The Effects of Normalisation and Data Leakage in Jet Tagging with A CNN

PHAS0056 Mini-Project

Lucas Curtin

Word Count:2726

UCL
Physics Department
UCL Physics Building Lab 2, London
January 2022

1 Introduction

Jets are collimated sprays of particles, produced from the hadronisation of quarks and gluons. Heavy particles that decay into heavy standard model particles such as W bosons and top quarks impart a Lorentz boost due to the conservation of momentum and difference in the masses. The jets from the decays of these W bosons and top quarks are therefore highly collimated and can be captured by a single jet. For a better understanding of the standard model, there is great merit in being able to distinguish between the jets from the decay of heavy standard model particles and the Quantum Chromodynamic (QCD) jets from lighter particles. The jets are measured in a detector calorimeter, where they leave their energy. The images are the transverse energies deposited by a jet in the calorimeter cells, with each image being centred on the jet axis, and the weight of each cell is the transverse energy deposited. Jets from background tend to have most of the transverse energy deposited in the centre, while those from W bosons, that are actually composed of two very nearby jets, present a two-prong structure. This substructure can be exploited with machine learning which has widespread uses in image classification. This can prove to be far more efficient than using physics inspired analysis of the images. This investigation aims to build and train a Convolutional Neural Network (CNN) with Keras, a python interface for artificial neural networks, that is capable of distinguishing between the jets produced by the decays of W bosons and the QCD jets (hereafter known as signal and background jets respectively). The effects of normalisation will also be analysed as this is common practice for image classification problems and to see if it would aid in discriminating between jets (jet tagging). A model will be designed and then trained on the unnormalised data, changing it by hand to improve its performance. The same model will then be trained and tested on normalised data and then compared to its initial performance. A common pitfall in machine learning will also be quantitatively investigated by exploring the effects of data leakage on the normalisation process.

2 Network Features

There are lots of features that can be applied to the model to affect its performance. The model itself will be a CNN but multiple layers will be introduced to aid the training process to provide more robust predictions.

2.1 Convolutional Neural Networks

CNNs are neural networks that contains at least one layer where a convolutional filter passes over an input matrix. This convolutional filter, also known as a kernel, is a small matrix that passes over the input matrix (which in this case will be the jet images). The response of the kernel i.e. the inner product when it is passing over a portion of the image is largest when that portion of the image is most similar to the kernel. The outputs of the kernel make up a feature map which is of a smaller dimension than the image itself as seen in the example in Figure 1. CNNs are often used for image classification because of how the kernel operates. If the kernel passes over the same sub-matrix of an image, it will give the same result. This can be very useful for spotting features in images such as circles or straight lines. In the case at hand, it could be a very useful feature to spot the two-prong structure often seen in the signal jets.

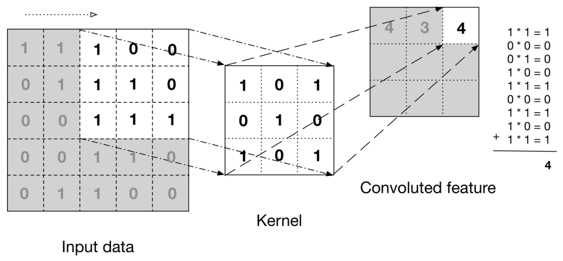


Figure 1: Example of the mechanism of convolutional filters[1]

2.2 Pooling Layers

Pooling layers reduce the dimensions of the inputted feature maps, say from the previously discussed convolutional layer. A max pooling layer, similar to the convolutional filter, will pass over the image but will instead just take the largest value in the sub matrix it is passing over. These largest values will then make up another feature map. By taking the largest feature out of regular regions in the input feature map, max pooling layers are very useful for noise reduction as mostly the prominent features will survive. By reducing the dimensions of the data flowing through the model and by reducing the amount of noise the model will need to discriminate from, max pooling can make the training process significantly easier.

2.3 Dropout Layers

Neural networks can often quickly begin to overfit to the training data, which is when the network becomes very good at predicting training images but perform worse on validation images that it hasn't seen before. A way to avoid this is by implementing dropout layers in the network. A dropout layer will cause some of a layer's outputs to be set to zero, as if some of the neurons that make up the layer were not there. Dropout layers can prevent complex co-adaptions on the training data and improve the model's performance as seen in other common image recognition tasks such as on the MNIST data set[2].

3 Pre-processing images

It is commonplace in image classification models to normalise the images before feeding them into the network. The jets images are made up of 40×40 pixels with a range of strength values corresponding to the transverse energy deposited at that point.

