Student Project Report

CERN Summer Student Program 2023

Impacts of Data Pre-processing on Jet Images Classification using Deep Learning

Abdalrahman Mohamed Elsheikh Eltahir

Under the supervision of:

Dr. Rachik Soualah

Abstract

Jet images are essential utilities in High Energy Physics (HEP) as they enable the extraction of crucial physics information from particle collisions. Jets are clusters of particles produced during high-energy collisions. Jet images are two-dimensional representations of these jets that preserve important details about the energy distribution and the jet substructure. When it comes to the classification of different types of jets, deep neural networks yield very promising results. Convolutional Neural Networks (CNNs) have proven to be particularly effective in achieving highly accurate classifications. One common process in deep learning is data pre-processing. Data pre-processing involves techniques such as (background) noise reduction, image normalization, and feature extraction. The performance of deep learning models can be significantly impacted by the quality of pre-processed data. Therefore, this study discusses the significance, impact and usefulness of data pre-processing techniques on the jet images classification performance of deep learning models. This study makes use of reconstructed jet images of proton-proton collisions at the CERN Large Hadron Collider (LHC). The signal events are jets at final states produced in top anti-top events, whereas the considered background contributions that mimic the signal events are jets that yield in QCD events.

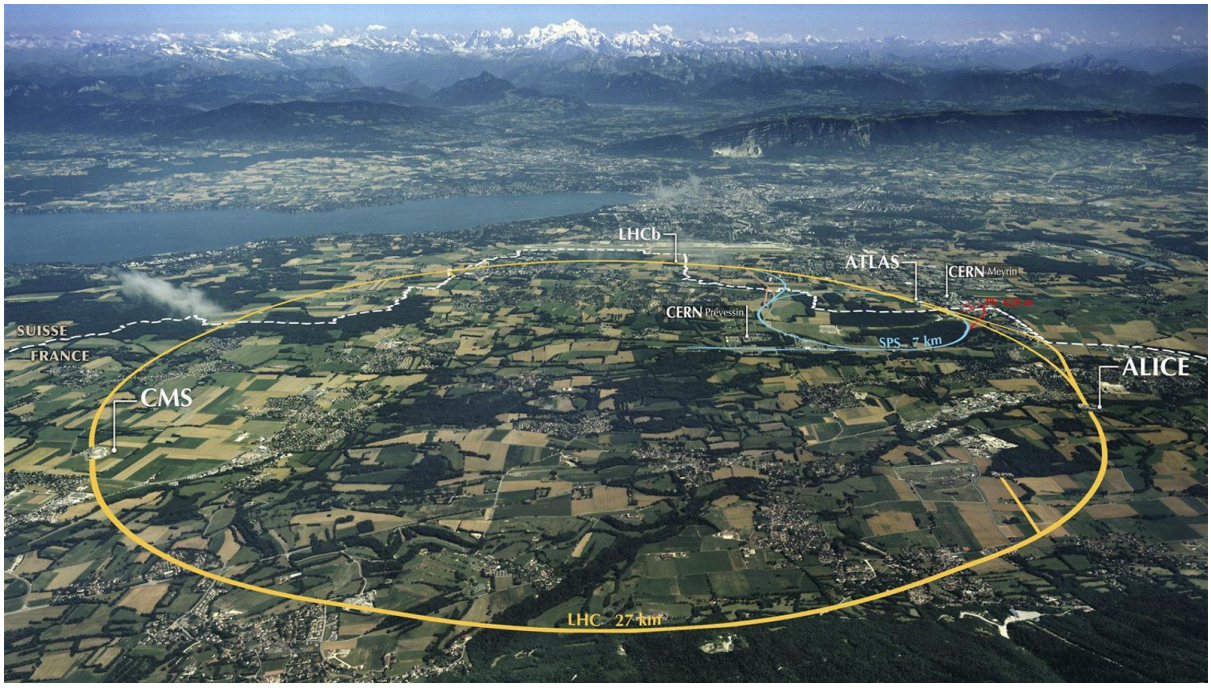
Contents

1. Introduction 4
2. Event Generation 6
3. Clustering 8
   3.1. SlowJet 8
   3.2. FastJet 8
4. Data Pre-processing 8
   4.1. Pre-pixelation or post-pixelation? 9
   4.2. Centering (Translation) 9
   4.3. Pixelation 10
   4.4. Normalization 16
      4.4.1. L^2 Normalization 16
      4.4.2. Min-max Normalization 16
   4.5. Rotation 17
   4.6. Reflection 17
5. Analysis 18
   5.1. Models 18
      5.1.1. Logistic regression 18
      5.1.2. Multilayer perceptron 18
      5.1.3. Convolutional Neural Network (CNN) 18
   5.2. Nature and Scheme of Analysis 19
   5.3. Performed Tests 19
      5.3.1. SlowJet & Centering on leading jet & histogram pixelation 19
      5.3.2. FastJet & Centering on leading sub-jet & histogram pixelation 20
      5.3.3. FastJet & Centering on leading sub-jet & ratio-of-areas custom pixelation 21

5.3.4. Rotation post-pixelation (jet-generation method) 22
5.3.5. Rotation post-pixelation (deepjets method) 22
5.3.6. Rotation pre-pixelation (jet-generation method) 23
5.3.7. Rotation pre-pixelation (jet-generation method) & Reflection 23
6. Discussion 24
7. Conclusion 26
8. Acknowledgements 27
9. References 27

