DØNote 4889 - Draft v1.0

A Neural Network b-tagging Tool

Tim Scanlon

Imperial College London - UK

August 23, 2005

Abstract

A Neural Network b-tagging tool has been developed which shows substantial improvement over the current b-tagging tools on Monte Carlo samples. The Neural Network b-tagger is trained and optimised on MC samples and shows fake rate reductions of 80% for a fixed signal efficiency and signal efficiencies increases ranging from 50% to 15% for a fixed fake rate over the current b-taggers.

1 Introduction

The main background at a hadron collider is multi-jet events. The identification of b-jets is essential in reducing this substantial background. Due to the b's relatively long lifetime b-hadrons can travel several millimetres before decaying.

There are three main ways to identify a b-jet.

1. Explicitly reconstruct secondary vertices from charged particle tracks.
2. Identify charged particles tracks which have large impact parameter significances[1] with respect to the primary vertex (PV).
3. Identification of a muon:- 10% of the time a b will decay into a muon which will typically have a large transverse momentum (p_T) with respect to the jet axis.

1.1 The Current b-taggers

There are currently four tools certified at DØ to identify ('tag') whether a jet was produced from a b or not. Three of the methods use charged particle tracks and the fourth uses the presence of a muon.

Counting Signed Impact Parameters (CSIP)[1] - Counts the number of tracks identified in a jet (tracks are matched to jets if they have a $dR < 0.5$)[2] which have a large impact parameter (IP) significance with respect to the primary vertex. Events must have at least 3 tracks with an IP significance greater than 2 or 2 tracks with an IP significance greater than 3 to be considered tagged.

Jet LIfetime Probability Tagger (JLIP)[2] - Combines the impact parameter information from all the tracks identified in a jet into one variable the Jet Lifetime Probability (JLIP Prob). JLIP Prob is the probability that all tracks originate from the primary vertex. The closer to zero the more likely that the jet originated from a b. The current operating points for the tagger range from 0.002 (very tight) to 0.04 (super loose).

Secondary Vertex Tagging (SVT)[3] - Uses tracks which are significantly displaced from the primary vertex to reconstruct secondary vertices. A jet is considered tagged if a secondary vertex is located within $dR < 0.5$ of the jet.

Soft Lepton Muon Tagging (SLT)[4] - Uses the identification of a muon within a jet to tag the jet.

[1] Impact parameter is the distance of closest approach to the primary vertex.
[2] $dR = \sqrt{d\eta + d\phi}$

Each of these taggers calculates several variables which contain valuable information on the likelihood of the jet originating from a b-quark. Used individually these variables provide powerful discrimination between uds and b-jets. However combining the variables using a multi-variant technique has the potential to provide much more discrimination than each of the constituent parts can achieve. It is important to note that such a tagger achieves all its power from the hard work that has gone into the tagging tools which provide the input variables.

1.2 Neural Networks

A Root based Neural Network (NN) package, TMultiLayerPercptron[5], which was inspired by the mlpfit package[6] was used to construct the NN's. For the training stage the NN package takes as input a tree of the examples to be trained upon, which in this instance is jets.

A multilayer perceptron (MLP) NN is a simple feed-forward network which consists of a layer of input nodes, one or more layers of hidden nodes and one layer of output nodes. The nodes which are called neurons are connected to each of the nodes in the consecutive layers by links called synapses that have a weight w_j and bias w_0 representing the strength of the signal between the two nodes. The neuron j of the hidden or output layer computes a linear combination x_j of the neurons in the previous layer y_i with a bias.

$$x_j = w_{0j} + \sum_i w_{ij} y_i \tag{1}$$

The output y_j of the neuron j is then a function of the input x_j. The function is either linear

$$y_j = x_j \tag{2}$$

or a sigmoid function

$$y_j = \frac{1}{1 + e^{-x_j}} \tag{3}$$

depending on which layer is processing the input. The different layers carry out the following operations.

Input Nodes - Receives its input from the external sample which is scaled and output to the nodes in the first hidden layer.

Hidden Layers - Outputs a sigmoid function of a linear combination of the inputs from the nodes in the previous layer.

Output Layer - Creates a linear combination of the inputs from the nodes in the previous layer which is then an output of the NN.

A NN is a linear combination of sigmoid functions and constructed in such a way to take advantage of two very important theorems involving the computation of linear combinations of sigmoids.

1. A linear combination of sigmoid functions can approximate any continuous function[7].
2. When trained with a desired output of 1 for signal and 0 for background, the approximate function of the input is the probability of the signal knowing the input[8].

Initially the weights for each of the synapsis are set randomly between -0.5 and 0.5. The NN output o_p is compared to the desired output t_p on a set of examples p. The training algorithms try to minimise the error on the training samples by altering the weights. The total error E on the training samples (training error) is given by

$$E = \sum_p \frac{1}{2}\omega_p(o_p - t_p)^2 \tag{4}$$

where ω_p is an event weight. All the training algorithms compute the first order derivative of the error with respect to the weights

$$\frac{dE_p}{dw_{ij}} = \sum_p \frac{de_p}{dw_{ij}} \tag{5}$$

where p is the set of examples and e_p is the error on each example. This is called back-propagation of the errors. A loop over all the examples is called an epoch. There are six learning methods implemented. They are

Stochastic - Uses the Robbins-Monro stochastic approximation method to update the weights after each sample[9].

Batch - Is the same as the stochastic algorithm but the weights are only updated after considering all the samples.

Line Search Methods - The following algorithms all rely on the theory of unconstrained minimisation [10]. Each of these methods works in a similar manner. For a set of examples t:

1. A direction $\vec{s_t}$ is computed from the gradient ∇E.
2. The α which minimises $E(\vec{w_t} + \alpha\vec{s_t})$ is found (this part is called the line search).
3. The weights are updated by $\vec{w_{t+1}} = \vec{w_t} + \alpha\vec{s_t}$
4. Goto 1)

The variations of the line search method differ in the way that step 1) is carried out. The simplest is the **steepest descent**[10] algorithm where $\vec{s_t} = -\nabla(E)$, there are two conjugate gradient methods using the **Polak-Ribiere**[10] and **Fletcher-Reeves**[10] updating formulas and the **Broyden, Fletcher, Goldfarb, Shanno (BFGS)**[10] algorithm which relies on the computation of $N_{weights} \times N_{weights}$ matrices.

2 Monte Carlo Samples

The Monte Carlo samples used in the testing and training of the NN are outlined in table (1).

Sample	**Number of Events**
$t\bar{t}$	390,000
$Z \rightarrow b\bar{b}$	300,000
$Z \rightarrow c\bar{c}$	210,000
$Z \rightarrow q\bar{q}$	250,000
QCD $p_T = 20-40, 40-80, 80-160, 160-320$	1,160,000
QCD $b\bar{b}$ $p_T = 20-40, 40-80, 80-160, 160-320$	570,000
QCD $c\bar{c}$ $p_T = 20-40, 40-80, 80-160, 160-320$	620,000

Table 1: MC Samples used in training and testing of the NN.

2.1 Data Processing

All the samples were processed using D0Correct v8.2 which corrects the electromagnetic, muons, jets and missing E_T objects with the latest corrections and selection cuts[11].

When this work was originally undertaken the b-tagging values of interest were not easily accessible. To access the variables an altered version of *d0root_example*[12] was used with altered versions of the taggers which allowed all the variables of interest to be accessed and output into root-tuples. This has now been rectified, and all the variables are now accessible via the *btags_cert*[13] package. The package version used in the development of the NN tagger are outlined in table 2.

Package	Tag
bc_csiptagger	v00-00-12 (A)
btags_cert	v00-02-09
d0root_analysis	v00-09-68
d0root_btag	v00-09-80
d0root_csip	v00-00-12 (A)
d0root_jlip	v00-00-21
d0root_tmb	p16cert-br-02

Table 2: Package versions used in development of NN tagger. (A) signifies code was altered from CVS version.

2.2 Physic Objects Selection

Primary vertices were reconstructed using a 2-pass probabilistic primary vertex algorithm[14]. Events were required to have a primary vertex reconstructed from 4 or more tracks with a $|z| < 60cm$.

The calorimeter jets were reconstructed using the 0.5 cone algorithm (JCCB) with fake jets and electron candidates rejected. The standard jet energy scale (JES) correction (without muon corrections) was applied to the jets. All jets were required to be taggable. The standard p14 taggability definition was used, which defines a taggable jet as a good calorimeter jet which is matched to a track jet within a cone of $dR < 0.5$. The track jet must have two charged particles including one with a $p_T > 1.0GeV$.

Jets were identified as being b-jets if they had a b-hadron or b-meson within $dR < 0.5$ of the jet. c-jets had a c-hadron or meson but no b-hadron or b-meson present. uds-jets had no heavy flavour at all present in the event.

2.3 Jet Samples

The MC jet samples produced after data processing and application of the physics object and jet cuts are outlined in table 3. The different energy QCD samples were merged into continuous p_T samples by weighting the different energy samples via the fall off in their jet p_T spectrums.

Sample	Number of Jets
$t\bar{t} \rightarrow b$	600,000
$Z \rightarrow b\bar{b}$	240,000
$Z \rightarrow c\bar{c}$	260,000
$Z \rightarrow q\bar{q}$	375,000
QCD uds (Merged)	500,000
QCD $b\bar{b}$ (Merged)	270,000
QCD $c\bar{c}$ (Merged)	300,000

Table 3: MC jet samples used in the development of the NN tagger.

3 Variables

The following variables were identified as potentially good discriminators between uds-jets and b-jets. All the jets in the variable plots are uds-jets from the QCD uds MC samples and b-jets from the QCD $b\bar{b}$ MC sample.

3.1 Jet Variables

In order to construct a generic tagger which was independent of any particular jet reconstruction algorithm, jet reconstruction algorithm specific variables like jet p_T were avoided. Figure 1 shows the p_T and η of the NN training and testing sample.

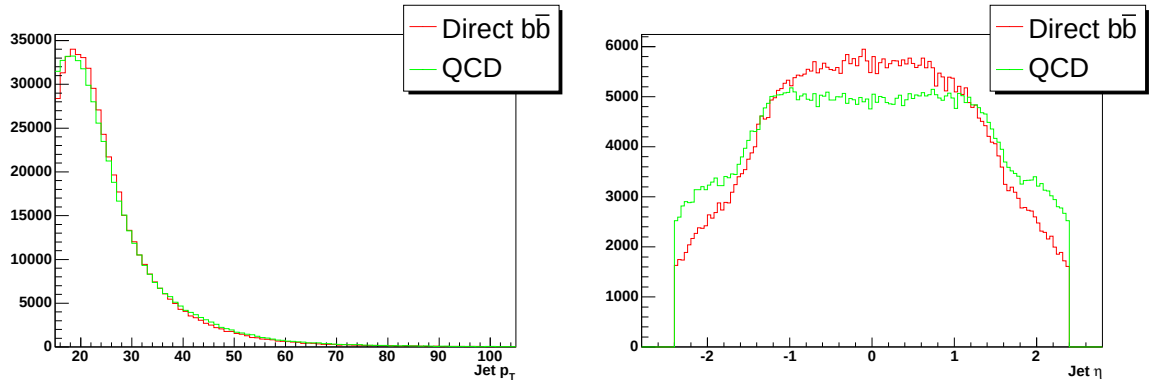

Figure 1: The p_T (left) and the η (right) of the training and testing samples.

3.2 Primary Vertex

The quality of the primary vertex (PV) is a very important indicator of the b-content of an event. The 2-pass probabilistic PV algorithm includes too many tracks originating

from displaced decays, resulting in PVs with can have larger errors when heavy flavour is present in an event. Variables which can be used to identify the quality of a PV are the χ^2_{dof} $(= \chi^2/(N_{Tracks} - 3))$ and the number of tracks (N_{Tracks}) which are shown in figure 2. However, for this version of the tagger the PV variables were not used in the NN due to possible, currently unstudied, correlations that may arise between jets when using a global event variable. However if these correlations can be shown to be small than these variables could be included in future versions of the NN tagger. Preliminary tests showed this to be a powerful variable.