3.1 Min-max normalisation

Min-max normalisation has been used in other machine learning image recognition studies such as on the MNIST data-set[3]. This could be accomplished by transforming the data with equation (1).

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{1}$$

This would cause the weakest pixel strength to have a value of 0 and the strongest pixel strength to have a value of 1 and spread out the other values between them with the same proportion as before. This however would not be practical for the task at hand. As seen in Figure 2, the axes of two signal jet images can have a difference in maximum pixel strength by nearly an order of magnitude. The spread of the pixel strengths across the data would be the same but large outliers would cause the majority of the signals data to be squashed close to 0. This accompanied with max pooling in the layers would mean meaningful information would likely be removed by the noise reduction feature. This form of normalisation will therefore not be considered.

3.2 Z-score normalisation

Z-score normalisation handles outliers in data far better than min-max normalisation with regards to what is fed to the model. Z-score normalisation makes the data have a mean of 0 and a standard deviation of 1 by using equation (2). The images won't share the same axes, which would be the case for min-max normalisation, but will be much closer to each other whilst being less distorted by outliers.

$$z = \frac{x - \bar{x}}{S} \tag{2}$$

Where $\bar{x}$ and S are the mean and standard deviation of the sample respectively. This transformation will reduce the distance between the majority of the data and the outliers which could help the CNN process the images. The axes will now be far closer to each other as seen in Figure 3. However again, normalisation in pre-processing can feasibly remove meaningful information that the model could learn from. It is possible that very high energy deposits in certain regions could be a meaningful characteristic in the discrimination between jets. With this in mind, Z-score normalised data will be compared to unnormalised data to see the effects it has on the training of the model.

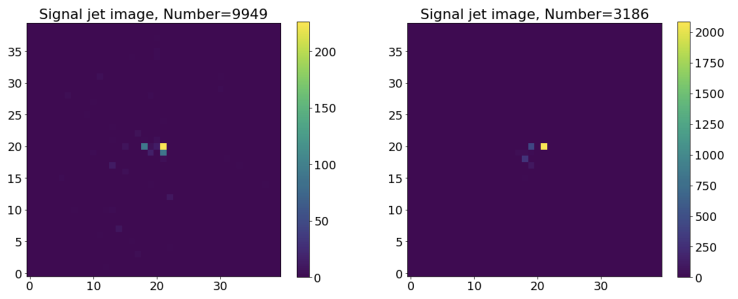


Figure 2: Two randomly chosen W boson decay jet images and their corresponding axes.

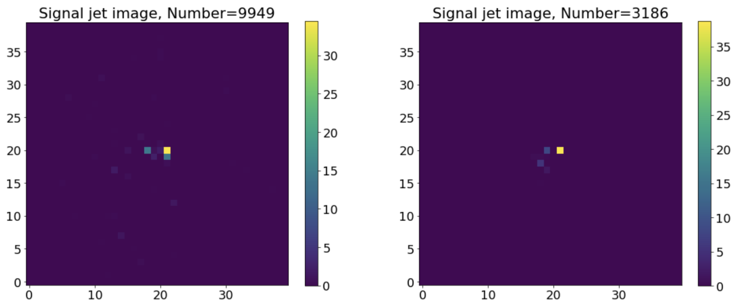


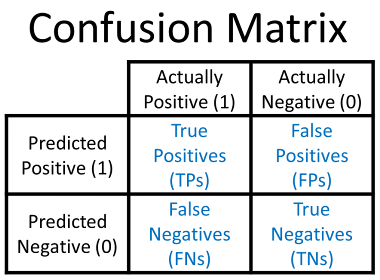
Figure 3: Same two jets images from Figure 2 after Z-score normalisation

4 Metrics

For the case of binary classification, the model can predict whether the data belongs to the positive or the negative class whilst the data is only actually part of either the positive or negative class. This leads to 4 possible outcomes of any one prediction by the model. The quantities of these outcomes are represented by a confusion matrix, as seen in Figure 4. This matrix can give valuable information for the competency of the model for different tasks whereas other metrics such as accuracy can sometimes be misleading such as when there is a skew in the size of the data classes.

