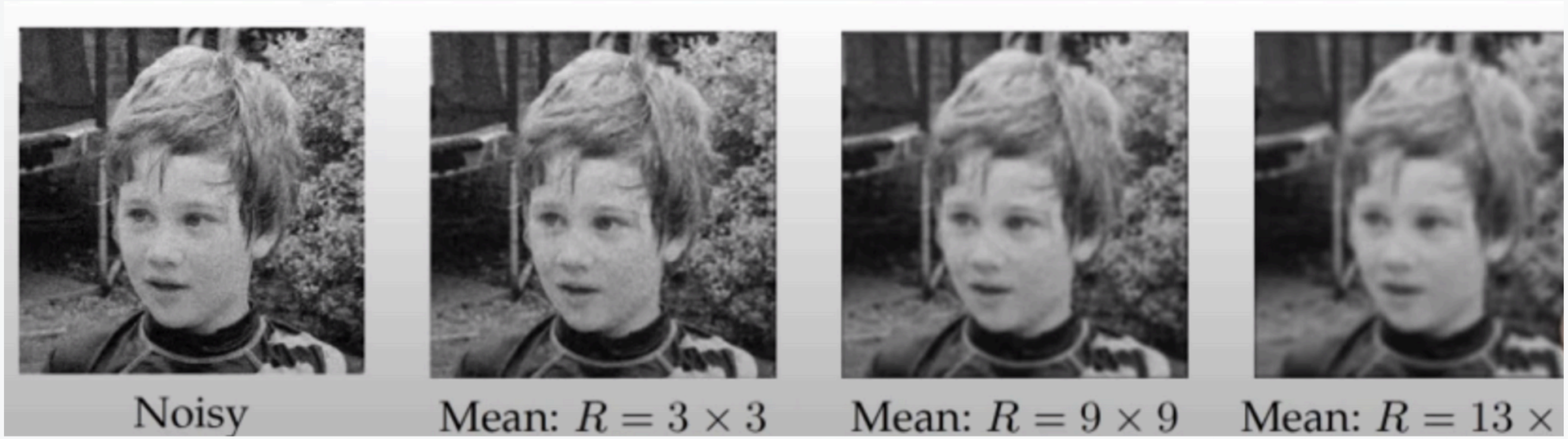
Due to illumination probelms

Noise is a fluctuation in pixel value and it is categoized by random variable
A random value probability distribution is an equation that links the statistical data with its probability of occurance

SOME METHODS OF IMAGE DENOISING:

1. MEAN FILTER

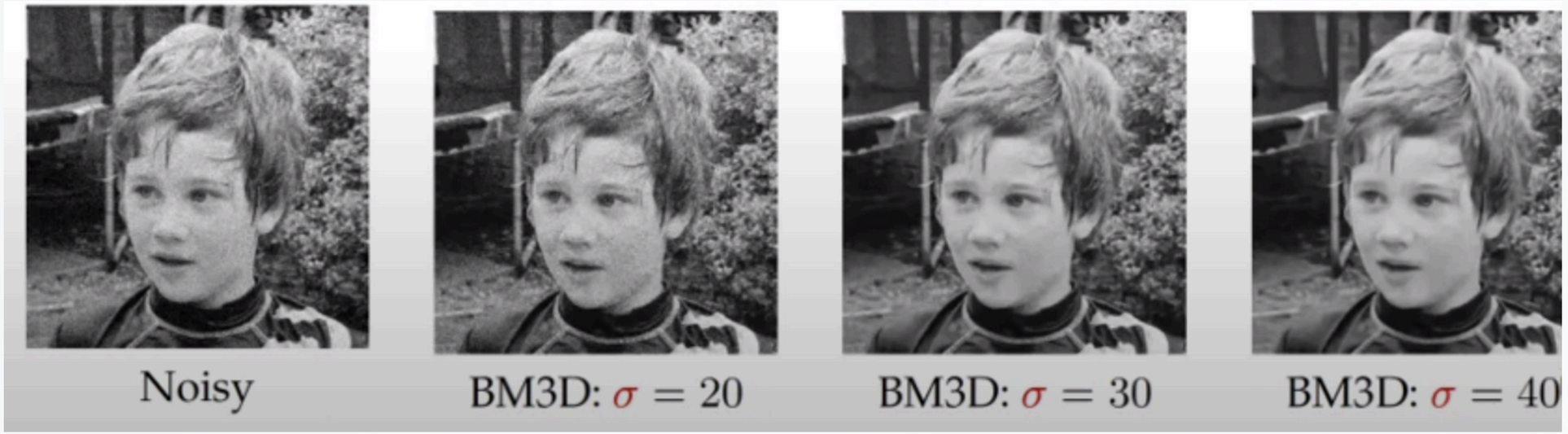
- Given an Input Image I, we define its pixels at position u,v as I(u,v).
- For each output pixel I’(u,v) we find the mean value with a local region R surrounding u,v.
- I’(u,v) = mean(I(u+i,v+j) | (i,j) are real numbers).
- Good in non-textured regions but blurs edges



CONT.

