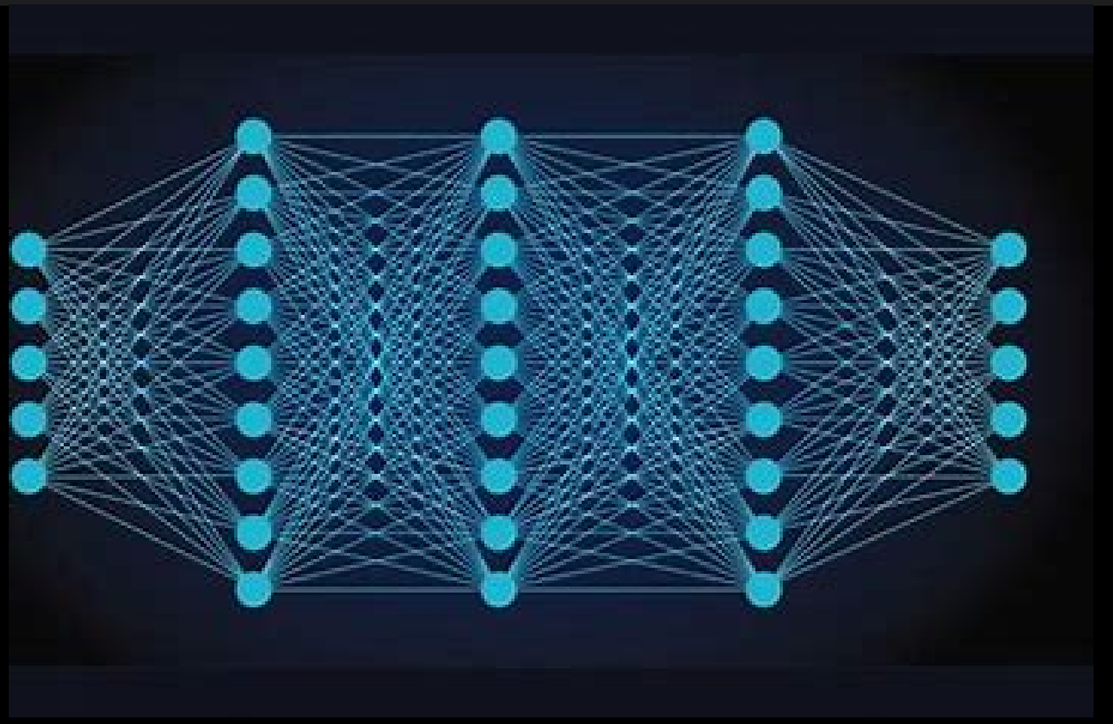
Artificial Intelligence 101

Mid - Term Evaluation

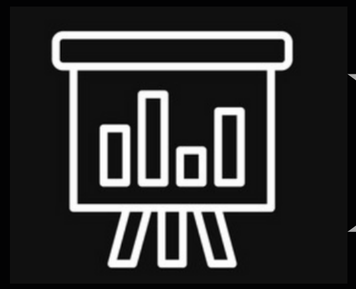
- Harnessing the power of interconnected layers to unravel mysteries in AI, from perceiving the unseen to understanding the unsaid.



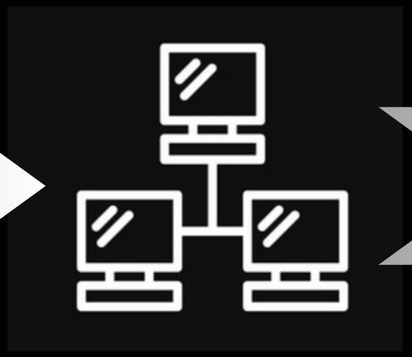
Watt Wizards
By Group - 6

Intro To ML

How does it work?



We give the data to our model



And the model makes predictions

Could I make it?

TYPES

Supervised Learning

The model learns from labeled data to predict outcomes.

Examples:
Regression: Predict house prices.
Classification: Email spam detection.

Semi-Supervised Learning

The model learns from a mix of labeled and unlabeled data

Example:
Text categorization with partially labeled documents.

Unsupervised Learning

The model finds hidden patterns or groupings in data without labels.

Examples:
Clustering: Customer segmentation.
Dimensionality Reduction: PCA for visualization.

Maximum Likelihood Estimation (MLE)

A statistical method to estimate the parameters of a model by maximizing the likelihood function.

Goal: Find parameter values (θ) that maximize the likelihood of observed data (X)

Steps-

Define a probabilistic model:

Example: Assume data follows a Normal distribution.

