🧠 Neural Network Code: Full Comments + Architecture Diagram

This document provides a complete walkthrough of the NeuralNet class implementation in C++, including a fully commented code excerpt and a diagram illustrating the network architecture used in the project.

# 📐 Neural Network Architecture

The network used in this project has the following structure:

* - Input Layer: 784 neurons (28x28 grayscale image pixels)
* - Hidden Layer: 128 neurons with ReLU activation
* - Output Layer: 10 neurons with softmax (one for each digit 0–9)

- Training: Loss computed via cross-entropy, optimized using manual gradient descent

Diagram:

[ Input (784) ]  
 |  
 V  
[ Hidden (128, ReLU) ]  
 |  
 V  
[ Output (10, Softmax) ]  
 |  
 V  
[ Predicted Digit ]

# 💬 Code Snippet with Comments

Below is a representative section of the core C++ neural network code with explanatory comments:

// Initialize the neural network with input, hidden, and output sizes

NeuralNet::NeuralNet(int inSize, int hiddenSize, int outSize)

{

// Resize and initialize weights and biases for input -> hidden

w1.resize(hiddenSize, std::vector<float>(inSize));

b1.resize(hiddenSize);

// Resize and initialize weights and biases for hidden -> output

w2.resize(outSize, std::vector<float>(hiddenSize));

b2.resize(outSize);

std::mt19937 rng(std::random\_device{}());

std::uniform\_real\_distribution<float> dist(-0.1f, 0.1f);

for (auto& row : w1) for (auto& w : row) w = dist(rng);

for (auto& b : b1) b = dist(rng);

for (auto& row : w2) for (auto& w : row) w = dist(rng);

for (auto& b : b2) b = dist(rng);

}

// Predict the digit from input vector

int NeuralNet::predictDigit(const std::vector<float>& input) {

std::vector<float> hidden(b1.size());

for (size\_t i = 0; i < b1.size(); ++i)

hidden[i] = relu(dot(input, w1[i]) + b1[i]);

std::vector<float> output(b2.size());

for (size\_t i = 0; i < b2.size(); ++i)

output[i] = dot(hidden, w2[i]) + b2[i];

auto probs = softmax(output);

return std::max\_element(probs.begin(), probs.end()) - probs.begin();

}