Artificial Neural Networks for ATLAS Data Analysis

2009 PPE Summer Project Report

Gavin Kirby
Matriculation No. 0502543
Supervisor: Dr. C. Collins-Tooth

June-July 2009

Contents

1 Introduction

In data analysis for high energy physics it is highly desirable to implement powerful computational methods. To this end it was deemed beneficial to develop an Artificial Neural Network (ANN or simply NN) system for processing Analysis Object Data (AOD) from the ATLAS experiment at CERN [1]. A Neural Net is a computational structure that aims to simulate the function of biological neural networks, i.e. those that operate in animal brains, with the advantage of great flexibility, since it can alter aspects of its structure to optimise its performace in analysing a given form of input. A Neural Net's structure consists of several layers of "neurons": one input layer, several "hidden layers" where the processing occurs, and one output layer. Each neuron is assigned a variable, and the process of training adjusts the "weight" on each neuron: this is the first process that occurs when the Neural Net software is run. The underlying mathematical principle is that of spectral decomposition, i.e. finding the principal axes of the input data sets (ntuples) in N-dimensional configuration space, hence allowing optimisation of the Neural Net for processing a particular kind of input data.

This project aimed to develop an NN system and fitting software for the analysis of data from inclusive Higgs searches at ATLAS involving a lepton trigger and Higgs decay to $b\bar{b}$. It also aimed to document this software, both its development and how it is to be used for data analysis - specifically separating signal from background and obtaining exclusion limits based on the anticipated luminosity. It was desirable that the documentation should allow the data analysis system to be used by those without advanced knowledge of CERN ROOT [2] or C++ programming, and should include a user-friendly guide to gaining access to relevant resources (TWikis, Grid, etc.).

One of the principal goals of the project was to document the steps that were taken to develop the code and hence produce a guide to using the Neural Net software; this documentation was to be added to the ATLAS group TWiki [3] in order to make it readily accessible to all members of the group. A TWiki is a structured wiki which provides an environment for collaboration and knowledge and document management; in many ways it is similar to a wiki (e.g. Wikipedia).

It was also decided to document the procedures that had to be followed to begin work on the software, including getting a CERN computing account and a Grid certificate (required for using Subversion). This documentation was intended to assist those who would have to follow these procedures in future.

2 Running the Neural Net

The Neural Net was constructed and run using ROOT's Toolkit for Multivariate Data Analysis (TMVA) [4], with a structure consisting of two hidden layers, of

N+1 and N nodes respectively, where N is the number of input variables. It was trained on sample ROOT ntuples that were generated by code obtained from the CERN Subversion repository. These files contained simulated data (as would be obtained empirically as output from ATLAS), which was processed by the Neural Net to isolate the underlying physical processes.

One of the most problematic issues that was encountered during the development of the Neural Net software was a recurring spike in the output plots. This was caused by the inclusion of events with no "sensible states" in the training and templating processes: events with "sensible states" are defined to be those in which the final particles and jets can be combined in at least one way to yield intermediate particles - clearly these are the only events which are of interest. These events were then stacked in the middle of the output plot (as being neither signal-like nor background-like), causing a tremendous distortion. To remedy this problem, it was necessary to remake the input ntuples to include an extra bit in the $my\_failEvent$ word for each event, which would indicate whether the corresponding event had zero "sensible states" or not. A constraint could then be added to the script to exclude those events without any sensible states from the training, and a bitmask $cutMask$ in another file was set to the value of the relevant bit (65536), together with an inversion of that bit, to ensure that these events would also be excluded from templating.

It was necessary to edit the script file that ran the Neural Net in order to make additional changes to the running procedure, including automatic allocation of the working directory (rather than having a hard-coded one that must be changed when the script is moved to another directory).

It was also discovered that there were errors in the values that had initially been assigned to the weights on the different processes. The initial attempt to correct this error failed, because entering the correct weights caused the signal to background ratio to become so low that the fit could not be completed. This was rectified by adding another bit to the $my\_failEvent$ word, 131072, which would indicate whether the event had at least 4 tightly-tagged b-jets. The fail code was then set to 196608 in order to restrict the training and fitting to these events and hence ensure a greater signal to background ratio, albeit at the cost of leaving few events remaining for fitting.

