Convolutional Neural Networks for Pile Up identification in ATLAS

Murtaza Safdari, Stanford University; Prof Ariel Schwartzman, SLAC/ Stanford University*

Abstract

This project explores the possibility of training Pile Up discriminators for ATLAS based on Convolutional Neural Networks, exploiting the structural differences in the momentum spread of Pile Up and Hard Scatter jets.

1 Introduction

The Large Hadron Collider (LHC) is a particle accelerator that collides two beams of protons at energies up to 14 TeV. Physicists can probe the most elementary of forces under conditions similar to that of the Universe at the time of the Big Bang. This is done by a careful study of all the particles that are produced in the proton - proton collisions that occur at the LHC using large detectors. The ATLAS detector is one such detector on the LHC,which studies the physics at the collider by careful measurements of all the final products that arise from the proton collisions [6]. These final products, however, don't show up as isolated items in the detector, but instead get converted into collimated high energy showers of particles.

These collimated streams of particles are called jets, and are crucial to any study of Particle Physics as the momenta and energies of these jets can tell us a great deal about the particle that created the jet, which in turn tells us something about the nature of the proton - proton collision that created the particle. As the proton's radius is merely a few femtometers, the LHC collides bunches of 10^9 protons every 25 ns to get an appreciable number of interesting collisions [1].

This proton bunching leads to several experimental complications, one of them being Pile Up (PU) [Fig. 1]; Pile Up is the phenomenon where false jets are registered in the detector equipment [7]. This could be as a result of several shower particles from different jets coincidentally hitting the detector in a localized spot, registering themselves as a new jet. This could also happen as a result of time delays in the detector which falsely register jets much after the actual hit. In addition to these issues, several other contributing factors also create these Pile Up jets. These PU jets can pollute the physics sample, hampering physics analyses and leading to potentially spurious results. Hence identifying these PU jets is crucial to the operation of experiments such as the ATLAS detector.

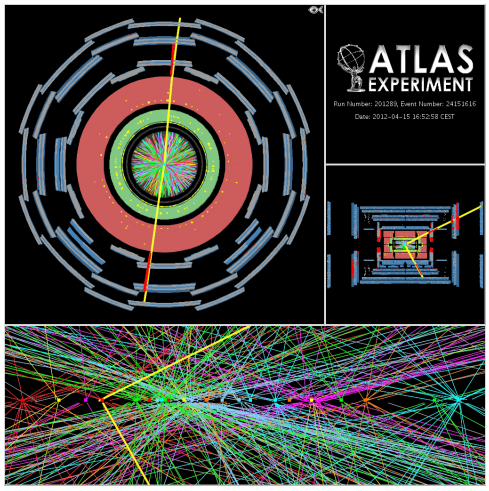

Figure 1: Typical bunch crossing at ATLAS. Each point in the lower image is a p-p collision; It is clear that shower particles from one collision wander across and interact with the detector in varied places, making it possible for false jets, that don't originate from an actual collision, to be registered in the detector equipment.

The aim of this project is to determine whether Machine Learning and Convolutional Neural Networks can be used to effectively identify PU jets in the central region[1] of the ATLAS detectors. This is done for a couple of reasons. First there is a clear baseline available in the forward region with which the results of this project can be compared[8]. Secondly, the central region is the most fruitful region of the detector in terms of physics analysis, which is why it has the best resolution[6]. Improving Pile Up identification in this region of the detector could therefore have a massive and immediate impact on the work done at ATLAS. This study will be limited by the similarity of QCD PU jets[2] to actual Hard Scatter (HS) Physics jets[3] [2], but the CNN method should be sufficient to discriminate against Stochastic PU jets[4].

2 Related Work and Evaluation Metrics

The ATLAS standard for discriminating between HS and PU jets in the central region is using the Jet Vertex Tag-

[1]Central Region in this report corresponds to pseudo-rapidity $\eta < 0.8$, which is related to θ w.r.t. z-axis by, $\eta = -\ln\tan\theta/2$

[2]PU jets due to Quantum Chromodynamics interactions

[3]Real jets due to interesting physics interactions

[4]PU jets made up of random collections of tracks and clusters

*Prof Ariel Schwartzman is not enrolled in CS231N

ger (JVT)[8][1]. The JVT[23] is based on the jet Rpt[24] variable as well as the pileup corrected jet-vertex-fraction (corrJVF) variable[22]. The jet Rpt is the sum of the p_T of the tracks from the primary vertex divided by the p_T of the jet, thus combining both calorimeter and tracking information. JVF[9] is the sum of the p_T of tracks from the primary vertex divided by the sum of all tracks. And corrJVF is a modified version of JVF which accounts for the total number of pileup tracks in the event. The JVT is constructed from the Rpt and corrJVF as a 2-D likelihood, using a k-nearest neighbour algorithm [10].

