Machine Learning – Artificial Neural Network – Report

Shlomi Ben-Shushan

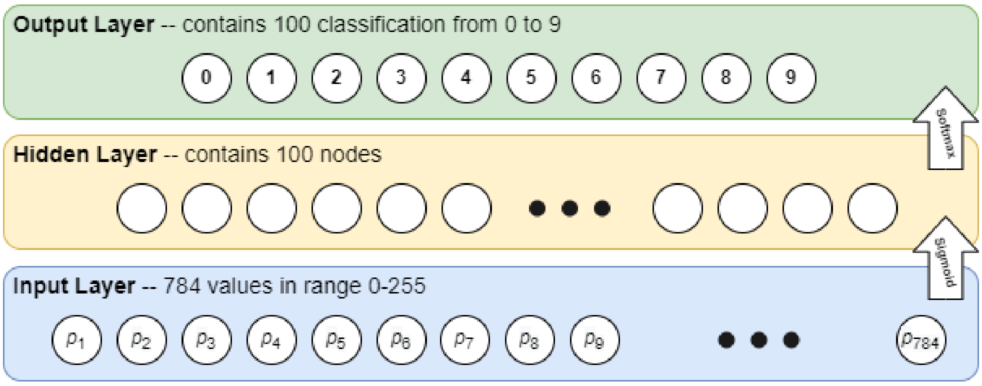
Summary

In this report I'll describe my model of a Neural-Network (NN) on MNIST dataset, I'll explain the network architecture, and I'll explain how I found the most efficient hyper-parameters.

Model Description

The inputs are grayscale images of 10 handwritten digits (0-9) and the goal is to train a classifier that classifies this data. To solve this, I've implemented the classifier as a multi-class neural network with one hidden layer, such that the transition from the input layer to the hidden layer is done by the Sigmoid activation function and then the transition to the output layer is done by Softmax function. The model trains on a training-set by running T epochs so that in each epoch the program shuffles the data-set, run forward-propagation, calculate the loss, run back-propagation and update parameters with gradient decent. This process results parameters that can be used by the model to predict the targets of new unseen objects with forward-propagation only.

Neural Network Architecture


Output Layer -- contains 100 classification from 0 to 9
0
1
2
3
4
5
6
7
8
9
Softmax
Hidden Layer -- contains 100 nodes
Sigmoid
Input Layer -- 784 values in range 0-255
p_1
p_2
p_3
p_4
p_5
p_6
p_7
p_8
p_9
p_{784}

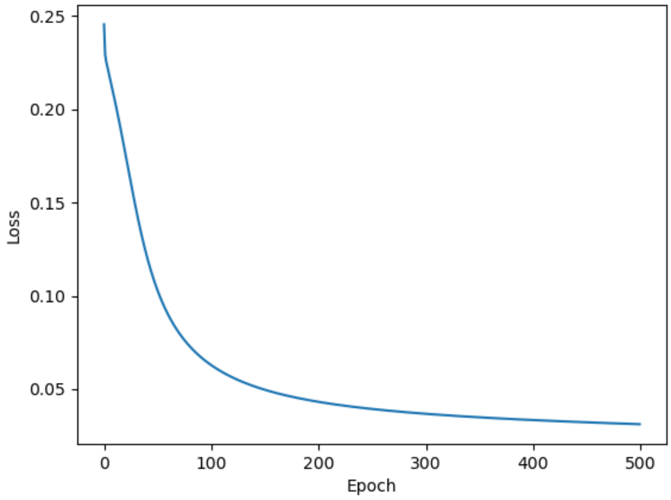
Hyper-Parameters

To find the hyper-parameters that makes the classifier most accurate I used cross-validation methodology on a diverse range of potential hyper-parameters combinations (code attached in $ex3.py$ file), and the results were: $\boldsymbol{500\ epochs}, \boldsymbol{\eta = 0.4}$ and hidden layer with $\mathbf{100}$ **nodes**.

Note: I have considered the number of inputs (784) and the number of outputs (10) as global constants.

Loss

The following graph describes the loss as a function of epoch. It shows clearly how the loss decents from epoch to epoch.


Loss
Epoch