Write the likelihood function

Maximize the likelihood (or log-likelihood)

Solve for μ and σ:

- $\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$ (mean).
- $\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)^2$ (variance).

Ex:- For Normal distribution:

$$L(\mu, \sigma | X) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x_i - \mu)^2}{2\sigma^2}\right)$$

CLASSIFICATION IN ML

One of the fundtamental task in ML which involves predicting the category or the class label of an input based on its features

REQUIREMENTS?

- Labelled Data
- Feature Selection and Representation
- Algo(like SVM ,logistic regression) and Evaluation matrices(like Euclidean matrice)

Linear Vs Non-Linear Classifiers

Linear classifiers assumes that classes can be seperated by a straight line or a hyperplane while non-linear creates a more complex curve and do not assume linear boundaries.

K-Nearest Neighbour

Non-Linear Classifer which classifes the input data by looking at k-nearest data points to it from the training data.The algorithm uses distance matrices (like Euclidean Distance) to measure the distance between data points.

Support Vector Machines(SVMs)

Linear Classifier which aims to find the optimum margin line(or Hyperplane) that seperates data points of different class.Unlike,Knn it does not stores the tarining data.

How SVM Works?

Finds the hyperplane w.x + b =0,that seperates the data,maximising the margin

For sepearble data,the optimisation and the constraints are :-

$$\min_{\gamma,w,b} \quad \frac{1}{2}||w||^2$$
$$\text{s.t.} \quad y^{(i)}(w^T x^{(i)} + b) \geq 1, \quad i = 1, \ldots,$$

TYPES

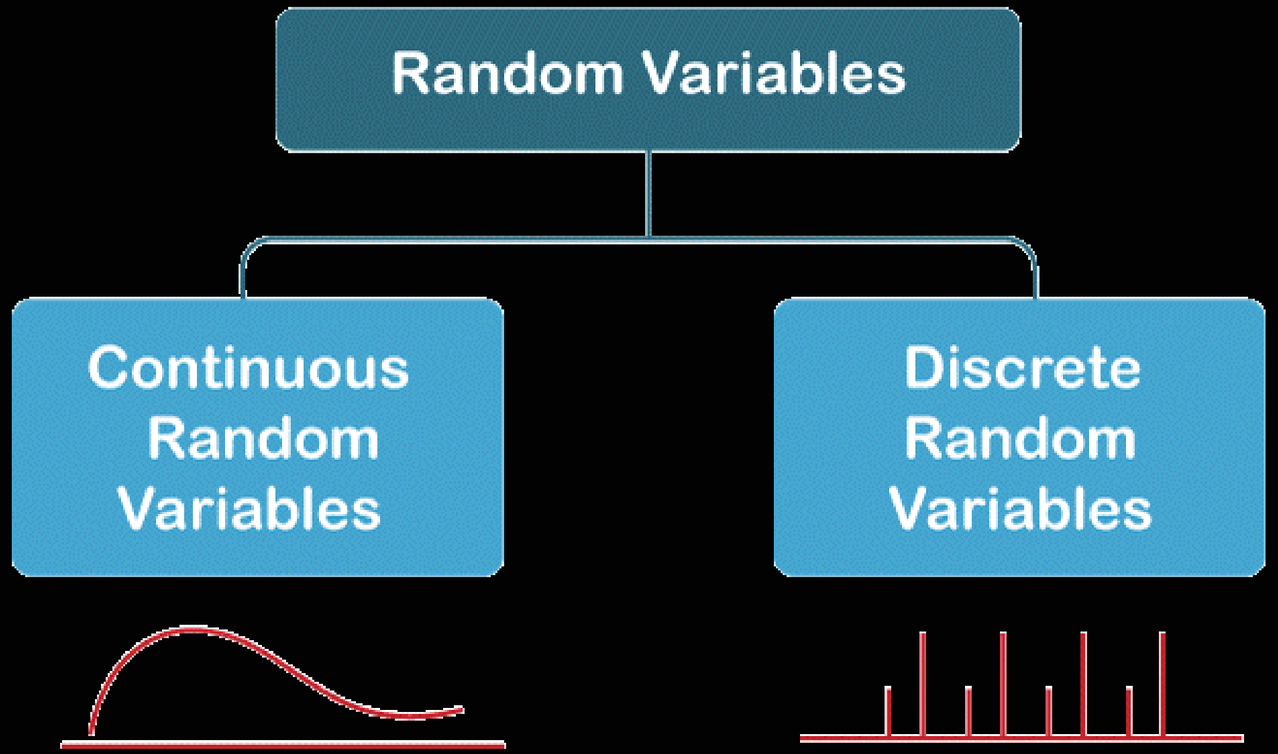
Hard SVM:-Assumes data is perfectly linearly seperable and finds a hyperplane with no misclassification.