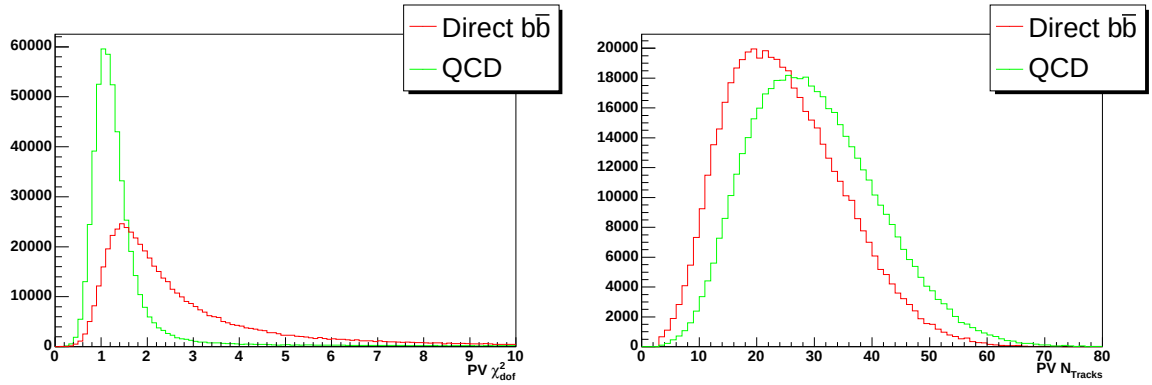

Figure 2: The χ^2_{dof} (left) and the N_{Tracks} (right) for the PV in the QCD $b\bar{b}$ and uds MC jets.

3.3 JLIP

The JLIP variables which were tested in the NN are listed below and are shown in figure 3.

JLIP *Prob* **-** The probability that the jet originated from the PV. The closer to 0 the more likely that it is a b-quark. If there was not enough information in the event to calculate a probability this value was set to 1.

JLIP $Prob_{Red}$ **-** The JLIP probability re-calculated with the most probable track removed from the calculation. This will identify jets where a miss-measured or fake track caused a miss tag of the jet. If there is not enough information to calculate the reduced probability the value was set to 1.

JLIP N_{Tracks} **-** The number of tracks used in the JLIP probability calculation. b-jets will typically have more tracks than uds-jets.

3.4 SVT

As multiple secondary vertices can be found in a jet, the secondary vertex with the highest decay length significance (DLS) (which is the most powerful SVT discriminator) was chosen

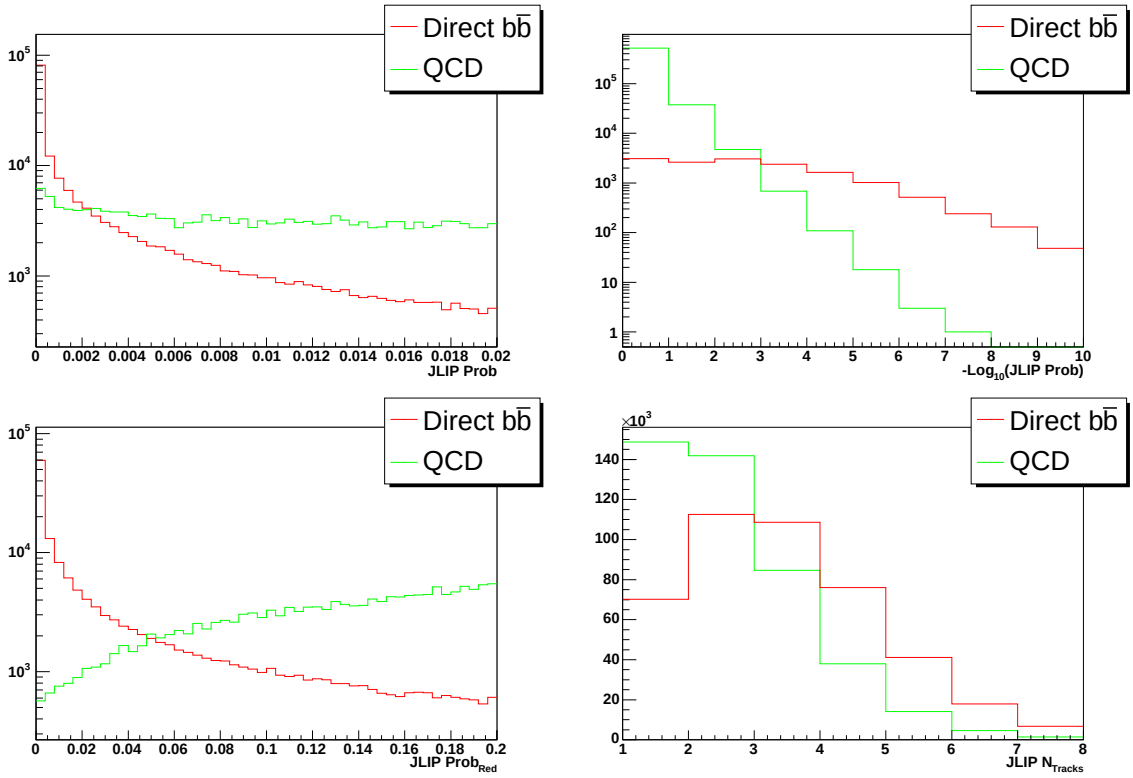

Figure 3: JLIP tagger variables Prob (top left), $-Log_{10}$(Prob) (top right), $Prob_{Red}$ (bottom left) and the N_{Tracks} (bottom right) in QCD $b\bar{b}$ and QCD uds MC jets, see the text for full definitions of variables.

to provide the input variables in the NN. The variables from the loose SVT (SVT_L) tagger which were tested in the NN are listed below and shown in figure 4. If no secondary vertex was found in the jet the SVT values were set to 0, apart from the SVT χ^2_{dof} which was set to 75 corresponding to the upper bound of χ^2_{dof} values.

SVT DLS - The decay length significance of the secondary vertex with respect to the primary vertex.

SVT χ^2_{dof} - The χ^2 per degree of freedom of the secondary vertex.

SVT N_{Tracks} - The number of tracks used to reconstruct the secondary vertex.

SVT $Mass$ - The p_T corrected mass of the secondary vertex. This is calculated from the combined rest mass of the tracks assuming all the tracks are pions, with the mass corrected for neutral particles.

SVT Num - The number of secondary vertices reconstructed in the jet.

3.5 Super Loose SVT

The current loose SVT tagger is not ideal for use in a multi-variant analysis. Although it finds a very pure selection of secondary vertices if it fails to reconstruct a secondary vertex no SVT information for the jet is available. For a NN it is better to have the maximal amount of information present and allow the NN to decide if the jet is signal or background. To increase the amount of information available a 'super loose' version of the SVT tagger was used (SVT_{SL}). The default values for the SVT_L tagger track selection and those used for the SVT_{SL} are shown in table 4. The variable plots for the SVT_{SL} are shown in figure 5. Although they don't show as much separation as the SVT_L tagger the extra information available per jet should more than counteract this effect. Figure 6 shows the efficiency of the SVT_L and SVT_{SL} taggers on the sample of QCD $b\bar{b}$ b-jets and the fake rate on the QCD uds-jet sample. Although the SVT_{SL} tagger is very impure, it provides information on $\sim 90\%$ of the b-jets compared to $\sim 65\%$ for the SVT_L tagger. This is very important as the SVT tagger provides by far the greatest number of variables and having information present for these variables is essential.

3.6 CSIP Variables

Potential variables from the loose CSIP tagger are listed below, and are shown in figure 7.

CSIP 3s - The number of tracks with a decay length significance greater than 3.

CSIP 2s - The number of tracks with a decay length significance greater than 2.

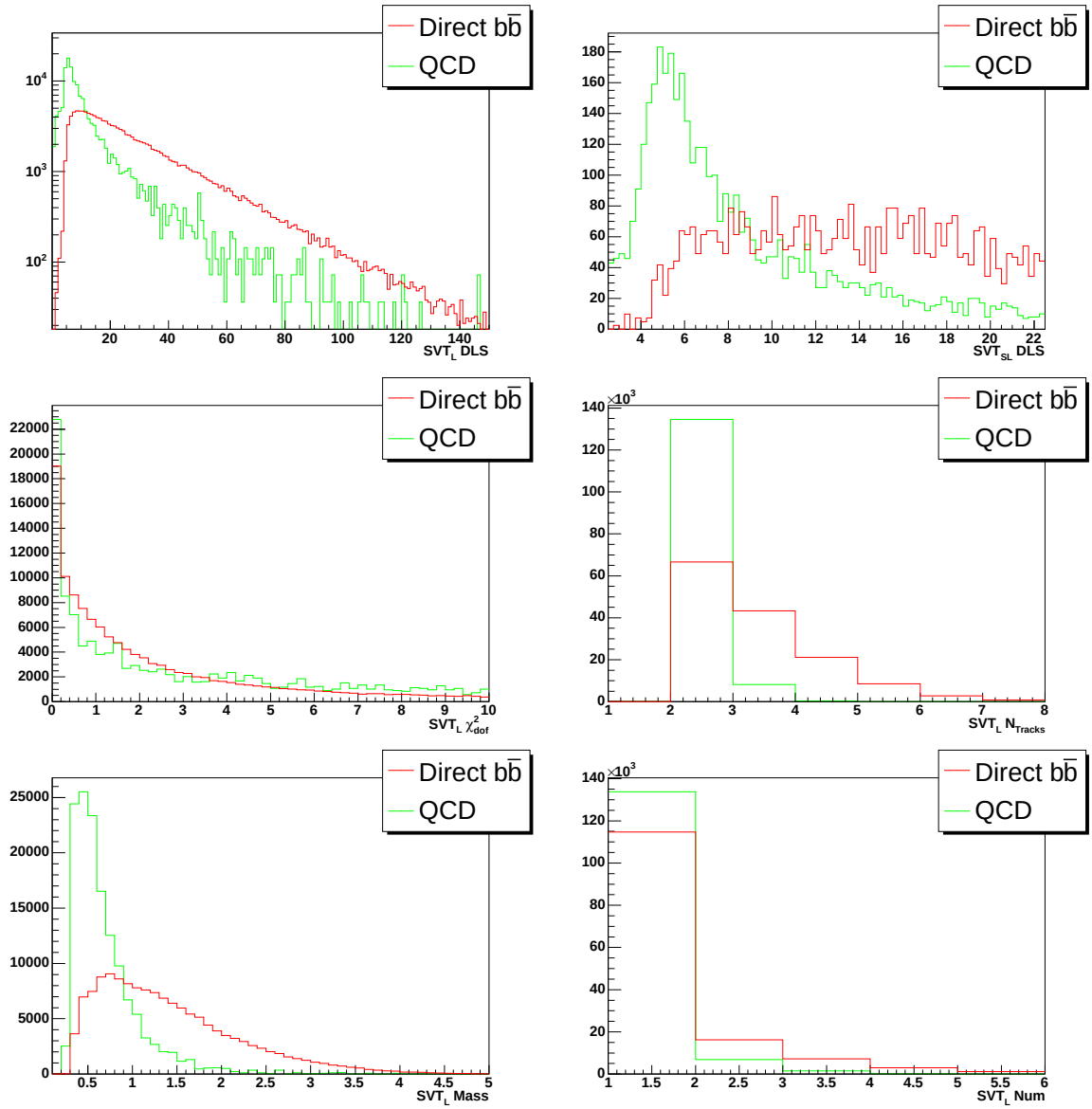

Figure 4: Variables for the loose SVT tagger, DLS (top left and right), χ^2_{dof} (middle left), N_{Tracks} (middle right), Mass (bottom left) and number of vertices (bottom right) in direct $b\bar{b}$ and QCD uds MC jets, see the text for a full description of the variables.

