San Jose State University SJSU ScholarWorks

Master's Projects

Master's Theses and Graduate Research

Spring 2013

Modular Approach to Big Data using Neural Networks

Animesh Dutta San Jose State University

Follow this and additional works at: https://scholarworks.sjsu.edu/etd_projects

Recommended Citation

Dutta, Animesh, "Modular Approach to Big Data using Neural Networks" (2013). *Master's Projects*. 315. DOI: https://doi.org/10.31979/etd.7kd8-w4us https://scholarworks.sjsu.edu/etd_projects/315

This Master's Project is brought to you for free and open access by the Master's Theses and Graduate Research at SJSU ScholarWorks. It has been accepted for inclusion in Master's Projects by an authorized administrator of SJSU ScholarWorks. For more information, please contact scholarworks@sjsu.edu.

CS298 REPORT

Modular Approach to Big Data using Neural Networks

Advisor: Dr. Chris Tseng

Animesh Dutta May 2013

A Writing Project Presented to The Faculty of the Department of Computer Science, San Jose State University. In Partial Fulfillment of the Requirements for the Degree: Master of Science

© 2013

Animesh Dutta

ALL RIGHTS RESERVED

The Designated Committee Approves the Project Titled

Modular Approach to Big Data Using Neural Networks

By

Animesh Dutta

Approved for the Department of Computer Science

San Jose State University

May 2013

Dr. Chris Tseng

Dr. Tsau Young Lin

Mr. Naresh Parmar

Department of Computer Science Department of Computer Science Member of Technical Staff, PayPal

ABSTRACT

MODULAR APPROACH TO BIG DATA USING NEURAL NETWORKS

Machine learning can be used to recognize patterns, classify data into classes and make predictions. Neural Networks are one of the many machine learning tools that are capable of performing these tasks. The greatest challenges that we face while dealing with the IBM Watson dataset is the high amount of dimensionality, both in terms of the number of features the data has, as well the number of rows of data we are dealing with. The aim of the project is to identify a course of action that can be chosen when dealing with similar problems. The project aims at setting up and experimenting with different strategies of training neural networks in order to reduce training time and increase prediction accuracy. The project will contrast the advantages and disadvantages of using these modular approaches and provide a completely open source implementation of the system.

ACKNOWLEDGEMENTS

I would like to thank Dr. Chris Tseng, my project advisor, for his continuous support and expert guidance throughout the course of the project. I would like to thank my committee members, Dr. T.Y. Lin and Mr. Naresh Parmar, for their feedback and suggestions. At last, I would like thank my friends for their encouraging support through the project.

Contents

| 1.1 Introduction and Problem Statement. 1.2 Project Goal. 1.3 Neural Networks 2. Project Design 2.1 Design Overview 2.2 Basic Implementation using Matlab Setting Up Matlab Importing the Data Importing Data from a text or CSV file Starting the Neural Network Toolbox Selecting the Data Neural Network Architecture Neural Network Architecture Neural Network Training Network Evaluation Saving Results Generating Scripts Saving the Results into Workspace 2.3 Performance Improvement Strategies Balanced Datasets Balanced Datasets Balanced Datasets Mini Batch Training Online Training with Preliminary Batch Training Multiple Experts Network 3. Project Implementation 3.1 Selecting an Open Source Neural Network Framework 3.2 Setting-Up Amazon EC2 Increasing the size of the storage device | 1. | Project Description | 8 |
|---|----|---|----|
| 1.2 Project Goal. 1.3 Neural Networks 2. Project Design 2.1 Design Overview 2.2 Basic Implementation using Matlab Setting Up Matlab Importing the Data Importing Data from a text or CSV file Starting the Neural Network Toolbox Selecting the Data Neural Network Architecture Neural Network Architecture Neural Network Training Network Evaluation Saving Results Generating Scripts Saving the Results into Workspace 2.3 Performance Improvement Strategies Balanced Datasets Balance Datasets Balanced Datasets | | 1.1 Introduction and Problem Statement | 8 |
| 1.3 Neural Networks 2. Project Design 2.1 Design Overview 2.2 Basic Implementation using Matlab Setting Up Matlab Importing the Data Importing the Data Starting the Neural Network Toolbox Selecting the Data Neural Network Architecture Neural Network Training Network Evaluation Saving Results Generating Scripts Saving the Results into Workspace Saving the Results into Workspace Balanced Datasets Balanced Datasets Balanced Datasets Balanced Datasets Balanced Datasets Batch Training Online Training with Preliminary Batch Training Multiple Experts Network 3.1 Selecting an Open Source Neural Network Framework 3.2 Setting-Up Amazon EC2 Increasing the size of the storage device | | 1.2 Project Goal | 9 |
| 2. Project Design 2 2.1 Design Overview 2 2.2 Basic Implementation using Matlab 2 Setting Up Matlab 2 Importing the Data 2 Importing Data from a text or CSV file 2 Starting the Neural Network Toolbox 2 Selecting the Data 2 Neural Network Architecture 2 Neural Network Training 2 Network Evaluation 2 Saving Results 2 Generating Scripts 2 Saving the Results into Workspace 2 2.3 Performance Improvement Strategies 2 Balanced Datasets 2 Balanced Datasets 2 Balanced Datasets 2 Mini Batch Training 2 Online Training with Preliminary Batch Training 3 Multiple Experts Network 3 3.1 Selecting an Open Source Neural Network Framework 3 3.2 Setting-Up Amazon EC2 2 Increasing the size of the storage device 4 | | 1.3 Neural Networks | 9 |
| 2.1 Design Overview 2 2.2 Basic Implementation using Matlab 2 Setting Up Matlab 2 Importing the Data 2 Importing Data from a text or CSV file 2 Starting the Neural Network Toolbox 2 Selecting the Data 2 Neural Network Architecture 2 Neural Network Training 2 Network Evaluation 2 Saving Results 2 Generating Scripts 2 Saving the Results into Workspace 2 2.3 Performance Improvement Strategies 2 Bottle-Neck Neural Networks 2 Balanced Datasets 2 Balanced Datasets 2 Mini Batch Training 2 Mini Batch Training with Preliminary Batch Training 3 Multiple Experts Network 3 3.1 Selecting an Open Source Neural Network Framework 3 3.2 Setting-Up Amazon EC2 2 Increasing the size of the storage device 4 | 2. | Project Design | 11 |
| 2.2 Basic Implementation using Matlab 1 Setting Up Matlab 1 Importing the Data 1 Importing Data from a text or CSV file 1 Starting the Neural Network Toolbox 1 Selecting the Data 1 Neural Network Architecture 1 Neural Network Training 1 Network Evaluation 1 Saving Results 1 Generating Scripts 1 Saving the Results into Workspace 1 2.3 Performance Improvement Strategies 1 Balanced Datasets 1 Balanced Datasets 1 Balanced Datasets 1 Mini Batch Training 1 Online Training with Preliminary Batch Training 1 Mini-Batch Training with Re-Sampling 1 Multiple Experts Network 1 3.1 Selecting an Open Source Neural Network Framework 1 3.2 Setting-Up Amazon EC2 1 Increasing the size of the storage device 1 | | 2.1 Design Overview | 11 |
| Setting Up Matlab Importing the Data Importing Data from a text or CSV file Importing Data from a text or CSV file Starting the Neural Network Toolbox Importing Data from a text or CSV file Selecting the Data Importing Data from a text or CSV file Neural Network Architecture Importing Data from a text or CSV file Neural Network Architecture Importing Data from a text or CSV file Neural Network Training Importing Data from a text or CSV file Neural Network Training Importing Data from a text or CSV file Neural Network Training Importing Data from a text or CSV file Neural Network Training Importing Data from a text or CSV file Saving Results Importing Data from a text or CSV file Saving Results Importing Data from a text or CSV file Saving Results Importing Experts Saving the Results into Workspace Importing Saving the Results into Workspace Saving the Results into Workspace Importing Saving text fraining Balanced Datasets Importing Saving Balanced Datasets Importing Saving Batch Training Importing Saving Mini Batch Training with Re-Sampling Importexperts Network Multi | | 2.2 Basic Implementation using Matlab | 12 |
| Importing the Data Importing Data from a text or CSV file. Starting the Neural Network Toolbox Importing the Data Selecting the Data Importing the Data Neural Network Architecture Importing the Data Neural Network Architecture Importing the Data Neural Network Training Importing the Data Neural Network Training Importing the Data Neural Network Training Importing the Data Network Evaluation Importing the Data Saving Results Importing the Data Generating Scripts Importent Strategies Saving the Results into Workspace Importent Strategies Bathelene Improvement Strategies Importent Strategies Balanced Datasets Importent Strategies Batch Training Importent Strategies Online Training Importent Strategies Mini Batch Training Importent Strategies Mini Batch Training Importent Strategies Mini Batch Training with Re-Sampling Importent Strategies Multiple Experts Network Importent Strategies 3.1 Selecting an Open Source Neural Network Framework Importent Strategies 3.2 Setting-Up Am | | Setting Up Matlab | 12 |
| Importing Data from a text or CSV file. Starting the Neural Network Toolbox. Selecting the Data Selecting the Data Neural Network Architecture. Neural Network Architecture. Neural Network Training. Selecting and Development Selecting Selecting Scripts. Saving Results. Selecting Selecting Scripts. Saving the Results into Workspace Selecting Sel | | Importing the Data | 13 |
| Starting the Neural Network Toolbox | | Importing Data from a text or CSV file | 16 |
| Selecting the Data 2 Neural Network Architecture. 2 Neural Network Training. 2 Network Evaluation. 2 Saving Results. 2 Generating Scripts. 2 Saving the Results into Workspace. 2 2.3 Performance Improvement Strategies 2 Bottle-Neck Neural Networks 2 Balanced Datasets. 2 Batch Training. 3 Online Training with Preliminary Batch Training. 3 Mini- Batch Training with Re-Sampling. 3 Multiple Experts Network 3 3.1 Selecting an Open Source Neural Network Framework 3 3.2 Setting-Up Amazon EC2 3 Increasing the size of the storage device. 4 | | Starting the Neural Network Toolbox | 17 |
| Neural Network Architecture | | Selecting the Data | 20 |
| Neural Network Training. 2 Network Evaluation. 2 Saving Results. 2 Generating Scripts. 2 Saving the Results into Workspace. 2 2.3 Performance Improvement Strategies 2 Bottle-Neck Neural Networks 2 Balanced Datasets. 2 Batch Training 2 Online Training. 3 Mini Batch Training with Preliminary Batch Training. 3 Mini- Batch Training with Re-Sampling. 3 Multiple Experts Network 3 3.1 Selecting an Open Source Neural Network Framework 3 3.2 Setting-Up Amazon EC2 3 Increasing the size of the storage device. 4 | | Neural Network Architecture | 22 |
| Network Evaluation. 2 Saving Results. 2 Generating Scripts. 2 Saving the Results into Workspace 2 2.3 Performance Improvement Strategies 2 Bottle-Neck Neural Networks 2 Balanced Datasets. 2 Batch Training 2 Online Training. 3 Mini Batch Training with Re-Sampling. 3 Multiple Experts Network 3 3.1 Selecting an Open Source Neural Network Framework 3 3.2 Setting-Up Amazon EC2 2 Increasing the size of the storage device. 4 | | Neural Network Training | 23 |
| Saving Results | | Network Evaluation | 25 |
| Generating Scripts. 2 Saving the Results into Workspace. 2 2.3 Performance Improvement Strategies 2 Bottle-Neck Neural Networks 2 Balanced Datasets. 2 Batch Training 2 Online Training. 3 Mini Batch Training with Preliminary Batch Training. 3 Multiple Experts Network 3 3.1 Selecting an Open Source Neural Network Framework 3 3.2 Setting-Up Amazon EC2 4 Increasing the size of the storage device. 4 | | Saving Results | 26 |
| Saving the Results into Workspace | | Generating Scripts | 27 |
| 2.3 Performance Improvement Strategies 2 Bottle-Neck Neural Networks 2 Balanced Datasets 2 Batch Training 2 Online Training 3 Online Training with Re-Sampling 3 Multiple Experts Network 3 3.1 Selecting an Open Source Neural Network Framework 3 3.2 Setting-Up Amazon EC2 4 Increasing the size of the storage device 4 | | Saving the Results into Workspace | 28 |
| Bottle-Neck Neural Networks 2 Balanced Datasets 2 Batch Training 2 Online Training 3 Mini Batch Training with Preliminary Batch Training 3 Mini- Batch Training with Re-Sampling 3 Multiple Experts Network 3 3. Project Implementation 3 3.1 Selecting an Open Source Neural Network Framework 3 3.2 Setting-Up Amazon EC2 3 Increasing the size of the storage device. 4 | | 2.3 Performance Improvement Strategies | 28 |
| Balanced Datasets 2 Batch Training 3 Online Training with Preliminary Batch Training 3 Mini- Batch Training with Re-Sampling 3 Multiple Experts Network 3 3.1 Selecting an Open Source Neural Network Framework 3 3.2 Setting-Up Amazon EC2 4 | | Bottle-Neck Neural Networks | 29 |
| Batch Training a Online Training a Mini Batch Training a Online Training with Preliminary Batch Training a Mini- Batch Training with Re-Sampling a Multiple Experts Network a 3. Project Implementation a 3.1 Selecting an Open Source Neural Network Framework a 3.2 Setting-Up Amazon EC2 a Increasing the size of the storage device. a | | Balanced Datasets | 29 |
| Online Training | | Batch Training | 30 |
| Mini Batch Training 3 Online Training with Re-Sampling 3 Multiple Experts Network 3 3.1 Selecting an Open Source Neural Network Framework 3 3.2 Setting-Up Amazon EC2 3 Increasing the size of the storage device. 4 | | Online Training | 30 |
| Online Training with Preliminary Batch Training Image: Constraint of the storage device. Mini- Batch Training with Re-Sampling. Image: Constraint of the storage device. Multiple Experts Network Image: Constraint of the storage device. Multiple Experts Network Image: Constraint of the storage device. Multiple Experts Network Image: Constraint of the storage device. | | Mini Batch Training | 31 |
| Mini- Batch Training with Re-Sampling. 3 Multiple Experts Network 3 3. Project Implementation 3 3.1 Selecting an Open Source Neural Network Framework 3 3.2 Setting-Up Amazon EC2 3 Increasing the size of the storage device. 4 | | Online Training with Preliminary Batch Training | 31 |
| Multiple Experts Network 3 3. Project Implementation 3 3.1 Selecting an Open Source Neural Network Framework 3 3.2 Setting-Up Amazon EC2 3 Increasing the size of the storage device. 4 | | Mini- Batch Training with Re-Sampling | 31 |
| 3. Project Implementation 3.1 Selecting an Open Source Neural Network Framework 3.2 Setting-Up Amazon EC2 Increasing the size of the storage device. | | Multiple Experts Network | 32 |
| 3.1 Selecting an Open Source Neural Network Framework 3.2 Setting-Up Amazon EC2 | 3. | Project Implementation | 32 |
| 3.2 Setting-Up Amazon EC2 | | 3.1 Selecting an Open Source Neural Network Framework | 32 |
| Increasing the size of the storage device | | 3.2 Setting-Up Amazon EC2 | 34 |
| | | Increasing the size of the storage device | 42 |

