**1. What is an Artificial Neural Network (ANN)?**

**Answer:**  
An Artificial Neural Network (ANN) is a computing system inspired by the biological neural networks of animal brains. It consists of interconnected nodes (neurons) organized in layers — input layer, hidden layer(s), and output layer. Each connection has a weight, and the network learns patterns by adjusting these weights.

**Counter-questions:**

* What are the types of neural networks?
* Why are neural networks called "artificial"?

**2. What is Bias in a Neural Network?**

**Answer:**  
Bias is an extra parameter in the neuron model that allows the model to shift the activation function curve left or right. It helps the model to better fit the data, even when input features are zero.

**Counter-questions:**

* How is bias different from weights?
* What happens if there is no bias?

**3. Can you explain the structure of a simple neuron model, the Perceptron?**

**Answer:**  
A perceptron is the simplest neural network. It takes multiple input signals, applies individual weights, sums them up, adds a bias, and passes the result through an activation function (usually step function) to produce output.

**Counter-questions:**

* What activation function is used in a perceptron?
* Why is bias important in a perceptron?

**4. What are Activation Functions, and why are they important?**

**Answer:**  
Activation functions introduce non-linearity into the network. Without activation functions, a neural network would behave like a linear regression model. They help the network learn complex patterns like images, languages, etc.

**Counter-questions:**

* Name a few activation functions.
* What happens if no activation function is used?

**5. What is the difference between Activation Function and Threshold Function?**

**Answer:**

* **Activation Function:** Any function (like ReLU, Sigmoid) that decides output value.
* **Threshold Function:** A specific type of activation function that outputs binary results (0 or 1) based on whether the input crosses a threshold.

**Counter-questions:**

* Give examples of threshold functions.
* Can a threshold function be non-linear?

**6. What is the McCulloch-Pitts Rule?**

**Answer:**  
The McCulloch-Pitts rule is the fundamental idea behind simple neuron models. It states that a neuron can either fire (1) or not fire (0) based on a weighted sum of inputs compared against a threshold.

**Counter-questions:**

* What assumptions are made in McCulloch-Pitts model?
* What kind of problems can it solve?

**7. What is the McCulloch-Pitts model of Logic Gates?**

**Answer:**  
The McCulloch-Pitts model simulates basic logic gates (AND, OR, NOT) by adjusting weights and thresholds properly. Each gate behavior is mimicked by setting rules for firing or not firing.

**Counter-questions:**

* How is an AND gate modeled in McCulloch-Pitts?
* Is it possible to model XOR with McCulloch-Pitts?

**8. What are the 7 Logic Gates?**

**Answer:**  
The 7 logic gates are:

* AND Output is **1 only if both inputs are 1**, otherwise 0.
* OR Output is **1 if at least one input is 1**, otherwise 0.
* NOT Output is the **inverse** of the input.
* NAND Output is **0 only if both inputs are 1**, otherwise 1.
* NOR Output is **1 only if both inputs are 0**, otherwise 0.
* XOR Output is **1 if inputs are different**, otherwise 0.
* XNOR Output is **1 if inputs are the same**, otherwise 0.

**Counter-questions:**

* Which gates are universal gates?
* Which gate cannot be realized directly by McCulloch-Pitts?

**9. What is the McCulloch-Pitts Model and its Function?**

**Answer:**  
It models neurons as simple binary units that fire based on a threshold rule. Its function is to represent decision-making processes using basic logical functions.

**Counter-questions:**

* What limitations does the McCulloch-Pitts model have?
* Can it handle memory or feedback?

**10. What is the difference between McCulloch-Pitts and Perceptron?**

**Answer:**

* McCulloch-Pitts uses **fixed weights** (usually 1) and **binary output** with no learning.
* Perceptron allows **learning** by adjusting weights and can handle more flexible inputs.

**Counter-questions:**

* Can Perceptron solve XOR problems?
* Why is learning important in perceptron?

