CMR Engineering college

UGC Autonomous

IV BTech - I - Semester End Examinations (Regular) - Dec - 2025

SUBJECT: Deep Learning

SUBJECT CODE: R22CY701PC

Branch : CSC

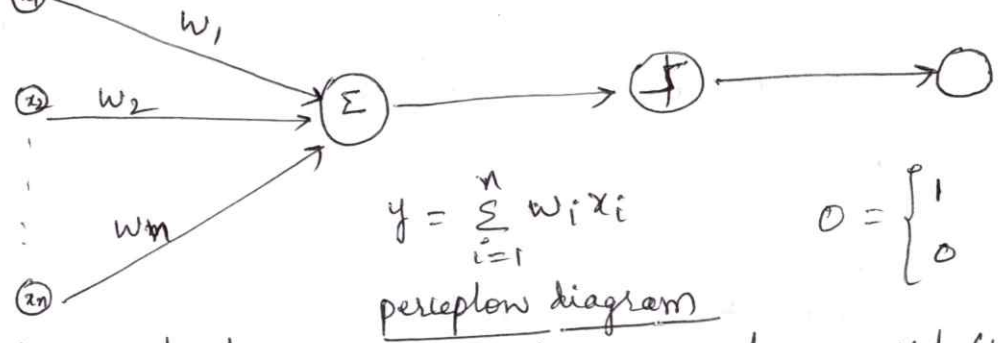
Time: 3 Hrs | Max Marks: 60

Note:- This question paper contains two parts A and B
part A is compulsory which carries 10 marks. Answer all questions in part A
part B consists of 5 units. Answer any one full question from each unit. Each question carries 10 marks and may have a, b, c as sub questions.

PART - A

1 (a) Define perceptron.

(A) perceptron is a basic unit in Artificial Neural Network. It is a single layer neural network designed for binary classification tasks.



perceptron diagram

perceptron takes input data (x_1 to x_n) with corresponding weights (w_1 to w_n) and activation function to output a result

(b) state any two uses of Semi Supervised Learning?

(A) Two uses of Semi-supervised learning are fraud detection in finance and medical image analysis in health care.

(C) Define fixed weight competitive Net?

(A) Fixed weight competitive networks are neural networks where neurons compete based on a fixed set of weights to enhance contrast or select the strongest activation. These networks are used for tasks like pattern recognition and clustering.

(d) what is meant by Adative Resonance Theory (ART)

(A) Adative Resonance Theory (ART) is a neural network model allowing a system to learn new information without forgetting old knowledge. It uses unsupervised learing to perform tasks like pattern recognition and clustering.

(e) what is the main purpose of ~~the gradient decent algorithm~~ output layer in a feed forward network

(A) ~~The main purpose of the gradient decent algorithm is to find the minimum of a function, typically a loss function or cost function in machine learning.~~

(A) The main purpose of the output layer in a feed forward Network is to produce the final predictions or results of the network by transforming the processed data from the previous layer.

It generates a final output / prediction of the task.

(f) state the purpose of the gradient decent algorithm.

(A) The purpose of the gradient decent algorithm is to minimize a given cost or loss function (error) by iteratively adjusting the parameters of a model.
It finds the optimal set of parameters (like weights and biases) to reduce the error between models predictions and the actual target values.

(g) what is meant my parameter norm penalty?

(A) parameter norm penalty is a regularization technique in deep learning that adds a penalty term to the loss function based on the magnitude (norm) of the model's parameter, encouraging smaller parameter values to prevent overfitting.
These penalties are typically applied to the weights not the biases of the network.

(h) why is data augmentation used during training?

(A) Data Augmentation is a technique in deep learning that artificially expands a dataset by creating modified versions of existing data by applying various transformations such as a rotation, flipping or cropping to generate new dataset.
Data augmentation is used during training to reduce overfitting, enhance model generalization and improves the performance

i) What is the advantage of Adam optimizer.

(A) The main advantage of the Adam optimizer are faster convergence, automatic adaptive learning rates for each parameter and robustness. It efficiently handles sparse or noisy gradient, computationally efficient and requires little memory. It is suitable for large datasets and complex models.

j) List the names of Second order methods?

1. Newtons Method
2. BFGS (Broyden Fletcher Goldfarb Shanno)

Second order optimization methods in deep learning utilize information from the Hessian matrix to guide the optimization process.