4.1 F1 Score

F1 score is the weighted average of precision and recall. Precision is the ratio of correctly predicted positives to the total number of positive predictions (correct and incorrect) and can be calculated by equation (3). Recall, also known as sensitivity, is the ratio of correctly predicted positives to the total number of observations that actually belong to the positive class and can be calculated by



Confusion Matrix

	Actually Positive (1)	Actually Negative (0)
Predicted Positive (1)	True Positives (TPs)	False Positives (FPs)
Predicted Negative (0)	False Negatives (FNs)	True Negatives (TNs)

Figure 4: Confusion matrix for predictions of a model on a data set[4]

using equation (4). F1 score, which can be calculated using equation (5) takes false positives and negatives into account so is a better measure when there's an imbalance of data between the classes. The 'Matthews Correlation Coefficient' (MCC) has proven to be a more reliable statistic than both F1 score and accuracy for certain models[5]. The data that will be used for this experiment is equally balanced however, so accuracy should be a perfectly reasonable metric to use.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{3}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{4}$$

$$\text{F1 Score} = \frac{2 \times (\text{Recall} \times \text{Precision})}{\text{Recall} + \text{Precision}} \tag{5}$$

4.2 AUC-ROC Curve

A 'Receiver Operator Characteristic' (ROC) curve is a probability curve that plots how the model performs at different decision thresholds i.e. at what probability a prediction is classed as a positive or a negative classification. To calculate the ROC curve, the true positive rate (TPR) and the false negative rate (FNR) are required. The true positive rate is synonymous for recall as defined in equation(4). The true negative rate, also known as specificity, is the proportion of the negative class that got correctly classified as defined in equation6. The false positive rate (FPR), which is equal to $1 -$ Specificity, is the proportion of the negative class that was incorrectly classified by the model. The ROC curve plots the TPR against the FPR at a range of threshold values. The 'Area Under The Curve' (AUC) score is a summary of the ROC curve and varies from $0.5 \rightarrow 1$ where 0.5 would represent a model randomly classifying the data and 1 would represent a model perfectly classifying the data.

$$\text{Specificity} = \frac{TN}{TN + FP} \tag{6}$$

5 Method

5.1 Data preparation and leakage

The data consisted of: 10,000 signal images and 10,000 background images. This data was combined, randomly shuffled, and then split into training and testing data (80% and 20% respectively). This was done so when the model was being evaluated, it was making predictions on data it hadn't seen before rather than remembering what it's classification was as it had already seen that image before. When testing the effects of Z-score normalisation, the data was normalised differently in two different methods for two separate investigations. For the first, the data was normalised twice: once just on the training portion and again just on the testing portion; this was done to avoid **data leakage**. If all the data was normalised and then split into the two groups, the model would gain information on the testing data despite only seeing the training data because the training data's normalisation had been affected by the testing data's mean and standard deviation. This can lead to over-estimates in the model's predictive abilities as it can learn about the testing data without being trained on it. A model was designed for and optimised on the unnormalised data. The same model structure was used and then trained on the normalised data twice: once when the data was normalised all together and once when the data was normalised *after* it had been split in the training and testing portions, as to avoid data leakage. This was done in the hopes of quantitatively analysing the effects of data leakage.

5.2 Network Architecture

The initial layout of the model was based on another CNN designed for jet tagging[6]. The hyperparameters of the model (features of the model that are not altered by training) such as number of layers and size of filters were set by hand. There are algorithmic methods to complete this, such as with Scikit-learn's 'GridSearchCV', but these would have been unreasonably time intensive. Succinctly, the final structure of the model was:

[Dropout $\rightarrow$ Conv $\rightarrow$ ReLU] $\rightarrow$ [Dropout $\rightarrow$ Conv $\rightarrow$ ReLU $\rightarrow$ MaxPool] $\times$ 2 $\rightarrow$ [Dropout $\rightarrow$ FC $\rightarrow$ ReLU] $\rightarrow$ Sigmoid

Where 'Conv' represents a convolutional layer as described in 2.1 with ReLU activation functions. Activation functions determine a neuron's output based on its input and ReLU was used as "the usage of ReLU helps to prevent the exponential growth in the computation required to operate the neural network"[7].

FC represents a fully connected layer of neurons which then all outputted to one neuron with a sigmoid activation. The sigmoid function, as seen in Figure 5, was used as the activation function for the final neuron as it would output a value between 0 and 1. This was very fitting for the task at hand of classifying binary images as the value outputted by the model between 0 and 1 would represent the model's probability of which class the inputted image belonged to.