3 Using Code from Subversion

Subversion (SVN) [5] is a software tool that was used to create and maintain an online repository of the software, allowing development to be centralised and coordinated much more effectively. Snapshots of code in development are released as "tags", with each subsequent tag representing a later phase in the development of the software. New files can easily be added to projects, and redundant files can be removed. Several different tagged releases were made of the Neural Net code throughout its development, as changes were made to files

to add and improve functionality, and new files were committed.

At each stage, development was carried out by checking out the most recent tagged release from the Glasgow ATLAS repository, then editing it as required. Subversion automatically tracked the changes, making subsequent commits simple. Both the Glasgow and CERN Subversion repositories were used - the former for the Neural Net code, and the latter for the code that was used to create the ntuples containing input data.

4 Data Analysis

The final outputs of a Neural Net run were a set of plots which show the distinction between the signal and the background processes - see Figs. 1 and 2.

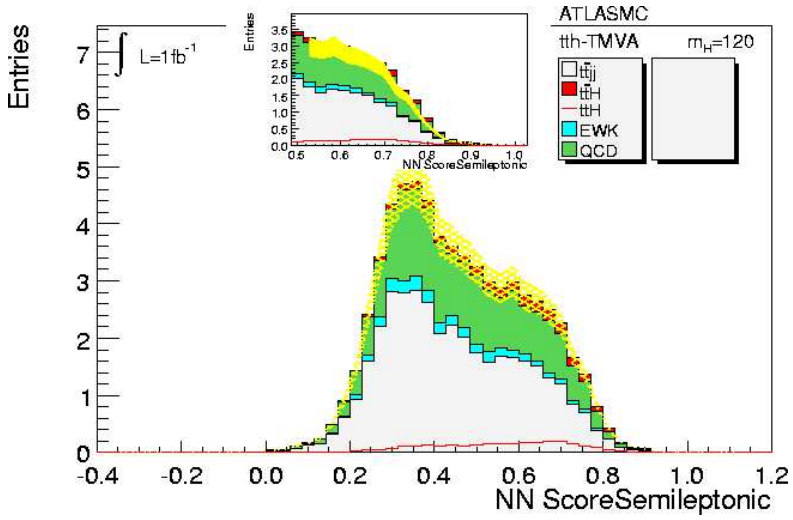

Figure 1: Output Fit - Original Weights. Signal events towards 1, background towards 0.

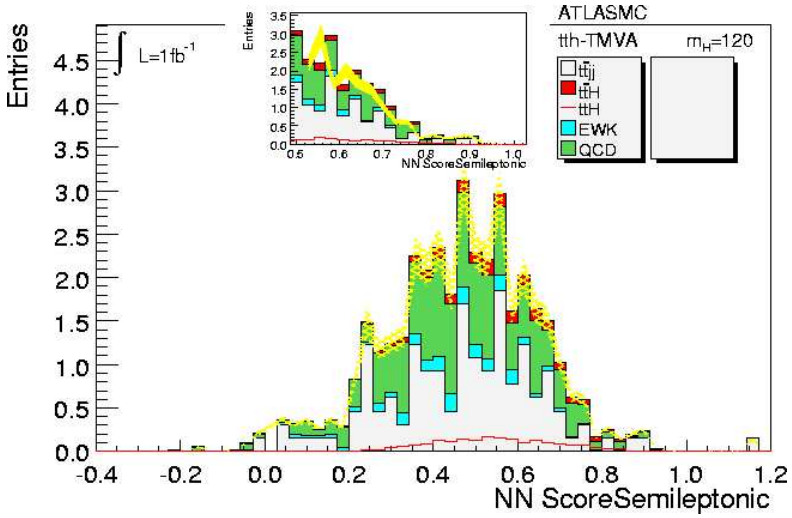

Figure 2: Output Fit - Fixed Weights. Signal events towards 1, background towards 0.