Newston method

This method directly uses the Hessian matrix and the gradient to determine the optimal step direction and size by minimizing a quadratic approximation of the loss function.

BFGS

This method approximates the inverse Hessian

PART-B

2. Explain how supervised learning networks differs from unsupervised learning networks in ANN and basic models of ANN

(A) Supervised learning Networks

In Supervised learning Networks train the model with labelled data.

Two techniques are using in supervised learning

1. classification
2. Regression

classification

This technique is cured for classification of data

EX:- Spam prediction, Tumor prediction, Image classification

Algorithms :- KNN, Decision tree (ID_3), Random Forest, logistic regression, Linear Regression

These networks are trained on datasets where each input is associated with a corresponding correct output label

For example in an image classification task an image of a dog would be labelled as dog

The goal is to learn a mapping function from inputs to outputs allowing the network to accurately predict the output for new unseen inputs.

The network adjusts its weights and biases based on the error between its predicted output and actual output.

Regression :- This is used for continuously changing data prediction Ex:- stock price predictio

Unsupervised Learning Network

In unsupervised learning network train the model with unlabelled data.

Two techniques are using in unsupervised learning

(1) clustering
(2) Association

clustering

clustering is an unsupervised machine learning technique that groups similar data points together into cluster based on their characteristics.

Association

This technique is used to discover interesting relation ships or if-then patterns between variables in large datasets

Differences between supervised learning and unsupervised learning networks

Feature	supervised learning networks	unsupervised learning Networks
Data	Labelled	unlabelled
objective	predict outputs, classify data	Discover patterns structures.
Tasks	classification Regression	clustering, Dimensionality Reduction, Associati
feedback	Explicit error feedback (from labels)	Implicit (Based on data structure)

Basic Models of ANN (Artificial Neural Network)

1. Feed Foward Neural Networks (FNN)
2. Recurrent Neural Networks (RNN)
3. Long short term Memory (LSTM)
4. Convolutional Neural Networks (CNN)
5. Generative Adversial Networks (GANs)
6. Auto Encoders

Feed Foward Neural Network (FNN)

In FNN information flows in a single direction from the input layer through any hidden layers to the output layer without any loops or cycles. It is used for tasks like classification, regression and pattern recognition and learns by adjusting weights through algorithms like back propagation

2. Recurrent Neural Networks (RNN)

RNN model is designed for processing sequential data such as text, speech and time series, by using feedback loop that allows it to remember previous inputs and use that memory to make better predictions

Ex:- Speech Recognition, NLP, Time Series Forecasting.

Long short term Memory (LSTM)

LSTM and GRU (Gated Recurrent Unit) are advanced types of RNN designed to address issues with remembering long & short term dependencies.

CNN (Convolutional Neural Network)

CNN is designed to analyze visual data like images and videos. CNN use specialized layers Convolutional pooling and fully connected layers to automatically extract features from data, recognize patterns and classify objects.

Generative Adversial Networks (GANs)

GAN consist of two competing networks, a generator and a discriminator that work together to generate new, realistic data.

Auto Encoders

unsupervised networkes that learn efficient data encodings by compressing input data and then reconstructing it. They are often used for dimensionality reduction and anomaly detection.

3. Describe the architecture and working principle of a Bidirectional Associative Memory (BAM) network.

(A) Bidirectional Associative Memory (BAM) is a type of Recurrent Neural Network that can store and recall pairs of associated patterns, even if they are different in size

It has two fully connected layers that communicate bidirectionally.