The tracks[18][19][20] mentioned above are paths of charged particles as they travel through the magnetic field and ionize the silicon tracker inside the ATLAS detector[6]. They are very precise and for that reason we can identify which interaction point they originated from, their primary vertex. Clusters[12][21], which will be mentioned shortly, are the energy deposits of both charged and neutral particles in the calorimeter. It's impossible to distinguish which interaction point a cluster came from because of the resolution of the calorimeter. That is why sometimes we do not know for sure if a jet is PU or not. Tracks and Clusters form the input data for the networks explored in this model, as described in Section 3 ahead.

For the purposes of the study conducted in this report, the jet Rpt variable serves as a good proxy for the JVT, and shall serve as the baseline against which network performance will be measured. In addition to jet Rpt, a baseline Neural Network has also been trained using jet Rpt and p_T as input features. This is theoretically a more challenging baseline to work with, as it uses p_T information to improve predictions.

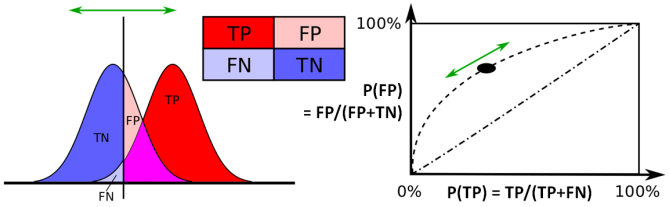

Figure 2: Receiver Operating Characteristic curve [14]

The primary metric used to gauge the performance of a discriminator in ATLAS is the Receiver Operating Characteristic (ROC) curve[15][16][17]. Figure 2 offers a neat illustration of the principle of ROC curves. Given some discriminating variable, one starts by plotting the distribution of said variable for truly Positive (P) and Negative (N) cases[5]. Given the distribution of the discriminator, one places either a lower or upper limit on the variable, and classifies everything on either side as P and N. This will mean that some cases will have been correctly classified as P (TP), some will have been correctly classified as N (TN), while others will have been misclassified as P (FP) or N (FN)[6]. Using these four numbers, one can deduce the efficiency of TP and FP events (Figure 2), and this forms a point on the ROC curve. Then by varying the value of the lower/upper limit, different points on the ROC curve can be explored.

ROC curves are used in ATLAS as discriminators are used at the start of any physics analysis to make sure that the physicist is working with good data. These ROC curves thus help the physicist determine the appropriate tool and upper/lower limit to use for their specific analysis. In addition to ROC curves, traditional metrics like Accuracy are also helpful in illustrating the merits of a discriminator tool. Hence in this report both ROC curves as well as accuracy shall be used to evaluate the performance of models.

3 Dataset and Features

Our dataset has been mined from the massive xAOD containers that hold most of the ATLAS simulation data. It consists of $\sim 4.10^5$ detector level jets which contain the following information:

- The boolean variable isPU, which tells us whether a jet is PU or HS. This forms our truth label.
- The reconstructed transverse momenta p_T of all the clusters[12] that belong to a jet; these contribute to the values of the pixels in the first channel in the image.
- The η and ϕ values of each cluster, effectively the (x, y) co-ordinate of each cluster in the image plane
- The p_T, and (η, ϕ) coords for all the tracks leading into a jet, separately for HS and PU tracks. These form the second and third channel in the image.
- The true transverse momentum p_T of the whole jet
- Reconstructed jet p_T, used to place the momentum cut on the data
- Event Weights, to ensure the network has an equal number of PU and non-PU (HS) jets to train on
- Jet η, which is used to select forward jets by $\eta > 2.5$, as well as the jet ϕ.
- The jet Rpt; the sum of the p_T of the tracks from the primary vertex divided by the p_T of the jet

The data has been split into 80% training, 10% CV, and 10% test sets. Only central jets with $|\eta| < 0.8$ are taken for uniform detector response, and with $p_T \in [20, 30]$ GeV are considered to wash out any p_T dependence. This is done since it was is that the jet energy scale can depend on the η position of the jet [3], and imposing $|\eta| < 0.8$ requires the jets to be in the central region of the detector where the detector non-uniformities are not as significant.