**Q11. What are the types of McCulloch-Pitts Neural Model?**

**Answer:**

The McCulloch-Pitts model is a simple mathematical model of a neuron. It works based on binary inputs and outputs (only 0 or 1).

There are three main types based on logic gates:

* **AND Model**
  + Fires (output = 1) only if **all** inputs are 1.
  + Else, output = 0.
* **OR Model**
  + Fires (output = 1) if **any one** input is 1.
* **NOT Model**
  + Inverts the input. If input is 1, output is 0 and vice versa.

**Important points:**

* The model uses a **threshold function**.
* It cannot solve problems that are **not linearly separable** (like XOR).

**Counter-Questions that can be asked:**

* Why can't McCulloch-Pitts solve XOR?
* What is a threshold function?

**📚 Q12. What is Delta Learning Rule?**

**Answer:**

The Delta Learning Rule is a supervised learning method where the weights are updated based on the difference between the expected (target) output and the actual output.

The formula is:  
Δw = η \* (Target Output – Actual Output) \* Input

Where:

* Δw = change in weight
* η = learning rate (small constant like 0.1 or 0.01)
* Target Output = expected output
* Actual Output = output given by neuron
* Input = input value corresponding to that weight

**Steps:**

1. Calculate the output of the neuron.
2. Find the error (Target - Output).
3. Multiply error with input and learning rate.
4. Update weights using Δw.

**Delta Rule is mainly used in:**

* **Adaline**
* **Backpropagation networks**

**Counter-Questions that can be asked:**

* What is the role of the learning rate η?
* In which networks is the Delta Rule applied?

**📚 Q13. What is Adaline and Madaline?**

**Answer:**

* **Adaline (Adaptive Linear Neuron):**  
  A single-layer neural network where the output is not thresholded immediately.  
  It uses continuous values and minimizes the error using the **Mean Squared Error (MSE)**.  
  Weights are updated using the **Delta Rule**.
* **Madaline (Multiple Adaptive Linear Neurons):**  
  A network of multiple Adalines arranged in layers (multi-layer structure).  
  It was one of the **first multi-layer networks** and uses an algorithm called **Madaline Rule** for training.

**Key Difference:**

* Adaline is single-layer, Madaline is multi-layer.
* Madaline can solve problems that are not linearly separable.

**Counter-Questions that can be asked:**

* What training rule does Madaline use?
* How is Madaline different from Adaline?

**📚 Q14. Is CNN a Perceptron?**

**Answer:**

No, a **CNN (Convolutional Neural Network)** is **not** a perceptron.

* **Perceptron:**  
  A very basic model — input weighted sum passed through an activation (step) function to produce output.
* **CNN:**  
  A very deep and complex structure consisting of:
  + **Convolution layers** (feature extraction)
  + **Pooling layers** (downsampling)
  + **Fully connected layers** (classification)

CNNs are inspired by the way animals perceive visual information.

**Counter-Questions that can be asked:**

* What is the main operation in CNNs?
* Which layers make CNNs powerful?

**📚 Q15. What is the difference between Perceptron and Adaline?**

**Answer:**

| **Feature** | **Perceptron** | **Adaline** |
| --- | --- | --- |
| Activation | Step function (binary) | Linear (continuous) |
| Error Calculation | After activation | Before activation |
| Learning Rule | Perceptron rule | Delta rule |
| Output | 0 or 1 | Continuous |

**Example:**

* In Perceptron, after weighted sum, apply step function → output 0 or 1.
* In Adaline, weighted sum is used directly for weight update.

**Counter-Questions that can be asked:**

* Which one is better for convergence?
* Why does Adaline use continuous output?

**📚 Q16. What is Widrow-Hoff Rule?**

**Answer:**

Widrow-Hoff Rule is another name for the **Delta Rule**.

* It updates weights based on the error between expected and actual output.
* It was developed for the **Adaline** model.

