CSCI 447 — Machine Learning

Project #4

Assigned: November 4, 2024
Design Document Due: November 15, 2024
Project Due: December 6, 2024

Introduction

This assignment requires you to apply three different population-based (evolutionary or swarm) approaches to train a feedforward neural network and then compare the results of population-based training to your implementation of backpropagation. For this particular assignment, you will focus only on training the weights of a pre-defined neural network, thus the only difference between the various implementations for a given problem will be the learning algorithm itself.

Starting with an untrained neural network, you will train the weights of the network using each of the following algorithms (separately):

- Backpropagation
- Genetic algorithm with real-valued chromosomes
- Differential evolution
- Particle swarm optimization

For each of these algorithms, you will conduct appropriate hyperparameter tuning studies to determine the best parameter settings (e.g., learning rate, momentum, inertia, etc.) prior to running the head-to-head comparisons. You should plan to use the method developed as part of Project 2 for tuning. The rest of the project will implement a cross-validation experiment similar to experiments run in the previous three projects.

Data Sets

For this assignment, you will use three classification datasets and three regression data sets that you will download from the UCI Machine Learning Repository. The data sets you will use are the following. They can also be found in the Assignments area under Content in Brightspace. Each data set has two files: a .NAMES file and a .DATA file. (Note that the Wine data set, should we use it, actually has two .DATA files.) The .NAMES file explains the data, providing information on all of the columns, including their types and what they represent. The .DATA file is the data itself. You do not need to be able to process the various data files in an entirely autonomous fashion, but you should try to generlize the processing as much as possible. The data sets you will use are as follows:

- Breast Cancer [Classification]

  `https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Original%29`

  This breast cancer databases was obtained from the University of Wisconsin Hospitals, Madison from Dr. William H. Wolberg.

- Glass [Classification]

  `https://archive.ics.uci.edu/ml/datasets/Glass+Identification`

  The study of classification of types of glass was motivated by criminological investigation.

- Soybean (small) [Classification]

  `https://archive.ics.uci.edu/ml/datasets/Soybean+%28Small%29`

  A small subset of the original soybean database.

- Abalone [Regression]

  `https://archive.ics.uci.edu/ml/datasets/Abalone`

  Predicting the age of abalone from physical measurements.

- Computer Hardware [Regression]

  `https://archive.ics.uci.edu/ml/datasets/Computer+Hardware`

  The estimated relative performance values were estimated by the authors using a linear regression method. The gives you a chance to see how well you can replicate the results with these two models.

- Forest Fires [Regression]

  `https://archive.ics.uci.edu/ml/datasets/Forest+Fires`

  This is a difficult regression task, where the aim is to predict the burned area of forest fires, in the northeast region of Portugal, by using meteorological and other data .

Issues

When using these data sets, be careful of some issues.

- Some of the data sets have missing attribute (i.e., feature) values. When this occurs in low numbers, you may simply edit the corresponding values out of the data sets. For more occurrences, you should do some kind of "data imputation" where, basically, you generate a value of some kind. This can be purely random, it can be sampled according to the conditional probability of the values occurring, given the underlying class for that example, or it can just be the mean or median value of the feature. The choice is yours, but be sure to document your choice.

- Preprocessing attributes involve the following. For categorical variables, you should one-hot-code. For numerical variables, you should either apply min-max normalization (usually appropriate when using the logistic function for the hidden nodes) or z-score normalization, i.e., $z = (x - \mu)/\sigma$ (usually appropriate when using the hyperbolic tangent for the hidden nodes). You do not need to normalize the outputs for regression. For classification, you should assume one output per class. This is recommended even with two-class problems.

- Other hyperparameters to be tuned include the learning rate, the momentum term (if you use momentum), and the batch size (if you use minibatches). For the population-based algorithm, you will have to tune things such as population size, crossover rate (GA), mutation rate (GA), scaling factor (DE), binomial crossover probability (DE), inertia (PSO), cognitive update rate (PSO), and social update rate (PSO). Plan to use the tuning procedure that was outlined and implemented as part of Project 2.

- With respect to the network architectures, you should use the "best" architectures for each data set from Project 3. This means you do not need to worry about tuning the number of hidden nodes per layer since you already did that.

Requirements

Design Document

You are required to submit a design document outline the design of your programming assignment. Details of what should go into your design document is the same for all of the programming assignments in this course. You can find the details of what needs to go into this document in the Content area under Administrative.

Programming

Your assignment consists of the following steps:

- Download the six (6) data sets from the UCI Machine Learning repository. You can use the links above or download them from Brightspace. You can also find this repository at `http://archive.ics.uci.edu/ml/`.
- Pre-process the data to ensure you are working with complete examples (i.e., no missing attribute values).
- Implement feedforward neural network training using a genetic algorithm for training the weights. Use your GA to train the 0, 1, and 2-hidden layer networks you produced from Project 3.
- Implement feedforward neural network training using a differential evolution procedure for training the weights. Use DE to train the 0, 1, and 2-hidden layer networks you produced from Project 3.
- Implement feedforward neural network training using a particle swarm optimization algorithm to train the weights. Use PSO to train the 0, 1, and 2-hidden layer networks you produced from Project 3.
- Develop a hypothesis focusing on convergence rate and final performance of each of the implemented algorithms.
- Compare the results of all three of your population-based algorithms and backpropagation (which should have been implemented previously), after tuning. You may reuse networks trained during Project 3 on the data sets given above as long as the folds for cross-validation are the same.. Your experimental design should apply 10-fold cross validation.

