📄 NeuralNet.hpp and NeuralNet.cpp: Line-by-Line Explanation

This document explains each function and key logic in the NeuralNet.hpp and NeuralNet.cpp files. The neural network implementation is written entirely from scratch, supporting one hidden layer with ReLU activation, softmax output, and a basic training loop with backpropagation and gradient descent.

# 📘 NeuralNet.hpp

This header file declares the structure of the NeuralNet class and its key public and private methods.

Key Members & Methods:

* - NeuralNet(int inSize, int hiddenSize, int outSize): Constructor to initialize weights and biases.
* - int predictDigit(const std::vector<float>& input): Forward pass to return the index of the highest output (predicted digit).
* - float train(const std::vector<float>& input, const std::vector<float>& target, float lr): One training step using backpropagation.
* - void save(const std::string& path): Save weights and biases to binary file.
* - void load(const std::string& path): Load weights and biases from binary file.

# 💻 NeuralNet.cpp

This file implements the neural network methods, including weight initialization, forward propagation, backward propagation (training), and serialization for saving/loading models.

Important Concepts and Functions:

* - dot(a, b): Computes the dot product of two vectors.
* - relu(v): Applies ReLU activation to each element in a vector.
* - softmax(v): Converts logits to probabilities for classification.
* - NeuralNet::NeuralNet(): Initializes weights and biases with small random values.
* - NeuralNet::predictDigit(): Runs a forward pass and returns the predicted class index.
* - NeuralNet::train(): Performs forward pass, calculates loss, and updates weights via backpropagation.
* - NeuralNet::save(): Serializes all weights and biases to a binary file.
* - NeuralNet::load(): Reads weights and biases from a binary file into memory.

Each method is implemented using simple loops and vector operations to maintain portability and understandability.

Note: The implementation does not use external libraries such as Eigen or PyTorch. All matrix operations are manually implemented.