Architecture

Two Layers

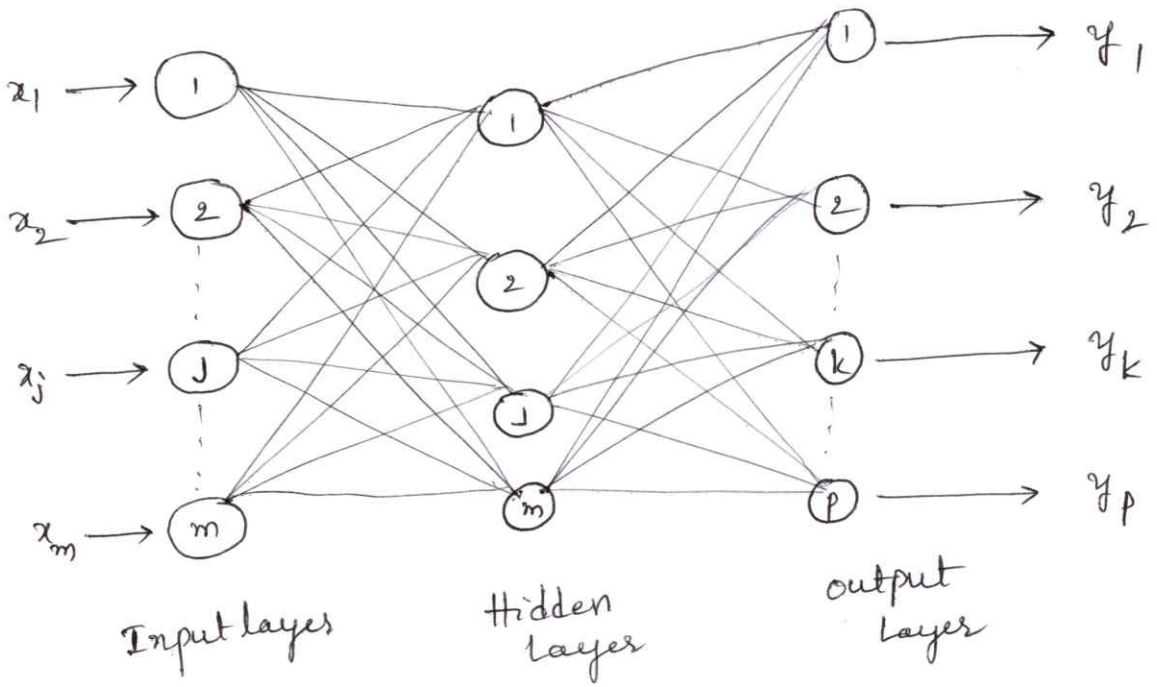
A BAM consists of two layers of neurons typically named layer X and Y

Full interconnectivity

Every neuron in layer X is connected to every neuron in layer Y and vice versa.

Symmetric weight matrix

The connections between the layers are represented by weight matrix. The connection from X to Y is defined by a matrix W and the connection from Y to X is defined by the transpose of that matrix W^T



Working principle

1. Encoding associations

During the learning phase, pairs of bipolar patterns are stored by summing their outer products to form the weight matrix W. This creates a stable memory of the paired data.

2. Forward and backward propagation

when an input is presented it is fed into one layer (ex layer X)

3. Iterative recall

1. The input from X is propagated through the weight matrix W to layer Y. The neurons in Y are updated based on their input.
2. The output from Y is then propagated backward through the transpose weight matrix W^T to layer X. The neurons in X are updated again

4. Convergence

This forward and backward propagation of signals continues iteratively. The network state evolves until it reaches a stable state.

5. Recalling patterns

The final stable state represents the recalled pattern pair, which is the correct association for the given input was incomplete or distorted.

4. Discuss the architecture and working principle of a Maxnet with the help of a neat diagram.

(A) The Maxnet is a type of fixed weight recurrent competitive neural network that uses a winner-take-all (WTA) mechanism to identify the neuron with the largest initial input value. It does not require a training process, as its weights are predefined and fixed.

Architecture

A Maxnet typically consist of the following components

1. Input Layer

Receives the initial input vector. The number of neurons in this layer correspands to the dimension of the input vector.

Competitive Layer (Hidden Layer)

It is the core of the Maxnet. All Neurons with in this layer are fully interconnected. This layer operates on a feedback (recurrent) mechanism.

Self excitation

Each neuron has a positive, self-connecting weight, typically set +1, to maintain its own activation.

Mutual inhibition

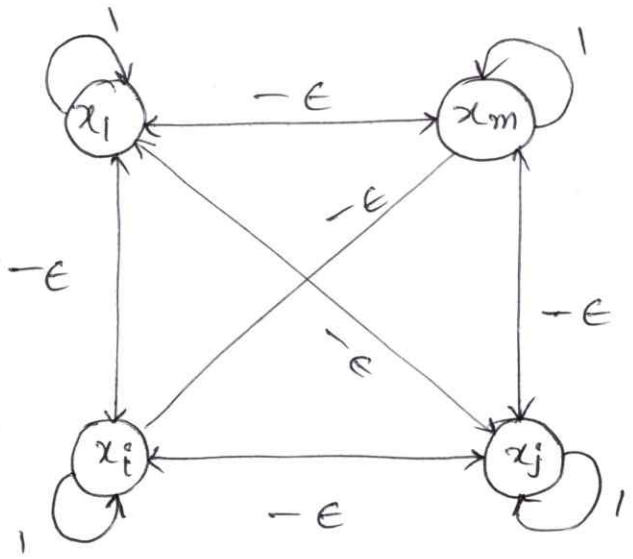
Each neuron has a negative, inhibitory connection to every other neuron in the layer, with a fixed weight typically denoted as $-\epsilon$ (epsilon).