The data is adapted into images by representing all the momentum deposits that make up a jet in the form of a 2-D planes (η - ϕ plane), where each pixel value represents the transverse momentum p_T deposited in that pixel. Images are formed using the cluster p_Ts, HS track p_Ts, and PU track p_Ts binned in the $\eta - \phi$ plane. The resolution along both the η and ϕ axes is 0.1, and the span along each axes is 1.0 (radians), which means that each layer in an image is a 10 x 10 matrix. CNNs can then be used to learn the structure of these images and learn to discriminate between PU and non-PU jets with reasonably good accuracy.

[5] In the context of this report, the Positive case corresponds to a PU jet, and the Negative case corresponds to a HS jet.

[6] literally, True Positive, True Negative, False Positive, False Negative.

To better visualize the data, Figure 3 and Figure 4 are shown. However it should be noted that individual images are very sparse and resemble samplings from these averaged distributions. The x-y axes here are the η and ϕ axes respectively, which are essentially the usual 3-D polar angles represented on a 2-D plane. The color of each pixel is a measure of the total transverse momentum p_T that is deposited within that pixel, making the entire image a 2-D histogram of the momenta p_T, where the first layer corresponds to cluster p_T, the second to HS track p_T, and the third to PU track p_T.

Figure 4 clearly highlights the structural differences that exist between the averaged PU and HS jets. This is difference in behavior of stochastic PU jets and actual Hard Scatter jets in momentum space means that there are indeed structural properties that can be learned from visualizing the jets in the $\eta - \phi$ plane.

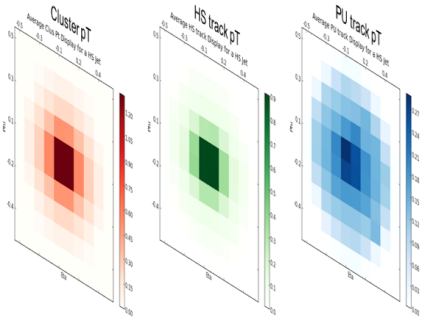

Figure 3: Averaged image of HS jets in the (η, ϕ) plane.

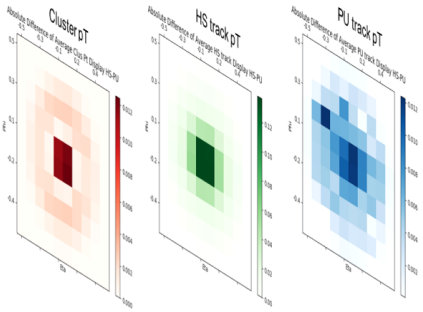

Figure 4: Absolute difference in the averaged HS & PU jets.

As part of the preprocessing, each image is reflected across the orthogonal axes to produce four copies of the image, each rotated by 90 degrees. This was done to help the network learn the rotation invariance of the jets in the $\eta - \phi$ plane. Another augmentation was done by dividing the images by the averaged sum of the values of the pixels channel-wise. By doing so, the momentum dependence of the data was further removed, leaving only structural information in the images to be learned by the CNNs. p_T dependence of the learned network would bias the network to data belonging to that specific p_T range, rendering it non-generalizable to other physics processes. This is why great care is taken to remove p_T dependence from the data.

4 Methods

Several different approaches were taken at training discriminators to tell PU jets from HS ones. Across all the models, Cross Entropy Loss is used as it tries to accumulate the probability distribution on the true labels, making the output of the network a good discriminator, as opposed to margin losses which settle once a margin is achieved. The Adam[11] optimizer is used to move towards convergence due to its efficient incorporation of gradient information into the step corrections, particularly its ability to handle sparse gradients appropriately. The Keras[5] machine learning libraries were used for designing the various networks and training the models discussed in this report. Note that the ReLu activation function was used with all the Convolutional Layers mentioned in this report.

4.1 Baseline Neural Network

The first step in the analysis was to train a Baseline NN based on the jet Rpt variable, as well as the jet p_T. This is expected to do better than the jet Rpt alone as the jet p_T is tied to the jet's identity as PU or HS. While this is not a strong correlation, it is certainly expected to improve the performance slightly. The structure of this model was guided by the universal approximation theorem which says that a single layer neural network can approximate any function to arbitrary precision [4]. This motivated the choice of a simple sequential fully connected network with two inputs, namely jet Rpt and jet p_T, only one hidden layer with 5 nodes before the output node. This network performed better than the jet Rpt alone, as expected, and this is visualized in Figure 9 and Table 1.