Soft SVM:-Allows for misclassifiaction and finds a hyperplane with maximum margin but with a trade off between maximising the margin minimizing misclassification.

Intro to Random variable

In probability, a real-valued function, defined over the sample space of a random experiment, is called a **random variable**.

Example:
Consider the random experiment of tossing a coin S={H,T}
Define Random variable X(H)=1, X(T)=0



Discrete variable	Continuous variable
Countable support	Uncountable support
Probability mass function	Probability density function
Probabilities assigned to single values	Probabilities assigned to intervals of values
Each possible value has strictly positive probability	Each possible value has zero probability

PDF Transformation - CDF Apporach

Given!!!

C.R.V. X with pdf fX(x) and Y = u(X) , u(X) need to be invertible

To find PDF of Y

PDF of Y

$$f_Y(y) = \frac{d}{dy} F_Y(y)$$

$$f_Y(y) = \frac{d}{dy} F_X(u^{-1}(y))$$

$$f_Y(y) = f_X(u^{-1}(y)) \cdot \frac{d}{dy} u^{-1}(y)$$

$$f_Y(y) = f_X(u^{-1}(y)) \cdot \left| \frac{du^{-1}(y)}{dy} \right|$$

CDF of Y

$$F_Y(y) = P[Y \leq y]$$

$$F_Y(y) = P[u(X) \leq y]$$

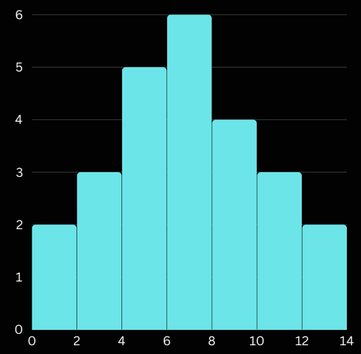
$$F_Y(y) = P[X \leq u^{-1}(y)]$$

$$F_Y(y) = F_X(u^{-1}(y))$$

Result

$$f_Y(y) = f_X(u^{-1}(y)) \cdot \left| \frac{du^{-1}(y)}{dy} \right|$$

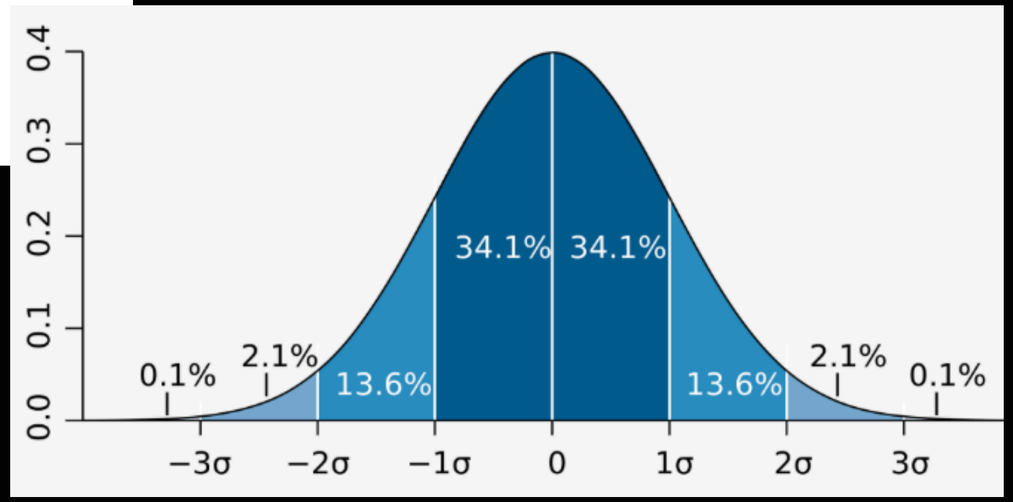
Central Limit Theorem



1 Lindeberg-Levy Theorem

Theorem 1 *Let* $\{X_n\}$ *be a sequence of i.i.d. r.v.s such that* $\mathbb{E}[X_k] = 0$, $Var[X_k] = \sigma^2 < \infty$ *and define* $S_n = \sum_{k=1}^{n} X_k$. *Then as* $n \to \infty$,

$$\frac{S_n}{\sigma\sqrt{n}} \xrightarrow{d} \tau \text{ where } \tau \sim N(0,1).$$



1 All Xi's are independent of each other.

2 Each Xi is drawn from the same distribution.

3 Finite Variance ($0<Var(Xi)<\infty$).

Cumulative Distribution Function:

Function that describes the probability that a random variable is less than or equal to a given value. It is usually represented as $F(x)=P[X\leq x]$.

