Neural Networks: Backpropagation Implementation Report

Chun-Chan (Bill) Cheng
Texas A&M University
College Station, TX
aznchat@tamu.edu

1. INTRODUCTION

The purpose of this project is to implement backpropagation algorithm described in the textbook and evaluate the confidence interval of the UCI data using the ten fold cross validation.

2. THE PROGRAM

2.1 Files Turned In

```
backPropAlgorithm.py
Case/
  data.txt
  structure.txt
  Different_Test_Case/
      test cases....
```

2.2 How to Compile the Program?

Inside the zip file there is a `backPropAlgorithm.py`. In order to run the script, run `python backPropAlgorithm.py -number_of_hidden_nodes -test_file_to run.` (e.g `python backPropAlgorithm.py 3 balance.txt`). More information can be seen by typing `python backPropAlgorithm.py -help`. Not that all test files are located in `Case/`.

Below is an example of a `structure.txt`:

```
buying,maint,doors,persons,safety,class
vhigh,high,med,low
vhigh,high,med,low
2,3,4,5more
2,4,more
low,med,high
unacc,acc,good,vgood
```

In the first line of the `structure.txt` is the name of the different attributes, the line always ends with a the element class. This hierarchy is to distinguish different attributes and the classifier. The first element of the first element maps to the second line of the `structure.txt`, in other words, `vhigh,high,med,low` is the attribute values of `buying`.

ACM ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

2.3 High Level Overview of Program

The program starts with reading in the structure file and data set file. While reading the data set files, **when it finds a missing attributes it randomly selects a value and inserts it into the missing cell**. The randomly selection of attribute values to insert was chosen is due to the fact that randomly selecting them is similar is putting weight on different attributes according to their numbers in the data set.

Next, it loops through the first tenth of the data in the data set, training the neural networks with the rest of nine-tenth data, and validates the one-tenth data. The program stops when the maximum iteration is met or the error rate is below a threshold.

In the last step the program prints out the iteration and error rate and calculates the mean and standard deviation.

2.4 Pseudo Code

Data: UCI Data Set
initialize variable
data = read_in_data()
normalize_data = normalize(data)
for *the i^{th} chunk of the 10 spitted data set* **do**
  training_data = all data except the i^th data
  training_target = all target except the i^th target
  validation_data = the i^th data
  validation_target = the i^th target data
  for *loop in max loop* **do**
    error = train(training_data, training_target)
    if *error < min error* **then**
      break
    end
  end
  output = run(validation_data)
  validate errors = (validate(validation_data, output))
end
confident_interval_calculation(cross_validation_errors)

Algorithm 1: Backpropagation Algorithm

2.5 Normalization of Input and Outputs

The input normalization was implemented by normalizing all the inputs to less than one. For example if a data set had `sunny,hot,high,weak`, this would be normalized to `[0.3333333333333333, 0.3333333333333333, 0.5, 0.5]` and the output, `no`, would be normalized to `[0,1]`

3. DATA SETS

3.1 Car Evaluation

This data set is composed of six attributes: buying price, maintenance price, number of doors, number of persons that can fit, size of trunk (lug_boot in database), and the safety of the car. The purpose of this data set is to evaluate if a car is good or not. The classifier are as follows: unacc, acc, good, v-good.

3.2 Balance Scale

This data set is composed of four attributes: left-weight, left-distance, right-weight, and right-distance.This data set was generated to model psychological experimental results. Each example is classified as having the balance scale tip to the right, tip to the left, or be balanced. The correct way to find the class is the greater of (left-distance * left-weight) and (right-distance * right-weight). If they are equal, it is balanced.

3.3 Congressional Voting

This data set is composed of sixteen attributes: handicapped-infants, water-project-cost-sharing, adoption of the budget resolution, physician feefreeze, el salvador aid, religious groups in schools, anti satellite test ban, aid to nicaraguan contras, mx missile, immigration, synfuels corporation cutback, education spending, superfund right to sue, crime, duty free exports, export administration act south africa.

