Implementing Optimization Algorithms on Neural Networks

Background:

Neural networks or Artificial Neural Networks are a subset of machine learning and are at the heart of deep learning algorithms. Their name and structure are inspired by the human brain, mimicking the way that biological neurons signal to one another. They are comprised of node layers, containing an input layer, one or more hidden layers, and an output layer. Each node, or artificial neuron, connects to another and has an associated weight and threshold. If the output of any individual node is above the specified threshold value, that node is activated, sending data to the next layer of the network. Otherwise, no data is passed along to the next layer of the network.

Neural networks rely on training data to learn and improve their accuracy over time. However, once these learning algorithms are fine-tuned for accuracy, they are powerful tools in computer science and artificial intelligence, allowing us to classify and cluster data at a high velocity. There are various optimization algorithms out there that help the Neural Network in improving the accuracy of the network.

Proposal:

The goal of our project is to determine which of the following optimization algorithms will suit best for our neural network that will be built from scratch, without using any advanced packages(numpy only), and based on the size of the dataset. To test, we shall consider three datasets having at least 1000+ observations with different number of features calling them Small(at least 5 features), Medium(between 10 and 15) & Large(More than 15) datasets. The optimization techniques that we will be taking into consideration are Backpropagation, Variable learning rate, Steepest Gradient Descent(SGD) with momentum, Conjugate Gradient Algorithm, Levenberg-Marquardt(LM) algorithm, Bayesian Regularization followed by LM algorithm with momentum.

To view the performance of the network, we will be using Sum Squared Error(SSE) and Mean Square Error and we will try to judge convergence time(if time line permits).

Repository Source

GitHub Repo: Follow this LINK to view our GitHub repository.

TIMELINE OF COMPLETION:

Project Discussion (10/04/2023) - Meeting notes:
First meet for objective clarification, project scope clarification and expectation alignment. The following are the to-do's for each member of the group:-
Aditya Nayak -
Dataset Collection - Requirement of 3 Datasets { Features < 5 (Small), 5 < Features < 15 (Medium), Features > 15 (Large) }
Selection Tasks - 1. Data Cleaning 2. Feature selection/Feature reduction using PCA/ EDA/ SVD/ Random Forest - For final features usable in model.
To implement - Variable Learning Rate, SGD with Momentum
Aditya Kumar -
Push already developed backpropagation code with SGD for 1-S-1 network to main branch for usability by other members.
Generalize backpropagation code with SGD for custom number of layers, custom transfer function, custom number of neurons for project progress.
To implement - Levenberg-Marqadt algorithm, Bayesian regularizaton, LM algorithm with momentum.
Medhasweta Sen -
Generalize backpropagation algorithm along with Aditya Kumar.
To implement - Conjugate Gradient algorithm
Project Discussion (17/04/2023) - Meeting notes:

Medhasweta Sen -
Implement PCA and other dimension reduction techniques on datasets selected by Aditya Nayak to get final usable number of features.
Aditya Kumar -
Continue backpropagation generalization - multiple layers, multiple ransfer functions. Confirm PCA usage and benefits.
Aditya Nayak -
Continue dataset cleaning and PCA -