The plots seen in these figures allow one to discriminate between the signal (e.g. $t\bar{t}H$, H→$b\bar{b}$) and background ($t\bar{t}jj$, $t\bar{t}b\bar{b}$ (EWK), $t\bar{t}b\bar{b}$ (QCD)) processes based on the separation between the peaks; the more distinct the signal and background peaks are, the more significance there is. The NNScore variable indicates how signal-like (close to 1) or background-like (close to 0) an event is. In Fig. 2 it can be seen that there is a higher signal to background ratio but a lower total number of events (because of the cut criteria), hence the plot is noticeably less smooth. In addition, there is an output text file which contains information about the number of times Standard Model cross-section that can be excluded, for example (with fixed weight values):

```
120 -2 sigma: 4.17079 NSig=2.04552 NBkg=35.7963 NData=0
120 -1 sigma: 5.33905 NSig=2.04552 NBkg=35.7963 NData=0
120   median: 7.62376 NSig=2.04552 NBkg=35.7963 NData=0
120 +1 sigma: 10.6662 NSig=2.04552 NBkg=35.7963 NData=0
120 +2 sigma: 14.6113 NSig=2.04552 NBkg=35.7963 NData=0
```

There are also plots which show the values of $\log 1 - \frac{s}{b}$ for 1000 pseudoexperiments - examples of these can be seen in Figs. 3 and 4. These plots indicate how significant random fluctuations are in the output data, i.e. how probable it is that an apparent signal is the product of stochastic processes.

It is also important to be able to measure the efficiency of a Neural Net, since optimal performance depends on running it for a number of cycles which will allow it to train sufficiently on the sample data to be able to recognise the key features of the distribution functions (relating to the underlying physical processes) without becoming over-sensitive to the particularities of the specific data sets on which it has been trained. Thus there are two hazards relating to a non-optimal number of training cycles: undertraining, which causes the

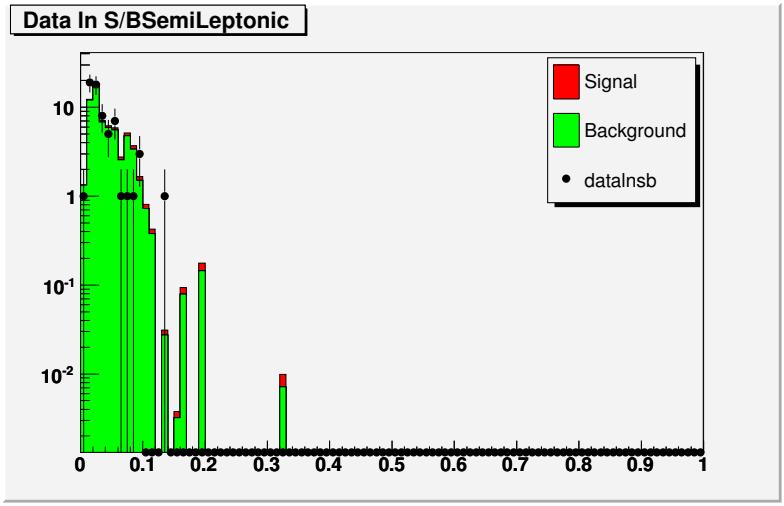

Figure 3: Logarithmic Plot of Signal to Background Ratio - Original Weights

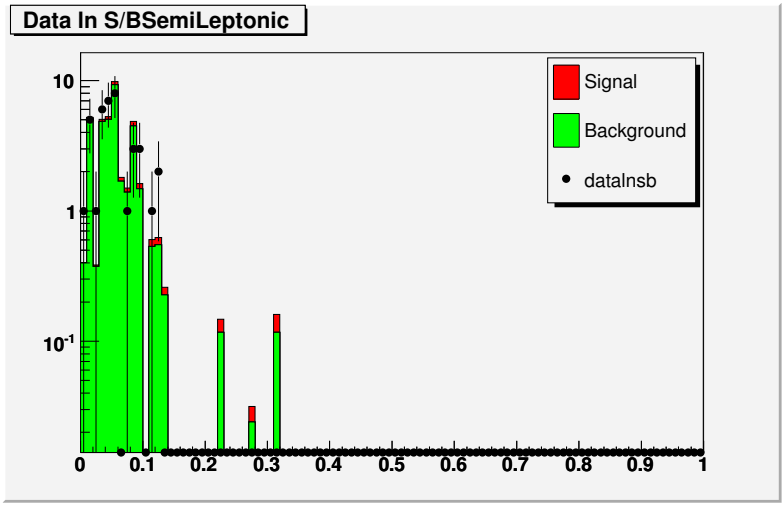

Figure 4: Logarithmic Plot of Signal to Background Ratio - Fixed Weights

Neural Net to be insufficiently sensitive to the physically significant features of input data, and overtraining, which causes the Neural Net to exaggerate the importance of minor features present in the sample data to which it has been exposed.