Properties:

Never decreasing
$F(x)<=F(y)$ for $x<y$

F is right continous
$F(x)=F(x^+)$

$\lim_{x\to-\infty} F(x)=0$

$\lim_{x\to\infty} F(x)=1$

f(x) must satisfy these 2 conditions. For discrete in place of integration summation take place

. We require:

$$f(x) \geq 0 \quad \text{for all } x.$$

. We require:

$$\int_{-\infty}^{\infty} f(x)\,dx = 1.$$

Relationship between pdf and cdf

The cdf is obtained from the pdf through integration:

$$F(x) = P(X \leq x) = \int_{-\infty}^{x} f(t)\,dt \quad \text{for all } x.$$

The pdf is obtained from the cdf through differentiation:

$$f(x) = F'(x).$$

Expectation (mean):

E(X)pectation of a Discrete Random Variable

$$E(X) = \sum_x xP(x)$$

E(X)pectation of a Continuous Random Variable

$$E(X) = \int_{-\infty}^{\infty} xP(x)dx$$

σ^2 = ∑ (Xi – u)^2/n

- *n is the total number of observation*

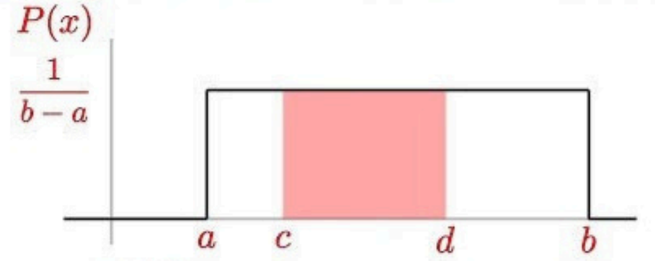
σ= E(X²)−(E(X))²

- σ denotes variance
- u=E(x)

Properties of Expectations

1. E[c] = c
2. E[aX + b] = aE[X] + b
3. $\sigma^2(ax + b) = a^2\sigma^2$
4. E[g(x)] = $\int g(x)dF(x)$

Uniform Distribution



$Mean: \quad \mu = \frac{a+b}{2}$

$S.D.: \quad \sigma = \sqrt{\frac{(b-a)^2}{12}}$

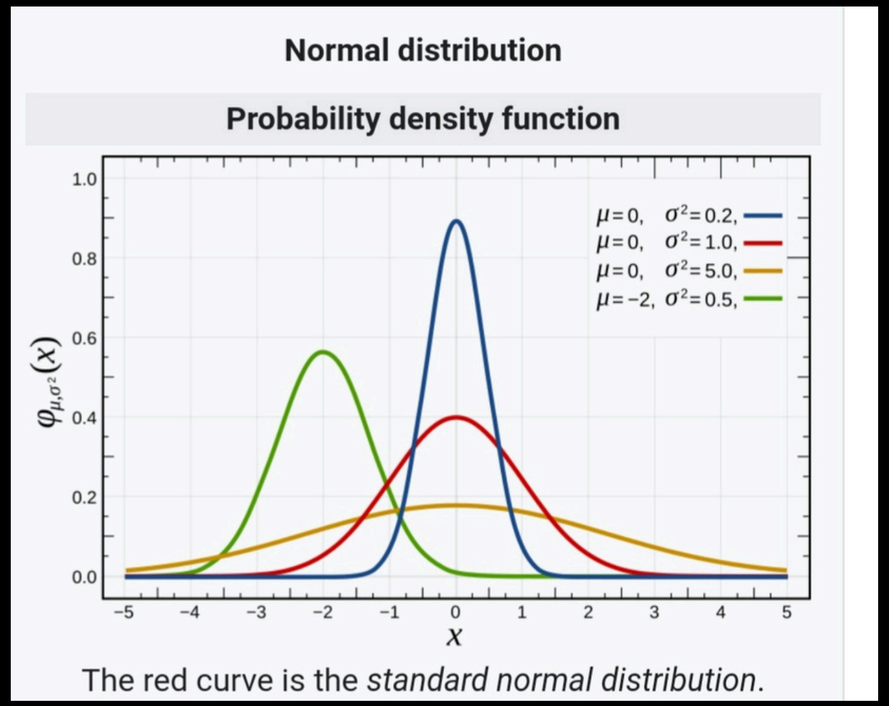
Probability

$P(c \le X \le d) = \frac{d-c}{b-a}$

Probability Density Formula for Normal Distribution

$$f(x, \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

- In standard normal distribution u=0 and σ=1
Symmetric about origin



The red curve is the *standard normal distribution*.

