Fully Connected Artificial Neural Network (ANN) Report

23B2158

1. Architecture and Naming Convention

Neural Network Architecture

This implementation describes a fully connected artificial neural network (ANN) for the MNIST dataset classification task. The architecture:

- **Input Layer:** 784 neurons for flattened 28x28 grayscale images. Input features are normalized to [0, 1] to improve convergence.
- **Hidden Layers:** Three layers with 256, 128, and 64 neurons, respectively. Tanh activation is used by default but can be switched to ReLU or Sigmoid.
- **Output Layer:** 10 neurons representing digit classes (0-9). Softmax activation outputs class probabilities.

Naming Conventions

Parameters:

- Weights: $W1, W2, \ldots, Wn$ for each layer.
- Biases: $b1, b2, \ldots, bn$.

Intermediate Outputs:

- $Z^{[i]}$: Linear output for layer i.
- $A^{[i]}$: Activation output for layer i.

Rationale for Naming

The naming aligns with standard deep learning conventions:

- Superscripts indicate layer index.
- W, b, Z, and A follow common mathematical notations for clarity.

2. Equations for Forward Pass

Forward Propagation

For layer i:

- **Linear Transformation:**

$$Z^{[i]} = W^{[i]} A^{[i-1]} + b^{[i]}$$

- **Activation:**

$$A^{[i]} = g(Z^{[i]})$$

  g can be Tanh, ReLU, or Sigmoid. Tanh is the default for hidden layers.

Output Layer

The softmax activation function computes class probabilities:

$$A_j^{[L]} = \frac{\exp(Z_j^{[L]})}{\sum_k \exp(Z_k^{[L]})}$$

Dropout Regularization

Dropout reduces overfitting during training:

$$A^{[i]}_{\text{dropout}} = \frac{A^{[i]} \cdot \text{Mask}}{1 - \text{dropout rate}}$$

Where *Mask* is a binary vector indicating active neurons.

3. Gradient Calculation Equations

Backpropagation

Gradients are computed layer by layer, starting from the output layer:

- **Output Layer:**

$$\begin{aligned} dZ^{[L]} &= A^{[L]} - Y \\ dW^{[L]} &= \frac{1}{m} \cdot dZ^{[L]} A^{[L-1]^T} + \lambda W^{[L]} \\ db^{[L]} &= \frac{1}{m} \cdot \sum dZ^{[L]} \end{aligned}$$

- **Hidden Layers:** For layer i:

$$\begin{aligned} dZ^{[i]} &= (W^{[i+1]^T} \cdot dZ^{[i+1]}) \odot g'(Z^{[i]}) \\ dW^{[i]} &= \frac{1}{m} \cdot dZ^{[i]} A^{[i-1]^T} + \lambda W^{[i]} \\ db^{[i]} &= \frac{1}{m} \cdot \sum dZ^{[i]} \end{aligned}$$

Weight Updates

Dual optimization combines Adam and RMSprop:

- **Adam Update:**

$$\begin{aligned} m_t &= \beta_1 m_{t-1} + (1 - \beta_1) g_t \\ v_t &= \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \\ \hat{m}_t &= \frac{m_t}{1 - \beta_1^t}, \quad \hat{v}_t = \frac{v_t}{1 - \beta_2^t} \\ \Delta\theta^{\text{adam}} &= \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \end{aligned}$$

- **RMSprop Update:**

$$\Delta\theta^{\text{rmsprop}} = \frac{g_t}{\sqrt{s_t} + \epsilon}$$

- **Combined Update:**

$$\Delta\theta = \frac{\Delta\theta^{\text{adam}} + \Delta\theta^{\text{rmsprop}}}{2}$$

4. Presentation and Results

Data Handling

- Dataset: MNIST, downloaded via `torchvision.datasets`.
- Preprocessing: Normalized pixel values to [0, 1], one-hot encoded labels.
- Splitting: 80% training, 20% validation, 100% of test data (stratified).

Training and Testing

- **Early Stopping:** Training stops if validation loss does not improve for 10 epochs.
- **Accuracy:** Achieved 93.46% test accuracy.

Visualization

- Loss trajectories for training and validation were plotted.
- Predicted labels are displayed alongside original images in a grid.

Enhancements

- Dual optimization combines Adam and RMSprop for better convergence.
- Dropout regularization mitigates overfitting.