One of the most significant additions made to the Neural Net software in this project was a macro that produced training plots to indicate how the Neural Net performance varied with the number of training cycles used, which would allow the optimum number of cycles to be determined. The macro also produced a pair of plots comparing the training and testing outputs for the signal and background respectively - see Figs. 7 and 8.

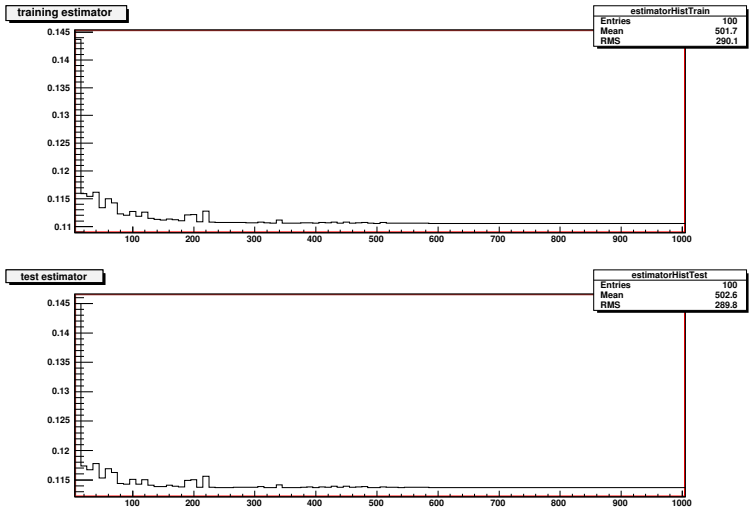

Figure 5: NN Performance against No. of Training Cycles - Original Weights

Figures 5 and 6 show how the Neural Net would perform with a varying number of training cycles: with few cycles, there are rapid changes in performance, but this quickly stabilises until, at an optimal value, it flattens out. Here it is clear that, after about 500 cycles, the graph is almost flat, meaning that further cycles add little to performance, and may indeed lead to overtraining. 1000 training cycles had been used by default, but on inspection of Figs. 5 and 6 it appeared that this number was far beyond the optimum, hence there was a danger that the Neural Net was being overtrained on the input data.

5 Conclusion

At the end of the project, the Neural Net was in an acceptable working condition for processing ATLAS AOD ntuples, with a body of documentation to allow new users to familiarise themselves with it. Subsequent work on the Neural Net would involve refining its operation, with a view to making runs more efficient