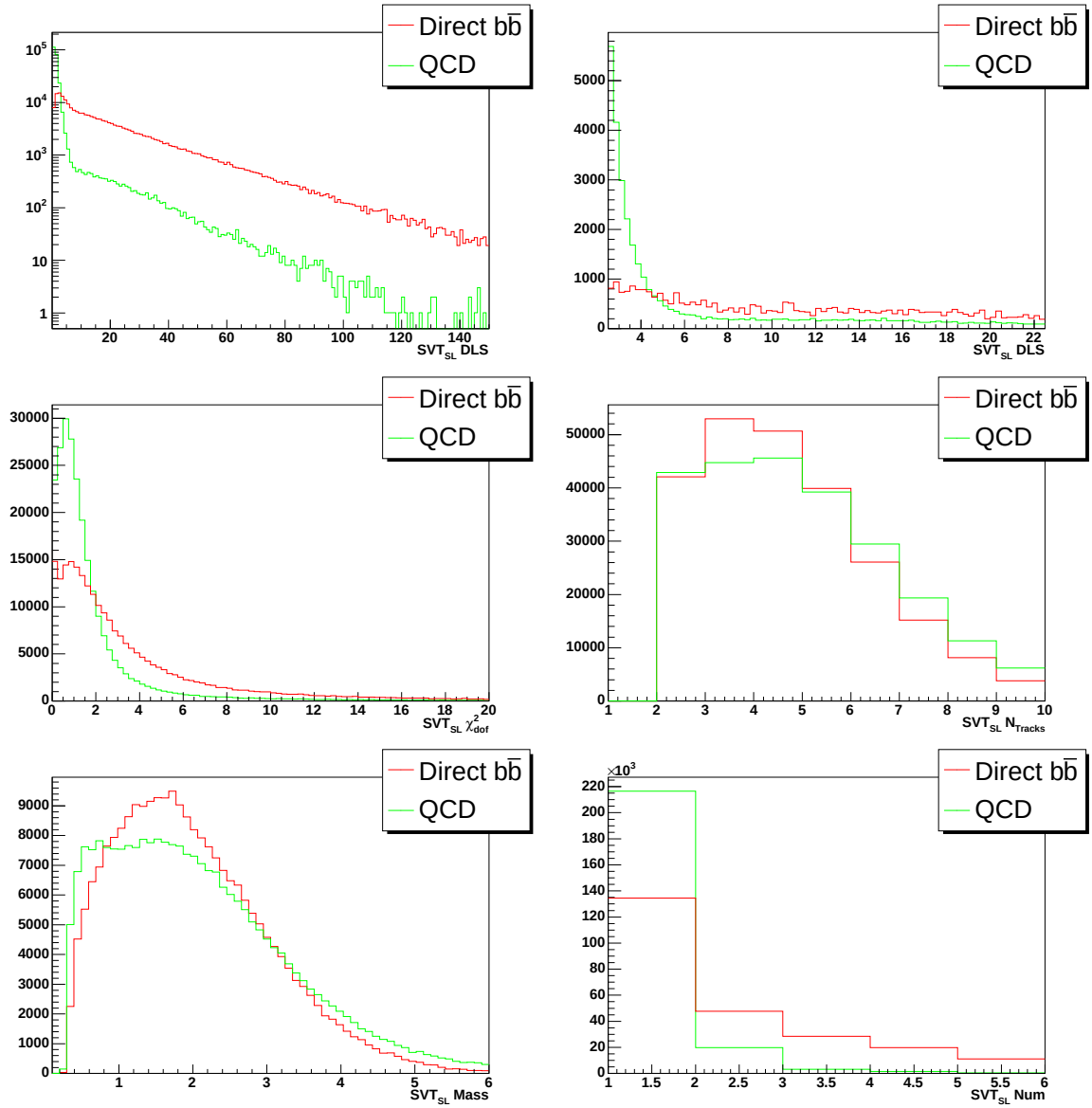

Figure 5: Variables for the SVT_{SL} tagger, DLS (top left and right), χ^2_{dof} (middle left), N_{Tracks} (middle right), Mass (bottom left) and number of vertices (bottom right) in direct $b\bar{b}$ and QCD uds MC jets, see the text for a full description of the variables.

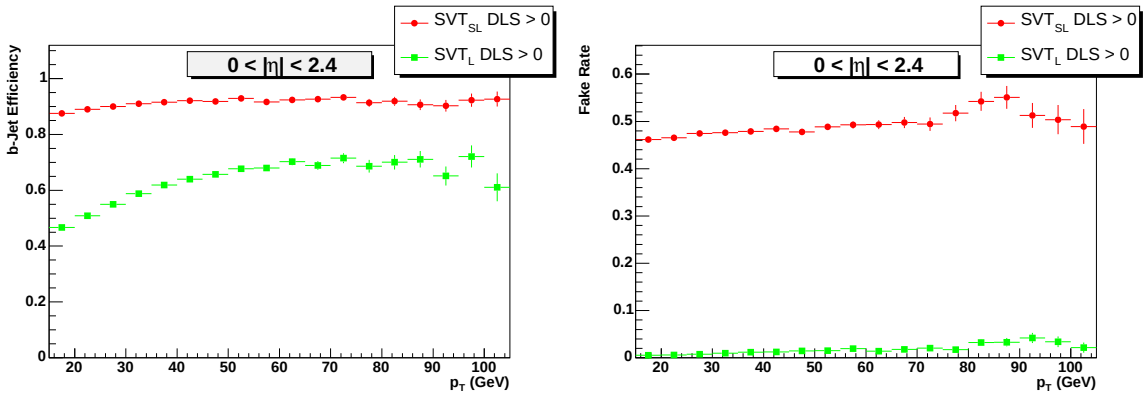

Figure 6: SVT_{SL} and SVT_L efficiencies in QCD $b\bar{b}$ and QCD uds MC jets.

Track Cuts	SVT_L	SVT_{SL}
χ^2	10	15
IP_{sig}	3	0.0
p_T	1.0	0.5
Num SMT Hits	2	2

Table 4: The track selection cuts used by the SVT_L and SVT_{SL} taggers. The tracks are selected by the χ^2, impact parameter significance (IP_{sig}), transverse momentum (p_T) and the number of SMT hits.

CSIP 3w - The number of tracks with a negative decay length significance greater than 3 and $d\phi < 1.15$ with respect to the jet.

CSIP 2w - The number of tracks with a negative decay length significance greater than 2 and $d\phi < 1.15$ with respect to the jet.

All the CSIP variables are small integer values which are not ideal inputs for a NN which works best with continuous values spread over a range. So the CSIP variables were used to create a single variable which was spread over a greater range. Replacing one variable with four is also advantageous as it reduces the number of variables and simplifies the NN. The weights were determined by using twice the tracks significance with the negative significance tracks being weighted by half that value. Several other weighting techniques were tested with this being found to be the most effective.

$$CSIP\ Comb = 6 \times 3s + 4 \times 2s + 3 \times 3w + 2 \times 2w \tag{6}$$

3.7 SLT

No input from the SLT was used in the NN. The reasons for this were twofold. Firstly the SLT is used in the System8 efficiency measurement to measure the taggers efficiency on real data[15]. The System8 measurement requires two taggers which are uncorrelated and using the SLT as a NN input would correlate the output of the two taggers. Secondly, from a performance perspective the SLT returns no output for the majority of events and a variable which is zero for the majority of events has very little benefit.

4 Analysis Method

Six attributes need studying to produce an optimised NN:

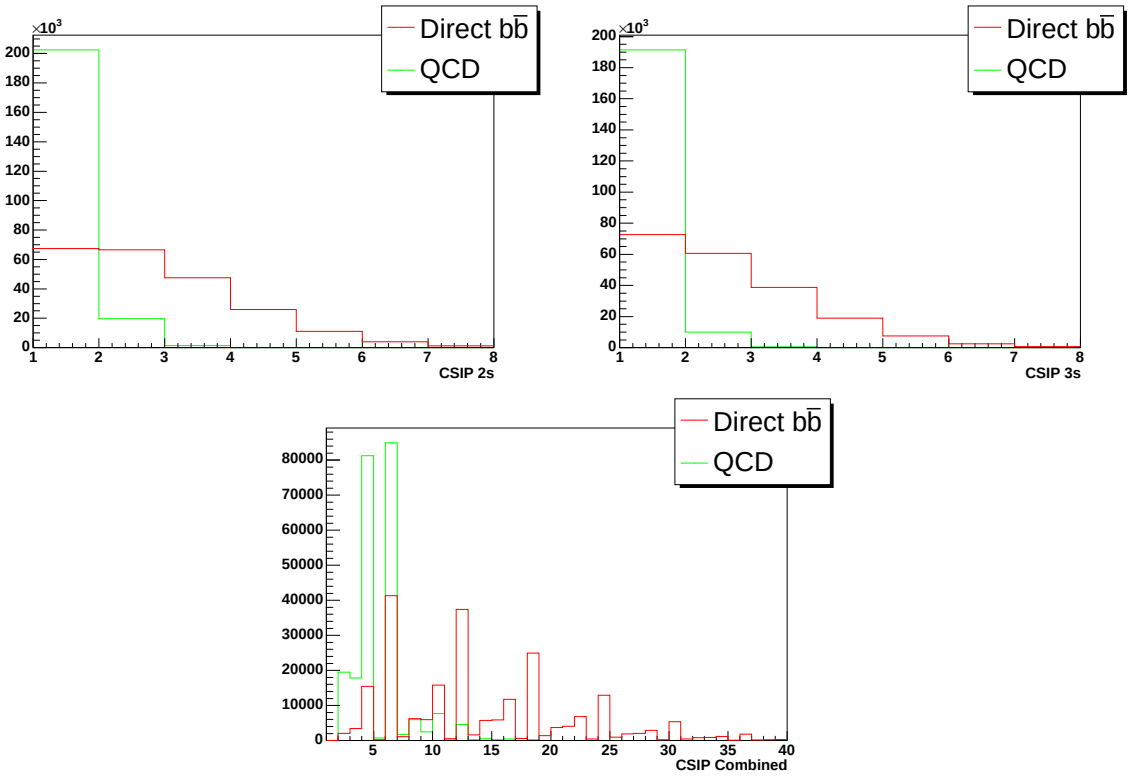

Figure 7: Variables from the loose CSIP tagger, the number of tracks with a significance of 2 (top left), the number of tracks with a significance of 3 (top right) and the combination variable (bottom) in QCD $b\bar{b}$ and *uds* MC jets. See the text for a full description of the variables.

1. **Variables** - How many input variables and which input variables to use.
2. **Structure** - The number of nodes in the hidden layer and the number of hidden layers.
3. **Training Algorithm** - Using the wrong algorithm could lead to the best function not being found or excessive CPU usage to find the optimal solution.
4. **Training Samples** - The choice of samples to train the NN with, this choice encompasses such sample features as p_T spectrum, b-production method and the generator used to produce the MC.
5. **Number of Training Epochs** - Too many could lead to over training on a sample, too few could result in the optimal function not being found.
6. **Jet Cuts** - The selection of events to input into the NN.

The most important attribute is the input variables, which was optimised first by selecting interim values for the other attributes as discussed below. The other values were then re-optimised after the selection of the input variables.

4.1 Performance Benchmark

A benchmark was established to rank the performance of the NNs. The NN is designed to maximise the b-efficiency for a give fake rate, so with this in mind, the benchmark was selected as the fake rate achieved for a fixed b-efficiency. For all the following studies the performance of the NN was determined by the fake rate at four fixed b-efficiencies 75%, 70%, 60 % and 50%.

4.2 Variable Selection Method

The most time consuming part in constructing a NN is selecting the input variables. The greater the number of variables the greater the amount of information and therefore the better the discrimination, but for simplicity's sake it is desirable to have the minimum number of variables possible. Another problem is the selection of which variables to use together. Although variables on their own might not have much effect, in combination with another variable a powerful pattern could emerge.

It would be very time consuming to test all the possible combination of variables. If we were to test all the different possible NNs for the 14 variables identified above it would take the construction of 16,368 different networks to identify the optimal solution. This is obviously impractical. One solution to this problem is to individually identify the most powerful variables. If we have an initial NN with n variables and a list of m variables we want to rank in order of power, the following procedure can be carried out to identify the most powerful variables.

1. Individually add each of the m variables to be tested to the initial n variable NN so we have m NNs each with $n+1$ variables.

2. Identify the variable whose addition improved the NN's performance the most, and add this variable permanently to the n variable NN.

3. Goto step 1) and and test each of the remaining $m-1$ variables with the new $n+1$ variable NN.

To identify the most powerful variables the above procedure was carried out. An initial two variable NN was identified by testing every combination of variables in a two input NN. The remaining variables were then ranked in order of power by examining the fake rate at the 70% b-efficiency operating point.

Although this method has its limitations, such as possibly missing an optimal combination of variables that did not have any discrimination when added separately, it is a very effective way of systematically identifying the best variables and the optimal number to use for a specific operating point.

4.3 Training Algorithms

Figure 8 shows the comparative training curves for the six different training algorithms for a simple NN consisting of JLIP Prob, SVT DLS, CSIP Comb and SVT χ^2_{dof} for a fixed amount of CPU processing time. The BFGS training algorithm appears to be the most consistent and systematic method to minimise the error on the fit.