**Formula:**  
Δw = η \* (Target – Output) \* Input

**Importance:**

* First step towards more powerful training methods like backpropagation.

**Counter-Questions that can be asked:**

* Which model introduced Widrow-Hoff Rule?
* Is Widrow-Hoff supervised or unsupervised?

**📚 Q17. What is the difference between Hebb and Delta Rule?**

**Answer:**

| **Feature** | **Hebb Rule** | **Delta Rule** |
| --- | --- | --- |
| Idea | "Neurons that fire together wire together" | Minimize error |
| Output consideration | No output comparison | Depends on output error |
| Used for | Unsupervised learning | Supervised learning |
| Formula | Δw = η \* input \* output | Δw = η \* (target - output) \* input |

**Counter-Questions that can be asked:**

* Which rule suits supervised learning?
* Where is Hebb’s rule still used?

**📚 Q18. What is ReLU in CNN?**

**Answer:**

* ReLU stands for **Rectified Linear Unit**.
* It is an activation function used widely in CNNs and deep networks.

**Formula:**  
f(x) = max(0, x)

**Meaning:**

* If x > 0, output = x.
* If x ≤ 0, output = 0.

**Why important?**

* Introduces non-linearity.
* Speeds up convergence compared to sigmoid/tanh.

**Counter-Questions that can be asked:**

* What is the disadvantage of ReLU (Dying ReLU problem)?
* Alternatives to ReLU?

**📚 Q19. What is Hebb’s Rule?**

**Answer:**

**Hebb’s Rule:**  
"If two neurons are activated simultaneously, the connection between them is strengthened."

**Mathematically:**  
Δw = η \* input \* output

Where:

* Δw = change in weight
* η = learning rate
* input/output = signals of two connected neurons

**Use:**

* Used in unsupervised learning.
* Foundation for associative memory models.

**Counter-Questions that can be asked:**

* Is Hebb’s Rule supervised or unsupervised?
* Where is Hebbian learning used?

**📚 Q20. What is the Hebbian Formula?**

**Answer:**

**Formula:**  
Δw = η \* input \* output

**Meaning:**

* The weight change is proportional to the product of the input and the output.
* Strengthens connections between co-active neurons.

**Example:** If input is 1 and output is 1, weight increases.  
If input or output is 0, no change.

**Counter-Questions that can be asked:**

* What happens if only one neuron is active?
* In which neural models is this formula used?

**Q21. What is the Learning Rate (η)?**

**Answer:**

Learning rate (denoted as η) is a small positive number that controls **how much** the weights are updated during training.

* If η is **too small**, learning is **slow**.
* If η is **too large**, learning becomes **unstable** (may overshoot the correct solution).

**Typical Values:**  
η = 0.1, 0.01, 0.001

**Example:**  
Suppose weight change is calculated as 0.5, and η = 0.01,  
then actual change = 0.01 × 0.5 = 0.005 (small, stable update).

**Counter-Questions:**

* What happens if η is set too high?
* How can learning rate be adapted during training?

**📚 Q22. What is Convergence in Neural Networks?**

**Answer:**

Convergence means the network's **error becomes very small or stops changing significantly** during training.

In simple words:

* The network has learned well.
* Further training will not change weights much.
* Predictions become stable.

**When does convergence happen?**

* When weights reach optimal/near-optimal values.
* When validation error stops decreasing.

**Counter-Questions:**

* What factors affect convergence speed?
* What if convergence happens too early (underfitting)?

**📚 Q23. What is the Activation Function?**

**Answer:**

Activation Function decides **whether a neuron should fire or not**, based on input sum.

It **adds non-linearity** to the model, allowing it to learn complex patterns.

**Types:**

* Step Function (early models)
* Sigmoid
* Tanh
* ReLU
* Leaky ReLU
* Softmax (for multiclass classification)

**Example:**  
Sigmoid activation:  
f(x) = 1 / (1 + exp(-x))

**Counter-Questions:**

* Why is non-linearity important?
* Which activation is better for hidden layers?