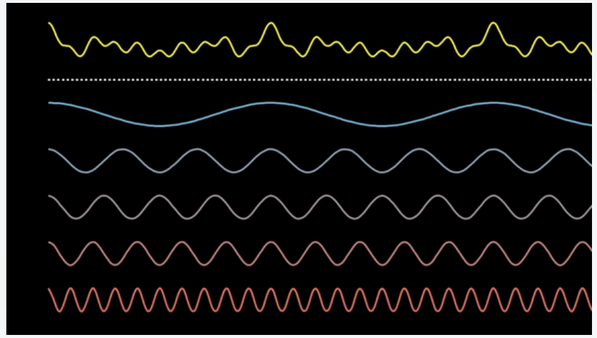
2. NON-LOCAL FILTER

- Block matching(hence BM) is done across whole image.
- Similar blocks are formed into 3D cubes.
- 3D blocks are filtered together and put back into Image.
- Great method until about 2010.

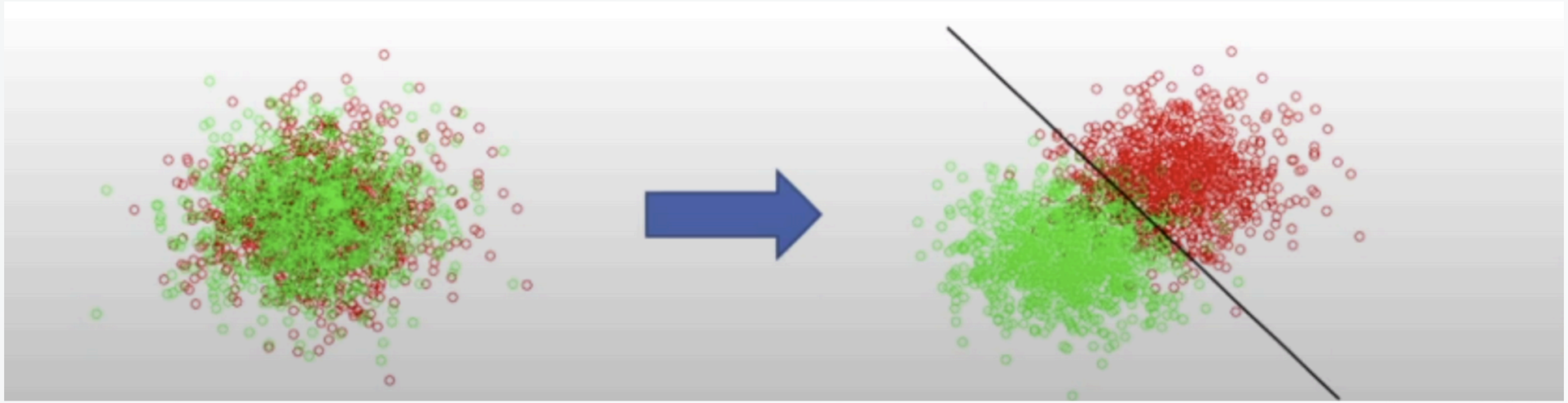


CONT.

3. TRANSFORM DOMAIN DENOISING



- New representation where the signal and noise are better separated.
- Denoising is achieved by suppressing noise coefficients while preserving the signal coefficients.