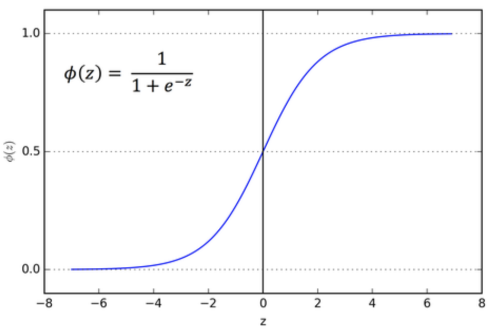


Figure 5: The Sigmoid Function [8]

6 Training

3 models were trained in total: one on the unnormalised dataset, one on the dataset that had its training portion and testing portion independently z-score normalised, and one on the dataset that was z-score normalised and then split into training and testing portions. These will hereafter be referred to as Model 1, Model 2 and Model 3 respectively. Model 1 was optimised on the unnormalised dataset and the exact same structure made up both Model 2 and Model 3 for comparison of the effects of normalisation. They were all compiled in the same form. The loss function, how the model calculates how well it is doing, chosen was 'binary cross-entropy'. The optimiser, how the loss function is reduced, was 'Rmsprop' and the metric, what the model is seeking to get better at, was 'accuracy'. These parameters were chosen by iterating through multiple options to see how they performed all together, as one loss function could lead to poor training with the wrong optimiser.

7 Results

Each model was trained and tested 3 times on their respective dataset to find an average of their performance metrics. These metrics can be found in Table 1 with one standard deviation of error. The best performing model with regards

Table 1: Model metrics comparison.

$Metrics$	Model 1	Model 2	Model 3
AUC Score	0.875 ± 0.001	0.870 ± 0.003	0.869 ± 0.004
Test Accuracy	0.779 ± 0.001	0.794 ± 0.009	0.795 ± 0.002
Test Loss	0.485 ± 0.003	0.455 ± 0.011	0.466 ± 0.007
F1 Score	0.762 ± 0.029	0.799 ± 0.007	0.790 ± 0.007

to AUC score was Model 1. This score was from Model 1's ROC-AUC curve as seen in Figure 6. Model 1 was not the best with regards to the other parameters. This could have been due to sporadic and inconsistent training of Model 2 and Model 3. For example, the absolute error of Model 2's test accuracy is 9 times larger than that of Model 1's. The training process of Model 2 is plotted below in Figures 7 and 8. It is clear the training process is far more jagged, likely due to random guesses, than that of Model 1's training process as seen in Figures 9 and 10. Model 1's predictions on both the training and testing data were plotted in histograms, as seen in Figure 11 and 11 to visualise how Model 1 responded to the background and signal jets. The blue histograms represent the probability outputs of the model for different background jets and the orange histograms analogously for the signal jets. It can be seen that Model 1 was far better at detecting the background jets from the signal jets, suggesting the signal jets had a more complex substructure to understand. This is further represented by the confusion matrix for Model 1 as seen in Figure 13. The proportion of background jets (BG) correctly predicted by Model 1 was $\frac{2058}{2058+434} = 0.825$. The proportion of signal jets correctly predicted was $\frac{1964}{1964+544} = 0.783$.

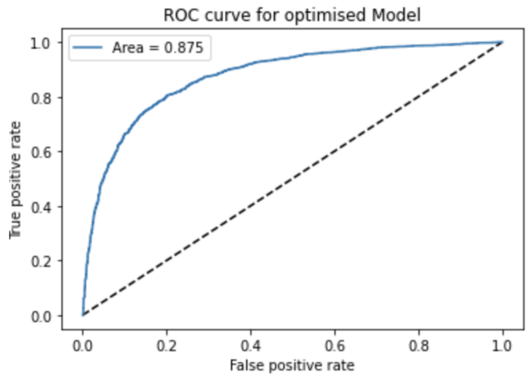


Figure 6: Model 1's ROC-AUC Curve for the testing data.

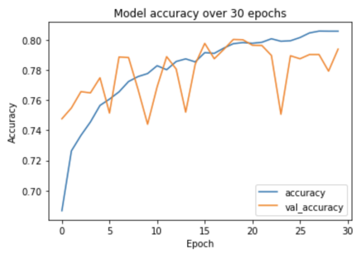


Figure 7: Model 2's accuracy for normalised data

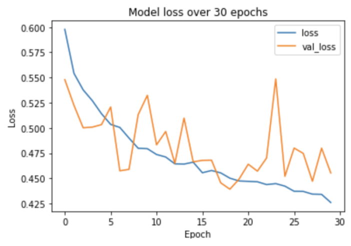


Figure 8: Model 2's loss for normalised data

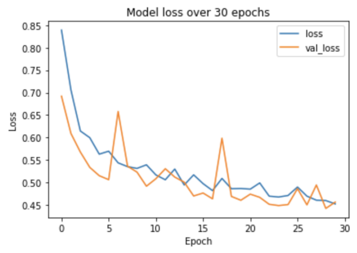


Figure 9: Model 1's accuracy for unnormalised data

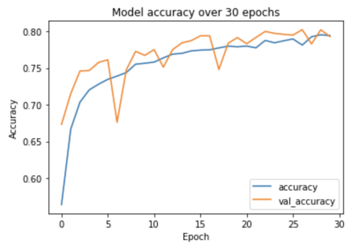


Figure 10: Model 1's loss for unnormalised data

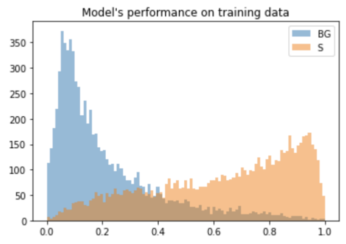


Figure 11: Histogram for Model 1 trained on normalised training data.