**📚 Q24. What is the difference between Linear and Non-Linear Activation Functions?**

**Answer:**

| **Linear Activation** | **Non-Linear Activation** |
| --- | --- |
| Output is directly proportional to input. | Output changes non-linearly with input. |
| f(x) = x | f(x) = sigmoid, tanh, ReLU, etc. |
| Cannot model complex data | Can model complex patterns |
| Not useful for deep networks | Essential for deep learning |

**Counter-Questions:**

* Can a network with only linear activations solve XOR?
* Give an example of a non-linear activation.

**📚 Q25. What is Gradient Descent?**

**Answer:**

Gradient Descent is an optimization technique used to **minimize the loss (error)** by updating weights.

**Idea:**

* Calculate gradient (slope) of loss function.
* Move weights in the **opposite direction** of the gradient.

**Types:**

* Batch Gradient Descent
* Stochastic Gradient Descent (SGD)
* Mini-batch Gradient Descent

**Formula:**  
new weight = old weight – η × gradient

**Counter-Questions:**

* What are types of gradient descent?
* Why do we move opposite to the gradient?

**📚 Q26. What is Local Minima?**

**Answer:**

In optimization, a **local minima** is a point where the loss is **lower than neighboring points**, but **not necessarily the lowest overall**.

**Problem:**

* If gradient descent gets stuck in local minima, training might stop without finding the best model.

**Example:**  
Imagine a bowl with multiple dips; local minima is a small dip, global minima is the deepest dip.

**Counter-Questions:**

* How can we avoid getting stuck in local minima?
* Is ReLU better than sigmoid for avoiding local minima?

**📚 Q27. What is Epoch?**

**Answer:**

An epoch is **one complete pass** over the **entire training dataset**.

**Example:**  
If you have 1000 samples, one epoch means training has seen all 1000 once.

* Usually, training happens for **many epochs** (10, 50, 100, etc.)
* After each epoch, weights are updated.

**Counter-Questions:**

* What happens if we train for too many epochs?
* Difference between batch and epoch?

**📚 Q28. What is Overfitting?**

**Answer:**

Overfitting happens when a model **learns the training data too well**, including noise and outliers, and **performs badly on new data**.

**Symptoms:**

* Very low training error
* High validation/test error

**Solutions:**

* Early stopping
* Regularization
* Dropout
* Using more data

**Counter-Questions:**

* What causes overfitting?
* What is Regularization?

**📚 Q29. What is Regularization?**

**Answer:**

Regularization is a technique to **prevent overfitting** by **penalizing large weights**.

**Two Common Types:**

* **L1 Regularization (Lasso):** adds sum of absolute weights.
* **L2 Regularization (Ridge):** adds sum of squared weights.

**Regularized Loss:**  
Loss = Original Loss + Regularization Term

**Counter-Questions:**

* What is difference between L1 and L2?
* How does regularization affect training?

**📚 Q30. What is Dropout in Neural Networks?**

**Answer:**

Dropout is a **regularization technique** where during training, randomly selected neurons are **ignored ("dropped out")**.

**Purpose:**

* Reduces overfitting.
* Forces the network to not depend too heavily on any one neuron.

**How it works:**

* During each iteration, randomly turn off neurons with probability p (like 0.5).

**Counter-Questions:**

* What is typical dropout rate?
* Does dropout happen during testing too?

**Q31. What is Batch Size in Neural Networks?**

**Answer:**

Batch size is the **number of samples** processed **before the model updates its weights**.

**Example:**  
If batch size = 32, model will look at 32 samples, calculate loss, update weights once.

**Types:**

* **Batch Gradient Descent:** batch size = all data
* **Stochastic Gradient Descent:** batch size = 1
* **Mini-Batch Gradient Descent:** batch size = small portion (like 32, 64)

**Counter-Questions:**

* How does batch size affect training speed?
* Why is mini-batch training popular?

