CSCI 4962 Project Report

Siddha Kilaru

Neural Networks as Numeric Solvers

Introduction

The goal of this project is to explore how neural networks can be used to approximate solutions to differential equations. The paper "Physics Informed Deep Learning" by Maziar Raissi, Paris Perdikaris, George Em Karniadakis is the inspiration for this project. In the paper, they introduce the concept of physics informed neural networks, and use it to find solutions to nonlinear partial differential equations. In this project, I use the methods detailed in the paper, but with simpler differential equations. Specifically, I focus on first order and second order ordinary differential equations (ODE), which have analytical solutions. In reality, there is no use of approximating a differential equation with an analytical solution; however, for the purposes of this project I want to compute the accuracy of the approximated solution. In addition to using the methods outlined in paper, I also experiment with different model hyperparameters and see how it affects the runtime, accuracy, etc. Finally, I compare approximate solutions derived from neural networks with traditional methods such as Euler's methods and higher order Runge-Kutta methods.

Unsupervised Approach

Consider the simple differential equation $y' = x$ with initial condition $y(0) = 1$. Also assume we use a neural network to approximate the solution in a supervised learning setting. The set of features would be the x points, but the set of labels is not immediately obvious. One could say the set of labels are a finite set of y points. There are couple problems however. First, if the differential equation has no solution, then the labels are attainable. Secondly, if we use real world observed data there might be too granular and contain too much noise. As a result, a supervised learning model is not completely appropriate for this task. The paper mentioned above uses an alternative method that makes use of unsupervised learning. The main idea is for the neural networks to learn the gradient of the differential equation. Moreover, for a given set of labels containing x points, each x points replicates the gradient described by the differential equation. In the case for $y' = x$, the gradient at each point should be equal to itself. One way to do this is by devising an error function as follows:

$$\alpha\Big(\sum_{i-0}^{n}(\hat{y}_i' - x_i)^2\Big)^{1/2} + \beta(\hat{y}_0 - 1).$$

Above, α and β are hyperparameters. The first term in the sum enforces the ODE constraint, and the second term enforces the intitial condition. This error function can be used by a neural network to find an approximate solution. In the sections below, I experiment with different differential equations along and their corresponding error functions.

Approximating First Order ODEs

Approximating Second Order ODEs

Comparison to Traditional Methods