| 3.3 Setting-Up Fast Artificial Neural Networks on Amazon EC2 | 45 |
|---|----|
| Pre-requisites: | 45 |
| Installation | 45 |
| 3.4 Data Preparation | |
| 3.5 Training the Neural Network Using FANN | |
| 3.6 Training Experiments using FANN | |
| 4. Measuring Performance Using Time and Accuracy | 53 |
| 4.1 Prediction Using FANN | 53 |
| 4.2 Prediction and Performance Results | 54 |
| Mini batch training on the entire data set | 54 |
| Training on a balanced dataset | 54 |
| Training Using RMSProp | 55 |
| Training on Falsified Data | 56 |
| Varying the Number of Neurons for a Constant Dataset | 57 |
| Training Time for Networks with different Hidden Neurons | |
| Comparing Prediction Time of Networks with Different Hidden Neurons | |
| 4.3 Conclusion | 59 |
| 5.0 Future Work | |
| 6.0 References | 60 |
| 7.0 Appendix | 62 |
| 7.1 Script to Change Raw CSV Data to FANN Format | 62 |
| 7.2 Script to Generate Unlabeled Data from CSV file | 63 |
| 7.3 Script to Calculate Mean Squared Error | 64 |

Table of Figures

| Figure 1: Neural Network Training Phase | 12 |
|---|----|
| Figure 2: Predicting with a Neural Network | 12 |
| Figure 3: Matlab Start Screen | 13 |
| Figure 4: Setting the Current Folder | 14 |
| Figure 5: Selecting the Data | 14 |
| Figure 6: Import Data Wizard | 15 |
| Figure 7: Matlab workspace populated with variables | 16 |

| Figure 8: Data in a Matlab Worksheet | 17 |
|---|----|
| Figure 9: Neural Network Start Screen | 18 |
| Figure 10: Neural Network Pattern Recognition Tool | 19 |
| Figure 11: Select Data for Training | 20 |
| Figure 12: Dividing data into validation and test data | 21 |
| Figure 13: Neural Network Architecture and Hidden Neurons Setting | 22 |
| Figure 14: Training the Neural Network | 23 |
| Figure 15: Neural network being trained | 24 |
| Figure 16: Network Evaluation | 25 |
| Figure 17: Neural Network Results | 26 |
| Figure 18: Neural Network Diagram | 27 |
| Figure 19: Script generated for the Neural Network | 27 |
| Figure 20: Results in the workspace | 28 |
| Figure 21: Simplest method to train a Neural Network | 29 |
| Figure 22: Bottleneck Neural Network | 29 |
| Figure 23: Creating Balanced Datasets | 30 |
| Figure 24: Mini Batch Training | 31 |
| Figure 25: Amazon Web Services Console | 35 |
| Figure 26: Amazon EC2 Dashboard | 35 |
| Figure 27: Elastic IP page | 36 |
| Figure 28: Selecting an Elastic IP for EC2 | 37 |
| Figure 29: Create New Instance | 37 |
| Figure 30: Choose an AMI | 38 |
| Figure 31: Instance Details | 38 |
| Figure 32: Advanced Instance Options | 39 |
| Figure 33: Storage Device Configuration | 39 |
| Figure 34: Naming the device | 40 |
| Figure 35: Creating a new Key Pair | 40 |
| Figure 36: Selecting Security Group | 41 |
| Figure 37: Instance Summary for Review | 41 |
| Figure 38: Associating Elastic IP to Instance | 42 |
| Figure 39: Creating a snapshot | 43 |
| Figure 40: Creating a Snapshot | 44 |
| Figure 41: Attach a Volume | 44 |
| Figure 42: Batch Training | 54 |
| Figure 43: Training on Balanced Data Sets | 55 |
| Figure 44: Training Using RProp on MiniBatch: MSE vs Epochs | 56 |
| Figure 45: Training on Data with High Noise: MSE vs. Epochs | 57 |
| Figure 46: Varying Hidden Neurons: MSE vs. Epochs | 58 |
| Figure 47: Training Time | 58 |
| Figure 48: Prediction Time | 59 |

1. Project Description

1.1 Introduction and Problem Statement

The project derives itself from the IBM Great Minds Challenge. The project involved using machine learning algorithms to assign TRUE/FALSE labels to question/answer pairs that had been broken down into feature vectors, or series of numbers that represent the data of the question/answer pairs.

IBM Watson analyzes the feature vectors of a potential answer and assigns a TRUE if it believes the answer is correct, and a FALSE if it believes the answer is incorrect. Similarly, the project focused on the creation of a machine learning algorithm that can assign these TRUE/FALSE labels to a series of question/answer feature vectors.

Out of the many machine learning tools, I chose to use neural networks. Neural networks are used in machine learning for the prediction of data. In order to make predictions, neural networks need to be trained and retrained in order to increase their prediction efficiency.

The data used to train neural networks is generally represented as feature vectors, the feature vector comprises of features that are values representing a certain feature/property of the data. These features cumulatively determine the final value for the data. The neural network is thus trained using these feature vectors and fine-tuned so that neural network output matches the correct result. When the neural network reaches a stage where its prediction rate is within a certain desired error percentage that is reasonably low, then the neural network is said to have been trained.

In case of big data, the dimension of data is a factor that limits the efficiency of neural networks. The data can have a large number of features; for example IBM Watson data has 342 features.

These features can be visualized as columns when comparing with a row and column based data format. Secondly, there might be a large volume of data, that is, a large number of rows of data. In both these cases, proper training of neural networks becomes a challenging process. My project aims at finding out and comparing different methods to train neural networks using big data in order to make a successful prediction. The project will involve analyzing the data in order to determine how it can be used to train the neural network for making successful predictions using big data and over the cloud.

1.2 Project Goal

The primary goal of the project will be to increase prediction rates and secondly to speed up processing/turnaround times. The process will involve chopping up data both based on rows and columns, controlling the amount of data used for training, splitting feature vectors or using feature vector reduction techniques, and then training the neural network and finally integrating and validating the results achieved.

The project also aims at finding out an open source framework that can successfully be run on the cloud in order to accommodate the large amount of data volume and processing required for training neural networks.

1.3 Neural Networks

Artificial neural networks are modeled on neural networks present in the brain. Biological neural networks are an interconnection of neurons. Similarly, artificial neural networks are an interconnection of nodes or neurons.

A neural network consists of 3 main components, the input layer, the output layer and the hidden layer. The hidden layer consists of nodes that are interconnected. The connections of these nodes have weights assigned to them, which in turn determine the output. Different algorithms are used to adjust these weights during the training process so that the desired output is achieved.



Figure: Neural Network Architecture

The data used to train neural networks is divided into the inputs and the target output.

- At the start the neural network is assigned random weights for its connections.
- The output achieved by using the given input is compared to the target output.
- The weights are adjusted to reduce the difference between the target and output to the minimum.

- This process is repeated until a low enough difference is achieved.
- This is a stopping condition known as the desired error
- Another stopping condition is the maximum number of training epochs.

