Jet Tagging

Lai Wei Sheng

Student Number: 19107650

Page Number: 10

I. Introduction

Convolutional Neural Network (CNN) is a class of deep learning neural network technique that is widely used in image classification and recognition for its high accuracy [1]. One of the first successful application of CNN was to recognize handwritten digits from MNIST dataset [2]. It works by using filters for feature extractions while using the concept of weight sharing to reduce the number of parameters that needs training compared to fully connected dense layers. It then uses logistic regression for training of parameters. Ever since then, there has been many improvements on its architecture by adding new layers with different visualization techniques. Implementation of CNN has observed huge breakthrough in many fields.

Modern particle accelerators such as the Large Hadron Collider (LHC) can produce complicated streams of particles from collisions of high energy particle beams at speed close to the speed of light. These streams of particles produced from collision called jets are detected and studied for the contribution of the ATLAS experiment. The detection of these particles is for the understanding and searching of new physics beyond the Standard Model.

To study such system, it is important to discriminate jets produced from heavy particle decays from those QCD initiated quark/gluons jets (jet tagging). For instance, the separation of jets produced from hadronic decays of W bosons (which is referred as signal jet for the remainder of text) and background jets produced by hadronization of light quarks and gluons. These jets are detected by their energy through a detector calorimeter which can be viewed as image for CNN image classification.

The model architecture of CNN for jet tagging is built from our understanding of jet physics. Trial and error for best hyperparameter of CNN is used due to lack of computing power for best hyperparameter scan

II. METHOD

20000 jet image data is downloaded from [3] by UCL High Energy Physics Group. These are 40x40 pixelated image centred on the jet axis with each pixel representing weight of transverse energy detected by calorimeter cells. CNN can be used for image classification as background jets tend to have most of their transverse energy deposited in the centre while images for signal jets would present a two-prong structure, composed of two very nearby jets. First, these images are labelled with a target vector as [0,1] represents a signal jet and [1,0] represents a background jet. The array

20000 image data arrays are reshaped from [40, 40] to a flattened [40, 40, 1] to include a colour channel. Then the 20000 image data is split into 3 different portions with 15000 dataset for training, 2500 validation dataset and 2500 dataset for testing and shuffled to ensure each dataset contains a good proportion of signal jets and background jets

Loss function

First a loss function to quantify how well our network is performing needs to be chosen. Our network works by minimising the loss function using Stochastic Gradient Descent (SGD). For instance, there is mean squared logarithmic error, mean squared error, categorical cross entropy, and binary cross entropy. Binary cross entropy which is more appropriate for discrete classification is used as our network would be classifying jet images [4]:

$$Loss = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot \log\big(p(y_i)\big) + (1-y_i)\cdot \log\,(1-p(y_i)) \tag{1}$$

where y is the label (signal jet or background jet) and $p(y)$ is the predicted probability of image being a signal jet for all images N.

Optimizer

Next, an optimizer algorithm needs to be chosen for minimisation of our loss function. The adaptive version of SGD called Adam optimizer which attempts to determine the optimal learning rate for training as the vanilla SGD is too sensitive to a determined learning rate. [5] [6]

Architecture

CNN is implemented into our network for faster learning with lower parameters compared to the vanilla neural network. The best hyperparameters has been chosen through improving different architectures found from different papers. The final architecture used consists of first two convolutional layers with 32 filters. Then it was paired with a max pooling layer with a 3x3 down sampling. Then, 3 convolutional layers with 64, 64 and 256 filters, with a max pooling layer with a 2x2 down sampling. All filter sizes are 3x3 with a dropout layer with rate 0.2 is added after each max pooling layer to minimise overfitting. The output is then flattened to be fully connected with a 256 nodes neural network with another dropout layer with rate 0.2 is added. Rectified Linear

Units (ReLU) was chosen as an activation function for all layers before. Finally, a dense layer with 2 nodes with SoftMax activation is added for output of predicted probability for each label. Early stopping is implemented monitoring our validation loss during epochs with a tolerance of 5 epochs to prevent our network from overfitting. The early stopping layer ensures that our validation loss is always decreasing during epochs training and set to use the best network trained throughout all epochs. Model is then compiled with specified loss function and optimizer mentioned before. Different model architecture was used for training for comparison as shown in Fig. 1. Model was trained using non normalised and normalised data for comparison. Then a different model with “zero-padding” added to each convolutional layer to account for any limitations on boundaries. Another model was built with reference to a paper on jet tagging which is referred as Model Deep Learning Edition (DLE) for the remaining text [7].