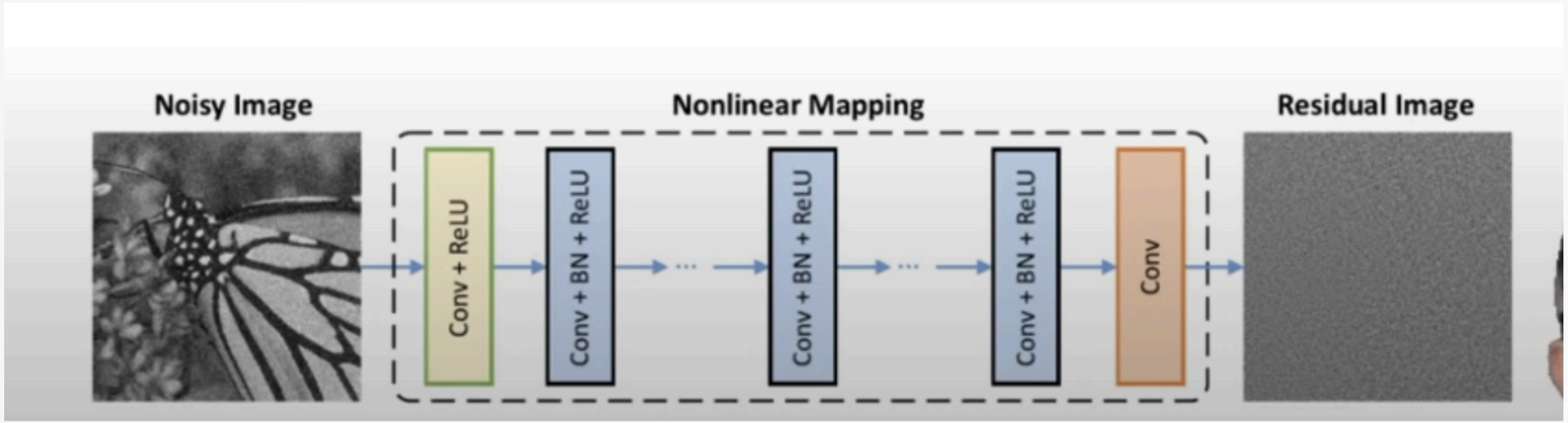


CONT.

4. NEURAL NETWORK METHOD:

- Trained networks use a database of clean and noisy images to train.

- ReLU – Rectified linear Unit(max (O, value))



MACHINE LEARNING MODEL ON IMAGE DENOISING USING NEURAL NETWORK:

REQUIRED LIBRARIES:

- TensorFlow and Keras are essential for building and training neural networks.
- NumPy is used for numerical operations, especially in manipulating arrays.
- Matplotlib is used for visualizing images.

```
from tensorflow.keras.datasets import mnist
from tensorflow.keras.layers import Conv2D, MaxPooling2D, UpSampling2D
from tensorflow.keras.models import Sequential
import numpy as np
import matplotlib.pyplot as plt
```

LOAD AND PREPROCESS DATA:

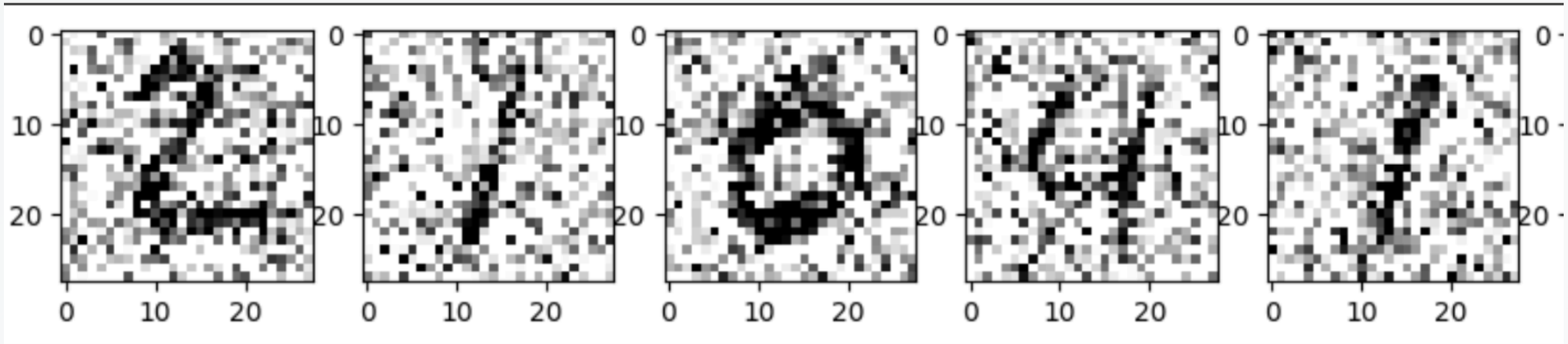
- MNIST dataset contains grayscale images of handwritten digits (0-9).
- Images are normalized to have pixel values in the range [0, 1] to aid convergence during training.
- Reshaping is necessary to match the input shape expected by the convolutional layers (height, width, channels)

```
(x_train, _), (x_test, _) = mnist.load_data()
x_train = x_train.astype('float32') / 255.
x_test = x_test.astype('float32') / 255.
x_train = np.reshape(x_train, (len(x_train), 28, 28, 1))
x_test = np.reshape(x_test, (len(x_test), 28, 28, 1))
```