The value of ϵ must be positive and smaller than $1/H$ where H is the total number of neurons in the competitive layer, to ensure stability and proper convergence.

output Layer

This Layer simply outputs the final activation values after the competition has converged. only one neuron (the winner) will have a non zero output.

Diagram Description



A neat diagram of the Maxnet architecture would show

- An input layer with M nodes, feeding into M nodes in a hidden (competitive) layer.
- In the competitive layer, each node has a self loop (positive weight) and connections to all other nodes (negative weights)
- An output layer that displays the final result after the iterative process.

Working principle

The process works as follows

1. Initialization :- An input vector is presented to the network. The initial activation of each neuron in the competitive layer is set to its corresponding input value.
2. Iteration

The network enters a feedback loop. In each

iteration, the activation of each neuron is updated based on its previous activation

1. The activation function used is typically a positive linear function $f(x) = x$ if $x > 0$ and $f(x) = 0$ if $x \leq 0$

2. The update rule ensures that neurons with larger initial inputs reinforce their activation while suppressing others.

competition

The continuous feedback with inhibitory weight causes a winner take all competition. Maximum initial value eventually converge to zero.

stopping condition

The process stops when only one neuron has a nonzero output.

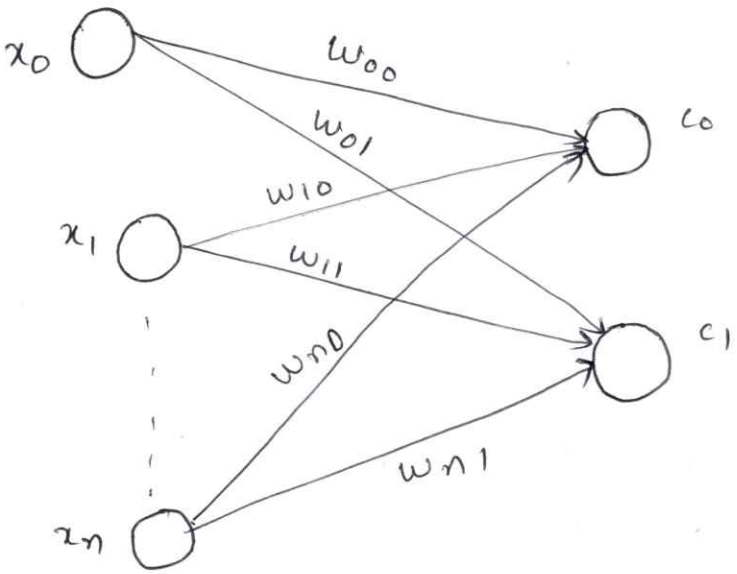
output

The single active neuron indicates the winner which corresponds to the neuron that received the largest initial input signal

5. Explain the structure and functioning of a Kohonen self organizing feature Map and discuss the applications of deep Learning.

(A) A self organizing Map (SOM) or Kohonen Map is an unSupervised neural network algorithm based

on biological neural models from the 1970s. SOM effectively maps high dimentional data to a lower dimensional grid enabling easier interpretation and visualization of complex datasets.



It consists of two primary layer

- Input Layer : Represents the features of the data
- output Layer : Arranged as a 2D grid of neurons with each neuron representing a cluster in the data.

Working of Self Organizing Maps (SOM)

1. Initialization

The weights of the output neurons are randomly initialized. These weights represent the features of each neuron and will be adjusted during training.

2. Competition

For each input vector SOM computes the Euclidean distance between the input and the weight vectors of all

neurons. The neuron with the smallest distance is the winning neuron.

$$D(j) = \sum_{i=1}^{n} (w_{ij} - x_i)^2$$

where

$D(j)$ is the distance for neuron

j and n is the number of features

3. Weight update

The winning neuron's weights are updated to move closer to the input vector. The weights of neighboring neurons are also adjusted but with smaller changes

$$w_{ij}^{(new)} = w_{ij}^{(old)} + \alpha \cdot (x_i - w_{ij}^{(old)})$$

where α is the learning rate

and x_i is the input feature.