Model Name	Model Structure
Model CNN	Final model as mentioned before
Model Norm	Same model structure but trained using normalised data
Model Padding	Same model structure but with zero padding added.
Model DLE	Model architecture built with reference to [7]

Figure 1: List of models trained with different structures for comparison

Model Training

Training of model is done using the 15000 training dataset and it was validated using the 2500 validation dataset throughout the epoch training. Number of epochs was set to be 20 with batch sizes set to be 100. A smaller batch size would give a noisier estimate of gradient which would be beneficial for model to not get stuck in a poor local minimum

The training was done using a Nvidia GeForce GTX1050 with Max-Q design using the Nvidia Cuda platform readied with Cuda drivers, toolkit and cuDNN library. This was to ensure GPU training was utilised as shorter time was used for training of network. The runtime of training was measured to be around 1 minute using datetime module of python [8][9].

Pre-processing of image

An attempt of pre-processing was done on the image, normalizing all image to have pixel intensities between 0 and 1 (dividing by np.max).

Evaluation of model

As the hyperparameters of architecture was obtained through trial and error, an evaluation method was needed for comparison of models. The models were first evaluated to test for its accuracy and loss value using the 2500 testing dataset. But as our models is a classifier, a more appropriate evaluation method using Receiving Operating Characteristics (ROC) graph was implemented [10]. ROC graph is a method to visualise and select best classifiers based on their performance by summarising all the confusion matrices for our model that each threshold produced. Confusion matrices have a structure as shown in Fig. 2.

	Actual Positive	Actual Negative
Predicted Positive	True Positive (TP)	False Positive (FP)
Predicted Negative	False Negative (FN)	True Negative (TN)

Figure 2: Structure of a confusion matric for classifying model with a set threshold.

The true positive rate (TPR) or recall of model which is the fraction of all actual positive that are predicted positive is as below:

$$TPR = \frac{TP}{TP + FN} \tag{2}$$

where a false positive rate (FPR) or fall-out which is the fraction of all actual negative that are predicted positive would be:

$$FPR = \frac{FP}{FP + TN} \tag{3}$$

The precision of model is the fraction of all positive predictions that are true positive, defined as below:

$$precision = \frac{TP}{TP + FP} \tag{4}$$

An ROC graph is a plot of true positive rate against false positive rate. The area under ROC curve is the AUC ROC score which will be used for comparing different classifying models. A higher number of AUC ROC would suggest a better classifying model. AUC ROC score of 1 suggest that the classifying model is able to perfectly distinguish signal jets and background jets. Another method of evaluation is by comparing f1 scores of different models. F1 score is the harmonic mean of precision and recall as shown below [11]:

$$F1 = 2 \cdot \frac{(precision \cdot recall)}{(precision + recall)} \tag{5}$$

which is a better evaluation as it takes into account for both precision and recall of model.

III. Result and Analysis

Five fresh runs of model training and evaluation was done and the scores from each run was recorded in Fig. 3 for comparison. An average of all runs is calculated. Theoretically, model training using pre-processed (normalised) image would result in a better model, but this was not the case for every run. Normalising the image results in a slightly better model on average. Use zero padding would result in a lower evaluation score which indicates a worse fit. Model DLE have a way lower accuracy than any other models as it has significantly lower number of parameters. In [7], the model was trained using 8 million datasets for a good enough evaluation score.

Model Name	Average Test Accuracy	Average Test Loss	Average AUC ROC score	Average F1 score
Model CNN	0.7898	0.4644	0.8670	0.7893
Model Norm	0.7940	0.4530	0.8724	0.7971
Model Padding	0.7783	0.4577	0.8456	0.5455
Model DLE	0.5239	0.6883	0.5473	0.4017

Figure 3: Table of average evaluation score for each model with different architecture with Model Norm with the best performance.

The training of Model Norm was visualised with plots of accuracy against epochs and loss value against epochs as shown below in Fig. 4.

