Neural Network

1. Feedforward Neural Network (FNN) Structure and Activation Functions

Ans: Basic Structure of FNN:

Composed of an input layer, one or more hidden layers, and an output layer

Information flows in one direction (forward) from input to output

Fully connected between layers (each neuron connects to all neurons in next layer)

No cycles or loops in the network architecture

Also called Multi-Layer Perceptron (MLP)

Purpose of Activation Functions:

Introduce non-linearity to enable learning complex patterns

Determine whether a neuron should be activated (fired) or not

Help the network learn from the error during backpropagation

Common activation functions: ReLU, Sigmoid, Tanh, Softmax (for output layer)

Without activation functions, the network would just be a linear regression model

2. Convolutional and Pooling Layers in CNNs

Ans: Role of Convolutional Layers:

Extract spatial features from input data (especially images)

Apply filters (kernels) that detect patterns like edges, textures, shapes

Share weights across the entire input (translation invariance)

Reduce number of parameters compared to fully connected layers

Preserve spatial relationships between pixels

Purpose and Benefits of Pooling Layers:

Reduce spatial dimensions (width, height) of feature maps

Decrease computational complexity

Make detection of features somewhat invariant to scale and small translations

Common types: Max pooling (most common), Average pooling

Helps prevent overfitting by providing an abstracted form of the representation

Typically uses 2×2 or 3×3 windows with stride 2

3. RNNs and Sequential Data Handling Ans: Key Characteristic of RNNs:

Have "memory" through hidden states that persist between time steps

Contain cycles/loops in the network architecture

Can process variable-length sequential inputs

Parameters are shared across time steps

Handling Sequential Data:

Processes one element of the sequence at a time

Maintains a hidden state that carries information from previous time steps

The same weights are applied at each time step

Can theoretically capture long-term dependencies (though practically limited)

Output at each step can depend on current input and previous hidden state

Commonly used for time series, text, speech, and other sequential data

4. LSTM Components and Vanishing Gradient Solution

Ans: LSTM Components:

Cell State: The "memory" that flows through the network

Forget Gate: Decides what information to discard from cell state

Input Gate: Determines what new information to store in cell state

Output Gate: Controls what information to output based on cell state

Candidate Values: Potential new information to add to cell state

Addressing Vanishing Gradients:

The cell state provides a "highway" for gradients to flow through time

Gates regulate information flow, preventing uncontrolled shrinking of gradients

Additive updates to cell state (rather than multiplicative) help maintain gradient magnitude

Forget gate helps preserve important long-term information

Can maintain gradients over hundreds of time steps (vs. ~10 for vanilla RNN)

5. GAN Components and Training Objectives Ans: Generator:

Role: Creates synthetic data that resembles real training data

Input: Random noise vector (latent space)

Output: Synthetic sample (e.g., image, text)

Training Objective: Fool the discriminator into classifying its output as real

Typically uses transposed convolutions (for images) to upsample noise

Discriminator:

Role: Distinguishes between real and generated samples

Input: Either real data or generator output

Output: Probability that input is real

Training Objective: Correctly classify real and generated samples

Typically a CNN (for images) or other classifier architecture

Training Process (Adversarial Objective):

Two-player minimax game with value function V(G,D): min_G max_D V(D,G) = E[logD(x)] + E[log(1-D(G(z)))]

Generator tries to minimize log(1-D(G(z))) (equivalent to maximizing D(G(z)))

Discriminator tries to maximize logD(x) + log(1-D(G(z)))

In practice, alternate between updating discriminator and generator