Training and learning functions are algorithms that are used to automatically adjust the neural network's weights and biases. The goal of the training function is to reduce the error to a minimum, thus we want to find a minimum value of error. If we move in the direction of the gradient we will find the local maxima. This is known as gradient ascent. Thus if we move in the opposite direction we should be able to find the local minima. This technique is known as the gradient descent. Certain neural network functions can be given a parameter that can decide the rate at which the gradient descent is done. This parameter is known as the Learning Rate. In algorithms that change the weights and biases of neurons iteratively, the learning rate decides the step size that needs to be taken during descent. Thus having a very low learning rate we will have better accuracy for the network, but we will take a lot of time to reach to the minima due to the small step size. On the other hand, when the leaning rate is large, we might get close to the minima very fast but may just oscillate around the minima and never reach it. Thus the choice is a learning rate is an important parameter for neural network training.

2. Project Design

2.1 Design Overview

Neural networks are well equipped to handle classification and prediction problems. In our case, the data has two classes, true and false. In order to be able to classify the data a neural network needs to be trained to recognize the pattern and identify a class of data. After initial training, we

verify the neural networks classification by testing it on the data that it has been trained on already, this is known as the testing phase or the validation phase. If the network does not meet the required validation standards, then further training of the network is required. The accuracy of the network is predicted by measuring the Mean Squared Error (MSE).



Figure 2: Predicting with a Neural Network

Once the network has started making predictions below the desired error rate, the training is stopped and the network is ready to make actual predictions on real data.

2.2 Basic Implementation using Matlab

Setting up Matlab

Matlab provides a useful interface for creating and training neural networks. We require both Matlab and the neural network toolbox for the installation of Matlab. 1. After the setup, start Matlab to obtain a blank workspace and Matlab console.



Figure 3: Matlab Start Screen

2. The console shows up with an EDU prompt for the student version.

Importing the Data

- 1. Set the current folder to the desired folder.
- 2. The contents of the current folder will be displayed in the current folder window.



Figure 4: Setting the Current Folder

- 3. On the menu, go to File -> Import Data
- 4. Browse for the data that you want to import into the Matlab workspace.

| A <student version=""> MATLAB R2012a</student> | | - | - | Collegest of | to Manual Real To | | | _ | |
|--|--------|----------------------|----------------|----------------------------|--------------------|--------------------|--------------|--------------------------|---------|
| File Edit View Debug Desktop Windo | w Help | | | | | | | | |
| : 🛅 🖆 👗 🛍 🖄 🕫 🚺 🗐 🗐 | 🕜 Cun | rent Folder: E:\CS29 | 7 | ✓ | | | | | |
| Shortcuts C How to Add C What's New | | | | 0 | | | | | |
| Current Folder | x x | lorksnace | | | | | | | ±⊓ ₹ X |
| | a. 1 | | | | | | | | |
| | | 📣 Import Data | | | | | | _ | |
| Name A | | Look in: | CS297 | | - - | | | | |
| - data | - | | 0.00207 | | | | | | |
| E Chinmay | | | Name | | Date modified | Туре | Size | | |
| DhruvData | | Recent Places | 闄 data | | 12/20/2012 1:10 PM | File folder | | | |
| 🗉 퉬 lines20k | | | 📗 Matlab | | 12/4/2012 1:32 PM | File folder | | | |
| 🗉 퉲 IrgData | Ξ. | | scripts | | 12/15/2012 7:40 PM | File folder | | | |
| 🕀 🎍 mat20k | | Desktop | TGMC Wat | tson Technical Info Packet | 2/6/2013 11:53 PM | File folder | | | |
| 3gbsinglematrix.mat | | <u> </u> | unlabelled | | 12/18/2012 2:41 AM | File folder | | | |
| Animesh submission? | | | Basic_data | .xis | 11/18/2012 5:30 PM | Microsoft Excel 9/ | 922 KB | | |
| Animesh submission2.csv | | Libranes | labelled.bd | | 11/1/2012 12:3/ PM | Text Document | 3,156,772 KB | | |
| data_to_predict.mat | | | | | | | | | |
| 🖺 Fin1.m | | Computer | | | | | | | + D 7 X |
| 🔡 FinInp1.mat | | | | | | | | | |
| FinInp2.mat | | | | | | | | | |
| FinInp3.mat | | Network | | | | | | | |
| Finlinp4.mat | | | | | | | | | |
| Fininp3.mat | | | | | | | | | |
| FinInp7.mat | 1 | | | | | | | | |
| FinInp8.mat | | | | | | | | | |
| Fintar1.mat | | | | | | | | | |
| 🔡 Fintar2.mat | | | | | | | | | |
| Fintar3.mat | | | | | | | | | |
| Fintar4.mat | | | File name: | Fininp1.mat | | | 1 | Open | |
| Fintar5.mat | | | Files of type: | Recognized Data Files | | | | Cancel | |
| Fintar7.mat | | | | 1 | | | | | 2 |
| Fintar8.mat | | _ | _ | | | | _ | | |
| first50.txt | Ψ. | | | | | | | | |
| data (File Folder) | ^ | | | | | | | | |
| Start Ready | | | | | | | | | |
| , | | | | | | | | | |

Figure 5: Selecting the Data

4. Use the Import wizard to select data and create variables in the workspace.

| 📣 Ir | mport Wiz | ard | | | | | | | | | | |
|------|---|----------------|---------------------|----------|--------|--|--|--|--|--|--|--|
| Sele | ect variable | es to import u | sing checkbox | 25 | | | | | | | | |
| ۲ | Create var | iables matchiı | ng <u>p</u> review. | | | | | | | | | |
| 0 | Create vectors from each column using column names. | | | | | | | | | | | |
| 0 | C <u>r</u> eate veo | ctors from eac | h row using ro | w names. | | | | | | | | |
| Vari | ables in E: | \CS297\data\F | FinInp1.mat | | | | | | | | | |
| Imp | port | Name 🔺 | Size | Bytes | Class | No variable selected for preview. | | | | | | |
| | V | 🕂 FinInp1 | 50000x341 | 68200000 | single | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | III | | • | | | | | | | |
| H | lelp | | | < Back | Ne | ext > Finish Generate MATLAB code Cancel | | | | | | |

Figure 6: Import Data Wizard

5. Similarly, import all the required data into the Matlab workspace as variables.

6. The workspace should now be filled with the required variables that can be used within

Matlab.

| Morkspace | | | | |
|---|--|--------------------|--------------------|-----|
| <u>File E</u> dit <u>V</u> iew <u>G</u> | raphics De <u>b</u> ug <u>D</u> esk | top <u>W</u> ii | ndow <u>H</u> elp | Ľ |
| 1 🖻 🖬 🐿 🖬 🖉 | stac <u>k</u> : Base → 🐼 Sel | ect data to | o plot | - |
| Name 🔺 | Value | Min | Max | |
| FinInp1 | <50000x341 single> <50000x1 single> | <too 0</too | <too 1</too | |
| | | | | .:: |

Figure 7: Matlab workspace populated with variables

Importing Data from a text or CSV file

1. Select the desired CSV or text file using the File -> Import Data command.

2. This will open the CSV file as a worksheet within Matlab.

3. The worksheet can now be used to select the required data from the worksheet and then import

it into Matlab as a variable.

| 🛃 Imp | 🖳 Import - E:\CS297\data\50mb\partba.csv | | | | | | | | | | |
|--------------|---|----------|---|----|---------|--------|--------|---|---------|-----|---|
| <u>F</u> ile | Edit Debug Desktop Window Help | | | | | | | | | | |
| | | | | | | | | | | | |
| Import | Import as: Matrix A1:ME20431 Range: A1:ME20431 Replace Non-numeric with 0.0 + - Matrix Mezo431 | | | | | | | | | | |
| IMP | PORTED DATA | SELECTI | C | D | E | | G | Ц | T | 1 | |
| 1 | 0 3420 | 131x343) | 0 | 0 | L 0 | · 0 | 0 | 0 | -0 3282 | , 0 | |
| 2 | 37586 | 0 | 0 | 19 | -9.0330 | 0 | 0.6632 | 0 | 0.5202 | 1 | |
| 3 | 37586 | 0 | 0 | 0 | 0 | 0 | 0.6632 | 0 | 0 | 1 | |
| 4 | 37586 | 0 | 0 | 0 | 0 | 0 | 0.6794 | 0 | 0 | 0 | |
| 5 | 37586 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| 6 | 37586 | 0 | 0 | 22 | -9.0653 | 0 | 0 | 0 | 0 | 1 | |
| 7 | 37586 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | |
| 8 | 37586 | 0 | 0 | 44 | -9.1676 | 0 | 0 | 0 | 0 | 0 | |
| 9 | 37586 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| 10 | 37586 | 0 | 0 | 0 | 0 | 0 | 0.6632 | 0 | 0 | 0 | |
| 11 | 37586 | n | n | 45 | -9 1683 | n | 0.6632 | n | n | 0 | |
| pi | artba | | | | | | | | | | : |

Figure 8: Data in a Matlab Worksheet

4. By default, the entire data is selected for import.

5. On clicking the import button will import the data specified in the range to get imported as a variable into the Matlab workspace.

6. The range can be altered to select only specific data into the workspace.

7. The un-importable cells section can be used to specify rules on how to handle cells that are not in the correct format and cannot be imported into Matlab.

Starting the Neural Network Toolbox

1. Use the command "nnstart" in order to start the neural network interface.

| ♦ <student version=""> MATLAB R2012a</student> | and the second s | |
|--|--|----------|
| File Edit Debug Desktop Window Help | | |
| : 🛅 🖆 🌡 🐂 🖏 🤊 (* 🎝 🛃 🖹 🕘 (| Current Folder: C:\Users\Animesh\Documents\MATLAB 🔻 📟 🖻 | |
| Shortcuts 🗷 How to Add 🗷 What's New | | |
| Current Folder 🖛 🖬 🛪 🗙 | Workspace | × 5 ⊡ 1+ |
| 📙 « Documents 🕨 MATLAB 🛛 🔻 🔎 🖻 @+ | 🗃 📹 ங 🔚 Stack: Base 👻 💯 Select data to plot 🔹 | |
| 🗋 Name 🔺 | Name 🔺 Neural Network Start (nnstart) | |
| | Student Ve Student Ve Stude time Each of these wizards helps you solve a different kind of problem. The last panel of each wizard generates a MATLAB script for solving the same or similar problems. Example datasets are provided if you do not have data of your own. EDU>> Iput-output and curve fitting. Pattern recognition and classification. Qustering. Qustering. Qustering. Qustering Tool Unpurious Time series. | X 5 🗆 I+ |
| | | 012 |
| 4 Start | | OVR: |

Figure 9: Neural Network Start Screen

- 2. This brings up the neural network start Interface.
- 3. At this stage we can chose between the different kinds of neural network tools available to us.
- 4. Choose the Patten recognition and classification tool.



5. The Pattern Recognition screen provides an overview of the feed forward neural network that will be used for pattern recognition. Click "next" on this screen.