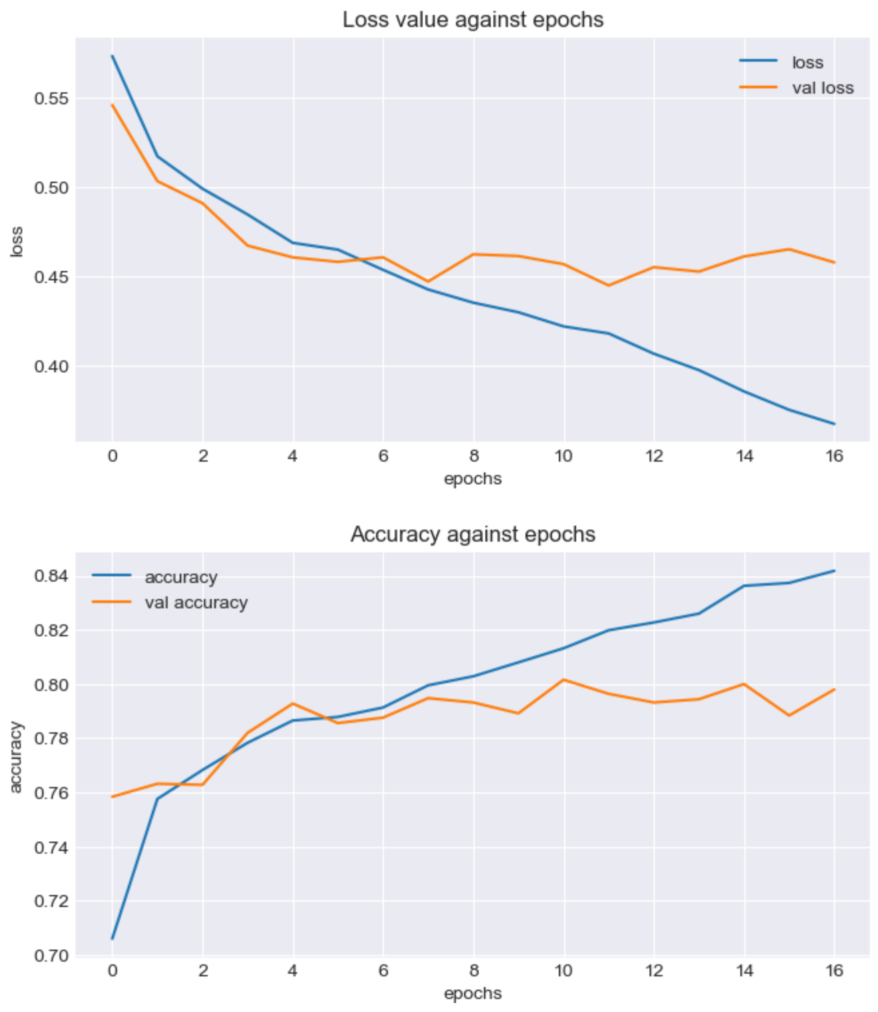

Figure 4: Plot of loss against epochs and accuracy against epochs for visualisation of Model Norm training

Then the ROC curve and a graph of tagging efficiency against background rejection for different architecture on the final run is plotted for comparison as shown in Fig. 5. The tagging efficiency against background rejection was plotted using log scale. As seen in both graph, area under curve for Model Norm is higher than the other models. This indicates that Model Norm has the best performance.