**📚 Q32. What is Loss Function?**

**Answer:**

Loss Function measures **how far the predicted output is from the actual output**.

It gives a numerical value representing model performance — lower is better.

**Examples:**

* MSE (Mean Squared Error) — for regression
* Cross-Entropy Loss — for classification

**Counter-Questions:**

* What are different types of loss functions?
* Difference between loss function and cost function?

**📚 Q33. What is Backpropagation?**

**Answer:**

Backpropagation is the **process of updating weights** by moving from **output layer back to input layer** based on error.

**Steps:**

1. Forward pass: calculate output.
2. Calculate loss.
3. Backward pass: compute gradients.
4. Update weights.

**Purpose:**  
Minimize loss by adjusting weights correctly.

**Counter-Questions:**

* Why do we need backpropagation?
* What is chain rule in backpropagation?

**📚 Q34. What are Weights and Biases in Neural Networks?**

**Answer:**

* **Weights:** Control strength of connection between neurons.
* **Biases:** Extra adjustable parameters added to shift output.

**Without bias:** Neurons output is fixed to pass through origin (0,0).  
**With bias:** Neurons can fit more complex patterns.

**Counter-Questions:**

* Why is bias important?
* How are weights initialized?

**📚 Q35. What is Weight Initialization?**

**Answer:**

Weight Initialization is the **method of setting the initial values of weights** before training begins.

**Why Important:**

* Good initialization → faster convergence.
* Bad initialization → slow or failed learning.

**Methods:**

* Random Initialization
* Xavier Initialization
* He Initialization

**Counter-Questions:**

* What happens if all weights are zero?
* Difference between Xavier and He initialization?

**📚 Q36. What is Vanishing Gradient Problem?**

**Answer:**

Vanishing Gradient happens when gradients become **very small**, causing **weights to barely update** during backpropagation.

**Problem:**

* Network stops learning.
* Common in deep networks with sigmoid/tanh activation.

**Solution:**

* Use ReLU activation.
* Use batch normalization.
* Careful weight initialization.

**Counter-Questions:**

* Why does sigmoid cause vanishing gradients?
* How does ReLU help?

**📚 Q37. What is Exploding Gradient Problem?**

**Answer:**

Exploding Gradient occurs when gradients become **very large**, causing **weights to update with huge steps**.

**Problem:**

* Model becomes unstable.
* Loss becomes NaN (not a number).

**Solution:**

* Gradient clipping
* Proper initialization
* Using normalized inputs

**Counter-Questions:**

* What is gradient clipping?
* Why deep networks suffer from this?

**📚 Q38. What is Early Stopping?**

**Answer:**

Early Stopping is a technique to **stop training when model performance stops improving** on validation data.

**Purpose:**

* Avoid overfitting.
* Save computational resources.

**How it works:**

* Monitor validation loss.
* If loss doesn’t improve for certain epochs → stop training.

**Counter-Questions:**

* How do you decide patience in early stopping?
* Can early stopping harm training?

**📚 Q39. What is the Cost Function?**

**Answer:**

Cost Function is **average loss** over the **entire training dataset**.

**Example:**  
If loss for one sample is 0.1,  
Cost = average loss across all samples.

**Difference:**

* **Loss** = error for a single example.
* **Cost** = average loss for all examples.

**Counter-Questions:**

* Is cost always minimized during training?
* Example of cost function?

**📚 Q40. What is the difference between Training Error and Validation Error?**

**Answer:**

| **Training Error** | **Validation Error** |
| --- | --- |
| Error on data the model trained on | Error on unseen validation data |
| Should decrease over epochs | Should decrease but can increase if overfitting |
| Lower training error ≠ good model | Lower validation error = better generalization |

**Counter-Questions:**

* What does it mean if validation error is much higher than training error?
* How can you reduce validation error?