VISUALIZE NOISY IMAGES:

- This step provides a visual understanding of how the added noise affects the appearance of the images.
- It helps in assessing the quality of the generated denoised images later.

```
plt.figure(figsize=(20, 2))
for i in range(1,10):
    ax = plt.subplot(1, 10, i)
    plt.imshow(x_test_noisy[i].reshape(28, 28), cmap="binary")
plt.show
```



MACHINE LEARNING MODEL ON IMAGE DENOISING USING NEURAL NETWORK:

REQUIRED LIBRARIES:

- TensorFlow and Keras are essential for building and training neural networks.
- NumPy is used for numerical operations, especially in manipulating arrays.
- Matplotlib is used for visualizing images.

```
from tensorflow.keras.datasets import mnist
from tensorflow.keras.layers import Conv2D, MaxPooling2D, UpSampling2D
from tensorflow.keras.models import Sequential
import numpy as np
import matplotlib.pyplot as plt
```

```
model = Sequential()
model.add(Conv2D(32, (3, 3), activation='relu', padding='same', input_shape=
(28, 28, 1)))
model.add(MaxPooling2D((2, 2), padding='same'))
model.add(Conv2D(8, (3, 3), activation='relu', padding='same'))
model.add(MaxPooling2D((2, 2), padding='same'))
model.add(Conv2D(8, (3, 3), activation='relu', padding='same'))


model.add(MaxPooling2D((2, 2), padding='same'))



model.add(Conv2D(8, (3, 3), activation='relu', padding='same'))
model.add(UpSampling2D((2, 2)))
model.add(Conv2D(8, (3, 3), activation='relu', padding='same'))
model.add(UpSampling2D((2, 2)))
model.add(Conv2D(32, (3, 3), activation='relu'))
model.add(UpSampling2D((2, 2)))
model.add(Conv2D(1, (3, 3), activation='relu', padding='same'))
```

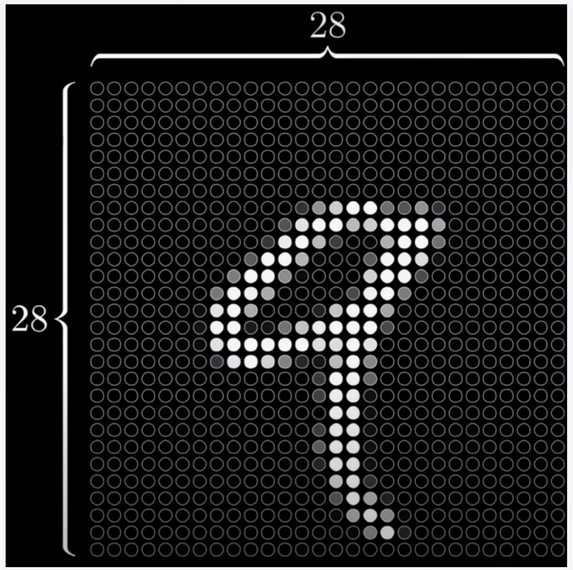
```
model.compile(optimizer='adam', loss='mean_squared_error')

model.summary()


model.fit(x_train_noisy, x_train, epochs=10, batch_size=256, shuffle=True,
 validation_data=(x_test_noisy, x_test))

model.evaluate(x_test_noisy, x_test)
```