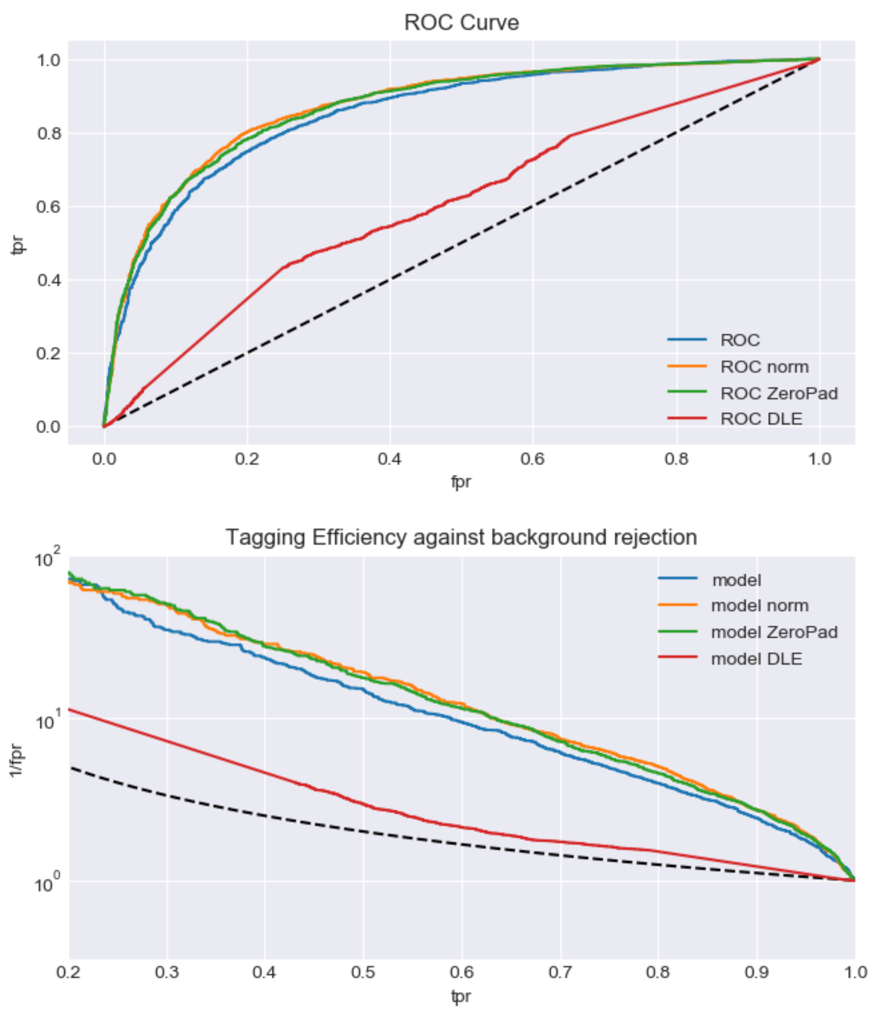

Figure 5: Plot of ROC curve and tagging efficiency against background rejection for all models with the dotted line being the plot for a random generated classifier.

A histogram of probability obtained from predictions with training dataset and testing dataset was plotted as shown.

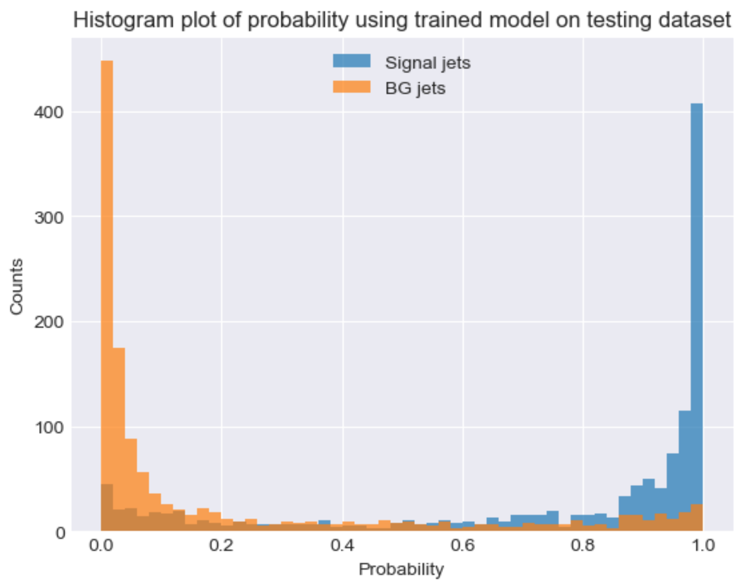

Figure 6: Histogram plot of probability predictions of Model Norm on testing dataset

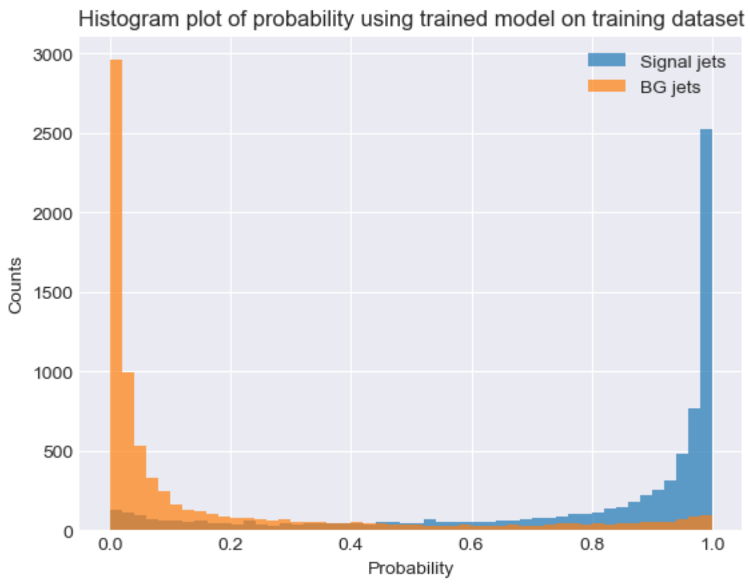

Figure 7: Histogram plot of probability predictions of Model Norm on training dataset

As shown in both Fig. 6 and Fig. 7, our model is approximately predicting a 1 to 1 ratio between signal jets image and background jets image which was the ratio of signal jets and background jets in the imported dataset. Also at probability equals to 0 or 1 there are a huge number of true predictions.