Paper

Write a very brief paper summarizing the results of your experiments. Your paper is required to be at least 5 pages and no more than 10 pages using the JMLR format. This page limit is all-inclusive of figures, tables, and references. However, the page limit does not include the Appendix listing work effort of the team. You can find templates for this format at `http://www.jmlr.org/format/format.html`. The format is also available within Overleaf. Make sure you explain the experimental setup, the tuning process, and the final parameters used for each algorithm.

- Title and author name
- Problem statement, including hypothesis
- Description of your experimental approach and software design
- Presentation of the results of your experiments
- A discussion of the behavior of your algorithms, combined with any conclusions you can draw
- Summary
- References (Only required if you use a resource other than the course content.)
- Appendix listing who did what on the assignment (remember equal work balance is expected)

Video

Create a video that is no longer than 5 minutes long demonstrating the functioning of your code. For the video, the following constitute minimal requirements that must be satisfied:

- The video is to be no longer than 5 minutes long.

- The video should be provided in mp4 format. Alternatively, it can be uploaded to a streaming service such as YouTube with a link provided.
- Fast forwarding is permitted through long computational cycles. Fast forwarding is *not permitted* whenever there is a voice-over or when results are being presented.
- Be sure to provide verbal commentary or explanation on all of the elements you are demonstrating.
- Provide sample outputs from one test fold showing performance on one classification and one regression network. Show results for the two hidden layer cases only but for each of the learning methods.
- Demonstrate each of the main operations for the GA: selection, crossover, and mutation.
- Demonstrate each of the main operations for the DE: crossover and mutation.
- Demonstrate each of the main operations for the PSO: pbest calculation, gbest calculation, velocity update, and position update.
- Show the average performance over the ten folds for one of the classification data sets and one of the regression data sets for each of the networks trained with each of the algorithms. (Note that results for all networks and all data sets are required in your paper.)

Group Work

All projects are completed in teams of two or three people. Team dynamics can be difficult in that there can be an unfair balance of work completed. As an attempt to avoid this issue, the following group work requirements are put in place.

- Include a summary of the tasks completed and percent level of effort for each member of the team in accordance with the following items. This document shall be included as part of your paper submission. The level of effort will be used to determine weighting of the project grade. Specifically, if a team member does less than 40% (for a two-person team) or 25% (for a three-person team), their grade will be adjusted downward to reflect their level of effort.
- There are three major components to each project—code, paper, and video. A division of labor where one person is responsible for code, one person for the paper, and one person for the video is *not allowed*. Everyone on the team is required to contribute to all three of the components.
- The coding can be subdivided based on the individual coding requirements (e.g., data preparation, algorithms, experimental framework, analysis code etc.). In this assignment, there are three new algorithms to implement, as well as a potential need to debug code from Project 3. There should be ample opportunity for everyone to code.
- The paper should be subdivided based on paper section. It is not sufficient for a member of the team to write the entire paper or just to proofread the paper. The goal is for each member of the team to contribute equal content to the final paper.
- The video should be recorded by a single person; however, all must contribute to the development of the video (e.g., script, setup for a particular item to be shown, editing of the final video). For the video, a different person should do the recording from project to project.

Submission

Submit your fully documented code with the outputs from running your programs, your video, and your paper. Your submission should include the following components and should be submitted as a "group" submission (i.e., only one submission for the entire group).

- This assignment requires you to submit a design document two weeks before submitting the program, video, and paper. Details on what needs to go into the design document are provided in the Content Area. Your design docment will be checked using the TurnItIn plagiarism checking system,. Submit to the "P1 Design Document" portion of the assignment. This document can be formatted however you wish but must be submitted as a PDF file. Your design document is worth 30% of the grade.

- Zip your source code files into a single zip file. Do not use alternative or archive formats like tar, gz, rar, etc. Be sure to include the team member contribution report with your code. Submit to the "P1 Code" portion of the assignment. The code must be well structured and fully commented. Code is worth 10% of the overall project grade.

- Submit a PDF of your report to the "P1 Paper" portion of the assignment. Brightspace is set up to accept PDF files only. Your report will be checked using the TurnItIn plagiarism checking system. The paper must be formatted per JMLR formatting requirements. All math should be included using the math editing capabilities of your document processing tool. Figures needs to be high quality. Do not use screenshots. Your paper is worth 40% of the grade.

- Submit either your video or a text file containing a link to your video in the "P1 Video" portion of the assignment. Note that a file is required to be submitted, so if you are using a streaming service, remember to submit that text file. Your video is limited to five minutes maximum. Anything that appears in the video beyond five minutes will be ignored. Use the video the satisfy the requirements above, rather than "walking through" the code. The video is worth 20% of the grade.