4.2 Pseudo CNN with full sized kernels and angular regularization

The next step was to train a simple CNN with just one convolutional layer with 100 kernels of size 10, which is the full size of the input images in the $\eta - \phi$ plane. This is akin to a fully connected layer, however in this case this was treated as a convolutional layer as a custom regularizer was designed to penalize not only the L2 norm of the learned weights, but also the L2 norm of the gradient of the learned weights in the polar θ direction in the $\eta - \phi$ plane. This is detailed below. The convolutional layer was followed by the final output single node.

4.2.1 Angular Regularizer

Given a weight matrix with the full size 10 of the input image, a regularization had to be imposed on the weights to impose angular invariance around the jet axis. This is equivalent to imposing a regularization on the gradient of the weights in the polar θ direction of the $\eta - \phi$ plane.

This invariance is expected due to the rotational invariance of the data about the jet axis, and must therefore be imposed on the weights learned by the network to ensure that only the useful structural properties are picked up by

the network. To do this, the initial 10×10 matrices are convolved with $row(1, 0, -1)$ and $column(1, 0, -1)$, and reduced to 8 x 8 images by dumping the edges. These two 8×8 matrices represent the gradient of the weight matrix along the η and ϕ directions, and can be combined under quadrature to get the total gradient at the point. Treating the two values of the gradient for each point as a 2-D vector, this vector can be dot producted by the unit vector along the polar θ direction, to find the component in that direction. Note that for a given point (x, y) in the $\eta - \phi$ plane, the unit vector along the polar θ direction divided by the radius is just $\frac{1}{r}(y, -x)$. This methodology is guided by the total derivative equation,

$$df(r, \theta) = \frac{1}{r}\frac{\partial f}{\partial \theta}d\theta + \frac{\partial f}{\partial r}dr$$

where only the coefficient of the $d\theta$ term is relevant for the penalty that was imposed here. This yields the magnitude of the gradient in the polar θ at the point, and so an 8×8 matrix of such magnitudes $\frac{\partial f}{\partial \theta}$ is constructed, and the L2 norm of this matrix is penalized.

4.2.2 Resulting weights

Some of the weights resulting from this model with the custom angular regularizer are visualized in Figure 5. As is evident, these weights have picked up the required angular invariance around the jet axis. Certain filters have picked up features reminiscent of the data having been quadrupled by rotating each image four times by 90 degrees. However, the improvement in performance over the vanilla CNN was marginal in both metrics (ROC curve and Accuracy) with an improvement of 0.1% at best. For this reason, the angular regularization approach was no longer deployed in the models that follow, as the weights seem to be learning the angular invariance naturally during the course of training.

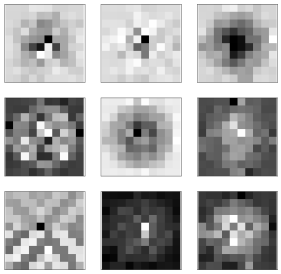

Figure 5: Random selection of learned full sized filters

4.3 Sequential Convolutional Neural Networks

Along with the pseudo CNN described above, two other architectures were explored. The first was a sequential CNN with a convolutional layer of 50 kernels size 3 with zero padding to keep the dimensions of the image intact. This was followed by a convolutional layer with 100 full sized 10×10 kernels, and the final output node.

The second was a sequential CNN with a convolutional layer of 50 kernels of size 2 and stride 2, which reduced the image to half its size. This was then followed by a convolutional layer with 100 full sized 5×5 kernel, and the final output node.

The idea behind training these models was to explore the possibility that downscaling the input images might improve the networks ability to learn the structures and have better overall performance compared to the network without downscaling. The results of the two were comparable (Figure 9 and Table 1), with the downscaling CNN outperforming the full sized CNN by only 0.04% points.

4.4 Wide Inception[13]-Inspired Convolutional Neural Network

The next step in the project was to explore the ability to learn from scaling the input images to different sizes and checking if the network was learning something hitherto inaccessible by the simple networks. This was done by performing different convolutions in parallel and combining the results from the different branches before connecting to the final output node. The structure of this model is illustrated in Figure 6.