4. Learning Rate Decay

The Learning rate α decreases over time allowing the map to converge to stable values.

$$\alpha(t+1) = 0.5\,\alpha(t)$$

5. Stopping condition

The training stops when maximum number of epochs is reached or when the weight Converge

Applications of Deep Learning

1. Image Recognition
2. Natural language processing
3. Autonomous vehicles
4. Healthcare
5. Fraud detection

It is used for tasks like identifying objects in images, translating languages enabling self driving cars to navigate, diagnosing diseases and detecting fraudulent transactions.

6. Demonstrate how Gradient Based Learning is applied in neural networks and explain the use of gradient descent in training them.

(A) Gradient based learning uses the gradient of a cost function to train neural networks by iteratively updating the networks parameters (weights and biases) to minimize errors.

The process involves forward propagation to make a prediction calculating the error with a cost function and then using back propagation to compute the gradient of a cost function with respect to each parameter.

→ Gradient descent then uses this gradient, which

indicates the direction of steepest ascent, to take small steps in the opposite direction (the negative gradient) to find the minimum of the cost function and improves the model accuracy.

Steps in Gradient Based learning in Neural Networks

step 1 :- Forward pass

- Input data is passed through the network
- Each layer computes

$$Z = wx + b \quad \text{and} \quad a = \phi(z)$$

step 2 :- Compute Loss

A cost function (or loss function) is used to measure the difference between the networks prediction and the actual target value.

step 3 :- Back propagation (Backward pass)

This process calculates how much each parameter (weight and bias) in the network. It uses chain rule to compute the gradient of the cost function

step 3 :- parameter update

The gradients are used by the gradient descent algorithm to update the weights and biases.

The use of gradient descent in training

- Gradient Descent overview

At each step determine the steepest direction

Gradient calculation

The gradient is a vector that points in the direction of the steepest increase of the loss function

Iterative process

The process is iterative

1. A prediction is made
2. The loss is calculated
3. Back propagation calculates the gradient of the loss w.r.t to each parameter.
4. Each parameter is updated using the formula
   . new - parameter = old parameter - learning
5. The process repeats until the loss is minimized or a set number of iterations is reached.

Learning Rate (α)

This is a crucial hyper parameter that controls the size of the step taken with each update.

The ultimate goal of this iterative process is to find the set of weights and biases that minimizes the loss function making the models prediction as accurate as possible.

7. Analyze the role of Hidden units and Hidden layers in enhancing the performance of deep learning models and back propagation algorithms.

(A) Hidden units and hidden layers are crucial to the performance of deep learning models by enabling the

network to learn complex, non linear patterns and hierarchical data representations. ~~that~~

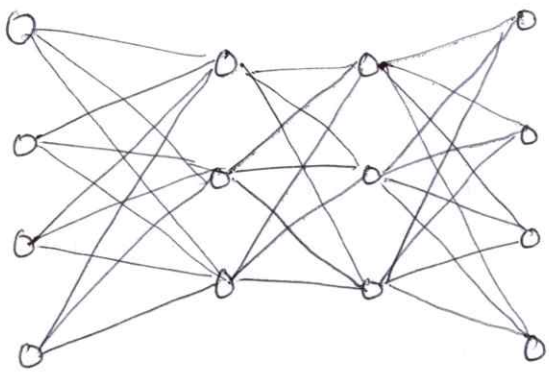
Role in Enhancing performance

- Learning Complex patterns and non linearity

Without hidden layers and their non-linear activation functions (like Relu or Sigmoid), a neural network could only learn ~~dear~~ linear relationships.

- Hierarchical feature Extraction

Hidden layers facilitate hierarchical learning where each successive layer builds upon the features extracted by the previous one



Input Layer Hiden Layers output Layer

Input Layer

Input Layer will take input and then pass the information to Hidden layers

Hidden Layer

This layer is the middle layer between input and output

Each neuron in a hidden layer adds weights to inputs, add bias,
passes the result through an activation function
(Relu, Sigmoid)
Determine patterns, transform data into a useful representation, learn complex relationships.

Output layer

The final layer of the neural network, it produces final result of the model.

