Hidden Layers Used in Deep Learning

Introduction :

In deep learning, a hidden layer is a layer between the input layer and output layer. These layers perform complex computations and help the model learn patterns, features, and representations from data. A neural network with more than one hidden layer is called a Deep Neural Network (DNN).

Consists of three primary types of layers:

- Input Layer
- Hidden Layers
- Output Layer

Types of Hidden Layers :

1. **Dense (Fully Connected) Layer :**
   Dense (Fully Connected) Layer is the most common type of hidden layer in an ANN. Every neuron in a dense layer is connected to every neuron in the previous and subsequent layers.
   - Role: Learns representations from input data.
   - Function: Performs weighted sum and activation.

2. **Convolutional Layer**
   Convolutional layers are used in Convolutional Neural Networks (CNNs) for image processing tasks. They apply convolution operations to the input, capturing spatial hierarchies in the data.
   - Role: Extracts spatial features from images.
   - Function: Applies convolution using filters

3. **Recurrent Layer**

Recurrent layers are used in Recurrent Neural Networks (RNNs) for sequence data like time series or natural
Language.

- Role: Processes sequential data with temporal dependencies.
- Function: Maintains state across time steps.

4. **Dropout Layer**

   Dropout layers are a regularization technique used to prevent overfitting.

   - Role: Prevents overfitting.
   - Function: Randomly drops neurons during training.

5. **Pooling Layer**

   Pooling Layer is used to reduce the spatial dimensions of the data, thereby decreasing the computational load and controlling overfitting.

   - Use Cases: Dimensionality reduction in CNNs

6. **Batch Normalization Layer**

   A Batch Normalization Layer normalizes the output of a previous activation layer by subtracting the batch mean and dividing by the batch standard deviation.

   - Use Cases: Stabilizing and speeding up training

Activation Functions Used in the Deep learning

1.Linear Activation Function

Formula:

- Output is same as input
- Used in regression output layer

- No non-linearity

1. **Binary Step Function**
   - Used in early Perceptron
   - Not used in modern deep learning (not differentiable)
2. **Sigmoid (Logistic) Function**
   - Output range: 0 to 1
   - Used in binary classification (output layer)
   - Problem: Vanishing gradient

4. Tanh (Hyperbolic Tangent)

- Output range: –1 to +1
- Zero-centered
- Still suffers from vanishing gradient

5.ReLU (Rectified Linear Unit)

- Most widely used
- Faster training
- Problem: Dead neuron problem

6. Leaky ReLU

- Solves dead neuron problem
- Small slope for negative values

7. Parametric ReLU (PReLU)

- A is learnable parameter
- Improved version of Leaky ReLU

8. ELU (Exponential Linear Unit)

- Smooth curve for negative values
- Reduces vanishing gradient
- Faster convergence than ReLU

9. SELU (Scaled ELU)

- Self-normalizing property
- Used in deep networks

10. Softmax

- Used in multi-class classification output layer
- Output range: 0 to 1 (probabilities)

To Find metrics,optimiser,losses

1. Loss Functions (Objectives to Minimize)

Regression Tasks:

- Mean Squared Error (MSE/L2 Loss): Used when outliers are not a major concern.
- Mean Absolute Error (MAE/L1 Loss): More robust to outliers.

Classification Tasks:

- Binary Cross-Entropy (Log Loss): For binary classification.
- Categorical Cross-Entropy: For multi-class classification.
- Sparse Categorical Cross-Entropy: Used when labels are integers rather than one-hot encoded.
- Hinge Loss: Used frequently in Support Vector Machines (SVMs).

2.Optimizers (Algorithms to Update Weights)

- Adam (Adaptive Moment Estimation): A popular, generally effective optimizer that combines the strengths of AdaGrad and RMSProp.
- SGD (Stochastic Gradient Descent): Updates parameters in the opposite direction of the gradient, often used with momentum.

- AdamW: A variant of Adam that implements decoupled weight decay, often used to improve generalization in Transformer models.
- RMSProp: Adaptive algorithm for handling sparse/noisy datasets.
- Adafactor: Memory-efficient optimizer used for very large-scale models.

3.Metrics (Evaluation Criteria)

Classification:

- Accuracy: Overall accuracy or sparse categorical accuracy for integer labels.
- Precision, Recall, F1 Score: Essential for imbalanced datasets.
- AUC-ROC: Measures performance across different thresholds.
- Confusion Matrix: Provides a detailed breakdown of correct/incorrect predictions.

Regression:

- MAE (Mean Absolute Error)
- MSE (Mean Squared Error)
- RMSE (Root Mean Squared Error)

NLP/Sequence:

- BLEU Score
- ROUGE Score
- Perplexity