The CDF of the standard normal distribution is denoted by the Φ function:

$$\Phi(x) = P(Z \leq x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} \exp\left\{-\frac{u^2}{2}\right\} du.$$

Marginal Distribution Function

Continuous Random Variable

If the joint distribution of the random variables X, Y is F[*x*, *y*], then

the marginal distribution of X is

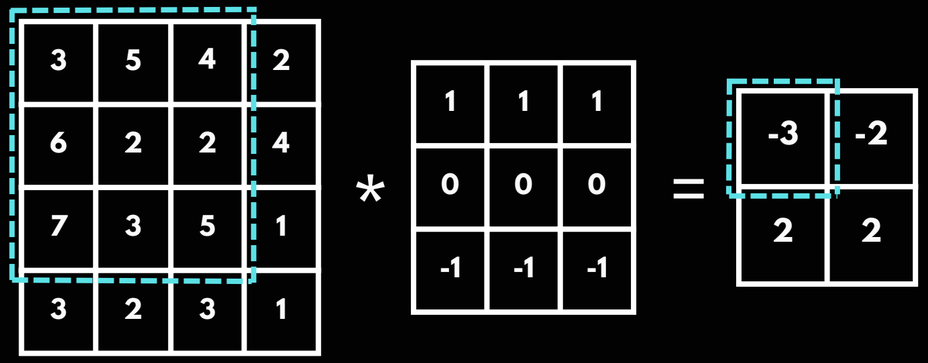
$$F_X(x) = \int_{-\infty}^{x} \int_{-\infty}^{\infty} f(x, y) dy dx$$

the marginal distribution of Y is

$$F_Y(y) = \int_{-\infty}^{y} \int_{-\infty}^{\infty} f(x, y) dx dy$$

Convolutional Neural Networks

Convolution



The filter slides over the input and computes the element-wise product between the filter and the portion of the input it covers.
Extracts local features and reduces dimensions.

Padding

Padding involves adding extra pixels (usually zeros) around the input image to control the spatial dimensions of the output after convolution.

Stride

Stride refers to the step size the filter moves while scanning the input data. Larger stride values result in filter moving more quickly across the input.

Pooling

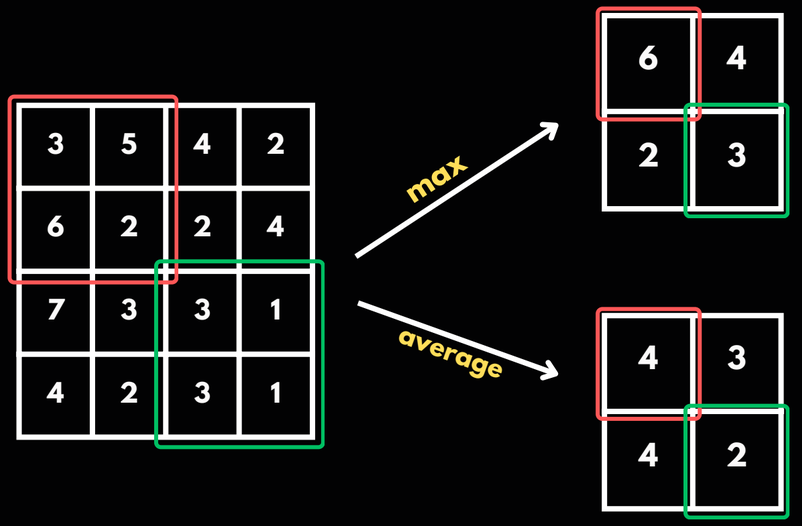
Pooling is used to reduce the size of an image or feature map while keeping important information. This helps the network learn more efficiently by focusing on the most important features and reducing unnecessary details.

Max Pooling

For each small region (like a 2x2 square), max pooling takes the largest number.