6. The next screen involves selecting data for neural network training. Inputs and target data needs to be selected.

7. This data can be selected from variables present in the Matlab workspace.

8. We must keep in mind that the number of samples of data must be the same for both the inputs and targets.

Selecting the Data

| A Neural Network Pattern Recognition Tool (nprtool) | |
|--|--|
| Select Data What inputs and targets define your pattern recognition problem? | |
| Get Data from Workspace | Summary |
| Input data to present to the network. | Inputs 'simpleclassInputs' is a 2x1000 matrix, representing static data: 1000 |
| P Inputs: simpleclassInputs • | samples of 2 elements. |
| Target data defining desire network output. Image: Comparison of the second s | Targets 'simpleclassTargets' is a 4x1000 matrix, representing static data: 1000 samples of 4 elements. |
| Samples are: Image: Image: Image: | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| Want to try out this tool with an example data set? | |
| Load Example Data Set | |
| | |
| To continue, click [Next]. | |
| Neural Network Start Welcome | 🗢 Back 🔍 Next 🔇 Cancel |

Figure 11: Select Data for Training

9. Next we divide the data into Training, Validation and Testing data.

10. The interface lets us specify the percentage of data we wish to divide.

| 📣 Neural Network Pattern Recognition Tool (nprtool) | | 3 |
|--|---|---|
| Validation and Test Data Set aside some samples for validation and testing. | | |
| Select Percentages Randomly divide up the 1000 samples: Training: 70% Validation: 15% • Testing: 15% • | 700 samples Explanation 700 samples Three Kinds of Samples: 150 samples Training: 150 samples These are presented to the network during training, and the network is adjusted according to its error. 150 samples Validation: These are used to measure network generalization, and to halt training when generalization stops improving. Image: These have no effect on training and so provide an independent measure of network performance during and after training. | |
| Restore Defaults | | |
| Change percentages if desired, then click [Next] to co | continue. | |
| Reural Network Start 🛛 👫 Welcome | 🗢 Back 🔍 Next 🥸 Cancel | |

Figure 12: Dividing data into validation and test data

11. The next screen lets us specify the number of hidden neurons in the hidden layer. The default

number is 10.

12. This screen also displays the architecture of the neural network.

Neural Network Architecture

| 📣 Neural Network Pattern Recognition Tool (nprtool) | | | | | | | |
|--|---|--|--|--|--|--|--|
| Network Architecture Set the dimensions of the self-organizing map's output layer. | | | | | | | |
| Hidden Layer | Recommendation | | | | | | |
| Define a pattern recognition neural network. (patternnet) | Return to this panel and change the number of neurons if the network does not perform well after training. | | | | | | |
| Restore Defaults | | | | | | | |
| Neural Network Hidden Layer Unput 2 10 4 | | | | | | | |
| Change settings if desired, then click [Next] to continue. | | | | | | | |
| Reural Network Start 🕅 Welcome | Sack Next Cancel | | | | | | |

Figure 13: Neural Network Architecture and Hidden Neurons Setting

13. The next screen is where we start training the neural network.

14. This screen also shows the Mean Squared error and the Percentage error.

15. The network can be retrained until we get a desired and low value for both these parameters.

Neural Network Training

| Neural Network Pattern Recognition Tool (nprtool) Train Network Train the network to classify the inputs according to the targets. | | | | |
|---|---|---|--|-----------|
| Train Network Train using scaled conjugate gradient backpropagation. (trainscg) | Results Training: | Samples 700 150 | ⊠ MSE - - | % %E - |
| Training automatically stops when generalization stops improving, as indicated by an increase in the mean square error of the validation samples. | Vesting: | 150 Plot Confusion | - Plot ROC | - |
| Training multiple times will generate different results due to different initial conditions and sampling. | Mean Squared E between outputs means no error. Percent Error in misclassified. Av 100 indicates ma | rror is the average sq s and targets. Lower dicates the fraction o value of 0 means no iximum misclassifica | uared difference values are better. Zo f samples which ar misclassifications, tions. | ero |
| Train network, then click [Next]. Neural Network Start KN Welcome | | 🗢 Baci | k Next | Cancel |

Figure 14: Training the Neural Network

16. The interface showing the neural network training appears when we click on the train button.

17. The interface provides the different algorithms that being used to train the neural network

and measure its performance.

| Neural Network Training (n | ntraintool) | , 🗉 🗾 | | | | | |
|---|---|----------|--|--|--|--|--|
| Neural Network | | | | | | | |
| Hidden | Output | | | | | | |
| Dutput 2 10 4 Output 4 | | | | | | | |
| Algorithms | | | | | | | |
| Data Division: Random (d Training: Scaled Conju Performance: Mean Square Derivative: Default (de Progress | lividerand) ugate Gradient (trainscg) ed Error (mse) faultderiv) | | | | | | |
| Enoch 0 | 14 iterations | 1 1000 | | | | | |
| Time: | 0.00.01 | 1 1000 | | | | | |
| Performance: 0.490 | 4.44e-07 | | | | | | |
| Gradient: 0.271 | 9.73e-07 | 1.00e-06 | | | | | |
| Validation Checks: 0 | 0 | 6 | | | | | |
| Plots | | | | | | | |
| Performance | (plotperform) | | | | | | |
| Training State | (plottrainstate) | | | | | | |
| Error Histogram | (ploterrhist) | | | | | | |
| Confusion | (plotconfusion) | | | | | | |
| Receiver Operating Chara | acteristic (plotroc) | | | | | | |
| Plot Interval: | 1 epochs | i | | | | | |
| ✓ Minimum gradient re | ached. | | | | | | |
| | Stop Training | Cancel | | | | | |

Figure 15: Neural network being trained

18. The data division algorithm specifies how the data was divided into the training, validation and testing sets. We use the default function that divides the samples randomly.

19. The training function specifies the function being used for training the neural network.

20. The performance is measured using the mean squared error, the lower the error, the better the performance.

21. An epoch of training is defined as a single presentation of all input vectors to the network.

The network is then updated according to the results of all those presentations.

Network Evaluation

| 📣 Neural Network Pattern Recognition Tool (nprtool) | |
|--|---|
| Neural Network Pattern Recognition Tool (nprtool) Evaluate Network Optionally test network on more data, then decide if network performance Iterate for improved performance Try training again if a first try did not generate good results or you require marginal improvement. Train Again Increase network size if retraining did not help. Matiust Network Size | rmance is good enough. Optionally perform additonal tests I Inputs: Targets: Samples are: No inputs selected. |
| Adjust Network Size | No targets selected. |
| Not working? You may need to use a larger data set. | Test Network MSE %E Plot Confusion Plot ROC |
| Select inputs and targets, click an improvement button, or click [Ne | ex]. |
| Reural Network Start 144 Welcome | 🗢 Back 🔍 Next 🔇 Cancel |



22. The next screen allows us to evaluate and retune the network if we are not satisfied with the performance of the network.

23. At this stage we have the option to change the number of hidden neurons.

24. We can also perform additional tests on the trained neural network by providing additional data for testing.

Saving Results

| 📣 Neural Network Pattern Recognition Tool (nprtool) | |
|---|-----------------------|
| Save Results Generate MATLAB scripts, save results and generate diagrams. | |
| Generate Scripts Recommended >> Generate scripts to reproduce results and solve similar problems: | Advanced Script |
| Save Data to Workspace | |
| Save network to MATLAB network object named: | net |
| Save performance and data set information to MATLAB struct named: | info |
| Save outputs to MATLAB matrix named: | output |
| Save errors to MATLAB matrix named: | error |
| Save inputs to MATLAB matrix named: | input |
| O Save targets to MATLAB matrix named: | target |
| Save ALL selected values above to MATLAB struct named: | results |
| Restore Defaults | 🔌 Save Results |
| Deploy the Network | |
| Generate a neural or Simulink diagram of the network: | link Diagram (gensim) |
| | |
| Save results and click [Finish]. | Next Finish |

Figure 17: Neural Network Results

25. We can now save the results of the neural network training to our workspace as variables.

26. We can view the architecture of the neural network by clicking on the Neural Network

diagram button



Figure 18: Neural Network Diagram

Generating Scripts

26. We can also generate a script that can be saved for later use.



Figure 19: Script generated for the Neural Network

Saving the Results into Workspace

27. Saving the results saves the results in the workspace.

| Morkspace | | - | - | | | | | |
|---|-----------------|---------|--------|--|--|--|--|--|
| <u>F</u> ile <u>E</u> dit <u>V</u> iew <u>G</u> raphics De <u>b</u> ug <u>D</u> esktop <u>W</u> indow <u>H</u> elp v | | | | | | | | |
| 🛅 📷 ங 💺 Stack: Base 🗸 🕼 Select data to plot 🔹 | | | | | | | | |
| Name 🔺 | Value | Min | Max | | | | | |
| error error | <4x1000 double> | -0.0331 | 0.0076 | | | | | |
| 📃 info | <1x1 struct> | | | | | | | |
| 😰 net | <1x1 network> | | | | | | | |
| 🖶 output | <4x1000 double> | 0 | 1.0000 | | | | | |
| 🛨 simpleclassInputs | <2x1000 double> | -0.3927 | 1.3993 | | | | | |
| 🛨 simpleclassTargets | <4x1000 double> | 0 | 1 | | | | | |
| 🛨 tout | <11x1 double> | 0 | 10 | | | | | |
| | | | | | | | | |
| | | | | | | | | |

Figure 20: Results in the workspace

28. Net is the network structure of the trained neural network that can be used later.

29. Output is the output that has been predicted by the trained neural network for the given input data.

2.3 Performance Improvement Strategies

The simplest way to train a neural network is to provide the complete data-set all together and train the neural network with the different training algorithms available. This is the standard training technique and is able to provide very accurate results for most cases. In the case of large amounts of data, the time taken to train a network and reach sufficient error can be very high. This is where we can optimize training in order to reduce the training time and try to retain the accuracy of the neural network.



Figure 21: Simplest method to train a Neural Network

Bottle-Neck Neural Networks

A Neural network which has an extra hidden layer with a very small number or neurons as compared to the input neurons is a bottleneck neural network. In [1] the author presents a bottleneck network that has a bottleneck that is equal to the range of the output values. The bottle neck network will thus have 2 hidden layers. The first hidden layer immediately present after the input layer will be the bottleneck layer, followed by a second hidden layer that connects to the output layer.



Balanced Datasets

In case of our data, the data contains a large number of false data samples and a relatively low number true data samples. Thus the network is hardly exposed to any true data samples during the entire training process. In order to make the training more effective we can create a new dataset that represents both the true and false cases equally. In general, [1] talks about creating a new dataset that has equal representation of all the separate classes. This technique also reduces the size of the dataset that is needed to train the neural network sufficiently.



Figure 23: Creating Balanced Datasets

Batch Training

In case of batch training the weights are updated after an entire pass of the training dataset. This produces better adjustment for weights at each epoch but each epoch takes a much longer time as the entire training data set needs to be read.

Online Training

Online training is a training technique that is suitable for large data-sets with a large number of variables. The weights are updated immediately after each training record is read. This technique

can reach the desired error rate much faster than the Batch Training method. However, the changes to the weights during the online training methods are small at each step.

Mini Batch Training

Mini Batch training is the same as batch training except that the data has now been divided into small modules during the data preparation stage. Thus the system updates its weight much more frequently, providing a middle path between the online training and Batch Training Methods.



Figure 24: Mini Batch Training

Online Training with Preliminary Batch Training

An alternative is to train the neural network with a significant portion of the dataset using the batch training technique. Then switch to adaptive training for the rest of the dataset. This method also provides a combination of the advantages of both batch training and online training. The batch training will allow the neural network to converge in the correct direction and then online training can fine-tune quickly to the desired error rate.

