Artificial Neural Networks (ANNs)

1. Activation Functions in ANNs

Activation functions introduce nonlinearity into the network, enabling it to learn complex patterns. Common activation functions include:

- **Sigmoid**
    - Formula: $\sigma(x) = \frac{1}{1+e^{-x}}$
    - **Properties:** Squashes input to (0, 1); historically used in binary classification but can suffer from vanishing gradients.
- **Hyperbolic Tangent (tanh)**
    - Formula: $\tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$
    - **Properties:** Output ranges from –1 to 1; zero-centered which often speeds up convergence compared to sigmoid.
- **Rectified Linear Unit (ReLU)**
    - **Formula:** f(x) = max(0, x)
    - **Properties:** Computationally efficient; helps mitigate the vanishing gradient problem but can lead to “dying ReLU” where neurons become inactive.
- **Leaky ReLU and Parametric ReLU (PReLU)**
    - Formula: $\text{Leaky ReLU}(x) = \begin{cases} x, & \text{if } x \geq 0 \\ \alpha x, & \text{if } x < 0 \end{cases}$ (with a small constant α, e.g., 0.01)
    - **Properties:** Addresses the dying ReLU issue by allowing a small, non-zero gradient when x < 0.
- **Exponential Linear Unit (ELU) and Scaled ELU (SELU)**
    - **Properties:** Aim to bring mean activations closer to zero, which can speed up learning and improve robustness.
- **Softmax**
    - **Properties:** Used in the output layer for multi-class classification, providing a probability distribution across classes.
    - Formula: $\text{softmax}(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}}$

---

2. Types of Layers in ANNs

Layers in ANNs are the building blocks that process inputs and pass them through the network. Common types include:

- **Input Layer:**
    - The first layer of the network that receives raw input data.
- **Fully Connected (Dense) Layer:**
    - Each neuron is connected to every neuron in the previous layer; used in many standard neural network architectures.
- **Hidden Layers:**
    - Layers between the input and output that perform complex transformations on the data. They can be fully connected or take other specialized forms.
- **Convolutional Layers:**
    - Used primarily in image and signal processing tasks; they apply convolution filters to capture spatial hierarchies.
- **Pooling Layers:**
    - Typically used after convolutional layers to reduce spatial dimensions (e.g., max pooling, average pooling).
- **Recurrent Layers:**
    - Employed in sequence processing tasks; examples include LSTM (Long Short-Term Memory) and GRU (Gated Recurrent Unit) layers.
- **Normalization Layers:**
    - Layers like Batch Normalization or Layer Normalization help stabilize and accelerate training.
- **Dropout Layers:**
    - Randomly drop neurons during training to prevent overfitting by ensuring the network does not rely too much on any one neuron.

---

3. Types of Optimizers

Optimizers adjust the weights of the network during training by minimizing the loss function. Popular optimizers include:

- **Stochastic Gradient Descent (SGD):**
    - **Description:** Updates weights using the gradient computed on a mini-batch of data.

    - **Variants:** Often used with momentum, which adds a fraction of the previous update to the current one for smoother convergence.

- **Momentum:**
    - **Description:** Accelerates SGD by navigating along relevant directions and dampening oscillations.
- **AdaGrad:**
    - **Description:** Adapts the learning rate for each parameter based on historical gradients; performs well for sparse data.
- **RMSProp:**
    - **Description:** Modifies AdaGrad by using a moving average of squared gradients to dampen the aggressive, monotonically decreasing learning rates.
- **Adam (Adaptive Moment Estimation):**
    - **Description:** Combines the ideas of momentum and RMSProp, keeping an exponentially decaying average of past gradients and squared gradients.
    - **Variants:** Adamax, Nadam (which incorporates Nesterov momentum) are some examples.

---

4. Types of Loss Functions

Loss functions measure the error between the network's predictions and the actual target values. The choice depends on the task:

- **Mean Squared Error (MSE):**
    - **Usage:** Commonly used in regression tasks.
        - Formula: $\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$.
- **Mean Absolute Error (MAE):**
    - **Usage:** Also used for regression; less sensitive to outliers compared to MSE.
- **Binary Cross-Entropy (Log Loss):**
    - **Usage:** Used for binary classification problems.
        - Formula: $L = -\frac{1}{n}\sum_{i=1}^{n}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)]$.
- **Categorical Cross-Entropy:**
    - **Usage:** Used for multi-class classification problems.
- **Hinge Loss:**

- **Usage:** Common in training classifiers such as support vector machines (SVMs) or in certain neural network contexts.

- **Huber Loss:**
  - **Usage:** Combines MSE and MAE, providing robustness to outliers while being differentiable at zero.