loss: 0.0237

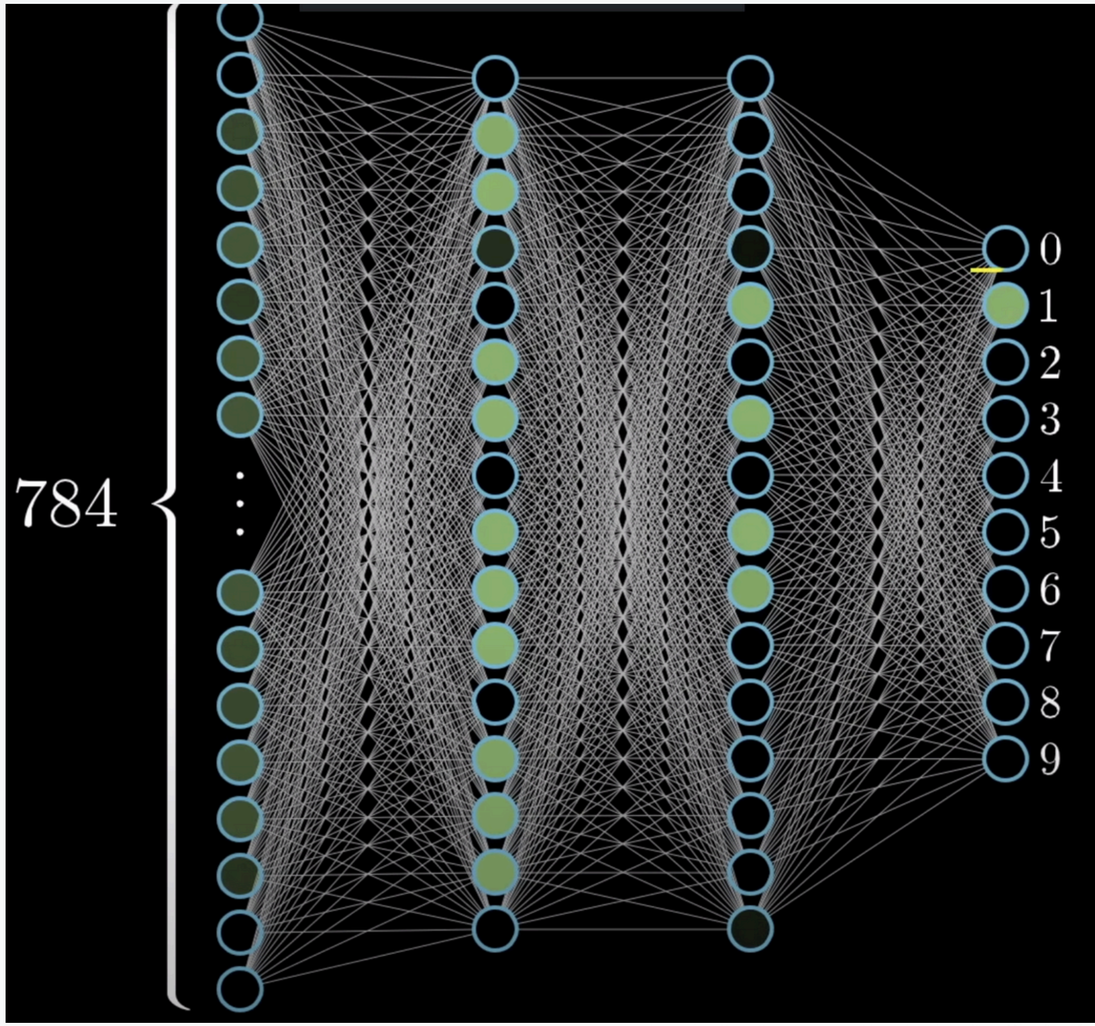
NEURAL NETWORKS:

