Activation Functions Role and Comparison.

Ans 1:

Role of Activation Functions:
Activation functions determine the output of a neural network node given an input or set of inputs. They introduce non-linearity into the network, enabling it to learn complex patterns and perform more sophisticated computations than simple linear models.

Comparison of Linear and Nonlinear Activation Functions:

Features	Linear Activation Functions	Nonlinear Activation Fuctions
Output	Proportional to Input	Non linear transformation of input
Decision Boundary	Can only learn linear boundaries	Can learn complex , nonliner, boundaries.
Stacking Layers	Mulitple Layers equivalent of single layer	Each Layer adds complexity
Examples	F(x)=x	RelU,Sigmoid,Tanh

Why nonlinear functions are preferred in hidden layers:
Nonlinear activation functions allow neural networks to approximate any continuous function (universal approximation theorem). Without them, no matter how many layers we add, the network would behave like a single-layer perceptron because the composition of linear functions is still linear. Nonlinearity enables the network to learn complex patterns in data.

Ans:2

Sigmoid Activation Function:

- Formula: $\sigma(x) = 1 / (1 + e^{-x})$
- Characteristics:
  - Outputs values between 0 and 1 (squashes input to (0,1) range)
  - Smooth gradient (differentiable everywhere)
  - Historically popular for binary classification
- Commonly used in:
  - Output layer for binary classification problems
  - Earlier neural networks (now largely replaced by ReLU in hidden layers)

Rectified Linear Unit (ReLU):

- Formula: $f(x) = \max(0, x)$
- Advantages:
  - Computationally efficient (simple operation)

    - Avoids vanishing gradient problem for positive inputs
    - Leads to sparse activations (many outputs become zero)
    - Accelerates convergence compared to sigmoid/tanh
- Challenges:
    - "Dying ReLU" problem (neurons can get stuck in inactive state)
    - Not differentiable at zero (though this is rarely an issue in practice)

Tanh Activation Function:

- Formula: $\tanh(x) = (e^x - e^{-x})/(e^x + e^{-x})$
- Purpose:
    - Outputs values between -1 and 1 (zero-centered)
    - Often performs better than sigmoid for hidden layers
- Differences from Sigmoid:
    - Output range (-1,1) vs (0,1)
    - Stronger gradients (derivative is steeper)
    - Zero-centered outputs help with learning in deep networks

Ans:3

Activation functions in hidden layers are crucial because:

1. They introduce nonlinearity, enabling the network to learn complex patterns and relationships in data.
2. They determine whether and how strongly a neuron should be activated based on inputs.
3. They affect the gradient flow during backpropagation, impacting learning dynamics.
4. Different functions can lead to different learning behaviors (e.g., ReLU avoids vanishing gradients for positive inputs).
5. They enable hierarchical feature learning - early layers learn simple features while deeper layers combine them into more complex representations.

Without activation functions in hidden layers, the network would simply be a linear transformation, no matter how many layers it has, severely limiting its representational power.

Ans:4

Choice depends on problem type:

Binary Classification:

- Sigmoid (outputs probability between 0 and 1)
- Example: Spam detection (spam/not spam)

Multi-class Classification:

- Softmax (outputs probability distribution over classes summing to 1)
- Example: Handwritten digit recognition (10 classes)

Multi-label Classification:

- Sigmoid for each output node (independent probabilities)
- Example: Image tagging (multiple possible tags)

Regression:

- Linear (unbounded output) for predicting continuous values
- Example: House price prediction
- ReLU if output should be non-negative
- Example: Predicting product demand (can't be negative)

Bounded Regression:

- Sigmoid or Tanh if output needs to be within specific range
- Example: Predicting normalized ratings between 0-1

Ans:5

Experimental Setup:

- Simple neural network (e.g., 2 hidden layers with 32 units each)
- Dataset: MNIST or similar benchmark
- Compare ReLU, Sigmoid, and Tanh in hidden layers
- Metrics: Training/validation accuracy, loss convergence, training time

Expected Observations:

Activation	**Convergence Speed**	**Final Performance**	**Potential issues**
ReLu	**Fastest**	**Good**	**Dying RelU possible**
Tanh	**Moderate**	**Good**	**Vanishing Gradients in deep nets**
Sigmoid	**Slowest**	**Lower**	**Severe vanishing gradients**