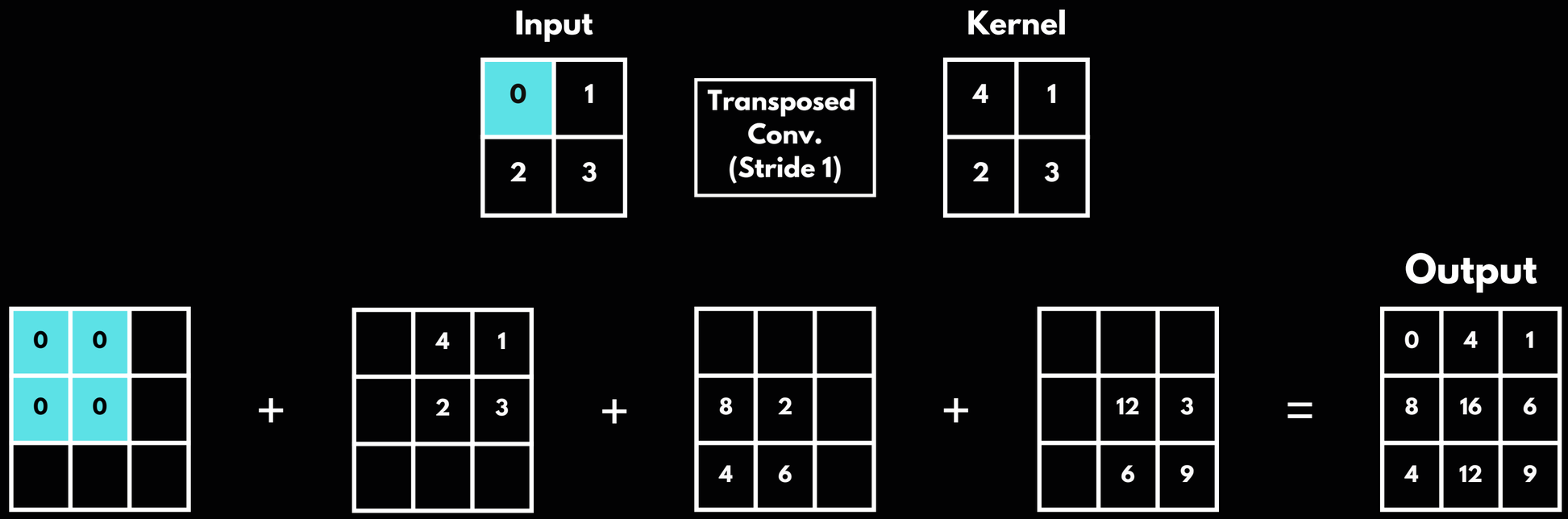
Average Pooling

For each small region (like a 2x2 square), average pooling takes the average.



Transposed Convolution

Transposed convolution can be thought of as the reverse of regular convolution. While convolution is designed to reduce spatial dimensions, transposed convolution is used to expand these dimensions. It "reverses" the effect of downsampling by generating more detailed information in a larger spatial area, allowing the network to recreate higher-resolution features from lower-dimensional representations.
It is often used for upsampling smaller feature maps, such as in image-to-image translation tasks, where the goal is to take a compressed or lower-dimensional representation and generate a high-resolution output.

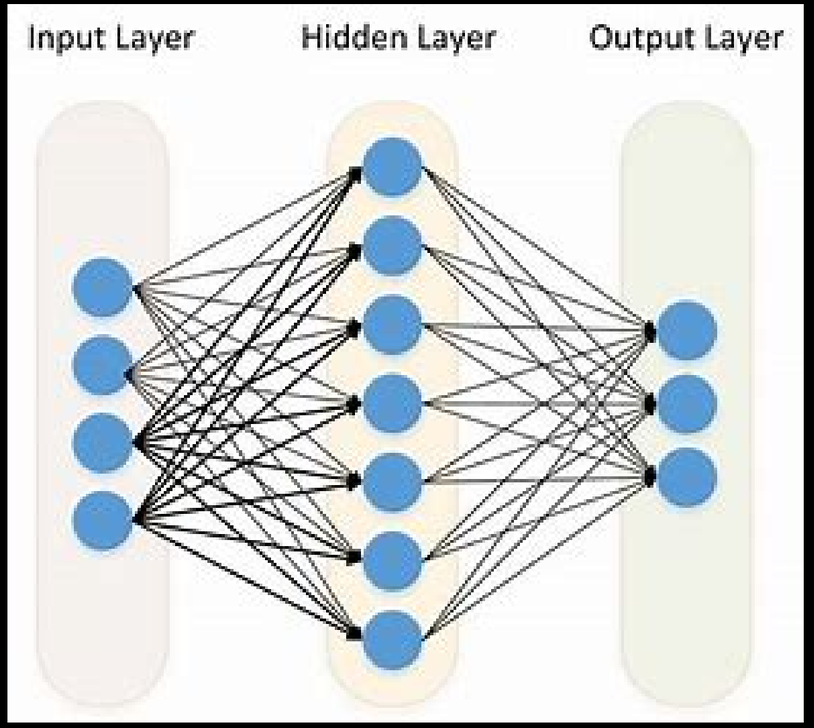


Neural Network

NN Models are computational models inspired by the brain's interconnected neurons for information processing.