Also, this data set includes votes for each of the U.S. House of Representatives Congressmen on the 16 key votes identified by the CQA. The CQA lists nine different types of votes: voted for, paired for, and announced for (these three simplified to yea), voted against, paired against, and announced against (these three simplified to nay), voted present, voted present to avoid conflict of interest, and did not vote or otherwise make a position known (these three simplified to an unknown disposition).

3.4 Nursery

This data set is composed of eight attributes: parents, has nurse, form, children, housing, finance, social, health. Nursery Database was derived from a hierarchical decision model originally developed to rank applications for nursery schools. It was used during several years in 1980's when there was excessive enrollment to these schools in Ljubljana, Slovenia, and the rejected applications frequently needed an objective explanation.

3.5 Hayes-Roth

This data set is composed of four attributes: hobby, age, educational level, marital status. This data set was from *Hayes-Roth, B., & Hayes-Roth, F. (1977). Concept learning and the recognition and classification of exemplars. Journal of Verbal Learning and Verbal Behavior, 16, 321-338.* This database borrowed the concepts of psychologists that were used in their laboratory experiments that aim to investigate human categorization in natural domain.

3.6 Statistics

Since in the last project, the Decision Tree, I had not fully implemented the pruning method so as you can see in Neural Network tables, all the stats are way better then the decision tree. The **training rates and momentum for Car evaluation and Hayes-Roth were 0.002, 0.9 respectively. The others' training rate and momentum were 0.0005 and 0.9 respectively.** All of the iteration all exceed the limit before converging to the threshold I set which was 0.01 except for the congressional voting, which has an average of 4000 epochs before it gets below the minimum error threshold (0.005).

Data Set	μ	σ	95% C.I.
Car Evalutaion	0.60	0.23	(0.45, 0.75)
Balance Scale	0.07	0.43	(-0.27, 0.33)
Congressional Voting	0.43	0.34	(0.22, 0.64)
Nursery	0.31	0.33	(0.10, 0.51)
Hayes-Roth	0.64	0.29	(0.46, 0.82)

Table 1: Statistics on Decision Tree Data Set

Data Set	μ	σ	95% C.I.
Car Evalutaion	0.89	0.023	(0.88, 0.92)
Balance Scale	0.93	0.03	(0.92, 0.96)
Congressional Voting	0.99	0.01	(0.99, 1.002)
Nursery	0.91	0.01	(0.90, 0.92)
Hayes-Roth	0.79	0.17	(0.67, 0.92)

Table 2: Statistics on Neural Network Data Set

The reason can be that because the standard back propagation method computed only $\frac{\partial E}{\partial w}$, the partial first derivative of the overall error function with respect to each weight in the network. Given these derivatives, we can perform a gradient descent in weight space, reducing the error with each step. It is straightforward to show that if we take infinitesimal steps down the gradient vector, running a new training epoch to recompute the gradient after each step, we will eventually reach a local minimum of the error function. However if we want to have a faster convergence, we can not take these small steps which may lead us to a local minimum rather than the global minimum, in this case we are stuck.

3.7 Momentum, Training Rate

It can be seen by testing that any training rates larger than 0.1 causes the neural networks to not converge, it stays at one spot until the maximum epochs is reached. The best training rate varies from data but is in the region of $0.0001, 0.001$. Anything higher than the region causes the converging to stall and anything lower than the region causes the convergence to slow down dramatically.

3.8 Hidden Nodes and Layers

I used the congressional-voting data set to test the number and layer of hidden node against the speed of convergence, since the congressional voting data set was the fastest to converge comparing to all other data sets. After increasing the hidden nodes in the neural network, it seemed to have little effect against the speed of the convergence while in the first iteration, the errors were large but the convergence speed was pretty much all the same. I have researched the literature, and the only reason for the decrease of performance when adding layers is attributed to the bad initial-

ization. However, this shouldn't apply here since I am doing pre-training.

4. CONCLUSION

In this project we have implemented a neural network using the back propagation algorithm and tested the network accuracy with UCI repository data sets. The overall outcome was a lot better than the decision tree for all the data sets except Hayes-Roth. This may be caused by the numerical attributes that Hayes-Roth has which is different from the other data sets.