Mini- Batch Training with Re-Sampling

Instead of training the entire dataset at once, mini batch training divides the data into smaller subsets. Instead of creating mini-batches by simply chopping down the data into smaller modules, we can decide on the module size and then populate that batch by randomly selecting the records from the entire data-set. This method is able to create a smaller statistical representation of the entire dataset and thus requires much less training time.

Multiple Experts Network

The multiple experts' network is, as the name suggests, a network consisting of multiple trained systems. The multiple experts' network combines different networks trained via different techniques to make predictions on a dataset. A gating network is used to combine the results obtained from each of the expert networks into a final result for the prediction.

3. Project Implementation

3.1 Selecting an Open Source Neural Network Framework

During CS297 I used the Matlab neural network toolbox to implement a neural network and predict the results for the output file. Matlab is commercial licensed software and the costs of obtaining a license for running Matlab over a cloud are very high.

I explored the following options for neural network packages that were completely open source and could be installed over the cloud without licensing issues.

1. GNU Octave Neural Network Package:

GNU Octave is an open source alternative to Matlab, its aim is to provide similar functionality to that of Matlab but remain as an open source project. It has many well written packages that

replicate Matlab functionality. However, the neural network package for Octave is very basic and only provides us with a single training function. The project is no longer actively supported and has compatibility issues with new versions of Octave.

2. Encog

Encog is an open source neural network framework. It has extensive documentation and examples available. It is primarily written in Java and maintained and developed by Heaton Research. The functionality provided by this framework is extensive. However, it is tied to commercial support and books as well. Thus, in order to get support for the project you need to purchase books that have been written for the framework.

3. Neuroph Studio

Neuroph is a Java library for Neural Networks. Neural networks can be implemented using simple Java programs. Neuroph seems more promising as it only requires basic Java knowledge and with a basic understanding of neural networks, the code is self-explanatory. The framework was primarily developed as a GUI for neural networks.

4. Fast Artificial Neural Network (FANN)

This is a neural network framework written in "C". It provides all the necessary functionality required for neural networks. It is mainly developed to run as a console application like a C program. It is completely open source.

Both Neuroph and Fast Artificial Neural Network (FANN) were good candidates for my project. I selected FANN since it is a C Framework and it promised faster runtime than Neuroph that is written in Java. Secondly, FANN libraries are simple C programs that can be modified as per my needs. This ability to customize was a major deciding factor for my selection.

3.2 Setting-Up Amazon EC2

The Amazon Cloud Compute, popularly known as EC2, is a remote machine that is easily scalable and provides computational power that can be scaled as per the needs of the program. Since we needed a large amount of main memory and computational power beyond the scope of personal laptops or home desktops, I decided to use amazon's ec2 for my computation. An EC2 instance is similar to a remote server that you can access remotely. Specifically for my experiments I used a machine that had 15gigabytes of main memory and 8 processing cores.

The procedure to setup an Amazon EC2 instance is as follows:

- 1. Sign up for the Amazon web services account.
- 2. Once the account is setup, navigate to the Web Services Console.





3. Select the EC2 option from the Dashboard. This will lead you to the EC2 dashboard.

| EC2 Dashboard | Resources | | | C | Account Attributes |
|--------------------|--|--|------------------------|---|---|
| Tags | You are using the following Amazon EC2 resou | You are using the following Amazon EC2 resources in the US East (N. Virginia) region: | | | |
| lago | 0 Running Instances | | 1 Elastic IP | | EC2-Classic |
| INSTANCES | 2 Volumes | | 1 Snapshot | | LG2-VFC |
| Instances | 1 Key Pair | | 0 Load Balancers | | Additional Information |
| Spot Requests | 0 Placement Groups | | 2 Security Groups | | |
| Reserved Instances | | | | | Getting Started Guide |
| | Create Instance | | | | All EC2 Resources |
| MAGES | To start using Amazon EC2 you will want to laur | | Forums | | |
| AMIs | To start using Amazon Eoz you will want to lad | to start using Amazon EC2 you will want to launch a virtual server, known as an Amazon EC2 instance. | | | |
| Bundle Tasks | Launch Instance | | | | Report an Issue |
| | Note: Your instances will launch in the LIS East (N. Vir | ninia) region | | | |
| ASTIC PLOCK STOPE | Note, four instances win faunch in the OS East (N. Virg | jinia) region | | | Popular AMIs on AWS |
| Volumes | Service Health | C | Scheduled Events | C | Marketplace |
| Snapshots | Service Statue | | US Each (N. Virginia); | | Debian GNU/Linux |
| · · | | | No events | | Provided by Debian |
| NETWORK & SECURITY | This service is operating normally | | | | Rating **** |
| Security Groups | Availability Zone Status: | | | | Free Software, pay only for AWS usage View all Operating Systems |
| Elastic IPs | us-east-1h; | | | | the an operanity operand |
| | do-cast-rb. | | | | MongoDB |

Figure 26: Amazon EC2 Dashboard

4. From the menu on the left select Elastic IP's under the Network and Security Tab. Elastic IP's can be associated with an account and then associated with the instance when the instance is

started. We need to associate an elastic IP as it takes some time to register the first time we set it up with our account.

| EC2 Dashboard | Alloc | ate New Address | | | | | ୯ 💠 🙆 |
|--------------------|--------|----------------------|-------------|--------|-----------|------------|--------------------------|
| Events Tags | Viewin | g: All Addresses | • (Se | earch | \supset | | Ҝ 1 to 1 of 1 Items 🔉 |
| INSTANCES | | Address | Instance ID | ENI ID | Scope | Public DNS | |
| Instances | | 54.225.192.162 | | | standard | | |
| Spot Requests | | | | | | | |
| Reserved instances | | | | | | | |
| IMAGES | | | | | | | |
| AMIs | | | | | | | |
| Bundle Tasks | | | | | | | |
| | | | | | | | |
| Volumes | | | | | | | |
| Snapshots | 0 Add | resses selected | | | | | |
| _ | Se | elect an address abo | ve | | | | |
| NETWORK & SECURI | | | | | | | |
| Elastic IPs | | | | | | | |
| Placement Groups | | | | | | | |
| | | | | | | | |
| | | | | | | | |

Figure 27: Elastic IP page

5. Click on the Allocate new Address button. It will prompt you for the type of Elastic IP (EIP) that you wish to allocate. Select EC2 from the drop down provided.



Figure 28: Selecting an Elastic IP for EC2

6. Navigate back to the EC2 dashboard, and under the create instance header, select launch new

instance.



Figure 29: Create New Instance

7. Select the classic wizard from the given options.



Figure 30: Choose an AMI

8. We select the Ubuntu server 12.04 LTS as our AMI (Amazon Machine Image)

| 🎁 Services 🗸 Edit 🗸 | | imesh 👻 N. Virginia 👻 Help 👻 |
|-------------------------------------|---|------------------------------|
| | Request Instances Wizard | Cancel 🔀 |
| EC2 Dashboard | CHOOSE AN AMI INSTANCE DETAILS CREATE KEY PAIR CONFIGURE FIREWALL REVIEW | |
| Tags | Provide the details for your instance(s). You may also decide whether you want to launch your instances as "on-demanc "spot" instances. | l" or |
| | Number of Instances: 1 Instance Type: M1 Extra Large (m1.xlarge, 15 GiB) | - |
| Instances | Launch as an EBS-Optimized instance (additional charges apply): | |
| Spot Requests Reserved Instances | Launch Instances | |
| | EC2 Instances let you pay for compute capacity by the hour with no long term commitments. This transforms whe commonly large fixed costs into much smaller variable costs. | at are |
| AMIs | Launch into: | |
| Bundle Tasks | Availability Zone: No Preference 👻 | |
| ELASTIC BLOCK STO | | |
| Volumes | © Request Spot Instances | |
| Snapshots | | |
| NETWORK & SECURI | | |
| Elastic IPs | | |
| Placement Groups | | |
| | | |
| | < Back | |
| © 2008 - 2013, Amazon Web Servio | | Feedback |

Figure 31: Instance Details

9. On this screen we change the Instance type to M1 Extra Large as this is the size of the instance

that we want to use.

| 🎁 Services 🕶 Edit 🗸 | Request Inst | ances Wizard | | | Cancel 🗙 | N. Virginia 🕶 | Help 🕶 |
|-------------------------------------|----------------------------------|---|--|--|----------|---------------|--------|
| EC2 Dashboard Events | CHOOSE AN AMI | INSTANCE DETAILS CREATE KEY PAIR CONFIC | URE FIREWALL REV | EW | | | |
| Tags | Number of Ins | tances: 1 A | vailability Zone: No | Preference | _ | | |
| INSTANCES | Advanced I | istance Options | | | | | |
| Instances | Here you can o Detailed Monit | hoose a specific kernel or RAM disk to use with yo oring or enter data that will be available from you | ur instances. You can a r instances once they l | also choose to enable CloudWatch aunch. | | | |
| Spot Requests Reserved Instances | Kernel ID: | Use Default 👻 | RAM Disk ID: | Use Default 👻 | _ | | |
| E _ | Monitoring: | Enable CloudWatch detailed monitoring for this in: (additional charges will apply) | stance | | | | |
| AMIS | User Data: | | | | | | |
| Bundle Tasks | as text | | | | | | |
| ELASTIC BLOCK STO | 🔘 as file | (Use shift+enter to insert a newline) base64 encoded | | | | | |
| Snapshots | Termination Protection: | Prevention against accidental termination. | Shutdown Behavior: | Stop 👻 | | | |
| | IAM Role: 🎯 | None 👻 | | | _ | | |
| Security Groups | | | | | _ | | |
| Elastic IPs | | | | | | | |
| | | | | | | | |
| © 2009 - 2012 Amazon Web Service | < Back | Conti | nue ▶ | | | Eag | dback |
| © 2000 - 2013, Amazon web Servic | | | | | | reet | Dack |

Figure 32: Advanced Instance Options

10. We do not need to change any specific details on the instance and can continue through this

screen as well as the storage configuration screen that comes next.

| 👔 Services 🗸 Edit 🗸 | Request I | nstances Wiz | zard | | | | Cancel X | imesh 👻 🛛 N | N. Virginia 🕶 | Help 🕶 |
|---|---|--|--|------------------|------------------------------|---|-----------|-------------|---------------|--------|
| EC2 Dashboard Events Tags | CHOOSE AN AN Number o Availabilit | II INSTANCE D | PETAILS CREATE P | CEY PAIR | CONFIGURE FIREWALL | REVIEW | | | | |
| INSTANCES Instances Spot Requests Comparison Instances | Storage Your instan volumes, c | Device Configure nce will be laund r edit the settin | guration hed with the follov gs of the root volu | ving stor me. | age device settings. Edit tl | nese settings to add EBS volumes, insta | nce store | | | |
| Reserved instances | Туре | Device | Snapshot ID | Size | Volume Type IOPS | Delete on Termination | | | | |
| IMAGES | Root | /dev/sda1 | snap-bcdff2f2 | 8 | standard | true | ^ | | | |
| AMIs | Ephemeral | /dev/sdb | instance store v | volume: (| ephemeral0 | 🔀 Re | move | | | |
| Bundle Tasks | | | | | | | - | | | |
| ELASTIC BLOCK STOI Volumes Snapshots | 0 EBS Vol | umes 1 Epł | nemeral | | | | 💊 Edit | | | |
| | | | | | | | | | | |
| Security Groups | | | | | | | | | | |
| Elastic IPs | | | | | | | | | | |
| Placement Groups | | | | | | | | | | |
| © 2008 - 2013, Amazon Web Servio | < Back | | | | Continue | | | | Fee | dback |
| | | | | | | | _ | | | |