The stochastic training algorithm which is seen to produce the very inconsistent and spiky result (which is a consequence of the its training algorithm), does actually produce a minimum error value in less CPU time. However, the BFGS algorithm was chosen as the Stochastic algorithm is unreliable and doesn't consistently improve the weights to lower error values when it finds a minimum.

4.4 Number of training epochs

Due to the large number of networks that required training the number of training epochs was selected to ensure that an optimal solution was achieved whilst minimising the amount of CPU time used. After several tests the number of training epochs was set to 500, which as shown in figure 8 is sufficient for the BFGS algorithm to find a stable solution.

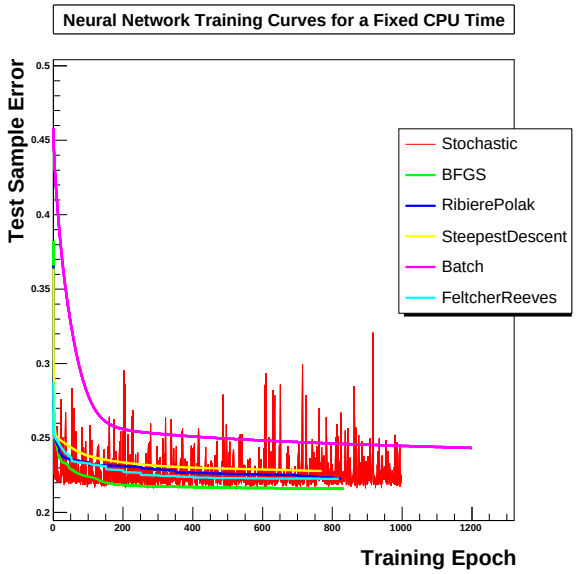

Figure 8: Training curves for the various training algorithms for a fixed amount of CPU time.

4.5 Training Samples

Trees of 270,000 signal (QCD $b\bar{b}$) and 470,000 background (QCD uds) sample jets, which have approximately the same p_T spectrum, weighted to be of equal number were used in the training and testing of the NN. The signal and background samples were split in half, with half used for the testing sample and the other half for the training sample.

4.6 Neural Net Structure

The basic neural network structure is $N_{input} : N_{hidden} : N_{output}$. The number of output nodes in this case is 1 and the number of input nodes is equal to the number of variables being tested. The number of hidden layers was chosen to be one, as one layer of sigmoid functions should be sufficient to model any function, and it is advantageous to keep the NN as simple as possible. The number of hidden nodes in this layer was initially set to twice the number of input nodes.

4.7 Jet Selection

One of the most important aspects of the NN is the selection of jets which the NN will be trained to separate. Too loose a selection will mean a loss of performance as the NN is separating signal from background which could have been done with a simple cut. However, too stringent a selection will cause higher fake rates at high efficiencies and limit the upper efficiency that can be achieved.

The jet selection was chosen to try and ensure the maximal number of b-jets whilst minimising the number of fake jets. The most powerful variable from each of the taggers was used to select the input jets, the JLIP Prob, SVT_{SL} DLS and CSIP Comb. Initial cuts were selected from inspection of the variables shown in figure 9. Jets were input into the NN which had JLIP Prob < 0.04, CSIP COMB > 8 or SVT_{SL} DLS > 2. This selection leaves 40,000 uds-jets and 225,000 b-jets for training.

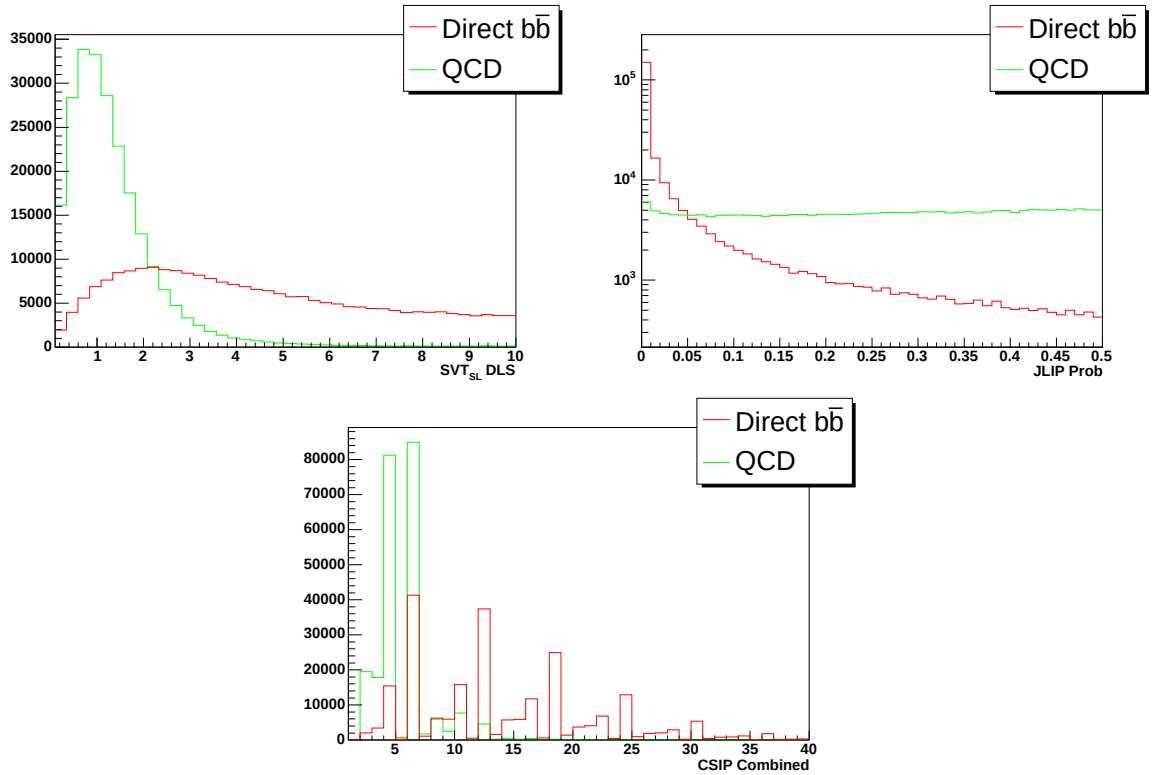

Figure 9: Distribution of the input selection parameters SVT DLS (top left), JLIP Prob (top right) and CSIP Comb (bottom) for QCD $b\bar{b}$ and QCD uds jets.

The efficiency for this selection on b-jets compared to requiring a loose JLIP, SVT or CSIP tag is shown in figure 10. The chosen criteria is demonstrated to have an efficiency of $\sim 90\%$ whereas the OR of the loose taggers has only a $\sim 80\%$ efficiency. This higher input b-efficiency of course comes at the cost of a higher input of uds-jets with a fake rate of 5 times the OR tagging input. However the higher uds efficiency can be advantages as it results in a larger sample of background jets for the NN to be trained upon.

5 Optimisation

So as not to bias the optimisation towards one particular MC b-sample all the following optimisations (except the variable optimisation) were carried out by training on the Pythia QCD $b\bar{b}$ MC and using the performance on the Alpgen $t\bar{t}$ MC to select the optimal operating point.

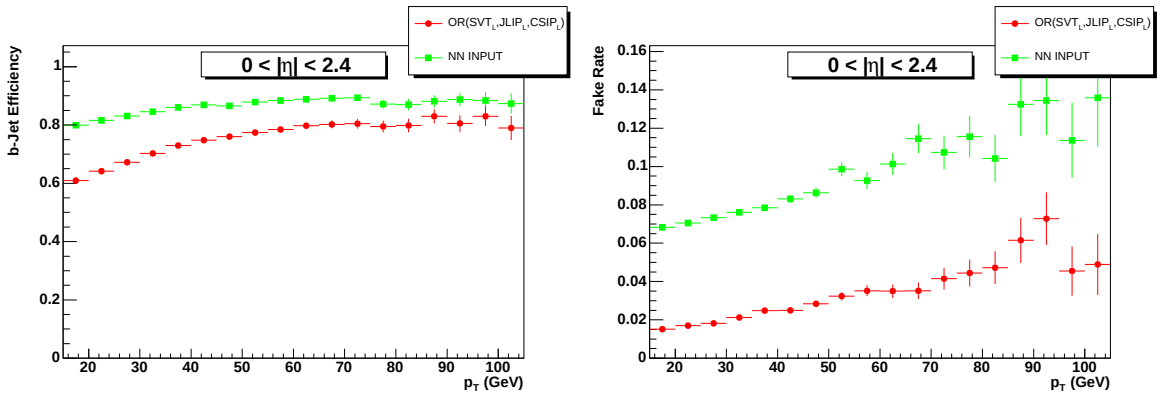

Figure 10: The b-efficiency (left) and fake rate (right) of the NN input criteria (specified in text) and an OR of the SVT, CSIP and JLIP taggers. The NN input has a much higher b-jet efficiency at the expense of much higher fake rate.

5.1 Variable Optimisation

The best variables were identified by selecting the variables which when added to a NN gave the greatest fake rate reduction for a signal efficiency of 70%. This method actually produced the same results as optimising for the signal efficiencies of 50%, 60% and 75%. The variables identified in order of power are shown in table 5. The variable optimisation results are shown in figure 11.

Rank	**Variable**
1	SVT_{SL} DLS
2	CSIP Comb
3	JLIP Prob
4	SVT_{SL} χ^2_{dof}
5	SVT_L N_{Tracks}
6	SVT_{SL} Mass
7	SVT_{SL} Num
8	JLIP $Prob_{red}$
9	SVT_{SL} dR

Table 5: NN nput variables ranked in order of power.

The curves in figures 11 highlight three potentially interesting NNs. The NN with 5 variables benefits from the majority of the improvements in performance and has the advantage of being the simplest. However, the NNs with 7 and 9 variables do show better performance but at the price of an increased number of variables.

Figure 12 shows the performance curves for the NN taggers using the 5, 7 and 9 variables identified above. The performance of the taggers shows a definite improvement when using

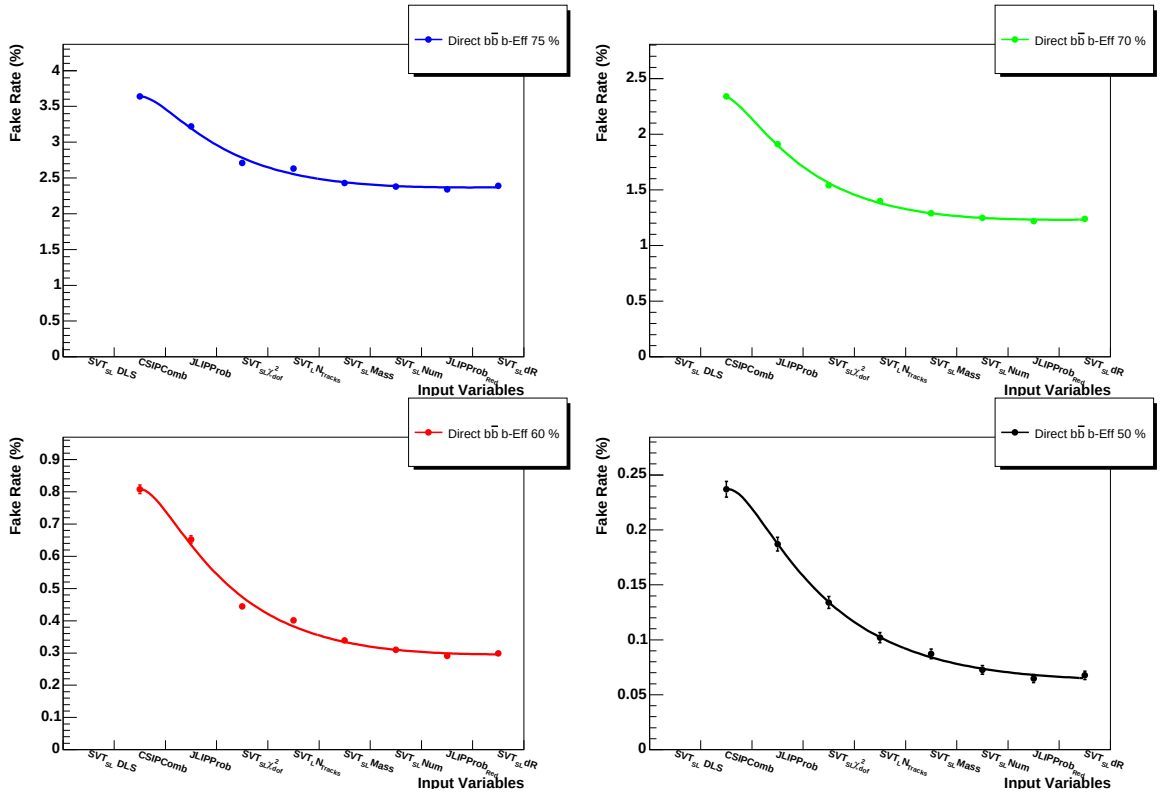

Figure 11: Fake rate for fixed signal efficiencies of 75% (top left), 70% (top right), 60% (bottom left), 50% (bottom right) as a function of additional NN variables. The NN variables were added to the NN in order of performance. The fit is intended to guide the eye only.

more than 5 variables. However using more then 7 variables has no increase in performance. In conclusion the 7 variable NN is identified as the optimal solution. The variables are SVT_{SL} DLS, CSIP Comb, JLIP Prob, $SVT_{SL}\chi^2_{dof}$, SVT_L N_{Tracks}, SVT_{SL} Mass, SVT_{SL} Num.

