**1. Explain the concept of a neural network: A neural network is like a human brain made up of layers of "neurons" or nodes. These neurons take inputs, process them using weights and activation functions, and pass the output to the next layer to make predictions or classifications.**

**2. What are the differences between supervised, unsupervised, and semi-supervised learning in deep learning?**

* **Supervised learning uses labeled data (input and correct output). Example: classifying images of cats and dogs.**
* **Unsupervised learning uses only input data without labels. Example: grouping similar customers using clustering.**
* **Semi-supervised learning uses a small amount of labeled data and a large amount of unlabeled data to improve learning.**

**3. Difference between ML and DL:**

* **Machine Learning (ML): Focuses on algorithms that learn from data and make predictions. Needs manual feature selection.**
* **Deep Learning (DL): Subset of ML that uses neural networks to automatically extract features and learn complex patterns.**

**4. Dataset description:**

* **Number of records: Total rows in dataset (e.g., 60,000 images in MNIST).**
* **No. of features: Columns excluding the target (e.g., pixel values).**
* **Data types: Integer, float, string etc.**
* **Preprocessing: Normalization, reshaping, encoding labels.**
* **Training-testing split: Usually 80-20 or 70-30.**
* **Target feature: The label (what we want to predict).**
* **Feature selection: DL auto-extracts important features during training.**

**5. What is sequential model, no. of input-hidden-output layers, no. of nodes, epoch:**

* **Sequential model: A model where layers are stacked in order.**
* **Input layer: Takes data (e.g., 784 nodes for 28x28 image).**
* **Hidden layers: Extract features (e.g., Dense layers).**
* **Output layer: Gives final prediction (e.g., 10 nodes for 10 classes).**
* **Epoch: One full cycle through training data.**

**6. Different loss functions, your program loss value, to improve result what you can do?**

* **Loss functions: Binary crossentropy, categorical crossentropy, MSE.**
* **Improving results: Add layers, increase epochs, use dropout, data augmentation, or better optimizer.**

**7. Forward and backpropagation in neural networks:**

* **Forward: Data goes layer by layer to get output.**
* **Backward: Error is calculated and weights are updated using gradients.**

**8. What is the vanishing gradient problem? In deep networks, gradients become too small while backpropagating, causing layers to learn very slowly. This happens often with sigmoid/tanh activations.**

**9. Structure of CNNs:**

* **Conv2D layers: Extract features from images.**
* **MaxPooling: Reduces image size.**
* **Flatten: Converts 2D to 1D.**
* **Dense: Fully connected layers.**

**10. What is activation function? Common ones? Why used in your code? Activation functions decide if a neuron should fire.**

* **Common ones: ReLU, sigmoid, tanh, softmax.**
* **In code: ReLU (fast and avoids vanishing gradient), softmax (for multi-class output).**

**11. What is batch normalization? It normalizes input of each layer to make training faster and stable.**

**12. Common data augmentation techniques:**

* **Flipping**
* **Rotation**
* **Zoom**
* **Cropping**
* **Shifting These help make model more general and avoid overfitting.**

**13. What is transfer learning? Using a pre-trained model (like VGG, ResNet) and re-training it on your dataset. Saves time and resources.**

**14. What are RNNs? Advantages and limitations?**

* **RNNs are used for sequence data like text.**
* **Advantage: Remember previous input.**
* **Limitation: Vanishing gradient and slow training.**

**15. What is attention mechanism in sequence-to-sequence tasks? It helps model focus on important parts of the input sequence, improving translation, text summarization, etc.**

**16. What are autoencoders? Applications? Neural networks that learn to compress and reconstruct data. Used in noise removal, anomaly detection, and data compression.**

**17. Challenges in deep learning and solutions:**

* **Overfitting: Use dropout, regularization**
* **Training time: Use GPUs**
* **Large datasets: Use data generators or augmentation**

**18. Training process of deep learning model:**

* **Load data**
* **Preprocess (normalize, encode)**
* **Split into train-test**
* **Build model**
* **Compile (loss + optimizer)**
* **Train (fit)**
* **Evaluate (test)**

**19. Hyperparameters vs model parameters:**

* **Hyperparameters: Set before training (epochs, batch size, learning rate)**
* **Parameters: Learned during training (weights, biases)**

**20. Popular DL frameworks:**

* **TensorFlow: Large community, production ready**
* **Keras: Easy-to-use**
* **PyTorch: Flexible and popular in research**

**21. Gradient descent optimizers:**

* **SGD: Simple, slow**
* **Adam: Combines momentum and RMSprop, faster and better**
* **RMSprop: Good for RNNs, adjusts learning rate**

**22. Example deep learning project:**

* **Project: Fashion MNIST classification**
* **Model: CNN with 3 Conv2D + pooling layers**
* **Training: 10 epochs, batch size 128**
* **Result: ~92% accuracy**

**23. Regularization in deep learning: Adds a penalty to loss to avoid overfitting. Example: L2 regularization, dropout.**

**24. Generative vs discriminative models:**

* **Generative: Create new data (e.g., GANs)**
* **Discriminative: Classify data (e.g., CNNs for image classification)**

**1. Explain the concept of a neural network:** A neural network is like a human brain made up of layers of "neurons" or nodes. These neurons take inputs, process them using weights and activation functions, and pass the output to the next layer to make predictions or classifications.