Figure 33: Storage Device Configuration

| 🎁 Services 🕶 Edit 🗸 | Request Instances Wizard | | Cancel X | n 👻 N. Virginia 👻 | Help 🕶 |
|--|---|--|--|-------------------|--------|
| EC2 Dashboard Events Tags INSTANCES Instances | CHOOSE AN AMI INSTANCE DETAILS CREATE REY PAIN Add tags to your instance to simplify the adminis case-sensitive key/value pair, are stored in the that help you organize, search, and browse your = Webserver. You can add up to 10 unique keys information, go to Tagging Your Amazon EC2 Res | t CONTROLER PREVAIL REVEW tration of your EC2 infrastructure. A form of metzi- soud and are private to your account. You can c resources. For example, you could define a tag v to each instance along with an optional value for ources in the EC2 User Guide. | adata, tags consist of a reate user-friendly names vith key = Name and value each key. For more | | |
| Spot Requests | Key (127 characters maximum) | Value (255 characters maximum) | Remove | | |
| Reserved Instances | Name | Demo Device | × | | |
| IMAGES AMIs Bundle Tasks ELASTIC BLOCK STOI Volumes Snapshots NETWORK & SECURT Security Groups Elastic IPs Placement Groups | Add another Tag. (Maximum of 10) | Continue | * | | |
| © 2008 - 2013, Amazon Web Servio | < Back | | | Feed | lback |

Figure 34: Naming the device

11. We can give a name to the device here or add any other parameters that we may find useful.

| i Services V Edit | Cancel 🔀 | mesh 👻 | N. Virginia 👻 | Help 👻 |
|--|-----------------|--------|---------------|--------|
| EC2 Dashboard Events Tags INSTANCES Instances Spot Requests Reserved instances INSTANCES INSTANCES INSTANCES Instances Spot Requests Reserved instances Chase Kay pair, effer a name and dick Create & Download Your Key Pair. You will be prompted to save the private k your computer. Note: You only need to generate a key pair once - not each time you want to deploy an Amazon EC2 instance or create a key pair, effer a name and dick Create & Download Your Key Pair. You will be prompted to save the private k your computer. Note: You only need to generate a key pair once - not each time you want to deploy an Amazon EC2 instance Choose from your existing Key Pairs Create a new Key Pair: Inter a name for your key pair.* Click to create your key pair.* Click to create your key pair.* Create & Download your Key Pair Snapshots | ey to ce. | | | |
| NETWORK & SECURI Security Groups Elastic IPs Placement Groups | | | | |
| © 2008 - 2013, Amazon Web Service | _ | | Feed | iback |

Figure 35: Creating a new Key Pair

12. On the next screen we need to create a new key pair, or use an existing one if present. This key will be used to access the Amazon EC2 instance and should be downloaded and stored in an accessible place on your local computer. Also remember to set the permissions on this file to 400.

| 🎁 Services 🗸 Edit 🗸 | Request Instances Wizard | Cancel 🗙 | imesh 🛩 | N. Virginia 👻 | Help 🕶 |
|---|--|----------|---------|---------------|--------|
| EC2 Dashboard Events Tags INSTANCES Instances Spot Requests Reserved Instances IMAGES AMIS Bundle Tasks ELASTIC BLOCK STOI Volumes Snapshots | CHOOSE AN AMI INSTANCE DETAILS CREATE KEY PAR COMPOUNE PREWALL REVIEW Security groups determine whether a network port is open or blocked on your instances. You may use an existing security or we can help you create a new security group to allow access to your instances using the suggested ports below. Add additional ports now or update your security group anytime using the Security Groups page. Choose one or more of your existing Security Groups SecOb0437 - default SecUrity Groups the suggested ports below. (Selected groups: sg-e4ad528f) | group, | | | |
| NETWORK & SECURI Security Groups Elastic IPs Placement Groups | Create a new Security Group Back Continue | | | 500 | lback |
| © 2008 - 2013, Amazon Web Servio | | _ | | Feed | lback |

Figure 36: Selecting Security Group

13. Select the default security group; we will only need the port for SSH for connecting to the

EC2 instance.

| Request Instances W | zard | Cancel X |
|---|---|--------------------|
| C2 Dashboard | | |
| vents | DETAILS CREATE REY PAIR CONFIGURE FIREWALL REVIEW | |
| ags Please review the inform | ation below, then click Launch. | <u>^</u> |
| AMI: | Obuntu Cloud Guest AMI ID ami-3fec7956 (x86 64) | |
| NSTANCES Name: | Ubuntu Server 12.04.1 LTS | |
| nstances Description: | Ubuntu Server 12.04.1 LTS with support available from Canonical | |
| pot Requests | (http://www.ubuntu.com/cloud/services). Edit | t AMI |
| Reserved Instances Number of Instances: | 1 | |
| Availability Zone: | No Preference | |
| MAGES Instance Type: | M1 Extra Large (m1.xlarge) | |
| MIs Instance Class: | On Demand Edit | t Instance Details |
| undle Tasks EBS-Optimized: | No | = |
| Monitoring | Disabled Termination Protection: Disabled | |
| LASTIC BLOCK STOL Tenancy: | Default | |
| folumes Kernel ID: | Use Default Shutdown Behavior: Stop | |
| inapshots RAM Disk ID: | Use Default | |
| Network Interfaces: | | |
| IETWORK & SECURI Secondary IP | | |
| ecurity Groups User Data: | | |
| lastic IPs IAM Role: | Edit | t Advanced Details |
| Placement Groups | | |
| Key Pair Name: | m1xlarge Edit | t Key Pair 👻 |
| (Pack | Launch | |
| Back | | |

Figure 37: Instance Summary for Review

14. On this screen we can review the instance details before we launch the instance.

15. Once the device is up and running, we can now associate the Elastic IP to this instance. The Elastic IP can then be used to access this device from terminal.

16. Go to the Elastic IP dashboard. Select the Elastic IP you wish to associate with the instance, select associate Address button at the top. The dashboard prompts for the Instance that you wish to associate with this Elastic IP.

| EC2 Dashboard | Allocate New Addres | Release | Address Associate Addres | S Disassociate Address | | ି 🔶 🖗 |
|--------------------|------------------------|-------------|------------------------------|-------------------------------|-------|-------------------|
| Events < | Viewing: All Addresses | | ▼ Search | | < < ⊥ | to 1 of 1 Items 📎 |
| INSTANCES | Address | Insta | Associate Address | Cancel 🗙 | | |
| Instances | 54.225.192.162 | | Select the instance to which | you wish to associate this IP | | |
| Spot Requests | | | address (54.225.192.162). | | | |
| Reserved Instances | | | Instance: You do not have an | y instances 🔻 | | |
| | | | | | | |
| AMIs | | | | Cancel Yes, Associate | | |
| Bundle Tasks | | | | | | |
| | | | | | | |
| ELASTIC BLOCK STO | | | | | | |
| Volumes | | | | | | |
| Shapshots | 1 Address selected | | | | | |
| NETWORK & SECURI | 🔮 Address: 54.2 | 225.192.162 | | | | |
| Security Groups | Address: | 54.225.1 | 92.162 | | | |
| Elastic IPs | Instance ID: | | | | | |
| Placement Groups | Scope: | standard | | | | |
| | Public DNS: | | | | | |
| | | | | | | |

Figure 38: Associating Elastic IP to Instance

17. Elastic IP addresses are released each time the instance is stopped and need to be associated with the instance when it starts.

Increasing the size of the storage device.

The default storage disk that is provided is an 8 GB device. To increase the storage capacity on this device, we can create a new volume as follows. This is an optional procedure that we can follow if we need more storage for our instance.

1. Select Volumes from the menu on the left under the header Elastic Block Storage.

| EC2 Dashboard | Create Volume | e Actions | * | | | | | | | | C 🕈 | 0 |
|--|---|-------------------------------|---|-----------|---------------|---|--------------|--|--------------|-----------|-----------|------------|
| Events Tags | Viewing: All Vol | lume: 💥 Delete V | /olume Volume | | | | | | < ≺ | 1 to 2 of | 2 Items | > > |
| INSTANCES | Name 🁒 | Vol 🎾 Detact | Volume | me Type | Snapshot | Created | Zone | State | Alarm Status | Attac | hment Ir | nformatio |
| Instances | empty | Sorce | Detach | lard | snap-bcdff2f2 | 2013-04-22T21:54:4 | 8 us-east-1c | available | none | | | |
| Spot Requests | empty | Create | Snapshot | lard | snap-17df424f | 2013-04-22T22:08:3 | 3 us-east-1c | 🥚 in-use | none | i-1bd | 90b76 (m1 | 1xlarge):/ |
| ELASTIC BLOCK STOL | 100 W 11111 | voi-ppode | UCZ | | | | | | | | | |
| Volumes | Details S | tatus Checks | Monitoring Ta | gs | | | | | | | | |
| ELASTIC BLOCK STOP Volumes Snapshots | Details S Volume II | tatus Checks | Monitoring Ta | gs | | Alarm Sta | tus: | none | | | | |
| ELASTIC BLOCK STOT Volumes Snapshots NETWORK & SECURI | Details S Volume II Capacity: | tatus Checks | Monitoring Ta Vol-bb8ae0e2 8 GiB | gs | | Alarm Sta Snapshot: | tus: | none snap-bcdff2f2 | | | | |
| Columes Snapshots NETWORK & SECURI Security Groups Elastic IPs | Details S Volume II Capacity: Created: | tatus Checks | Monitoring Tar vol-bb8ae0e2 8 GiB 2013-04-22 14:54 F | gs PDT | | Alarm Sta Snapshot: Zone: | tus: | none snap-bcdff2f2 us-east-1c | | | | |
| ELASTIC BLOCK STOT Volumes Snapshots NETWORK & SECURI' Security Groups Elastic IPs Placement Groups | Details S Volume II Capacity: Created: State: | tatus Checks): | Monitoring Tai Vol-bb8ae0e2 8 GiB 2013-04-22 14:54 P available | gs PDT | | Alarm Sta Snapshot: Zone: Attachmer | tus: nt: | none snap-bcdff2f2 us-east-1c | | | | |
| Volumes Snapshots NETWORK & SECURI' Security Groups Elastic IPs Placement Groups | Details S Volume II Capacity: Created: State: Volume Typ | tatus Checks): e: | Monitoring Ta Vol-bb8ae0e2 8 GiB 2013-04-22 14:54 F available standard | gs PDT | | Alarm Sta Snapshot: Zone: Attachmer IOPS: | tus: nt: | none snap-bcdff2f2 us-east-1c - | | | | |

Figure 39: Creating a snapshot

2. Select the default volume that is present and from the actions dropdown at the top, select create snapshot. Give the snapshot a name and description when prompted.