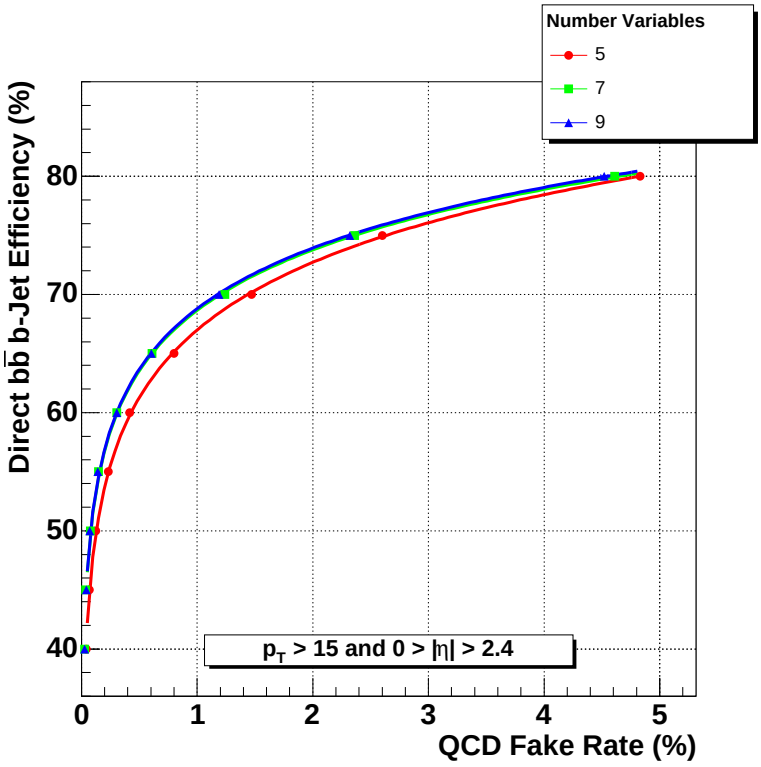

Figure 12: Performance curves for the 5, 7 and 9 variable NNs. The 7 and 9 variable NNs are shown to have better performance than the 5 variable NN, although there is no improvement in going from 7 to 9 variables.

5.2 Number of Training Epochs

The number of training epochs in the variable optimisation was set at 500. This was varied from 50 up to 2000 training epochs for training on the direct $b\bar{b}$ MC sample and tested on the aplgen $t\bar{t}$ MC sample. The results of which can be seen in figure 13. For each of the operating points the majority of the minimisation has been reached by $\sim$ 200 epochs, with additional small improvement seen thereafter. To ensure optimised NNs the number of training epochs was set to 1000.

5.3 Neural Network Structure

In the optimisation of the NN the assumption was made that the optimal number of hidden nodes was 2N, where N is the number of input nodes. To check this hypothesis the number of hidden nodes was varied from 7 through to 34. Figure 14 shows the effect on the fake

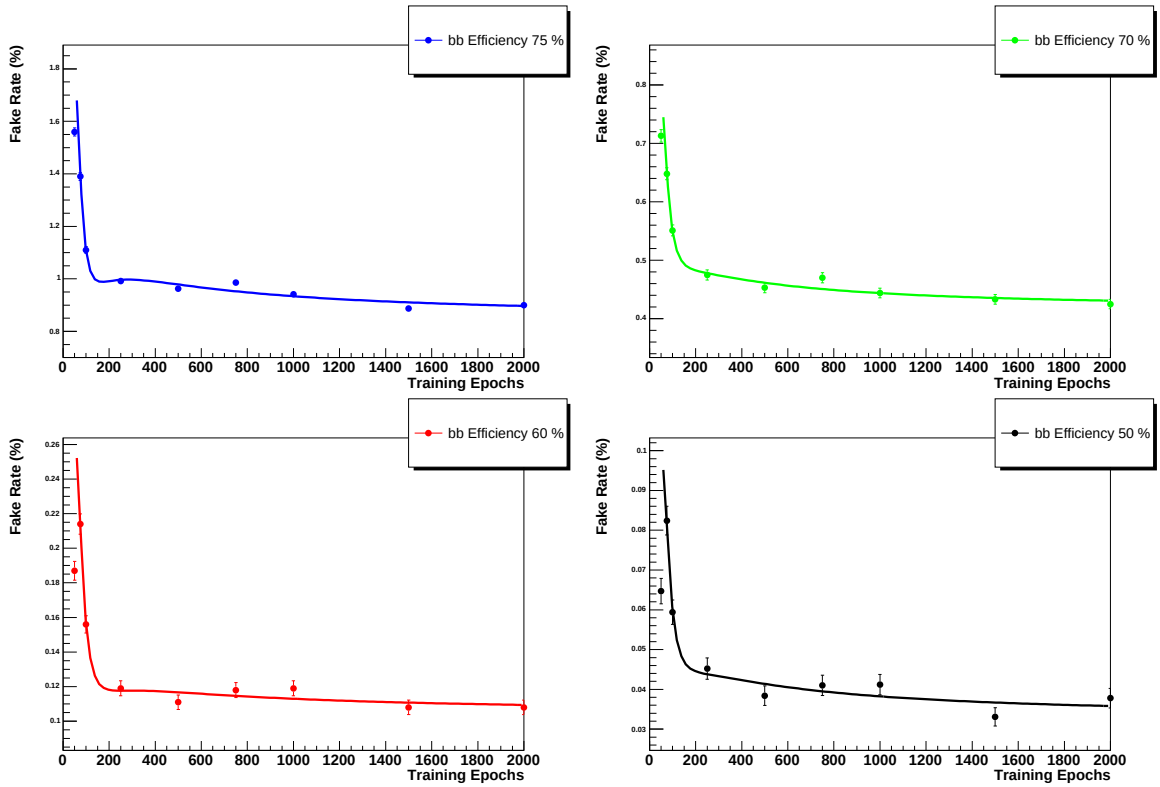

Figure 13: Fake rate for fixed signal efficiencies of 75% (top left), 70% (top right), 60% (bottom left), 50% (bottom right) as a function of number of training epochs. The majority of the minimisation is achieved by 200 epochs for each of the operating points with small increases in performance seen thereafter. The fit is intended to guide the eye only.

rate for fixed signal efficiencies. Each of the operating points shows a minimum fake rate for 24 hidden nodes, which was chosen as the optimal number.

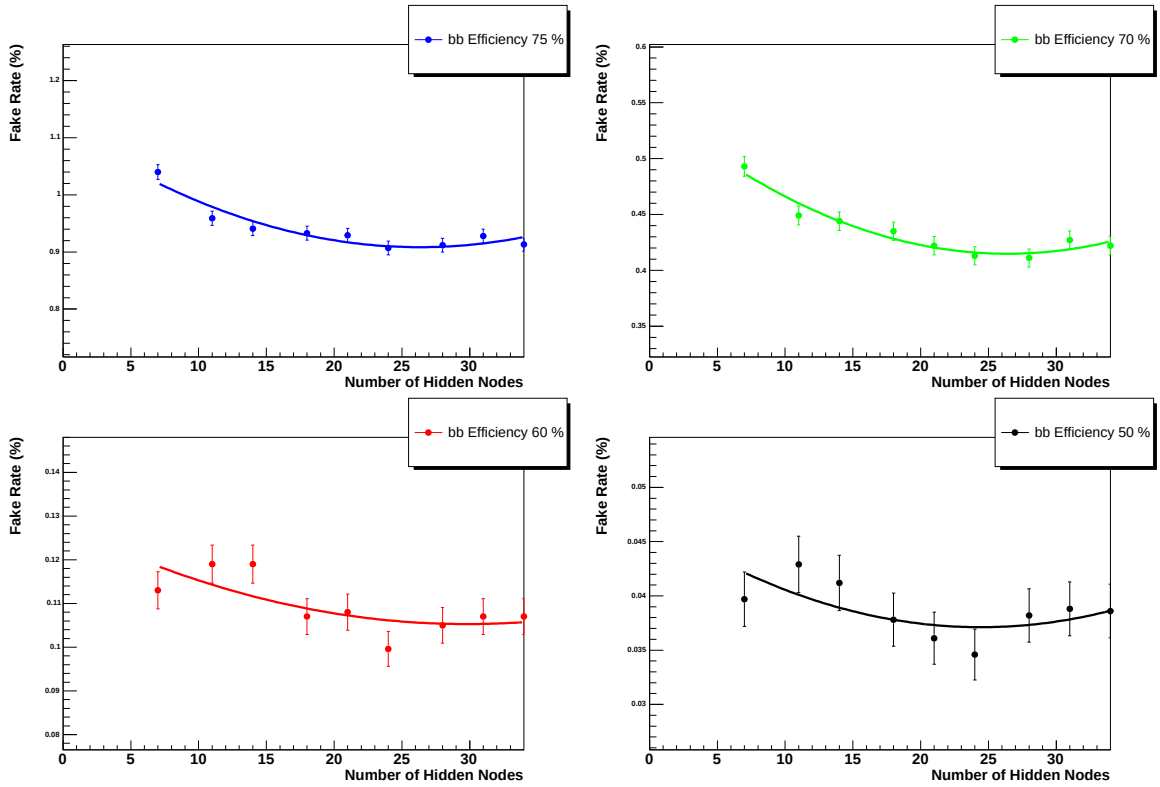

Figure 14: Fake rate for fixed signal efficiencies of 75% (top left), 70% (top right), 60% (bottom left), 50% (bottom right) as a function of the number of hidden nodes in the single hidden layer. A minimum can be seen at 24 hidden nodes for each of the operating points. The fits are designed to guide the eye only.

It should be possible to replicate any continuous function with only one hidden layer. However, the NN output shown in figure 15 is not constrained between 0 and 1. This demonstrates that a NN consisting of one hidden layer will not always produce a function between 0 and 1.

This occurs as the NN provides an approximated fit of the function and so this fit will not necessarily be constrained between 0 and 1. Also, in the MLP implementation of a NN, the output node is not a sigmoid function so the approximated fit will not be normalised by the output node. The consequence of this is a single layer NN may not produce a perfectly normalised function between 0 and 1.

This can be solved by adding an extra hidden layer with 1 node. This extra sigmoid should not change the performance of the NN but should normalise the output to ensure that it is always between 0 and 1. Figure 15 shows the output of such a NN, which retains the same shape in the regions > 0 and < 1 but with all the outlying points collected into the peaks at 0 and 1. The performance of this NN compared to the single later NN is shown in figure 16. There is a very slight improvement in the performance, but the two NNs basically have the same performance.

The single hidden layer of 24 nodes has achieved the optimal fit and the extra single layer has just normalised this fit correctly between 0 and 1. This is corroborated by the fact that adding more nodes into the second hidden layer does not improve the performance at all as shown in figure 16.