- **They consist of three main** layers:input, hidden and output layers.
- NN are widely employed in image recognition, speech processing and various ML tasks.

Training involves adjusting weights and biases among the neurons and using feature maps to easily analyze the data and the use of an activation function to achieve our result.



Backpropagtion

- Purpose: Backpropagation is used to train neural networks by optimizing weights and biases to minimize the error.

Process overview

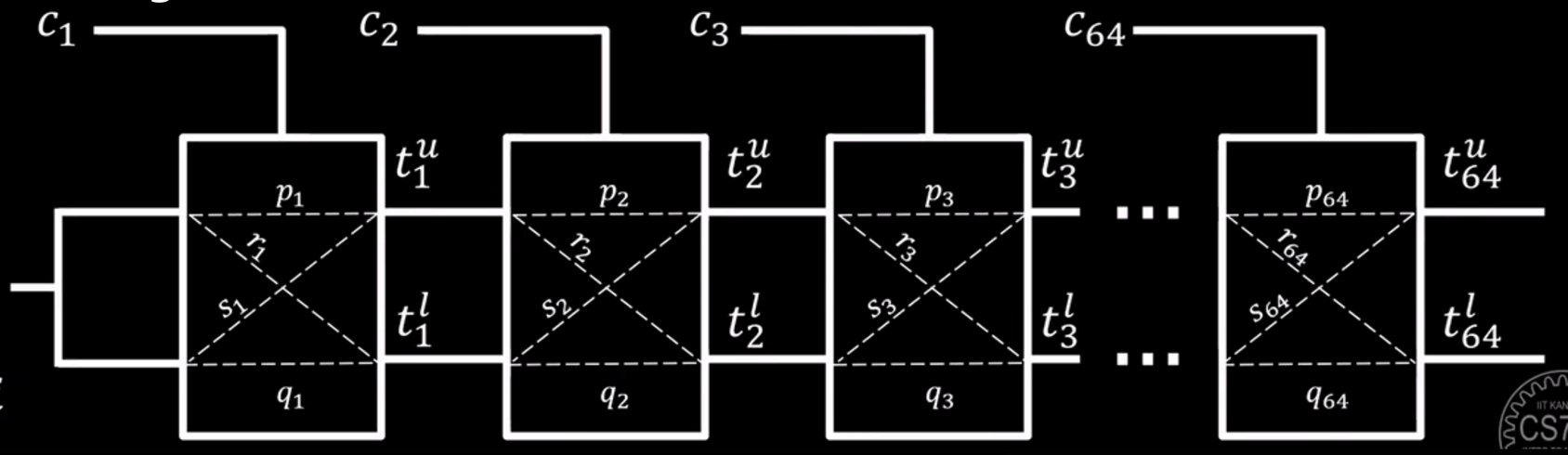
Forward Pass
Input flows through the network.

Loss Calculation
Compare predictions with actual values.

Backward Pass
Calculate gradients using the chain rule.

Weight Update
Adjust weights and biases using gradients and learning rate.

PUFs - Physically Unclonable Functions.



$$\Delta_i = t_i^u - t_i^l$$

$$\Delta_i = d_i \cdot \Delta_{i-1} + \alpha_i \cdot d_i + \beta_i$$

We can keep going on recursively

$\Delta_1 = \alpha_1 \cdot d_1 + \beta_1$ (since $\Delta_0 = 0$)

$\Delta_2 = \Delta_1 \cdot d_2 + \alpha_2 \cdot d_2 + \beta_2$: now plugin value of Δ_1 to get