3. Now create a new volume using the create volume button. When prompted, select standard in the device type, input the size you need. The availability zone should be the same as the zone of your instance. In the snapshot, select the snapshot that you just created. Go ahead and click "create".

| EC2 Dashboard | Create Volume Actions | ¥ | | | | ୯ 🏘 \Theta |
|--|--|---|-------------|-------------------------------|--------------|---------------------------|
| Events Tags | Viewing: All Volumes | ▼ (Search | | | > >ا | 1 to 2 of 2 Items 🔉 🔌 |
| INSTANCES | Name 🐄 Volume ID | Canacity Volume Type Spanshot Cre | ated Zone | State | Alarm Status | Attachment Information |
| Instances | empty Sol-bb8ae0 | Create Volume | Cancel 🗵 | available | none | |
| Spot Requests | empty vol-cdee84 | Volume Type: Standard - | | in-use | none | i-1bd90b76 (m1xlarge):/de |
| E IMAGES AMIS Bundle Tasks ELASTIC BLOCK STOI Volumes Snapshots NETWORK & SECURI | 1 Volume selected Volume: vol-bb8ae Details Status Checks Volume ID: Capacity: | Size: 100 GiB • (Min: 8 GiB, IOPS: Max: 2000 IOPS Availability Zone: us-east-tb • Snapshot: snap-17df424f - empty snap | Max: 1TiB) | none snap-bcdff2f2 | | = • • • |
| Security Groups | Created: | 2013-04-22 14:54 PDT | Zone: | us-east-1c | | |
| Elactic IDc | State: | available | Attachment: | | | |
| Elastic IPs Placement Groups | | standard | IOPS: | - | | |

Figure 40: Creating a Snapshot

4. Now we are ready to detach one volume and attach the new one to our instance. Make sure

that the instance is Stopped when we do this.

5. Select the old volume that we need to detach. From the actions drop down select detach

volume.

| EC2 Dashboard | dit 🗸 Cre | ate Volum | ne Actions | • | - | - | - | - | | Animesh 👻 | N. Virginia 👻 Help 👻 |
|---|--------------|--|---|----------------------------------|--|--|---|---|-------------------------------|--------------|---------------------------|
| Events Tags | Viewi | ng: All Va | olumes | ▼ (Search | | | | | | × < | 1 to 2 of 2 Items 📏 渊 |
| INSTANCES | | Name 🕫 | Volume ID | Capacity | Volume Type | Snapshot | Created | Zone | State | Alarm Status | Attachment Information |
| Instances | | empty | 🌍 vol-bb8ae0e | 2 8 GiB | standard | snap-bcdff2f2 | 2013-04-22T21:54:48 | us-east-1c | available | none | |
| Spot Requests | | empty | 📚 vol-cdee849 | Attach Vol | ume | | | Cancel 🗙 | in-use | none | i-1bd90b76 (m1xlarge):/de |
| MAGES AMIS Bundle Tasks ELASTIC BLOCK STOI Volumes Snapshots NETWORK & SECURT Security Groups | 1 Vo | lume sele Volume tails folume I apacity: | cted e: vol-bb8ae Status Checks D: | Volume: Instances: Device: | vol-bb8ae0e i-1bd90b76 - /dev/sda1 Linux Devices Note: Newer I through /dev/ here (and sho | 2 in us-east-1c m1xlarge (stopped : /dev/sdf through inux kernels may xvdp intermally, ev wn in the details) is | in us-east-1c /dev/sdp ename your devices to, en when the device ner /dev/sdf through /dev Cancel | /dev/xvdf ne entered /sdp. Yes, Attach | none snap-bcdff2f2 | | |
| Elastic IPs | 0 | reated: | | 2013-04-22 14 | 1:54 PDT | | Zone: | | us-east-1c | | |
| Placement Groups | 5 | otate: | no: | available | | | TOPS | : | | | |
| | P | roduct Co | des: | - | | | | | | | |
| © 2008 - 2013, Amazon Web | Service | s, Inc. or it | s affiliates. All rigl | nts reserved. F | Privacy Policy | Terms of Use | | | | | Feedback |

Figure 41: Attach a Volume

6. Now select the new volume that we just created. From the Actions dropdown, select Attach Volume.

7. Select the Instance that we need to attach the new volume. Set the device to /dev/sda1 in order

to make this the boot device.

8. The new volume has now been attached to the device and the instance now has increased disk space.

3.3 Setting-Up Fast Artificial Neural Networks on Amazon EC2

Pre-requisites:

- 1. Make sure that you have cmake installed:
 - sudo apt-get install cmake
- 2. Also install libgtk2.0-dev using the command

sudo apt-get install libgtk2.0-dev

Installation

Copy the Fast Artificial Neural Networks Directory to your Amazon EC2 machine using secure

copy (SCP).

3. Using the terminal go to the top level of the FANN directory and run the following commands

- 4. cmake .
- 5. sudo make install
- 6. sudo ldconfig

7. To confirm successful installation navigate to the examples directory and execute the following:

make runtest

8. This should result in the running of some sample program that trains and predicts on some sample data.

9. If you are able to see the results, then you have successfully configured FANN on your machine.

3.4 Data Preparation

The given data was in the Comma Separated Value format. The FANN framework requires that training data be in a specific format. For FANN training file, the first row of the file should contain three columns, the number of training samples provided, number of input variables and number of output variables. Starting from the second row, the row should contain the input variables that are delimited by space, the output for these set of input values should be in presented in the next row. Also, the framework only accepts numeric values, so the output values of true and false need to be interpreted as zero's and one's.

For creating balanced datasets we need to separate the true samples from the false samples and then create a new mini batch dataset from samples from both the true dataset and the false dataset. After the separation we can create the dataset we need for the experiments. We now have the following datasets:

- IBM Training Data Set (approx. 900,000 rows)
 - False Training Data
 - True Training Data (approx. 4000 rows)
- IBM Prediction Data Set (approx.. 400,000 rows)

- False Prediction Data
- True Prediction Data (approx. 2000 rows)

3.5 Training the Neural Network Using FANN

The overall IBM Watson dataset is a 3 GB CSV (comma separated value) file. This file has been separated into two parts for prediction and training purposes. This file has 1.3 million rows of data.

The training part is approximately 2/3's of the total file, which we refer to as the IBM Training Data set and 1/3 of the file is used for prediction, which we refer to as the IBM Prediction data set.

The simplest way of training a neural network is to simply supply the entire training data set at once and let the network train on it until a desired error is reached. The number of hidden neurons is kept constant for the experiment. There are 683 hidden neurons in the hidden layer, that is (2n+1) neurons, where n is the number of features.

This method is the base case for my experiments. The entire IBM Training set is used to train the network, and then the prediction is made on the entire Prediction dataset.

The fast artificial neural network is a library written in C. Thus for training a neural network using FANN can be done by writing a C program that uses the standard FANN functions. The Steps involved for creating and training a neural network using FANN are as follows:

1. An empty neural network structure can be created using the following in-built FANN data structure

struct fann *ann = fann_create_standard(num_layers, num_input, num_neurons_hidden, num_output);

where:

fann_create_standard = function for creating a neural network.

ann = name of the network.

Num_layers = number of layer in the neural network, this includes the input layer, the hidden layers, and the output layer.

Num_input = number of input variables, or neurons in the input layer.

Num_output = number of outputs, or neurons in the output layer.

 Optionally, we can set the activation functions for the different layers by using the "fann_set_activation_function_hidden" or "fann_set_activation_function_output" functions.

3. We can also set the training function at this stage.

4. We are now ready to train the neural network. We need to simply provide the file that contains the training data. We use the function

fann_train_on_file(ann, filename, max_epochs, epochs_between_reports, desired_error);

fann_train_on _file : function used for training the network on training data from a file.

Filename: the filename (string) that contains the training data.

Max_epochs: the maximum number of epochs we want the training to continue. This is a stopping condition.

Epochs_between_reports: the program will report error after this interval.

Desired_error: this is the condition to successfully stop training.

5. Once the training is complete we can save the network using the function

Fann_save (neuralnet, filename)

3.6 Training Experiments using FANN

The following training experiments were performed on the dataset. The aim of the training experiments was to reduce the training time and improve the accuracy of the final predictions made.

1. Train on IBM true training Data with 228 hidden neurons in the hidden layer.

Max epochs 500000. Desired error: 0.0049999999.

Table 1: Batch training of true data

Epoch Error 1 0.312028 2 0

Time Taken 11.790000 seconds

2. Train on all False Training Data with 228 hidden neurons in the hidden layer.

Max epochs 500000. Desired error: 0.0049999999

Table 2: Batch Training of False Data

Epoch Error

1 0.328547

2 0

Time Taken 2519.089844 seconds

3. Training Batches of IBM True training data followed by the entire training dataset and then the entire true training dataset again.

Max epochs 500000. Desired error: 0.00499999999.

Table 3: Batch training of true and false batches

- 1 0.995213
- 2 0.983665
- 3 0.136118
- 4 0.008193
- 5 0.619769
- 6 0.558391
- 7 0.004787

Time Taken 19636.681641 seconds

4. Training on Falsified data mixed with True data. When mixed in an equal percentage, the

network can never reach the desired error.

Table 4: Training on Falsified data

| Epoch | Er | ror |
|-------|----|----------|
| | 1 | 0.282506 |
| | 2 | 0.200073 |
| | 3 | 0.200073 |
| | 4 | 0.200073 |
| | 5 | 0.200073 |

| 6 | 0.300657 |
|----|----------|
| 7 | 0.173688 |
| 8 | 0.196442 |
| 9 | 0.199701 |
| 10 | 0.183086 |
| 11 | 0.361739 |
| 12 | 0.181087 |
| 13 | 0.187449 |
| 14 | 0.196585 |
| 15 | 0.183785 |
| 16 | 0.16917 |
| 17 | 0.196298 |
| 18 | 0.163221 |
| 19 | 0.160709 |
| 20 | 0.16001 |
| 21 | 0.176327 |
| 22 | 0.163362 |
| 23 | 0.159997 |
| 24 | 0.162346 |
| 25 | 0.159547 |
| 26 | 0.161851 |
| 27 | 0.160139 |
| 28 | 0.163535 |
| 29 | 0.160656 |
| 30 | 0.158725 |
| 31 | 0.158323 |
| 32 | 0.158957 |
| 33 | 0.160555 |
| 34 | 0.159454 |
| 35 | 0.159011 |
| 36 | 0.158669 |
| 37 | 0.159198 |
| 38 | 0.15932 |
| 39 | 0.158845 |
| 40 | 0.158657 |
| 41 | 0.158775 |
| 42 | 0.159072 |
| 43 | 0.160326 |
| 44 | 0.158104 |

| 45 | 0.158034 |
|----|----------|
| 46 | 0.158093 |
| 47 | 0.161786 |
| 48 | 0.161577 |
| 49 | 0.156076 |
| 50 | 0.155677 |

5. Training on a balanced data set:

Only the first 50 epochs are presented.