INSPIRED BY HUMAN BRAIN

.Neuron - Thing that holds a number bw 0 and 1



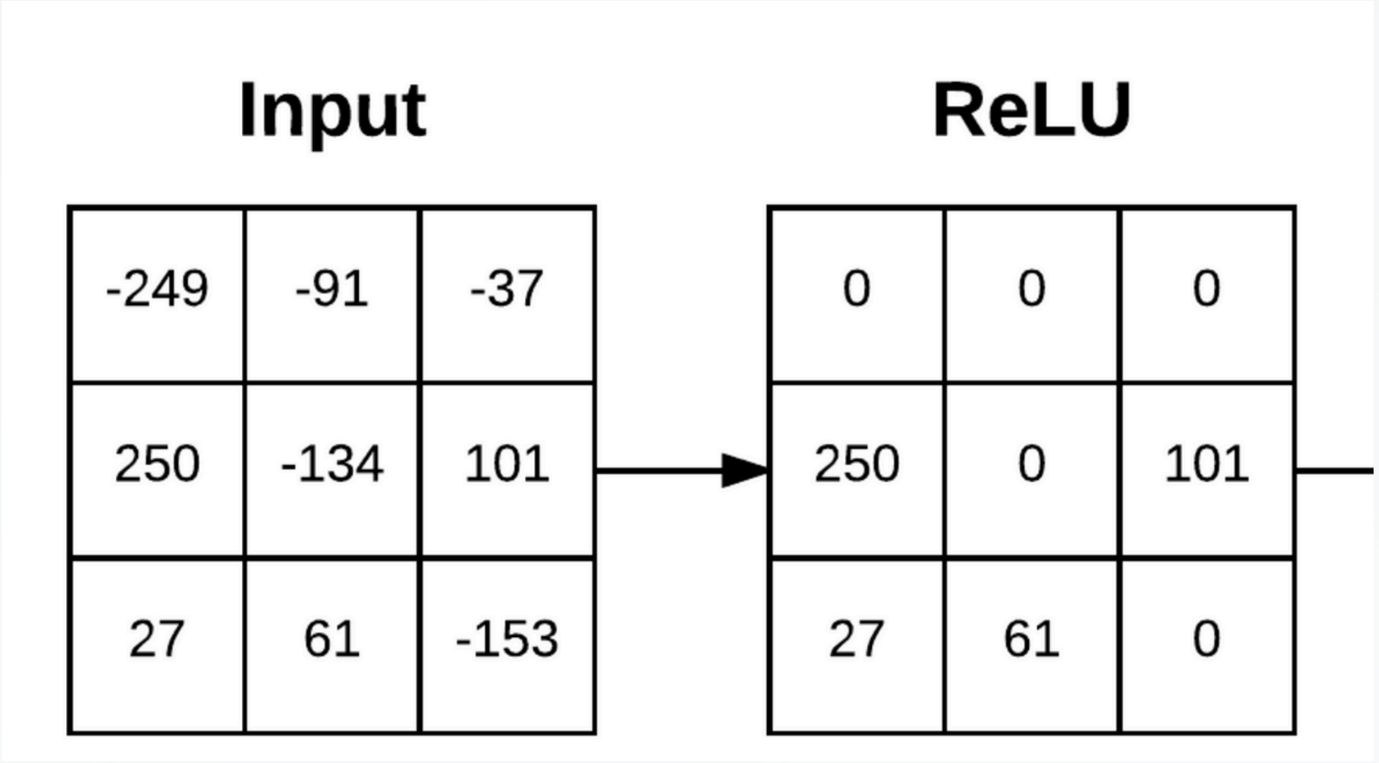
Each of the circle holds the grey scale value of the corresponding pixels