IV. Conclusion

With our given number of data points, and through trial and error, Model CNN is concluded to be the best model architecture built for jet tagging with good accuracy. Its evaluation score for 5 runs is shown as below.

	1st Run	2nd Run	3rd Run	4th Run	5th Run
AUC ROC score	0.8718	0.8741	0.8731	0.8709	0.8720
F1 score	0.7955	0.7956	0.7988	0.7951	0.8007

Figure 6: Scores for five runs of Model Norm training

There are many improvements that could be made to achieve better accuracy. First, more data points could be used for training of model. Also, more image pre-processing could be done with our knowledge of symmetry regarding jet physics. For instance, rotation, translation or flipping of image. Other than that, an LSTM network can also be added into the architecture with consideration of the momentum of jets. Also, the use of Principle Component Analysis (PCA) and Fisher's Linear Discriminant (FLD) in facial recognition could be implement in jet tagging model training as explained in [12]. Hyperparameter scan can also be done with better computing power to obtained the best architecture for model.

V. References

[1] N. Sharma, V. Jain and A. Mishra, "An Analysis Of Convolutional Neural Networks For Image Classification", *Procedia Computer Science*, vol. 132, pp. 377-384, 2018. Available: 10.1016/j.procs.2018.05.198

[2] Ciresan, Dan, Ueli Meier, and JurgenSchmidhuber, (2012) ""Multi -column deep neural networks for image classification." *2012 IEEE Conference on Computer Vision and Pattern Recognition.*

[3]"Index of /undergrad/0056/other/projects/jetimage", *Hep.ucl.ac.uk*, 2022. [Online]. Available: https://www.hep.ucl.ac.uk/undergrad/0056/other/projects/jetimage/. [Accessed: 16- Jan- 2022].

[4] k. Transfer, "How to choose cross-entropy loss function in Keras? - knowledge Transfer", *knowledge Transfer*, 2022. [Online]. Available: https://androidkt.com/choose-cross-entropy-loss-function-in-keras/. [Accessed: 16- Jan- 2022].

[5Diederik P. Kingma and J. Lei Ba, "ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION", 2015. Available: https://arxiv.org/abs/1412.6980. [Accessed 16 January 2022].

[6] T. Schaul, S. Zhang and Y. LeCun, "No More Pesky Learning Rates", 2013. Available: https://arxiv.org/pdf/1206.1106.pdf. [Accessed 16 January 2022].

[7] L. de Oliveira, M. Kagan, L. Mackey, B. Nachman and A. Schwartzman, "Jet-images — deep learning edition", *Journal of High Energy Physics*, vol. 2016, no. 7, 2016. Available: 10.1007/jhep07(2016)069 [Accessed 16 January 2022].

[8] M. Garland et al., "Parallel Computing Experiences with CUDA", *IEEE Micro*, vol. 28, no. 4, pp. 13-27, 2008. Available: 10.1109/mm.2008.57 [Accessed 16 January 2022].

[9] N. Vasilache, J. Johnson, M. Mathieu, S. Chintala, S. Piantino and Y. LeCun, "FAST CONVOLUTIONAL NETS WITH fbfft : A GPU PERFORMANCE EVALUATION", 2015. Available: https://arxiv.org/pdf/1412.7580.pdf. [Accessed 16 January 2022].

[10] T. Fawcett, "An introduction to ROC analysis", *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861-874, 2006. Available: 10.1016/j.patrec.2005.10.010 [Accessed 16 January 2022].

[11] Z. Lipton, C. Elkan and B. Narayanaswamy, "Thresholding Classifiers to Maximize F1 Score", 2014. Available: https://arxiv.org/abs/1402.1892. [Accessed 16 January 2022].

[12] J. Cogan, M. Kagan, E. Strauss and A. Schwarztman, "Jet-images: computer vision inspired techniques for jet tagging", *Journal of High Energy Physics*, vol. 2015, no. 2, 2015. Available: 10.1007/jhep02(2015)118 [Accessed 16 January 2022].