## 1. How does Batch Size impact Neural Network Training?

* **Batch size** determines how many samples are passed through the network at once before updating the model parameters.
* **Small batch sizes:** Noisy updates, better generalization, slower.
* **Large batch sizes:** Smoother updates, faster training, needs more memory.
* **Counter:** Mini-batch gradient descent is a compromise between batch and stochastic.

## 2. What are Hyperparameters in Neural Networks? How to Optimize Them?

* **Hyperparameters** are settings that define the network structure (e.g., number of layers, neurons) and training process (e.g., learning rate, batch size).
* **Optimization techniques:** Grid Search, Random Search, Bayesian optimization.

## 3. How is Cross-Validation used in Neural Network Training?

* **Cross-validation** (like k-fold) splits data into parts to test model performance across different sets and avoid overfitting.
* Useful especially when data is limited.

## 4. What are Autoencoders and Their Applications?

* **Autoencoders** are unsupervised neural networks that learn to compress (encode) data and then reconstruct (decode) it.
* **Applications:** Denoising, Feature extraction, Dimensionality reduction.

## 5. How are Neural Networks used in Image Recognition, NLP, and Predictive Analytics?

* **Image Recognition:** CNNs identify features like edges, textures.
* **NLP:** RNNs/LSTMs handle sequential text data.
* **Predictive Analytics:** Neural networks predict future values from patterns.

## 6. How do Reinforcement Learning and Neural Networks interact?

* **Reinforcement Learning (RL):** Agents learn from actions and rewards.
* **Neural Networks:** Act as function approximators (value function or policy).
* Used in Deep RL (e.g., AlphaGo).

## 7. What is the Vanishing Gradient Problem? How to Solve it?

* **Vanishing Gradient:** In deep networks, gradients become too small to update weights effectively.
* **Solutions:** Use ReLU activation, Batch Normalization, Residual Networks (ResNets), Proper initialization.

## 8. Explain Gradient Descent and its Variants: SGD, Momentum, Adam, RMSprop.

* **Gradient Descent:** Algorithm to minimize loss by adjusting weights.
* **SGD (Stochastic):** Updates per batch → faster, noisier.
* **Momentum:** Adds "velocity" to escape local minima.
* **Adam:** Adaptive Moment Estimation, combines Momentum + RMSprop.
* **RMSprop:** Adjusts learning rate based on moving average of gradients.

## 9. What is the difference between Adam and RMSprop Optimizers?

* **Adam:** Combines momentum and adaptive learning rates.
* **RMSprop:** Only uses adaptive learning rate based on recent gradients.

## 10. What is the difference between Downscaling and Upscaling in Neural Networks?

* **Downscaling:** Reducing image size (loss of detail, faster).
* **Upscaling:** Increasing image size (may introduce artifacts).

## 11. What are Generative Adversarial Networks (GANs) and How do They Work?

* **GANs:** Two networks (Generator and Discriminator) competing — one generates fake data, the other tries to detect fakes.
* Applications: Deepfakes, Art generation, Super-resolution.

## 12. How to Assess Neural Network Model Performance?

* Metrics: Accuracy, Precision, Recall, F1 Score, Loss, ROC-AUC.
* Techniques: Confusion Matrix, Validation Curves.

## 13. What is Bidirectional Associative Memory (BAM) and its Types?

* **BAM:** Hetero-associative memory neural model where input patterns can retrieve output patterns and vice versa.
* **Types:** Auto-BAM (same input/output), Hetero-BAM (different).
* Activation Function: Hard limiting (binary) function.
* BAM is **unsupervised** learning.

## 14. Why is Data Normalization Important before Training Neural Networks?

* Normalization (like Min-Max Scaling or Z-Score) brings data to a common scale, helping faster convergence and stability.