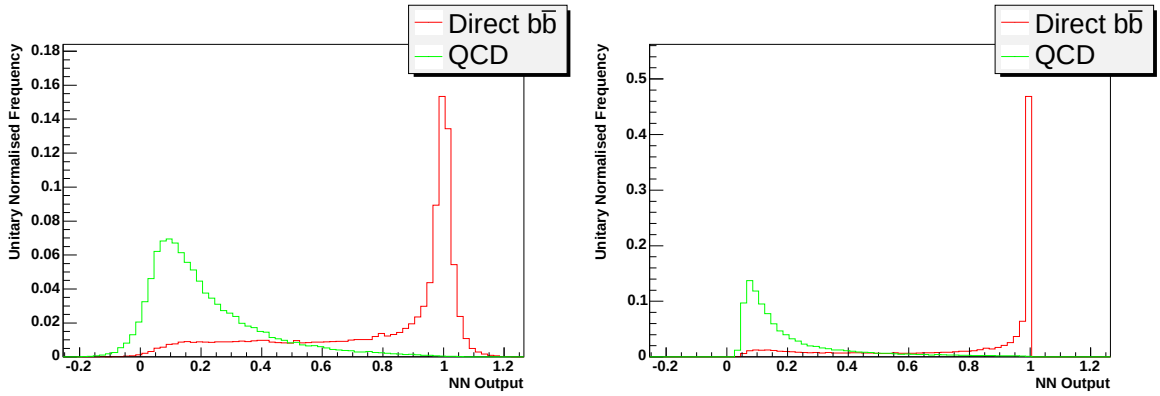

Figure 15: The NN output for the case of a single hidden layer 1:24:1 (left) and double hidden layer 1:24:1:1 (right) NN with one sigmoid in the extra hidden layer. The extra hidden layer constrains the NN between 0 and 1.

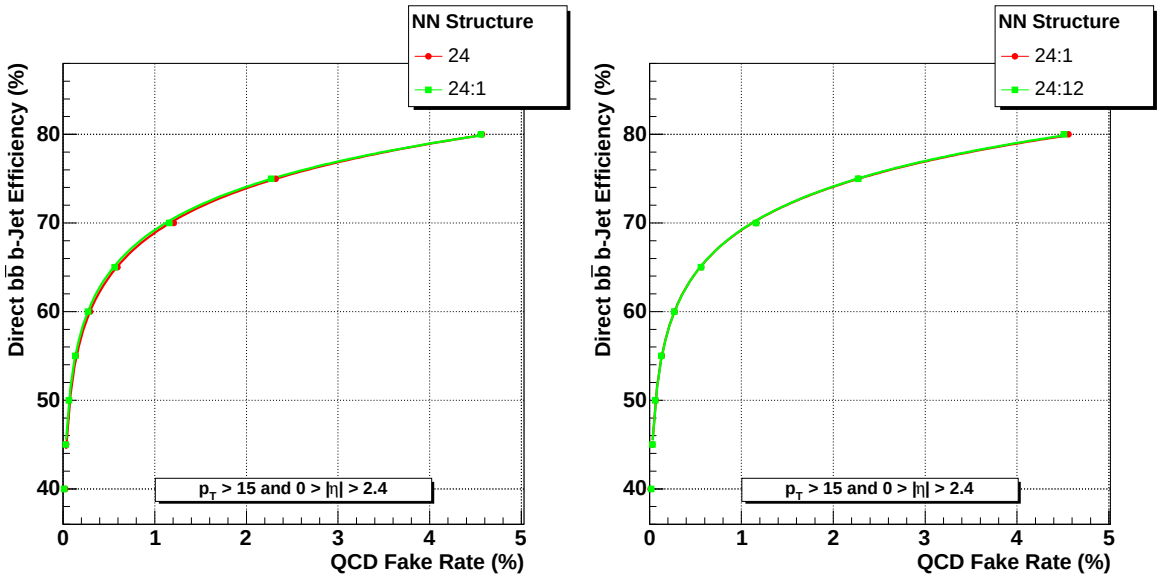

Figure 16: Performance comparison between a single hidden layer 1:24:1 and a double hidden layer with one extra sigmoid 1:24:1:1 NN (left) and a comparison between two NNs, one with a simple extra hidden layer and one with a more complex hidden layer (right).

5.4 Input Selection Cuts

The input selection cuts were optimised by considering each variable in turn, optimising the variables in order of SVT_{sl} DLS, JLIP Prob and CSIP Comb. The NNs were trained on the QCD $b\bar{b}$ and QCD pythia samples and the optimisation plots were produced from the high p_T Alpgen $t\bar{t}$ sample and cross checked with the direct $b\bar{b}$ sample to ensure there was no

sample, p_T or MC generator dependence in the optimisation. The background efficiencies for the operating points of 75%, 70%, 60% and 50% were used to identify the optimal cuts.

The SVT_{SL} DLS is the most important variable as it resides in a highly populated region. Small variations can make a large difference to the number of b and uds-jets input into the NN. The SVT_{SL} DLS cut was varied from 0.5 to 4.0. Figure 17 shows the effect on the fake rate as a function of the SVT_{SL} DLS cut. A cut value of 2.5 was chosen as it benefits from the vast majority of the gains in tightening this cut whilst being slightly conservative on the value due to the importance of this cut.

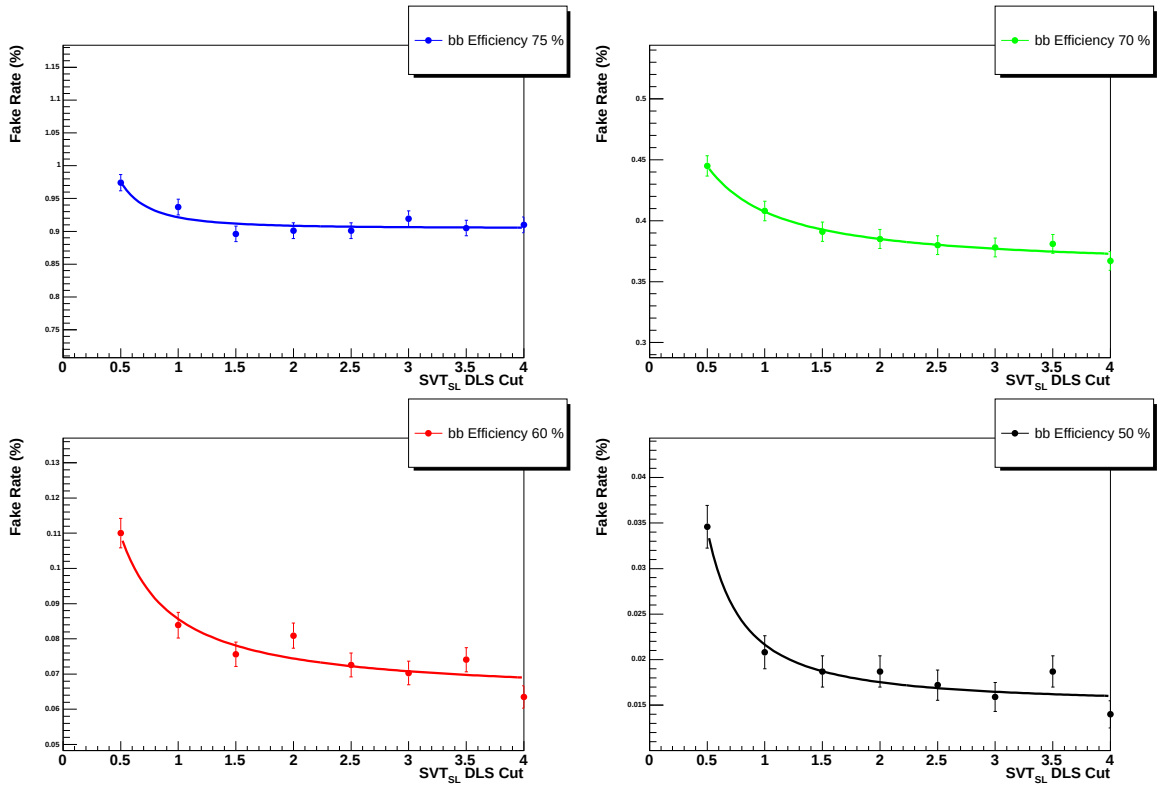

Figure 17: Fake rate for fixed signal efficiencies of 75% (top left), 70% (top right), 60% (bottom left), 50% (bottom right) as a function of the SVT_{SL} DLS cut on input jets. A conservative cut value of 2.5 benefits from the majority of the gain. The fits are designed to guide the eye only.

The JLIP Prob cut was altered from 0.005 to 0.4. The resulting background efficiencies are shown in figure 18. The optimal operating point appears to be 0.02, which substantially lowers the fake rate at the 50% and 60% operating points whilst not increasing the fake rates at higher efficiencies.

The CSIP Comb cut was varied between 2 and 14 and is shown in figure 19. Varying the cut value does not have much effect on the fake rate although for the operating points of 50 and 60% a minimum around 8 is seen

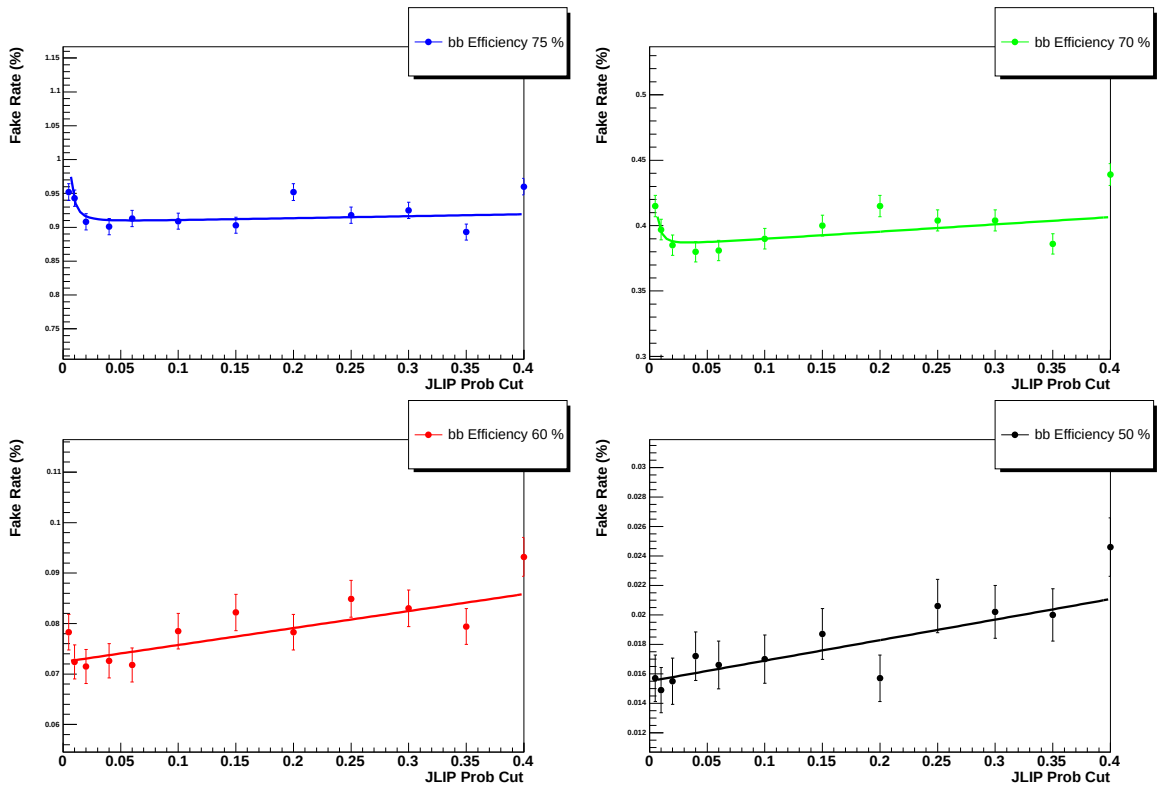

Figure 18: Fake rate for fixed signal efficiencies of 75% (top left), 70% (top right), 60% (bottom left), 50% (bottom right) as a function of the JLIP Prob cut on input jets. A cut value of 0.02 minimises the fake rate for the 50% and 60% operating points, whilst maintaining the fake rate for the 70% and 80% points. The fits are designed to guide the eye only.

