CSCI 447 — Machine Learning

Project #3

Assigned: October 9, 2024
Project Due: November 4, 2024

Introduction

This assignment requires you to implement several neural network training algorithms to perform classification and regression on several data sets from the UCI Machine Learning Repository. All of the networks will be densely-connected feedforward networks, meaning that the nodes will be laid out in layers, and every node in a given layer will connect to every node in the next layer. The hidden nodes will incorporate non-linear activation functions, such as the logistic or the hyperbolic tangent function. The output node(s) will depend upon whether the network is being used for classification or regression. For classification, softmax will be used at the output. For regression, a simple linear activation will be applied. Consideration should be given to using bias nodes in each of the layers of the network. In addition to implementing and testing the algorithms, you are required to write a research paper describing the results of your experiments.

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.
- You will need to determine the number of hidden nodes per layer via a tuning process. Note that you can use a rule of thumb that you have fewer hidden nodes than inputs, but be careful that this may not always work. In addition, when you go to two hidden layers, you should not need as many hidden nodes per layer as you needed in the one hidden layer case.
- Other hyperparameters to be tuned include the learning rate, the momentum term (if you use momentum), and the batch size (if you use minibatches). Plan to use the tuning procedure that was outlined and implemented as part of Project 2.

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 download from the URLs above or from Brightspace. You can also find this repository at `http://archive.ics.uci.edu/ml/`
- Pre-process each data set as necessary to handle missing data and normalize as needed.
- Implement a multi-layer feedforward network with backpropagation learning capable of training a network with an arbitrary number of inputs, an arbitrary number of hidden layers, an arbitrary number of hidden nodes by layer, and an arbitrary number of outputs. In other words, the number of inputs, hidden layers, hidden units by layer, and outputs should be furnished as inputs to your program. Be able to specify whether a node uses a linear activation function for regression or a softmax activation function for classification.

- The hidden nodes should use a sigmoid activation function (you may choose between logistic or hyperbolic tangent). Do not use ReLU or other similar activation functions. Remember that these choices affect the update rules because of having different derivatives. Implement learning such that momentum is provided as an option.
- Develop a hypothesis focusing on convergence rate and final performance of each of the different network architectures for each of the various problems.
- Test the MLP (backpropagation) algorithm on networks with 0, 1, and 2 hidden layers. 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 network and one regression network. Show results for each of the cases where you have no hidden layers, one hidden layer, and two hidden layers.
- Show a sample model for the smallest of each of your neural network types. This will consist of showing the weight matrices with the inputs/outputs of the layer labeled in some way.
- Demonstrate and explain how an example is propagated through a two hidden layer network of your choice. Be sure to show the activations at each layer being calculated correctly.

- Demonstrate the gradient calculation at the output for one classification network and one regression network.
- Demonstrate the weight updates occurring on a two-layer network for each of the layers for one classification network and one regression network.
- Show the average performance over the ten folds for one of the data sets for each of the types of networks. (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 is only one algorithm to implement, so care should be given to make sure the coding requirements are distributed equitably.
- 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.