Model Capacity and Generalization

The number of hidden units and layers determines the network capacity to learn

Backpropagation Algorithm

Backpropagation (training-examples, η , n_{in}, n_{out}, n_{hidden})
Each training example is a pair of the form $(\vec{x}, \vec{t})$ where $\vec{x}$ is the vector of network input values and $\vec{t}$ is the vector of target network output values.
η is the learning rate
n_{in} is the number of network inputs
n_{hidden} is the number of units in hidden layer
n_{out} is the number of output units
The input from unit i into unit j is denoted x_{ji} and the weight from unit i to unit j denoted by w_{ji}

- Create feed foward network with n_{in} inputs, n hidden hidden units and n_{out} output units
- Initialize all network weights to small random numbers (0.05)
- until the termination condition is met do

propagate the input forward through the network

1. Input the instance $\bar{x}$ to the network and compute the output o_u of every unit u in the network.

propagate the errors backward through the network

2. For each network output unit k, calculate its error term δ_k

$$\delta_k \leftarrow o_k (1 - o_k)(t_k - o_k)$$

3. For each hidden unit h, calculate its error term δ_h

$$\delta_h \leftarrow o_h (1 - o_h) \sum_{k \in \text{outputs}} w_{kh} \delta_k$$

4. update each network weight w_{ji}

$$w_{ji} \leftarrow w_{ji} + \Delta w_{ji}$$

where $\Delta w_{ji} = \eta \delta_j x_{ji}$

8. Apply your understanding to explain parameter norm penalities and demonstrate how L_1 and L_2 regularizations function with suitable example.

(A) When training a neural network, the model may overfit ie it memorizes the training data and perfoems poorly on new data

To reduce overfitting, we add a penalty to the loss function that pushes large parameter (weight values)

General form

$$\text{Loss}_{\text{regularized}} = \text{Loss}_{\text{original}} + \lambda \cdot \Omega(W)$$

where

W : weights of the model

$\Omega(W)$: penalty function

λ : Regularization strength (hyper parameter)

Larger $\lambda \Rightarrow$ stronger penalty.

Two most common norm penalties are L_1 norm and L_2 norm

L_1 Regularization (Lasso)

This technique is used to solve overfitting problem

The penalty term is $\Omega(W) = \sum_i |W_i|$

Effect :- . pushes weights exactly to zero
. creates sparse models
. useful for feature selection

updated Loss :

$$L = L_0 + \lambda \sum_i |W_i|$$

Example Let weights be

$$W = [0.5, -2.0, 0.1]$$

L_1 penalty

$|0.5| + |-2.0| + |0.1| = 2.6$

If $\lambda = 0.2$

$L = L_0 + 0.52$

During training $0.5 \Rightarrow 0.3$, $-2 \Rightarrow -1.5$, $0.1 \Rightarrow 0$

L_1 forces exact zeroes

L_2 Regularization (Ridge | Weight Decay)

This technique is used to solve over fitting problem.

Here penalty term is

$$\Omega(W) = \sum_i w_i^2$$

Effect

- Reduce weights smoothly toward zero
- Does not set weights exactly to zero
- Helps reduce model complexity by making weights smaller.
- prefered in neuralnetworks (called weight decay)

updated Loss:

$$L = L_0 + \lambda \sum_i w_i$$

Simple example

Suppose your original loss (without regularization) is

$$L_0 = (y - \hat{y})^2$$

And weights are $W = [2, -3]$

Then L_2 penalty $= \lambda (2^2 + (-3)^2)$

$= \lambda (4+9)$

$= 13\lambda$

If $\lambda = 0.1$

Total Loss = $L_0 + 1.3$

Weights become smaller during training

$2 \Rightarrow 1.8$

$-3 \Rightarrow -2.7$

Exact values depend on learning rate.

9. Demonstrate your understanding by writing short notes on

(a) Tangent Distance and Tangent propagation

(b) Manifold Learning

(A) Tangent distance, tangent propagation and manifold learning are related to deep learning techniques that assume high dimentinal data lies on a lower dimentinal manifold.

Tangent distance

Tangent distance measures similarity between data points by comparing the tangent spaces (local linear approximation) of their manifold. Manifold learning finds the structure.

It compares the local tangents of the manifolds to find measure of similarity often used for image classification

Tangent propagation

Tangent propagation uses the tangent space to create a more robust neural network by training it

to be invariant to transformations along the manifold.