CONVOLUTIONAL LAYERS

Activation Layer

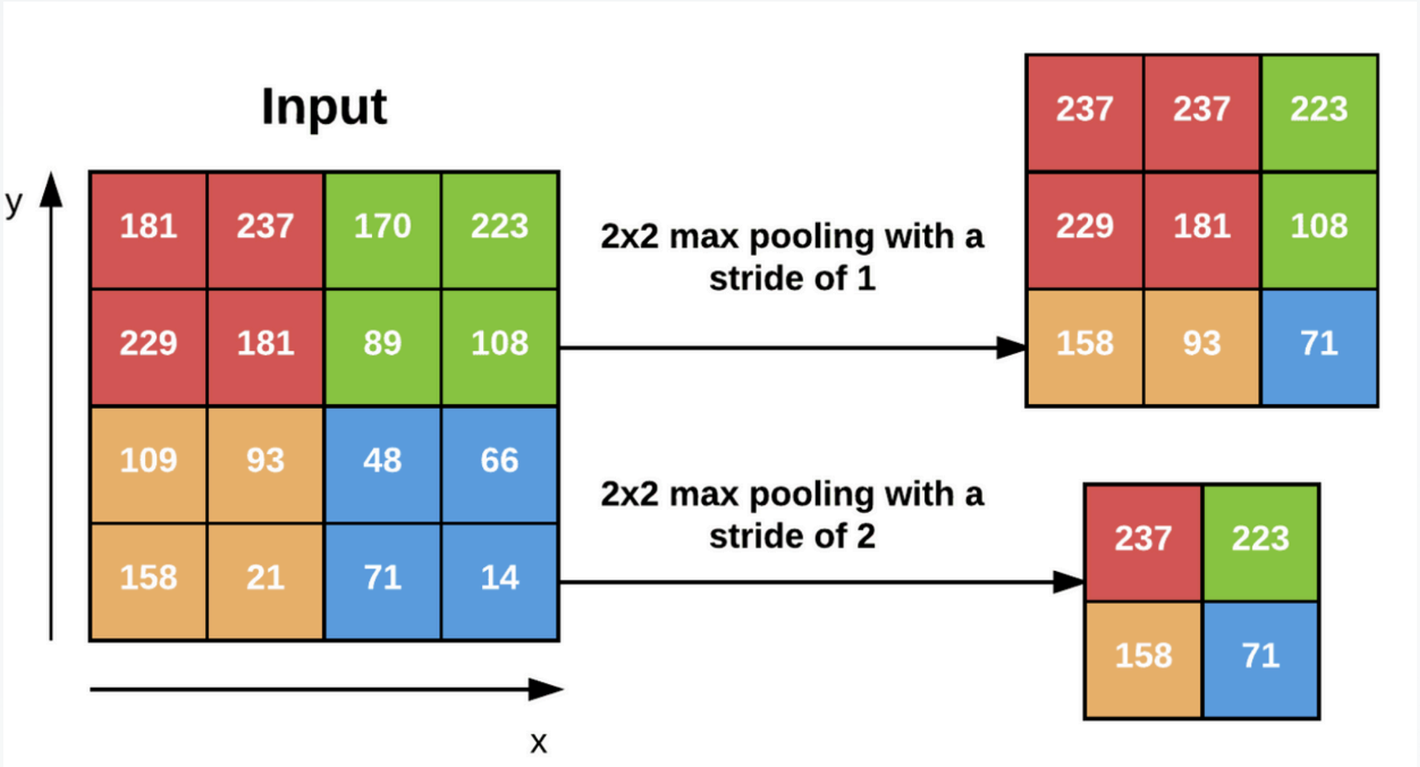
An activation layer accepts an input volume of size Winput×Hinput×Dinput and then applies the given activation function. Since the activation function is applied in an element-wise manner, the output of an activation layer is always the same as the input dimension, Winput = Woutput, Hinput = Houtput, Dinput = Doutput.



Pooling Layer

There are two methods to reduce the size of an input volume — CONV layers with a stride > 1 (which we've already seen) and POOL layers. It is common to insert POOL layers in-between consecutive CONV layers in a CNN architectures:

The primary function of the POOL layer is to progressively reduce the spatial size (i.e., width and height) of the input volume. Doing this allows us to reduce the amount of parameters and computation in the network — pooling also helps us control overfitting.