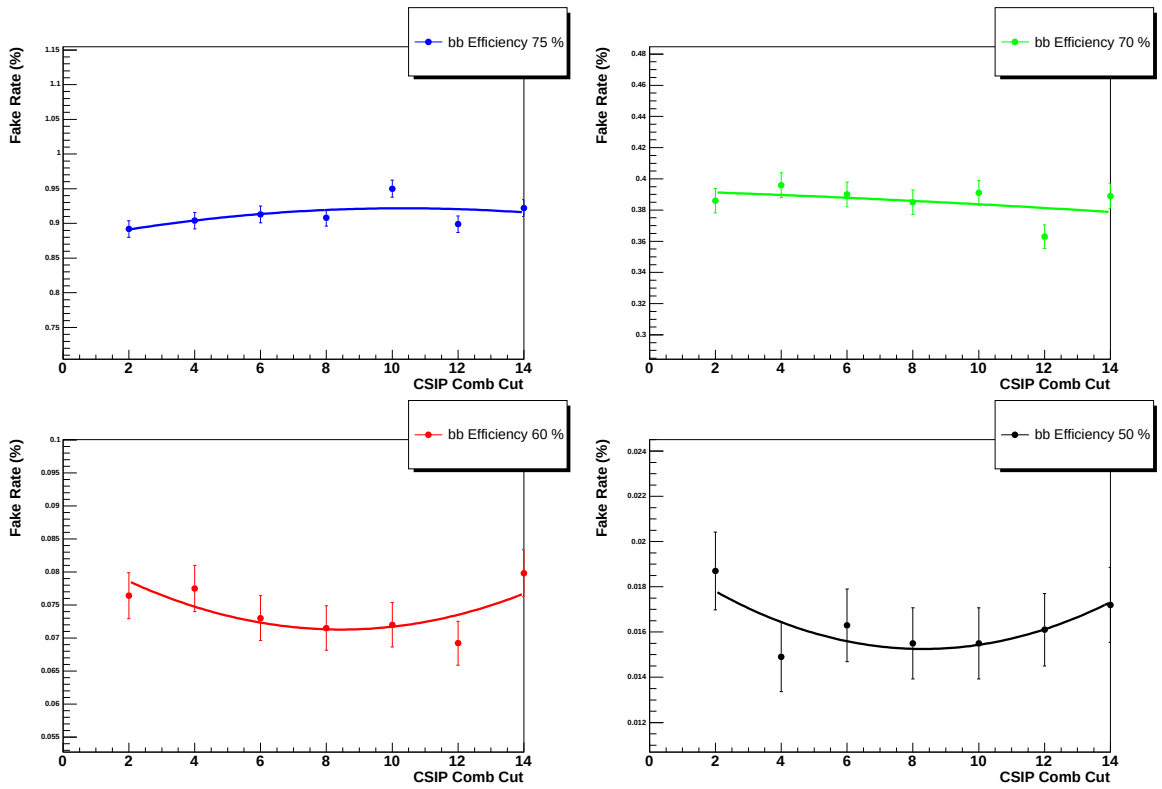

Figure 19: Fake rate for fixed signal efficiencies of 75% (top left), 70% (top right), 60% (bottom left), 50% (bottom right) as a function of the CSIP Comb cut on input jets. Varying the cut value does not have much effect on the fake rate although for the operating points of 50% and 60% a minimum around 8 is seen. The fits are designed to guide the eye only.

5.5 Optimised NN Parameters

The optimised parameter values for the NN tagger are shown in table 6.

Parameter	**Value**
NN structure	7:24:1:1
Input Variables (Ranked)	(1) SVT_{SL} DLS (2) CSIP Comb (3) JLIP Prob (4) SVT_{SL} χ^2_{dof} (5) SVT_{SL} N_{Num} (6) SVT_L N_{Tracks} (7) SVT_{SL} Num
Jet input selection cuts (failure results in NN output of 0)	SVT_{SL} DLS > 2.5 or JLIP Prob < 0.02 or CSIP Comb > 8
Number of training epochs	1000
Number of b-jets used in training (after jet selection cuts)	270,000 (220,000)
Number of fake jets used in training (after jet selection cuts)	470,000 (40,000)

Table 6: NN parameters.

6 NN Performance

The output from the optimised NN b-tagger on various signal and background samples is shown in figure 20, with the comparative signal and background output shown in figure 21. The NN b-tagger shows significant separation between signal and background samples and demonstrates similar performance on the different samples tested. Any differences can be attributed to the difference in p_T spectrums of the samples. The performance on c-jets is in line with the performance of other taggers, with a value of $\sim 20\%$ of the b-jet efficiency.

The performance of the loose version of the NN tagger which has a background rate of $\sim 1\%$ verses p_T and η for various samples is shown in figure 22. The NN tagger demonstrates similar performance on each of the samples and achieves an efficiency of $\sim 70\%$ for a fake rate of $\sim 1\%$.

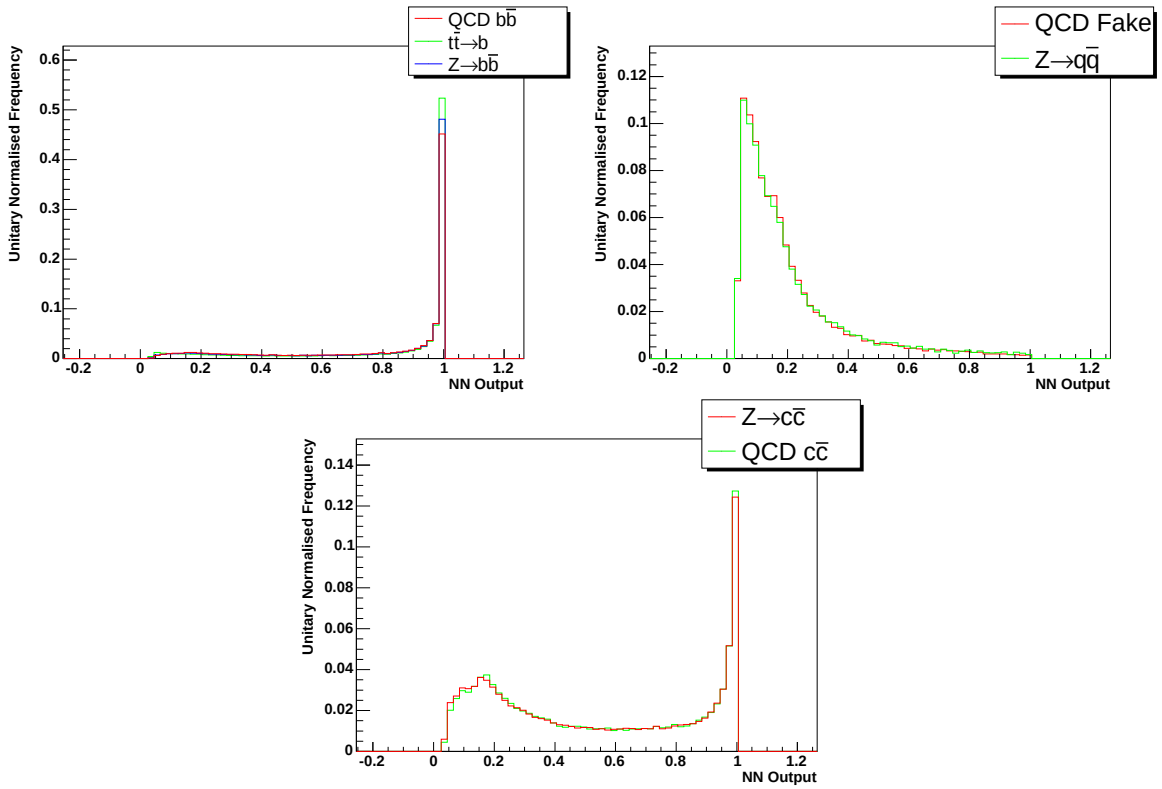

Figure 20: NN output for various b-jet (top left), uds-jet (top right) and c-jet (bottom) samples.

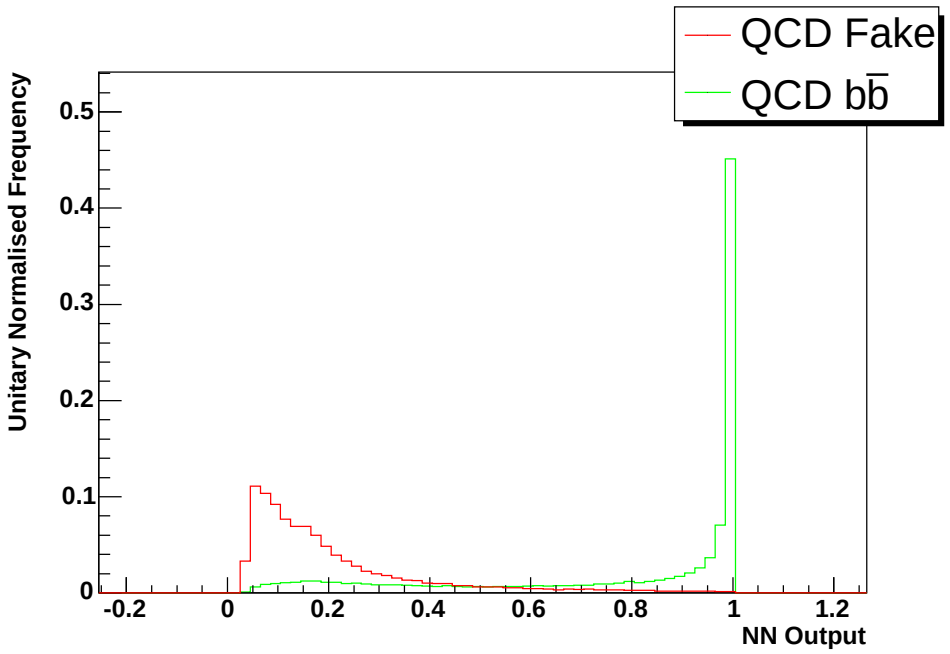

Figure 21: Comparative NN output for the QCD b-jet and QCD uds-jet sample.

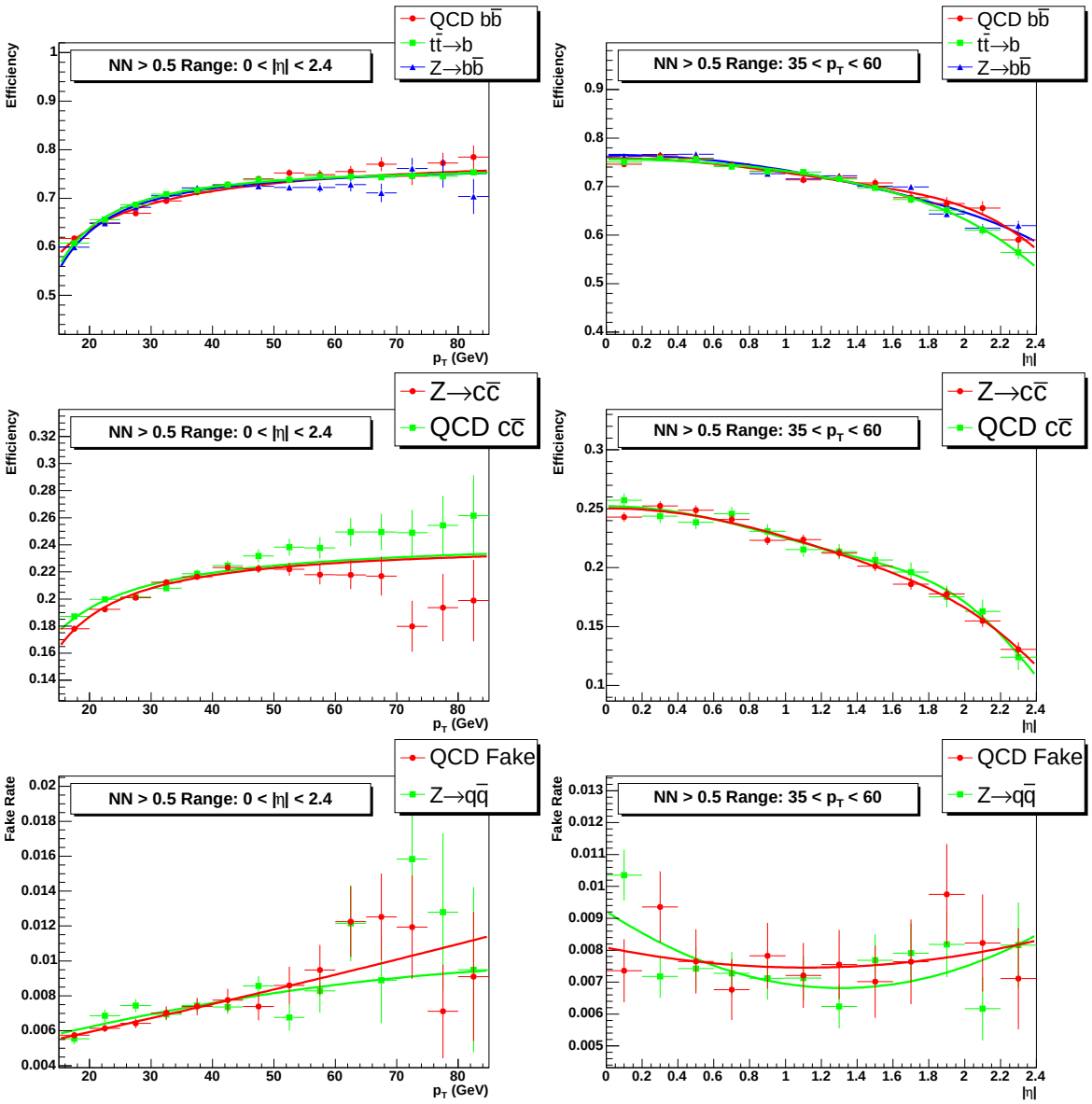

Figure 22: NN efficiency and fake rates for various b-jet (top), c-jet (middle) and uds-jet (bottom) samples in p_T (left) and η (right) projections. The fits are intended to guide to eye only.