**2. What are the differences between supervised, unsupervised, and semi-supervised learning in deep learning?**

* **Supervised learning** uses labeled data (input and correct output). Example: classifying images of cats and dogs.
* **Unsupervised learning** uses only input data without labels. Example: grouping similar customers using clustering.
* **Semi-supervised learning** uses a small amount of labeled data and a large amount of unlabeled data to improve learning.

**3. Difference between ML and DL:**

* **Machine Learning (ML)**: Focuses on algorithms that learn from data and make predictions. Needs manual feature selection.
* **Deep Learning (DL)**: Subset of ML that uses neural networks to automatically extract features and learn complex patterns.

**4. Dataset description:**

* **Number of records**: Total rows in dataset (e.g., 60,000 images in MNIST).
* **No. of features**: Columns excluding the target (e.g., pixel values).
* **Data types**: Integer, float, string etc.
* **Preprocessing**: Normalization, reshaping, encoding labels.
* **Training-testing split**: Usually 80-20 or 70-30.
* **Target feature**: The label (what we want to predict).
* **Feature selection**: DL auto-extracts important features during training.

**5. What is sequential model, no. of input-hidden-output layers, no. of nodes, epoch:**

* **Sequential model**: A model where layers are stacked in order.
* **Input layer**: Takes data (e.g., 784 nodes for 28x28 image).
* **Hidden layers**: Extract features (e.g., Dense layers).
* **Output layer**: Gives final prediction (e.g., 10 nodes for 10 classes).
* **Epoch**: One full cycle through training data.

**6. Different loss functions, your program loss value, to improve result what you can do?**

* **Loss functions**: Binary crossentropy, categorical crossentropy, MSE.
* **Improving results**: Add layers, increase epochs, use dropout, data augmentation, or better optimizer.

**7. Forward and backpropagation in neural networks:**

* **Forward**: Data goes layer by layer to get output.
* **Backward**: Error is calculated and weights are updated using gradients.

**8. What is the vanishing gradient problem?** In deep networks, gradients become too small while backpropagating, causing layers to learn very slowly. This happens often with sigmoid/tanh activations.

**9. Structure of CNNs:**

* **Conv2D layers**: Extract features from images.
* **MaxPooling**: Reduces image size.
* **Flatten**: Converts 2D to 1D.
* **Dense**: Fully connected layers.

**10. What is activation function? Common ones? Why used in your code?** Activation functions decide if a neuron should fire.

* **Common ones**: ReLU, sigmoid, tanh, softmax.
* **In code**: ReLU (fast and avoids vanishing gradient), softmax (for multi-class output).

**11. What is batch normalization?** It normalizes input of each layer to make training faster and stable.

**12. Common data augmentation techniques:**

* Flipping
* Rotation
* Zoom
* Cropping
* Shifting These help make model more general and avoid overfitting.

**13. What is transfer learning?** Using a pre-trained model (like VGG, ResNet) and re-training it on your dataset. Saves time and resources.

**14. What are RNNs? Advantages and limitations?**

* RNNs are used for sequence data like text.
* **Advantage**: Remember previous input.
* **Limitation**: Vanishing gradient and slow training.

**15. What is attention mechanism in sequence-to-sequence tasks?** It helps model focus on important parts of the input sequence, improving translation, text summarization, etc.

**16. What are autoencoders? Applications?** Neural networks that learn to compress and reconstruct data. Used in noise removal, anomaly detection, and data compression.

**17. Challenges in deep learning and solutions:**

* **Overfitting**: Use dropout, regularization
* **Training time**: Use GPUs
* **Large datasets**: Use data generators or augmentation

**18. Training process of deep learning model:**

* Load data
* Preprocess (normalize, encode)
* Split into train-test
* Build model
* Compile (loss + optimizer)
* Train (fit)
* Evaluate (test)

**19. Hyperparameters vs model parameters:**

* **Hyperparameters**: Set before training (epochs, batch size, learning rate)
* **Parameters**: Learned during training (weights, biases)

**20. Popular DL frameworks:**

* **TensorFlow**: Large community, production ready
* **Keras**: Easy-to-use
* **PyTorch**: Flexible and popular in research

**21. Gradient descent optimizers:**

* **SGD**: Simple, slow
* **Adam**: Combines momentum and RMSprop, faster and better
* **RMSprop**: Good for RNNs, adjusts learning rate

**22. Example deep learning project:**

* **Project**: Fashion MNIST classification
* **Model**: CNN with 3 Conv2D + pooling layers
* **Training**: 10 epochs, batch size 128
* **Result**: ~92% accuracy

**23. Regularization in deep learning:** Adds a penalty to loss to avoid overfitting. Example: L2 regularization, dropout.

**24. Generative vs discriminative models:**

* **Generative**: Create new data (e.g., GANs)
* **Discriminative**: Classify data (e.g., CNNs for image classification)