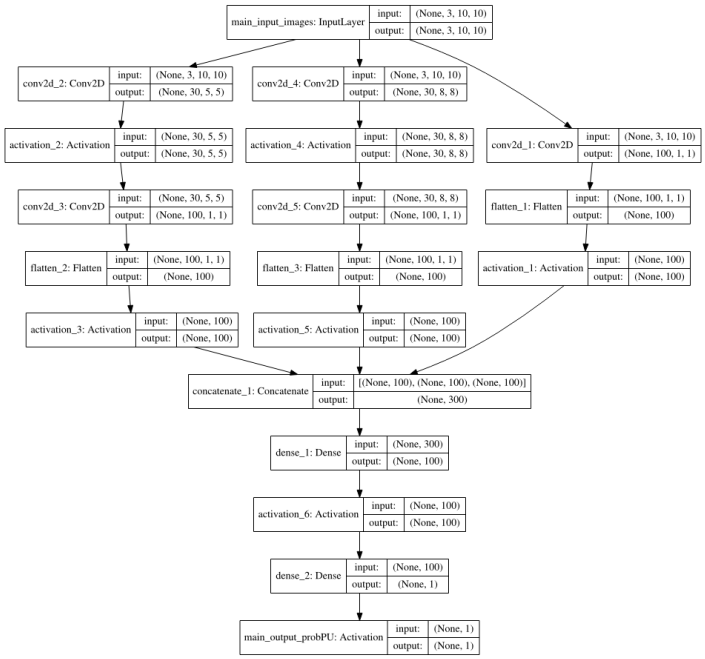

Figure 6: Architecture of wide Inception[13] inspired model

A sample of the learned weights is visualized in Figure 7. While the larger kernels can be seen as inverse jets, and thus make visual sense, the smaller weight matrices shown in Figure 7 have also picked up structural nuances of the jets. Whilst this may not be immediately obvious from the image, the vast multitude of filters in the branches makes sure all the details are learned. The outputs from these branches are all equally combined before being condensed down to the final output node.

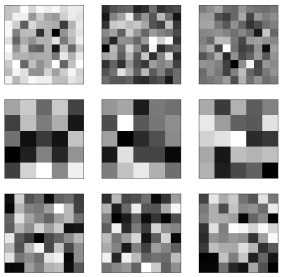

Figure 7: Random selection of weights from final full sized filters, each row corresponds to a different branch of the model

4.5 Models with Auxiliary Inputs

As a final exercise, the wide model discussed above, as well as the pseudo-CNN model, were modified with an auxiliary input, the jet p_T. As mentioned before, training networks on the jet p_T is expected to artificially improve the performance of the networks. Hence these two models were trained as a sanity check to ensure that this jump in performance would be observed. As is evident from Table 1, this is indeed the case.

It must be noted that training a discriminator on the jet p_T is not appropriate as it biases the model on the jet momentum scale. This renders the model non-generalizable to different physics processes that occur at different p_T scales. So while including the jet p_T as an added feature will improve the discriminator's performance, this must not be implemented in practice.

5 Results

All of the above models were trained to the brink of over fitting, and the results of the training are now presented for the test data.

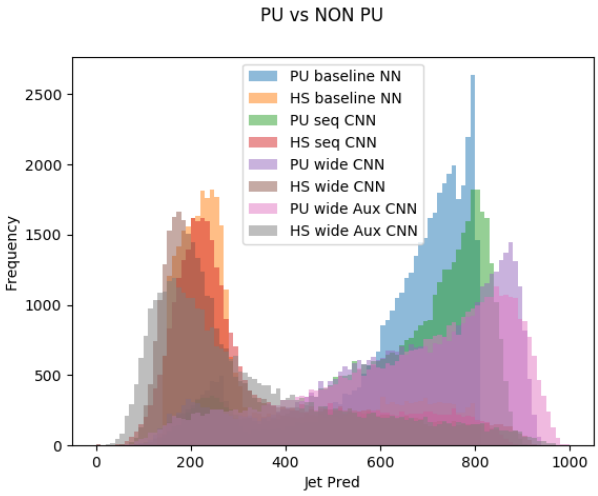

Figure 8: Distribution of output variables in HS/PU jets from tested models

Figure 8 is a visualization of the output of some of the networks times 1000. This plot neatly illustrates the separation the models achieve between the PU and HS jets in the data. The ROC curves are based on these plots in the way as described in Section 2. Here the x - axis represents the efficiency of selecting a HS jet, and the y - axis the efficiency of selecting a PU jet. These are identical to the efficiency of TP and FP respectively, as described in Section 2. The objective here is to maximize the efficiency of HS while minimizing the efficiency of PU. The ROC curves for all the listed models are shown below.