6.1 Tagger Comparison

The NN tagger performance curve compared to the JLIP tagger are shown in figure 23. A substantial improvement is demonstrated over the JLIP tagger with efficiencies for a fixed fake rate increasing by $\sim 50\%$ at a fake rate of 0.2% to $\sim 15\%$ at a fake rate of 4.0%. For fixed signal efficiencies we see a reduction in the fake rate to 20% of its initial value.

The performance of the NN tagger compared to the SVT, CSIP and JLIP taggers is shown in figure 24. The loose version of the SVT and CSIP taggers are used and the JLIP tagger is set to JLIP Prob < 0.006 to ensure similar fake rates on the direct $b\bar{b}$ and QCD samples. As shown the NN demonstrates substantial improvement over the individual input b-tagging tools over all η and p_T, although this improvement falls off at high p_T. However, at these higher p_T values the NN tagger has a substantially lower fake rate than the SVT and CSIP taggers.

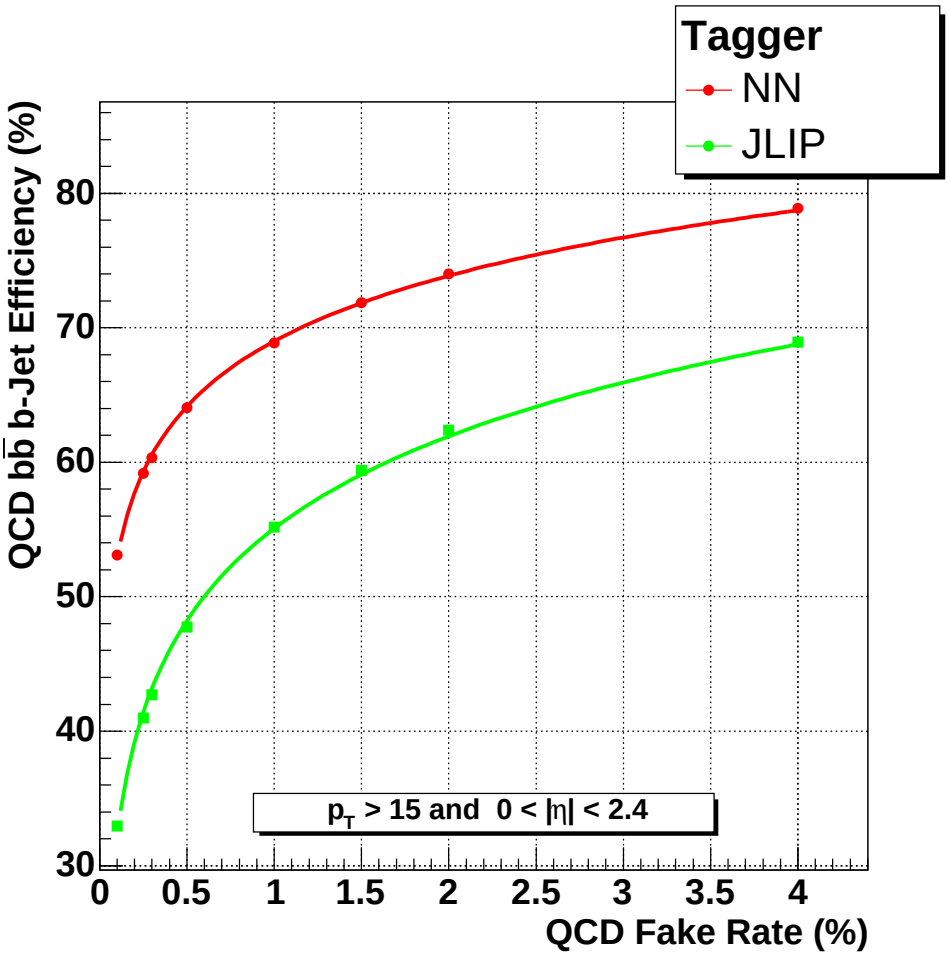

Figure 23: Performance curves for the NN tagger and the JLIP tagger.

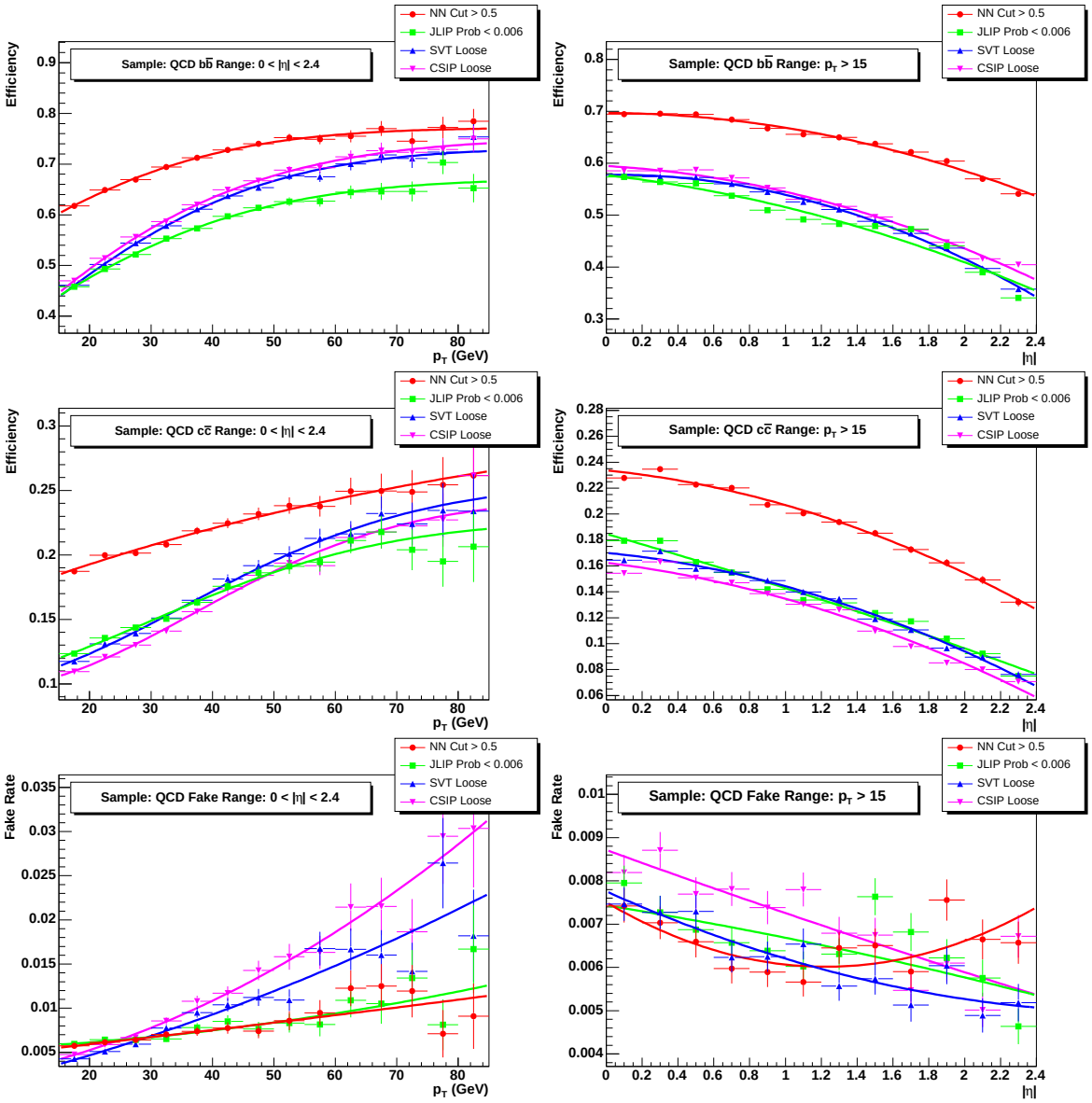

Figure 24: Comparative tag rates for the JLIP, CSIP, SVT and NN taggers on QCD b-jet (top), QCD c-jet (middle) and QCD uds-jet (bottom) samples in p_T (left) and η (right) projections. The NN tagger shows substantial improvements over the other taggers whilst also having much lower fake rates at higher p_T.

7 Conclusion

A powerful new b-tagging tool which combines the track based b-tagging tools in a NN has been developed and optimised on MC. This new b-tagging tool shows a substantial improvement over the constituent b-tagging tools. The improvement on MC has been shown to the reduce the fake rate by 80% for a fixed signal efficiency and increase the signal efficiency by up to 50%. The performance of the optimised NN b-tagger on real data has been measured in a second note[16].

References

[1] Lisa Chabalina, Regina Demina, Alexander Khanov, Flera Rizatdinova http://www-d0.fnal.gov/phys_id/bid/d0_private/certification/p14/CSIP/CSIP_v2.html

[2] D. Bloch, B. Clément, D. Gelé, S. Greder, Isabelle Ripp-Baudot, "Performance of the JLIP b-tagger in p14", DØNote 4348.

[3] L. Feligioni, M. Narain, P. Schieferdecker, A. Schwartzman "Update on b-quark Jet Identification with Secondary Vertex Reconstruction using DØReco version p14", DØNote 4414.

[4] K. Hanagaki, J. Kasper Soft Lefton (muon) tagging method http://www-d0.fnal.gov/phys_id/bid/d0_private/certification/p14/muonjet/SLT_v1.html

[5] TMultiLayerPerceptron http://root.cern.ch/root/html/TMultiLayerPerceptron.html#TMultiLayerPerceptron:description

[6] MLPfit: a tool for Multi-Layer Perceptrons http://schwind.home.cern.ch/schwind/MLPfit.html

[7] K.Hornik et al., "Multilayer Feedforward Networks are Universal Approximators", Neural Networks, Vol. 2, p359-366 (1989)

[8] D.W.Ruck et al., "The Multilayer Perceptron as an Approximation to a Bayes Optimal Discriminant Function", IEEE Transactions on Neural Networks, Vol. 1, p296-298 (1990)

[9] H.Robbins and S.Monro, "A Stochastic Approximation Method", Annals of Math. Stat. 22 (1951), p400

[10] R.Fletcher, Practical Methods of Optimization, second edition, Wiley (1987)

[11] E. Busato, F. Deliot, R. Hauser, J. Stark, R. Stroehmer, P. Verdier, M. Verzocchi, "D0Correct v8", DØNote 4646

[12] *d0root_example* Instructions http://www-d0.fnal.gov/d0dist/dist/releases/development/d0root_example/doc/

[13] *btags_cert* Instructions http://www-d0.fnal.gov/d0dist/dist/packages/btags_cert/devel/doc/

[14] Minimum Bias Probability selection of Primary Vertex http://www-clued0.fnal.gov/∼aran/d0work/mbpsel/how_to_use_PVSelect_MinBiasProb.html

[15] B. Clément, D. Bloch, D. Gelé, S. Greder, A.C. Le Bihan, I. Ripp-Baudot, "SystemD or how to get signal, backgrounds and their efficiencies with real data", DØNote 4159.

[16] T. Scanlon, M. Anastasoaie, "Performance of the NN b-tagging Tool on Pass 2 Data", DØNote 4890.