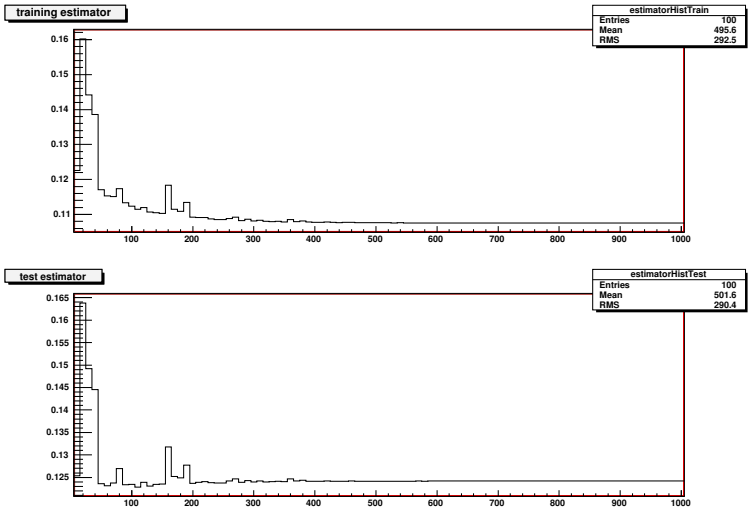

Figure 6: NN Performance against No. of Training Cycles - Fixed Weights

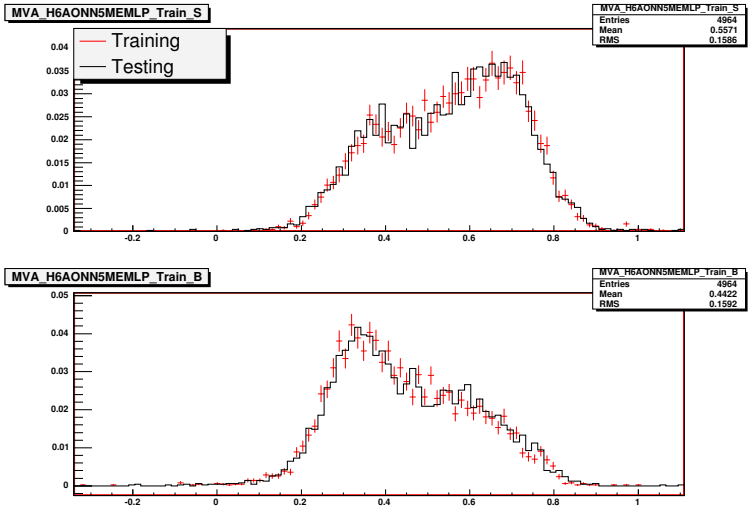

Figure 7: Training/Testing Comparison - Original Weights

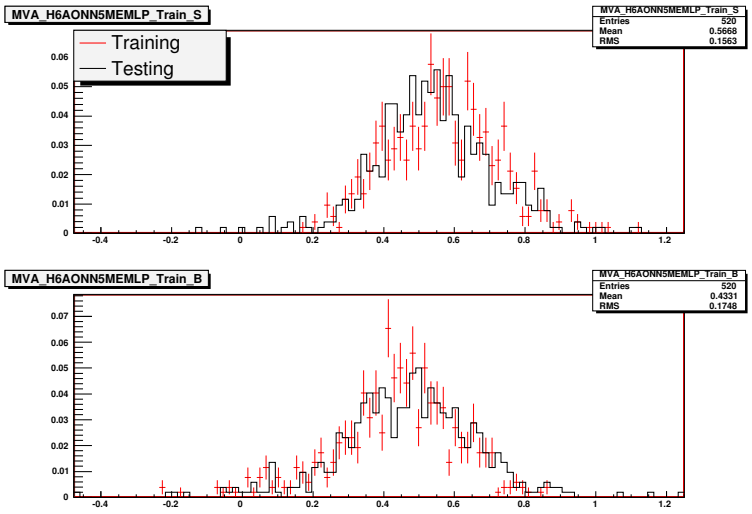

Figure 8: Training/Testing Comparison - Fixed Weights

and increasing the overall sensitivity, potentially allowing more exclusions to be made at higher confidence.

Such improvements to the Neural Net could be accomplished by careful consideration of the input variables to attempt to find the optimal subset - in principle, the principal component analysis should allow the Neural Net to identify the most significant variables, but a good choice of inputs would greatly improve processing time. There is also a potential overtraining problem when the number of training cycles is set to 1000, as can be seen in Figs. 5 and 6; some experimentation should be done to discover what the optimum number of cycles is.

It would also be advantageous to compare the results of the fits obtained through this Neural Net with those of other researchers, such as Junk [6]. In addition, it should be noted that the constraints that were imposed on the events in the input ntuples were strict, so that very few events were left over for training and fitting. This caused less than optimal output on the later runs with the fixed weights, a problem which could be solved by remaking the inputs to increase the total number of events in them.

6 Bibliography

[1] ATLAS Collaboration, *Expected Performance of the ATLAS Experiment*, CERN, 2008

[2] *ROOT User's Guide v.5.21*, CERN, 2008

[3] Glasgow ATLAS TWiki Webpage, URL:
https://ppes8.physics.gla.ac.uk/twiki/bin/view/ATLAS/AtlasDataAnalysis

[4] *TMVA Users' Guide*, CERN, 2007

[5] Collins-Sussman, B., Fitzpatrick, B.W. and Pilato, C.M., *Version Control with Subversion* (ebook), 2008

[6] Junk, T. *Sensitivity, Exclusion and Discovery with Small Signals, Large Backgrounds, and Large Systematic Uncertainties*, Fermilab, 2007