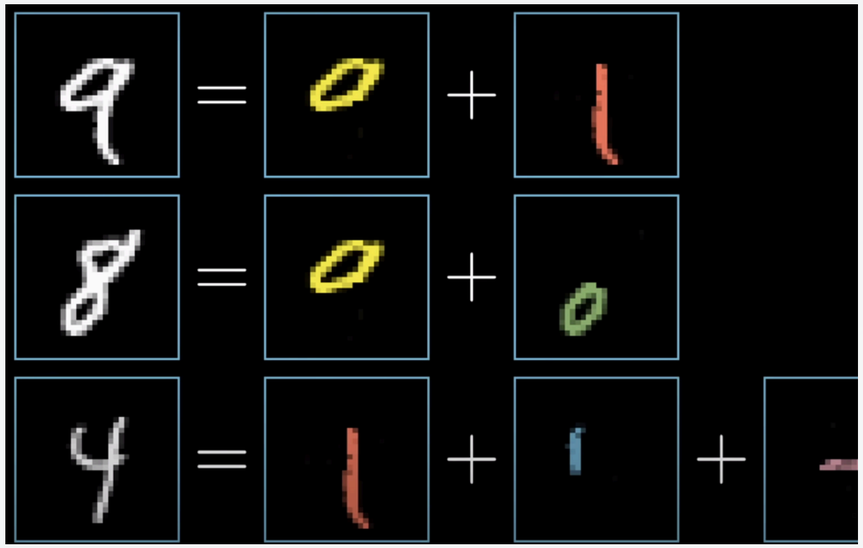


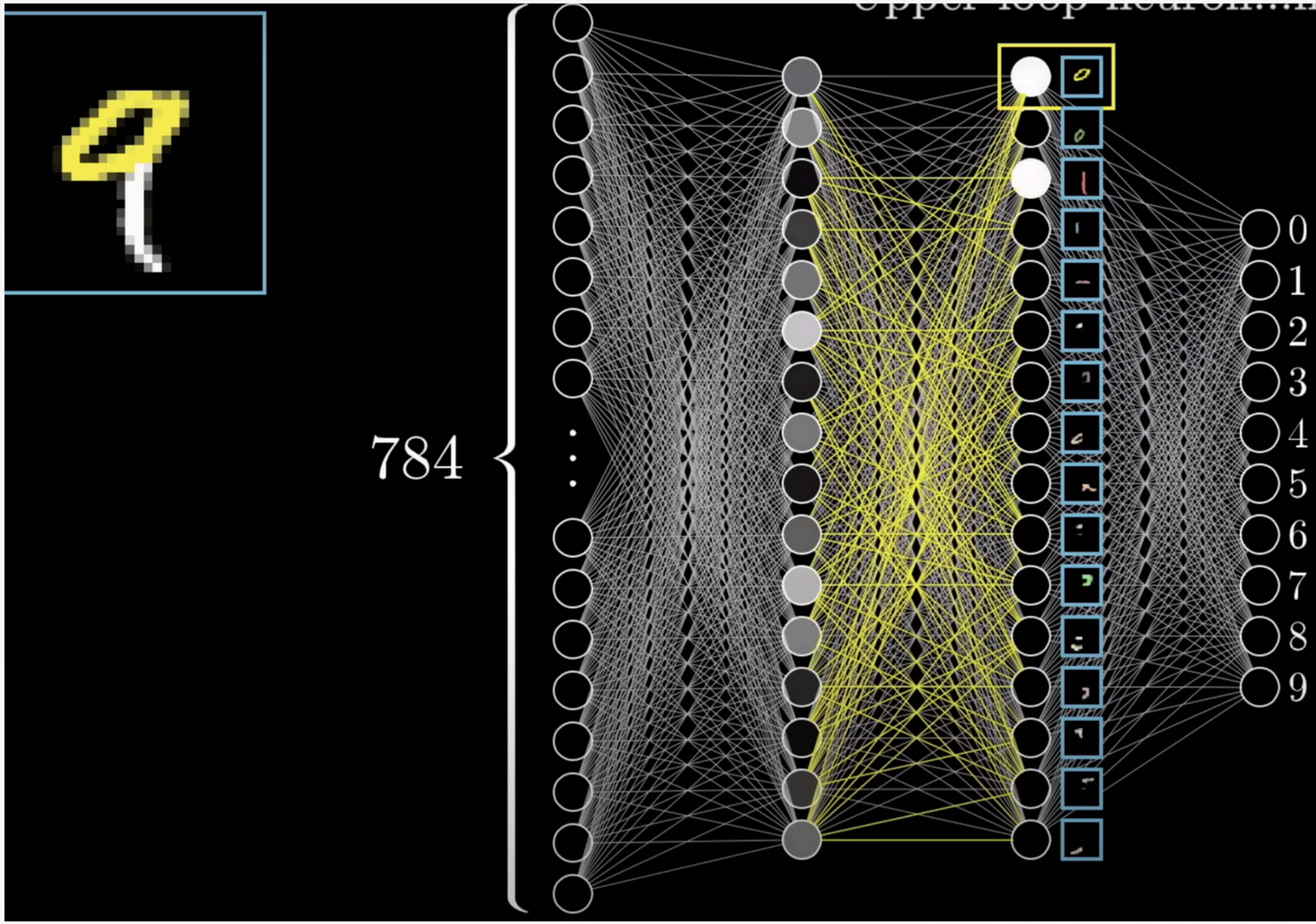
Fully Connected Layers

- Neurons in FC layers are fully connected to all activations in the previous layer, as is the standard for feedforward neural networks.
- FC layers are always placed at the end of the network (i.e., we don’t apply a CONV
- layer, then an FC layer, followed by another CONV layer.

A fully connected layer refers to a neural network in which each neuron applies a linear transformation to the input vector through a weights matrix. As a result, all possible connections layer-to-layer are present, meaning every input of the input vector influences every output of the output vector.

CONT.





CONCLUSION

The project report serves as a comprehensive guide to Digital Image Processing (DIP), covering fundamental concepts, techniques, and applications across industries. It emphasizes image denoising methods, from traditional approaches to advanced neural network-based models using TensorFlow and Keras. With real-world examples in medicine, satellite imaging, and security, the report highlights DIP's practical relevance. It emerges as a valuable resource for understanding DIP principles and techniques, particularly in the context of neural network-driven denoising, offering insights into its evolving landscape and applications.