Tangent propagation is a regularization technique that trains neural networks to be invariant to small transactions of the input data.

It uses the tangent space to propagate error gradients through the networks in a way that accounts for the manifold structure leading to better generalization. This is an alternative to more data intensive methods like data augmentation.

Manifold Learning

Manifold learning finds the structure. The manifold hypothesis states that high dimensional data

In Manifold hypothesis the idea that data points are not randomly scattered in high dimensional space but are concentrated on or near a lower dimensional manifold

For example images of a rotating object form a manifold.

It helps overcome the curse of dimensionality and can lead to better generalization for machine models.

Tangent distance

It is a method for comparing high dimensional

data points like images, by measuring distance between their underlying manifolds.

It calculates the distance by using local tangent at each point on the manifold, approximating the manifold as a linear space near that point.

It provides a more robust way to compare data that has been transformed.

10. Explain Various optimization strategies and meta algorithms used for efficient model learning

(A) optimization strategies and meta algorithms are crucial for efficient model learning, guiding the search for the best model parameters.

They are generally categorized into first order and second order methods.

- optimization strategies

These algorithms determine how the model's parameters are updated based on the loss function gradient.

- Gradient Descent (GD)

The foundational algorithm that iteratively adjusts parameters in the opposite direction of the gradient of loss function. This indicates the direction of steepest descent.

Stochastic Gradient Descent (SGD)

Instead of calculating the gradient over the entire dataset, SGD estimates the gradient using single randomly chosen example at each iteration. It speed up convergence on large data sets.

Meta Algorithms (Adaptive learning Rate Methods)

These methods enhance the core strategies by dynamically adjusting the learning rate for each parameter, which is a crucial hyper parameter that dictates the size of the steps taken during optimization.

Momentum

Accelerates Gradient Decent by incorporating information from previous updates. It helps the optimizer roll past shallow local minima and proceed faster down ravines, smoothing out updates.

AdaGrad (Adaptive Gradient)

Adapts the learning rate to the parameters performing larger updates for infrequent parameters and smaller updates for frequent ones. A common issue is that learning rate can become vanishingly small over time.

RMSprop (Root Mean Square propagation)

An important over AdaGrad that solves the vanishing learning rate problem by using a moving

average of squared gradients which allows learning rate to adapt dynamically without continuously decreasing.

Adam (Adaptive moment estimation)

It is most popular and effective algorithm

Adam combines the ideas of RMSprop (adaptive learning rate) and Momentum (using past gradients) It is generally robust and performs well in a wide range of deep learning Applications.

11. Describe the role of optimization in Computer Vision speech Recognition and natural language processing with suitable examples

(A) optimization plays a crucial role in computer vision speech recognition and Natural language processing (NLP) primarily by enabling the training and fine tuning of machine learning model to acheive high accuracy and efficiency.

The main function is minimize a loss function that measures the difference between models predictions and the actual data.

Computer Vision

In computer vision optimization is used to train deep neural networks (like CNN) to perform tasks such as image classification, object detection and segmentation.

optimization algorithms such as stochastic gradient descent (SGD) and Adam adjust the millions of parameters (weights and biases) of a neural network to minimize the classification error across the large dataset of images.

Ex:- Training an image classifier to distinguish between cats and dogs.

→ The model initially makes many errors. The optimization algorithm iteratively update the model parameters in the direction that reduces the error rate.

After optimization the model can accurately identify whether an input image contains a cat or a dog

Speech Recognition

optimization is vital for training models to accurately changes spoken language into text, even amidst back ground noise or variations in speaking styles.

→ It adjusts the parameters of acoustic models (often recurrent neural networks or transformers)

to minimize the edit distance or cross entropy loss between the models changed output and actual spoken words in a training audio clip.

Ex :- A voice assistant application (like Siri, Alexa) learns to understand users command.

Natural Language processing (NLP)

For NLP optimization drives the development of applications like machine translation, sentiment analysis and question answering system.

→ optimization algorithms are used to train large language models (LLM) by minimizing the probability of the model predicting an incorrect next word in a sentence.

Example:- Training a machine translation system to translate english sentence to Hindi

The optimization process fine tunes the model's vast number of parameters to minimize the difference between the models translated output and correct human translations provided in a training dataset.

- This results more fluid and contextually accurate translations.