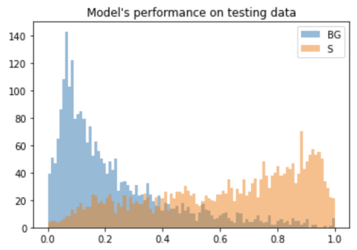


Figure 12: Histogram for Model 1 trained on unnormalised testing data.

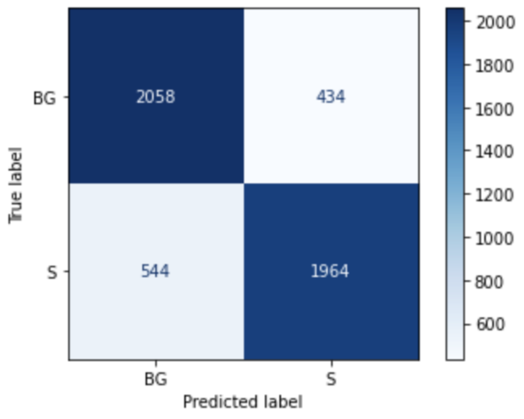


Figure 13: Model 1's confusion matrix for the testing data.

8 Conclusion

The Z-score normalisation process harmed the model structure's ability to train on the data as seen from the decrease in the AUC score of the models. This is possibly due to the fact that the model was in fact capable of learning valuable information from the relative intensities of the pixels and even outlier values didn't significantly affect its training. However, Model 1 was optimised on the unnormalised data and that structure was then applied on the normalised data with Models 2 and 3. This was done so the models could be directly compared however, perhaps a model with different hyperparameters could have been optimised better on the normalised data. If this experiment was to be repeated, each model should be independently optimised on its respective data to see the maximum capabilities of the CNN structure used. It was also expected that Model 3 would outperform Model 2, as the distribution of the training data contained information about the testing data which could possibly give over-estimates of Model 3's validation prediction. However again as seen in Table 1, Model 3 performed similarly to Model 2. Their AUC scores agree with each other however Model 3 has a larger loss than Model 2 so it likely trained on the data worse and would overall make worse predictions. The effect of data leakage was unable to be quantitatively analysed. This was due to other issues with Model 3's performance, likely due to the lack of repeated optimisation discussed above. The normalisation process appeared to harm the model's ability to classify the jets but again perhaps the optimisation of the model whilst being validated against the testing data could allow for an overall better performance. For improving models for future experiments, other forms of pre-processing could be done such as aligning features of the jet to aid the training process or point of interest finding such as the two prong structure of the signal jets (similar to finding the eyes for facial recognition) such as in previous experiments[9].

References

[1] "Cnn for deep learning: Convolutional neural networks," Jul 2021. https://www.analyticsvidhya.com/blog/2021/05/convolutional-neural-networks-cnn/.

[2] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[3] S. Ahlawat, A. Choudhary, A. Nayyar, S. Singh, and B. Yoon, "Improved handwritten digit recognition using convolutional neural networks (cnn)," *Sensors*, vol. 20, no. 12, p. 3344, 2020.

[4] R. Draelos, "Measuring performance: The confusion matrix," May 2019. https://glassboxmedicine.com/2019/02/17/measuring-performance-the-confusion-matrix/, Accessed 1/1/2022.

[5] D. Chicco and G. Jurman, "The advantages of the matthews correlation coefficient (mcc) over f1 score and accuracy in binary classification evaluation," *BMC genomics*, vol. 21, no. 1, pp. 6–6, 2020.

[6] L. de Oliveira, M. Kagan, L. Mackey, B. Nachman, and A. Schwartzman, "Jet-images — deep learning edition," *Journal of High Energy Physics*, vol. 2016, Jul 2016.

[7] Baeldung, "How relu and dropout layers work in cnns," Aug 2020.

[8] S. Sharma, "Activation functions in neural networks," Jul 2021.

[9] J. Cogan, M. Kagan, E. Strauss, and A. Schwarztman, "Jet-images: computer vision inspired techniques for jet tagging," *The journal of high energy physics*, vol. 2015, no. 2, pp. 1–16, 2015.