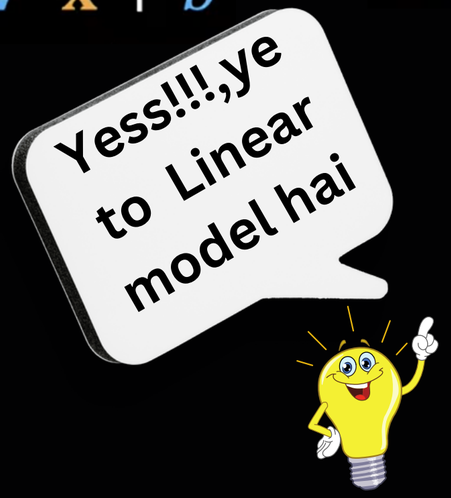
$$\Delta_2 = \alpha_1 \cdot d_2 \cdot d_1 + (\alpha_2 + \beta_1) \cdot d_2 + \beta_2$$

$$\Delta_3 = \alpha_1 \cdot d_3 \cdot d_2 \cdot d_1 + (\alpha_2 + \beta_1) \cdot d_3 \cdot d_2 + (\alpha_3 + \beta_2) \cdot d_3 + \beta_3$$

$$\Delta_4 = \alpha_1 \cdot d_4 \cdot d_3 \cdot d_2 \cdot d_1 + (\alpha_2 + \beta_1) \cdot d_4 \cdot d_3 \cdot d_2 + (\alpha_3 + \beta_2) \cdot d_4 \cdot d_3 + (\alpha_4 + \beta_3) \cdot d_4 + \beta_4$$

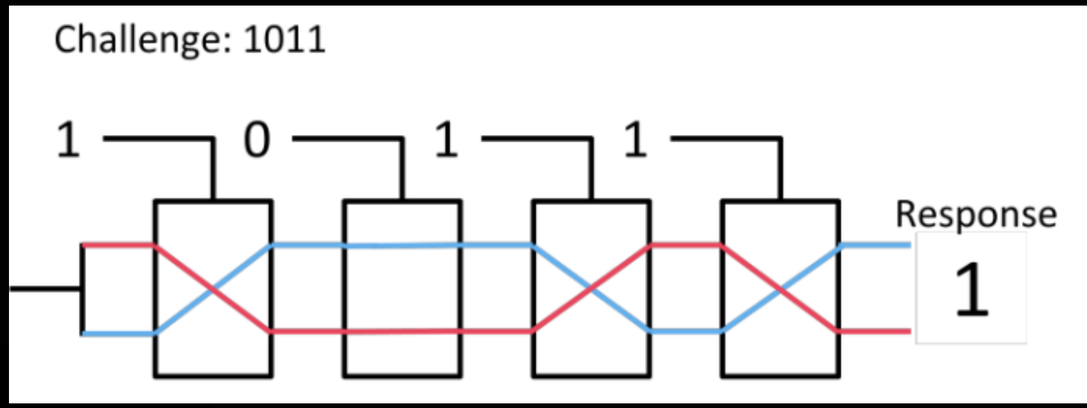
It turns out that

$$\Delta_{64} = \mathbf{w}^{\mathsf{T}}\mathbf{x} + b$$



$$\text{if } \Delta_{64} < 0 \rightarrow$$  , else 

A Simple Arbiter PUF.



where $\mathbf{x} = [x_1, \ldots, x_{64}], \mathbf{w} = [w_1, \ldots, w_{64}] \in \mathbb{R}^{64}$

$$x_i = d_i \cdot d_{i+1} \cdot \ldots \cdot d_{64}$$
$$= (1 - 2c_i) \cdot (1 - 2c_{i+1}) \cdot \ldots \cdot (1 - 2c_{64})$$
$$w_1 = \alpha_1$$
$$w_i = \alpha_i + \beta_{i-1} \text{ (for } i = 2,3, \ldots, 64)$$
$$b = \beta_{64}$$

Thus, response is simply $\dfrac{\text{sign}(\mathbf{w}^{\mathsf{T}}\mathbf{x}+b)+1}{2}$

$$d_i \stackrel{\text{def}}{=} (1 - 2c_i)$$
$$\alpha_i \stackrel{\text{def}}{=} (p_i - q_i + r_i - s_i)/2$$
$$\beta_i \stackrel{\text{def}}{=} (p_i - q_i - r_i + s_i)/2$$

Thank You!

Mentees:-

Kaushal Mehra, 230546
Rohit Singh, 230870
Sarthak Shekhar, 230929
Nitesh Kumar, 230707
Karan Kumar, 230533
Naman Patidar, 230679
Krishna Agarwal, 230574
Shital Niras, 230967
Vaibhav Chauhan, 231113
Akshat Dhote, 230095
Dishank Kumawat, 230377

Mentors:-

K.N. Joshua
Nikhil Jain
Aravind