Table 5: Training on the Balanced dataset

| Epochs | Error |
|--------|----------|
| 1 | 0.330194 |
| 2 | 0.125 |
| 3 | 0.625 |
| 4 | 0.12317 |
| 5 | 0.625 |
| 6 | 0.624676 |
| 7 | 0.124891 |
| 8 | 0.125002 |
| 9 | 0.116761 |
| 10 | 0.624089 |
| 11 | 0.623223 |
| 12 | 0.598423 |
| 13 | 0.122669 |
| 14 | 0.120844 |
| 15 | 0.107279 |
| 16 | 0.613381 |
| 17 | 0.609236 |
| 18 | 0.463886 |
| 19 | 0.113532 |
| 20 | 0.111817 |
| 21 | 0.09927 |
| 22 | 0.553946 |
| 23 | 0.538812 |
| 24 | 0.27922 |
| 25 | 0.102347 |
| 26 | 0.116385 |
| 27 | 0.115937 |

| 28 | 0.098564 |
|----|----------|
| 29 | 0.094998 |
| 30 | 0.093823 |
| 31 | 0.093888 |
| 32 | 0.097353 |
| 33 | 0.092338 |
| 34 | 0.085421 |
| 35 | 0.119162 |
| 36 | 0.1192 |
| 37 | 0.080527 |
| 38 | 0.081208 |
| 39 | 0.079357 |
| 40 | 0.073873 |
| 41 | 0.071581 |
| 42 | 0.069407 |
| 43 | 0.066854 |
| 44 | 0.066991 |
| 45 | 0.067838 |
| 46 | 0.064344 |
| 47 | 0.062108 |
| 48 | 0.06104 |
| 49 | 0.059682 |
| 50 | 0.058266 |

4. Measuring Performance Using Time and Accuracy

4.1 Prediction Using FANN

The FANN provides us with a function called "fann_run" that can predict results. Once we load a saved neural network, we can then use this function to provide input values for the trained neural network and provide us with an output. Once the output is stored into a file this file is then this file is checked for accuracy by comparing it with a target file using a python script. The following are the steps involved:

- 1. Load the trained neural network.
- 2. Provide input one row of inputs at a time

- 3. Save the output to a file, results.txt.
- 4. Provide the Target file and the results file to the python script mse.py
- 5. The script calculates the Mean Square Error and Percentage Error.

4.2 Prediction and Performance Results





Figure 42: Batch Training

| Time Taken | Mean Squared error |
|------------|--------------------|
| 8.37 hours | 0.258 |

Training on a balanced dataset

The balanced dataset was 1% the size of the entire dataset with a mix of 50% true and false data.



Figure 43: Training on Balanced Data Sets

| Time taken | Mean Square Error |
|------------|-------------------|
| 11.07 hrs. | 99.6126 |

The balanced dataset when exactly balanced performed as if it had only seen true data and classified everything as true. This is a case of failure.

Training Using RMSProp

The RMS prop algorithm is an algorithm that does not use the learning rate parameter directly. The learning rate at each neuron is calculated by dividing the learning rate by an average of the previous weights that this particular neuron has had.



Figure 44: Training Using RProp on MiniBatch: MSE vs Epochs

| Time Taken | Mean Squared Error |
|------------|--------------------|
| 1.86 Hours | 0.00397 |

Training on Falsified Data

In this case, the data was inserted with true values that had been changed to false.

In this kind of situation, the network was never able to converge towards the desired error rate.

This is due to that fact that the training data consisted of a large percentage of noisy data and the

system was unable to recognize a correct classification pattern.



Figure 45: Training on Data with High Noise: MSE vs. Epochs

Varying the Number of Neurons for a Constant Dataset

This graph provides a comparison of training times of neural networks with different number of hidden neurons. The training data size is 10,000 rows, which is 1% of the entire training dataset.

As we can see, when the hidden neurons are less than half the size of the input layer, the error bounces between a large range and eventually reaches the desired error. In the case of the hidden layer being around 2/3 the size of the input layer, the error smoothly tapers off towards the desired error.



Figure 46: Varying Hidden Neurons: MSE vs. Epochs

Training Time for Networks with different Hidden Neurons



Figure 47: Training Time

The training time increases considerably as the number of neurons in the hidden layer approaches the number of neurons in the input layer.



Comparing Prediction Time of Networks with Different Hidden Neurons

Figure 48: Prediction Time

4.3 Conclusion

While the standard training techniques took around a 100 epochs to converge towards the desired error, training using mini batches generally took under 50 epochs to make this convergence.

Most notable was the Mini-batch Training using RMS propagation algorithm that took nearly 35 epochs to converge. This also had considerable effect on the training time, and the training was able to finish in less than 2 hours. The prediction rates for this kind of training were also very high. The percentage error was 0.39 percent.

5.0 Future Work

We have explored ideas where we are splitting and resampling the data based on rows. An important work will be to divide the data based on columns or features. One particular interesting approach would be to train multiple networks on different features and combine their outputs using the Multiple Experts Network. For example, there can be 3 networks training on approximately 120 features each. These outputs can then be used to predict, the output of these 3 networks can then be combined in different ways. It can either be used to train another neural network, or use a function to combine the 3 outputs into single output.

6.0 References

 Empirical Modeling of Very Large Data Sets Using Neural Networks, Aaron J. Owens, DuPont Central Research and Development.

[2] Modular neural networks with applications to pattern profiling problems, H. Chris Tseng, Bassam Almogahed. San Jose State University.

http://www.sciencedirect.com/science/article/pii/S0925231208005444

[3] Gating Improves Neural Network Performance, Min Su, Mitra Basu, City University of New York.

[4]Feature Preparation in text categorization, Ciya Liao, Shamim Alpha, Paul Dixon Oracle Corporation

http://www.oracle.com/technetwork/database/enterprise-edition/feature-preparation-130942.pdf

[5]Feature selection for pattern classification problems, Li Zhang, Gang Sun, Jun Guo School of Information Engineering, Beijing University of Posts and Telecommunications http://ieeexplore.ieee.org/ielx5/9381/29791/01357202.pdf?tp=&arnumber=1357202&isnumber= 29791

[6] Unsupervised Feature Selection: A Neuro-Fuzzy Approach. Sankar K. Pal, Rajat K. De, Jayanta Basak, IEEE TRANSACTIONS ON NEURAL NETWORKS, VOL. 11, NO. 2, MARCH 2000

http://www.isical.ac.in/~rajat/publications/journal/00839007.pdf

[7] A neural Fuzzy System with Fuzzy Supervised learning

http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=537316&tag=1

[8] Efficient Approximation with Neural Networks: A comparison of Gate functions, Bhaskar Dasgupta, Georg Schnitger, Department of Computer Science, The Pennsylvania State University.

[9] Feature Selection Using Principal Feature Analysis, Ira Cohen Qi Tian Xiang Sean Zhou Thomas S. Huang, Beckman Institute for Advanced Science and Technology, University of Illinois at Urbana-Champaign.

[10] Overview of Amazon Web Services,

http://media.amazonwebservices.com/AWS_Overview.pdf

[11] Fast Artificial Neural Networks

http://leenissen.dk/rl/Steffen_Nissen_Thesis2007_Hyper.pdf

7.0 Appendix

7.1 Script to Change Raw CSV Data to FANN Format

```
#this script changes the ibm format to the required FANN format
# 1st line has totalrows inputs outpts, this appears at the top once
# 1st set of input
# Target output
# and so on, the entire file is space delimited
from xml.dom.minidom import parse, parseString
def processData(ipdoc,opdoc):
     print "Parsing started"
     print ipdoc
      ip = open(ipdoc,"r")
      op = open(opdoc,"w")
      csvdata= ip.readlines()
      for count,line in enumerate(csvdata):
            print ""
      op.write("%d 341 1\n" % (int(count)+1))
      op.close()
      op = open(opdoc, "a")
      for num,line in enumerate(csvdata):
            #print "Line number %d" %num
            data = []
            data = line.split(",")
            pre = []
            post = []
            for idx,item in enumerate(data):
                  if (idx < 342) and (idx > 0):
                        pre.append(item)
                        pre.append(" ")
                  else :
                        if (idx == 342):
                              item = item.rstrip()
                              if (item == "true"):
                                     #print "true"
                                     post.append("1 \n")
                              else:
                                     #print "false"
                                    post.append("0 \n")
            printList(pre,op)
            op.write("\n")
            printList(post,op)
      #print "NUM %d" %num
      ip.close()
      op.close()
```

```
def printList(lst,opfile):
    for item in lst:
        opfile.write(item)

if ___name__ == "__main__":
    ipdoc = "ibm2gb"
    opdoc = "ibm2gb.train"
    processData(ipdoc,opdoc)
```

7.2 Script to Generate Unlabeled Data from CSV file

```
#Takes as input the ibm csv file
#separates the results into a separate file
# gives us an unlabelled csv file
from xml.dom.minidom import parse, parseString
def processData(ipdoc,opdoc,opdoc2):
     print "Parsing started"
     print ipdoc
      ip = open(ipdoc,"r")
      op = open(opdoc,"w")
      op2 = open(opdoc2,"w")
      csvdata= ip.readlines()
      for num,line in enumerate(csvdata):
            #print "Line number %d" %num
            data = []
            data = line.split(",")
            pre = []
            post = []
            for idx,item in enumerate(data):
                  if (idx < 342) and (idx >= 0):
                        pre.append(item)
                        pre.append(",")
                  else :
                        if (idx == 342):
                              item = item.rstrip()
                              if (item == "true"):
                                     #print "true"
                                     post.append("1 \n")
                              else:
                                     #print "false"
                                    post.append("0 \n")
            printList(pre,op)
            op.write("\n")
            printList(post,op2)
```

```
ip.close()
op.close()
op2.close()

def printList(lst,opfile):
   for item in lst:
        opfile.write(item)

if ______ mame___ == "____main___":
        ipdoc = "ibmlgb"
        opdoc = "outlTest.data"
        opdoc2 = "outlTarget.data"
        processData(ipdoc,opdoc,opdoc2)
```

7.3 Script to Calculate Mean Squared Error

```
# this file will calculate the mean squared error and percent error
# given the results and the targets
# file also changes float results to binary from float if flag is set to 1
# if not then results remain in float
from xml.dom.minidom import parse, parseString
def processData(ipdoc,ipdoc2,flag):
      #print "Parsing started"
      #print ipdoc
      test = open(ipdoc,"r")
      target = open(ipdoc2,"r")
      testdata= test.readlines()
      targdata= target.readlines()
      testList =[]
      targList = []
      for num,line in enumerate(testdata):
            #print "Line number %d" %num
            data = line.rstrip()
            temp = float(data)
            if (flag == 1):
                  if (temp > 0.65):
                        temp = 1
                  else:
                        temp = 0
            testList.append(temp)
      for num,line in enumerate(targdata):
            #print "Line number %d" %num
            data = line.rstrip()
            targList.append(float(data))
      test.close()
```

```
target.close()
      calc mse(testList,targList)
def calc mse(testList, targList):
      sum = 0
      tot = 0
      for tar, res in zip(targList, testList):
           diff = tar - res
            sqr = diff * diff
            sum += sqr
            tot = tot+1
     mse = sum/tot
     percError = mse * 100
     #print (sum)
     #print (tot)
     print ("mean Squared error: %f" %mse)
     print ("Percent error: %f" %percError)
def printList(lst):
     for item in lst:
           print (item)
if __name__ == "__main__":
     ipdoc = "test.txt"
     ipdoc2 = "target.txt"
     processData(ipdoc,ipdoc2,1)
```