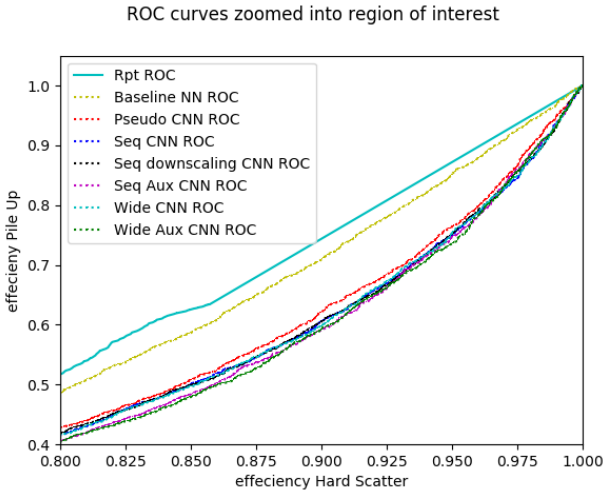

Figure 9: ROC curves for tested models

Figure 9 is the most important metric for evaluating the performance of a discriminator in ATLAS. The models explored in this report have all outperformed both the baselines; the jet Rpt variable as well as the artificially improved baseline. The models perform comparably amongst each other, for which reason the Accuracy of the model is used a secondary metric. The accuracies of each of the models tested are presented in Table 1 on the following page.

6 Discussion and Future Work

The trained CNNs outperform the baseline Rpt discriminant by 20 - 30 % in PU efficiency. They also outperform the baseline NN by similar margins.Since much of the physics analysis at ATLAS happens in the central region, these results have the potential to massively impact ATLAS Pile Up ID procedures.

It is particularly interesting to note the effectiveness of CNNs at a classification job intractable by human eyes alone. This could prove to be a highly useful observation as there are several other aspects of the ATLAS detector that could benefit from similar image analysis techniques, such as jet p_T Calibration.

The accuracies in Table 1 suggest models with jet p_T passed as auxiliary inputs perform the best. As explained in Section 4.5, these models cannot be used for the task. Nevertheless these results serve as a useful sanity check. Consequently the best network from all the architectures tried was the Wide Inception[13] inspired model, which learned

Models	**Accuracy**
ATLAS standard proxy, Rpt	**0.5005**
Baseline NN using jet Rpt and p_T	**0.6994**
Pseudo CNN with full sized kernels and angular regularization	0.7013
Sequential CNN with "Same" Conv2D followed by full sized Conv2D	0.7025
Sequential CNN with downscaling followed by full sized Conv2D	0.7029
CNN with parallel convolutions of 3×3, 5×5, 10×10 filters	**0.7036**
CNN with parallel convolutions and Auxiliary Input of jet p_T	0.7072
Sequential CNN with Auxiliary Input of jet p_T	0.7073

Table 1: The model accuracies

from different downscaled convolutions. This makes physical sense given the sparse nature of the input images, as scaling them down can reveal details in the structure otherwise inaccessible.

The next step in this analysis is to perform detailed studies of the learned weights. This is required to understand how and why these networks outperform the current standard. Additionally, the preprocessing of the images can be improved by duplicating the data more than just 4 times, with rotations finer than 90 degrees. This should make the task of learning the rotational invariance even simpler for the networks.

Furthermore the trained models will have to be tested on datasets from different p_T scales to ensure that the networks are truly generalizable. Once these thorough analyses are performed, a formal proposal to ATLAS needs to be made suggesting possible methods to incorporate the findings into its normal operations.

7 Acknowledgments

This project is possible thanks to the continued support of Chu-En Chang of the TID group, as well as the assistance of the SLAC ATLAS group, Prof Ariel Schwartzman, Dr Francesco Rubbo, Aviv Cukierman, et al.

The data for this project, as well as access to the SLAC GPU cluster, was provided by the permission of Prof Schwartzman.

References

[1] The ATLAS Collaboration. "Performance of pile-up mitigation techniques for jets in pp collisions at $\sqrt{s} = 8$ TeV using the ATLAS detector." arXiv 1510.03823v1.pdf (2016).

