Week 2 Assignment - Part 2 Report

Venkat Neelraj Nitta

College Of Professional Studies, Northeastern University

EAI6080: Advanced Analytical Utilization

Instructor: Mimoza Dimodugno PhD

Oct 3rd, 2025

A neural network is organized as layers of interconnected nodes (neurons). Each neuron receives inputs, applies a weighted sum, passes it through an activation function (like sigmoid), and outputs a value. The network typically includes Input layer: Receives raw data (e.g., pixel values for images); Hidden layers: Transform inputs through learned weights and nonlinear activations and Output layer: Produces predictions (e.g., class probabilities). Understanding this structure helps you visualize how data flows and transforms through the network, and why deeper architectures can capture more complex patterns.

To build and train a NN efficiently, you decompose the workflow into reusable functions like Forward propagation for computing activations layer by layer followed by Backward propagation calculating gradients of the loss with respect to weights, using the chain rule and finally Loss functions to measures prediction error for classification tasks. Parameter initialization is a critical step: weights are set to small random values to break symmetry, and biases are often initialized to zero. For example, in Python, weights can be initialized with `W = np.random.randn(input_dim, output_dim) * 0.01` and biases with `b = np.zeros((1, output_dim)).`

Neural networks are powerful tools for solving real-world problems by learning patterns from data. In image classification, inputs are pixel arrays reshaped into vectors, and the network learns to map these to class labels such as cat or dog. In text classification, inputs are word embeddings or token vectors, enabling the network to distinguish spam from non-spam or to determine sentiment polarity. Implementing vectorized operations with NumPy ensures that code is efficient and scalable for large datasets, which is crucial in practical scenarios.

In summary, this workflow taught me to structure a neural network, write modular code for forward and backward passes, initialize and update parameters, and apply these concepts to real-world tasks. Mastering these steps is foundational for building more advanced models and tackling complex machine learning problems.

References:

1. EAI_6080_Week2_Part2.ipynb