CS559 Project Report
Amie Roten

Introduction

Neural networks, though frequently alluded to in discussions of artificial intelligence and machine learning, are often considered opaque, “black box”-type algorithms to most outside of the field, until recently, myself included. As I became more exposed to these algorithms during my time at OHSU, and witnessed first-hand how useful and powerful they can be when applied to difficult problems, the more I wanted to experiment with and learn more about them. However, up until this class, and, if I’m being honest, this project, the underpinnings of the algorithm itself remained mysterious to me. If I wanted to start using these networks to solve real-world problems, I knew that I needed to understand what was happening “under-the-hood” as much as possible. So, I set out on this project in hopes of expanding my knowledge of the neural network algorithm, and specifically, how backpropagation is used to train the weights of a network. In addition, I wanted to build a framework that allowed the user to customize their network to suit the needs of their particular dataset, from changing the number of hidden layers, as well as the nodes in each layer, and also altering the objective function in order to solve regression, binary classification, and multi-class classification problems.

So, the remainder of this report will outline my approach to building the neural network framework, describe some pitfalls and challenges that came up in the learning process, as well as discuss some experiments using the framework on a selection of datasets with varying complexity. I will discuss the results of these experiments, as well as some interesting questions that arose during the process. I will conclude by discuss my impressions of the process, improvements that could still be made to the framework, and how I plan to use the knowledge gained during the process in future work.

Methods

As mentioned above, I knew that I wanted to have a flexible implementation for the neural network framework, that allows the user to adjust the following basic network structure parameters: number of hidden layers, number of nodes in each layer, activation for each layer, and objective/cost function used to train the network weights. This would allow the user (aka, me!) to not only test the performance and functionality of the network on a number of datasets, but also experiment with different network complexities/activation functions, and determine whether certain combinations were better suited for certain datasets and problem types.

Data

Since the framework was built to be very flexible, and allow for use on a number of different problem/dataset types, I needed to make sure that all possible conditions were exercised and determined to be functioning correctly. To do this, I decided to perform a number of experiments u

Results

Iris Dataset: Simple Multiclass Classification

Boston Housing Dataset: Simple Regression

Iris Dataset: Simple Multiclass Classification

Discussion

Conclusions