[2] Francesco Rubbo, "Jet pileup and ML." Talk given in an ATLAS Collaboration Meeting, slides at https://dl.dropboxusercontent.com/u/4890393/ jet-pileup_ML20160629.pdf (2016).

[3] The ATLAS Collaboration. "A measurement of the calorimeter response to single hadrons and determination of the jet energy scale uncertainty using LHC Run-1 pp-collision data with the ATLAS detector." arXiv 1607.08842v1 (2016).

[4] Alex Graves. *Supervised Sequence Labelling with Recurrent Neural Networks (Studies in Computational Intelligence).* Springer. (2012).

[5] Keras libraries, Deep Learning for Python. https://keras.io (2016).

[6] G. Aad *et al.* [ATLAS Collaboration], "The ATLAS Experiment at the CERN Large Hadron Collider," JINST **3**, S08003 (2008). doi:10.1088/1748-0221/3/08/S08003

[7] Menke, Sven. "Pile-Up in Jets in ATLAS " Talk given at the BOOST 2013, Flagstaff, AZ.

[8] The ATLAS Collaboration. "Tagging and suppression of pileup jets with the ATLAS detector" ATLAS-CONF-2014-018

[9] The ATLAS Collaboration. " Pile-up subtraction and suppression for jets in ATLAS" ATLAS-CONF-2013-083, 2013

[10] A. Hoecker, P. Speckmayer, J. Stelzer, J. Therhaag, E. von Toerne, and H. Voss. "TMVA: Toolkit for Multivariate Data Analysis" PoS ACAT (2007) 040, arXiv:physics/0703039.

[11] Diederik P. Kingma, Jimmy Ba. "Adam: A Method for Stochastic Optimization" arXiv:1412.6980 (2014).

[12] The ATLAS Collaboration. "Topological cell clustering in the ATLAS calorimeters and its performance in LHC Run 1." arXiv 1603.02934v2 (2016).

[13] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, Alex A. Alemi. "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning" ICLR 2016 Workshop, 2016.

[14] Sharpr. "ROC curves" Wiki Commons License https://commons.wikimedia.org/wiki/File:ROC_curves.svg

[15] Tom Fawcett. "An introduction to ROC analysis' ' Pattern Recognition Letters 27 (2006) 861874.

[16] Spackman, Kent A. (1989). "Signal detection theory: Valuable tools for evaluating inductive learning". Proceedings of the Sixth International Workshop on Machine Learning. San Mateo, CA: Morgan Kaufmann. pp. 160163.

[17] Department of Mathematics, University of Utah. "Using the Receiver Operating Characteristic (ROC) curve to analyze a classification model: A final note of historical interest". Retrieved May 25, 2017.

[18] Maaike Limper. "Track and vertex reconstruction in the ATLAS inner detector" PhD Thesis, Universiteit van Amsterdam, 12 Oct 2009. ISBN: 978-90-9024572-0

[19] The ATLAS Collaboration. "Track Reconstruction Performance of the ATLAS Inner Detector at $\sqrt{s} = 13$ TeV" ATL-PHYS-PUB-2015-018, 23 July 2015

[20] V Lacuesta. "Track and vertex reconstruction in the ATLAS experiment" Journal of Instrumentation, Volume 8, Number 2, February 2013, 1748-0221-8-02-C02035, http://stacks.iop.org/1748-0221/8/i=02/a=C02035

[21] Lampl, W ; Laplace, S ; Lelas, D ; Loch, P ; Ma, H ; Menke, S ; Rajagopalan, S ; Rousseau, D ; Snyder, S ; Unal, G. "Calorimeter Clustering Algorithms : Description and Performance". ATL-LARG-PUB-2008-002 ; ATL-COM-LARG-2008-003

[22] Miller, D W ; Schwartzman, A ; Su, D. "Jet-Vertex Association Algorithm". ATL-COM-PHYS-2008-008

[23] K G Tomiwa. "Performance of Jet Vertex Tagger in suppression of pileup jets and E_T^{miss} in ATLAS detector" IOP Conf. Series: Journal of Physics: Conf. Series 802 (2017) 012012, doi:10.1088/1742-6596/802/1/012012

[24] M Dobre on behalf of the ATLAS Collaboration. "ATLAS Detector Upgrade Prospects". IOP Conf. Series: Journal of Physics: Conf. Series 798 (2017) 012205 doi:10.1088/1742-6596/798/1/012205