## 15. What are the Common Challenges in Training Neural Networks?

* Overfitting, Underfitting, Vanishing/Exploding Gradients, Choosing Architecture, Hardware limitations, Data scarcity.

## 16. What are the Ethical Considerations in Neural Networks?

* Bias, Privacy, Transparency, Fairness, Accountability.
* Example: AI wrongly rejecting loan applications based on biased data.

## 17. What is Transfer Learning and Fine-Tuning in Deep Learning?

* **Transfer Learning:** Using a pre-trained model on a new task.
* **Fine-tuning:** Adjusting layers (especially later layers) of a pre-trained model to fit the new data.

## 18. What Tools and Libraries are Commonly Used for Neural Networks?

* **TensorFlow**, **PyTorch**, **Keras**, **Caffe**, **Theano**, **Scikit-learn**.

## 19. What is Feature Extraction in Deep Learning?

* Process where networks automatically find the best features (e.g., edges in images) instead of manually selecting them.

## 20. How do Weight Initialization, Momentum, and Learning Rate Affect Neural Network Training?

* **Weight Initialization:** Properly initialized weights help avoid vanishing/exploding gradients.
* **Momentum:** Accelerates SGD in the right direction.
* **Learning Rate:** Controls step size; too high = overshoot; too low = slow learning.

## 21. What are ART Networks (Adaptive Resonance Theory) and Their Types (ART1, ART2)?

* **ART Networks:** Solve stability-plasticity dilemma; learn new patterns without forgetting old ones.
* **ART1:** Binary inputs.
* **ART2:** Continuous inputs.

## 22. How do Neural Networks Learn Non-linear Decision Boundaries?

* By stacking multiple layers (deep learning), nonlinear activation functions (ReLU, Tanh) help model complex patterns beyond straight lines.

## 23. What is the Role of Learning Rate, Architecture Choices, and Hyperparameters on Model Performance?

* **Learning Rate:** Controls convergence speed.
* **Architecture Choices:** Depth, width of network affects model complexity.
* **Hyperparameters:** Tune to optimize model for different tasks.

## 24. How are Attention Mechanisms and Residual Networks (ResNets) used in Deep Learning?

* **Attention Mechanisms:** Focus on important parts of input (used in Transformers, NLP).
* **Residual Networks:** Skip connections solve vanishing gradient issue by allowing gradients to flow directly.

## 25. What is Data Augmentation and How Does it Help Neural Networks?

* Technique to artificially expand dataset by flipping, rotating, scaling images/text → improves generalization.

## 26. How do Neural Networks Handle Time-Series Data?

* RNNs, LSTMs, GRUs handle sequential patterns (e.g., stock prices, speech).

## 27. What are the Limitations of Neural Networks?

* Data-hungry, Require large compute, Interpretability issues ("black box"), Overfitting risks.

## 28. How are Neural Networks used in Recommendation Systems, Autonomous Driving, Fraud Detection, and Speech Recognition?

* **Recommendation:** Deep learning suggests products (Netflix, Amazon).
* **Autonomous Driving:** Object detection, decision making.
* **Fraud Detection:** Find patterns in financial transactions.
* **Speech Recognition:** Convert voice to text (Siri, Alexa).

## 29. What is Pruning in Neural Networks?

* Removing unnecessary neurons or connections to compress the model and make it faster without much loss in accuracy.

## 30. What are Siamese Networks and How Are They Used?

* **Siamese Networks:** Two identical networks share weights to compare inputs → used in Face Verification, Signature verification.

## 31. How Does Batch Normalization Help in Training Deep Neural Networks?

* **Batch Normalization:** Normalizes activations during training → faster convergence, higher learning rates possible, reduces internal covariate shift.

## 32. What are the Future Trends in Neural Networks?

* **Trends:** Explainable AI, Energy-efficient AI, Neural-symbolic systems, TinyML (Neural nets on edge devices), Advanced GANs and Transformers.