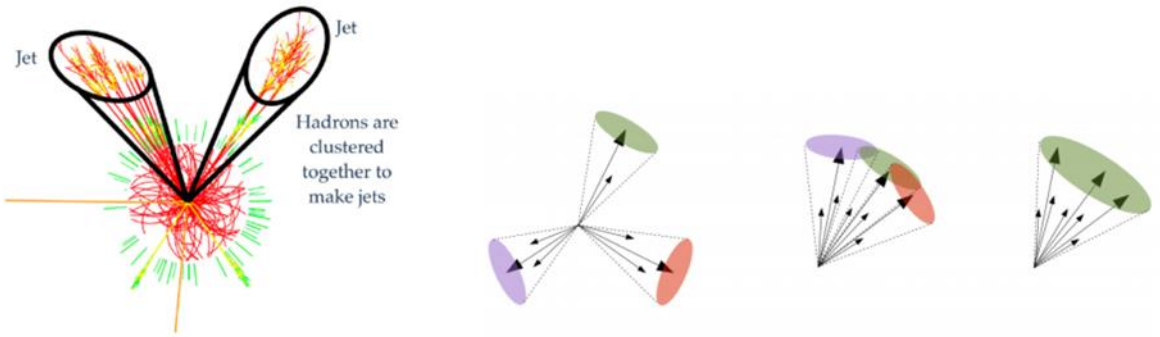
1. Introduction

High Energy Physics (HEP) experiments, conducted at state-of-the-art facilities like the CERN Large Hadron Collider (LHC), provide us with invaluable insights into the fundamental building blocks of the universe. Among the many tools and techniques employed in HEP, jet images [1] play an important role in the extraction of critical physics information from the aftermath of high-energy particle collisions.



Map of the LHC at CERN.

Jets, in this context, refer to the clusters of particles known as quarks and gluons, the fundamental constituents of matter, which are generated during these busy environments and extreme collisions. They offer a glimpse into the behavior of quarks and gluons inside the protons and provide a unique lens through which we can explore the universe's most fundamental forces and particles. However, raw jet data from these experiments is complex, high-dimensional, and challenging to analyze in direct and simple analysis schemes.



Typical cone-shaped jet reconstruction examples (left). Jets from the decay products grow more colinear as the momentum of a decaying particle increases (right). [Ref.]

In recent years, the application of deep learning has demonstrated promise in jet images classification, outperforming traditional techniques [2]. However, it is of importance to note that quality of the data provided to these models plays a pivotal role in determining their performance. This brings us to the idea of data pre-processing, which is a fundamental stage in the deep learning pipeline. Techniques such as noise reduction, image normalization, and feature extraction can generally impact the classification accuracy of deep learning models significantly.

The objective of this study is to delve into the significance of data pre-processing in the context of jet image classification using deep learning. We aim to explore not only the importance of these pre-processing techniques but also their impact and usefulness in enhancing the performance of deep learning models in the context of jet images.

2. Event Generation

Event generation has been done through mainly two packages: Pythia [3] and MadGraph [4]. MadGraph is responsible for the physics process generation. Pythia is then used for even generation of the hadronization process of parton showers, which are the cascading emissions of quarks and gluons as they evolve from high-energy initial states to stable hadrons.

The following commands were used in MadGraph to generate 50,000 signal events for jets coming from top quark pairs (top / ttbar jets):

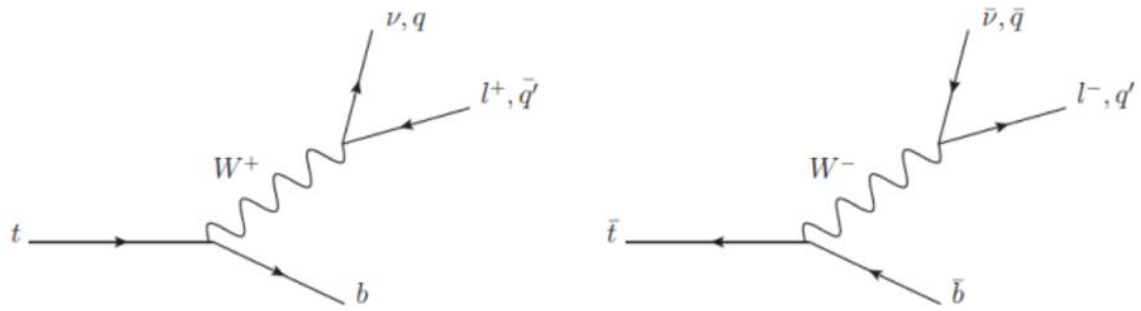
```
generate p p > t t~              Generating the top anti-top events
                                 from proton-proton collisions at 13
                                 TeV, the considered beam energy in
                                 this study.

output ttbar_2jets               Creating the output directory:
                                 "ttbar_2jets" (signal).

Launch                           Start the simulation.

done

set nevents 50000                The considered number of events in
                                 this study.

decay t > w+ b, w+ > j j         Force top quark to decay into w+
                                 boson and b-jet where the w+ boson
                                 should explicitly decay into
                                 hadronic mode.

decay t~ > w- b~, w- > j j       Force anti-top quark to decay into
                                 w- boson and anti-b-jet where the
                                 w- boson should explicitly decay
                                 into hadronic mode.

set pt_min_pdg {{ 6: 490 }}      Minimum pT cut for pdg ID=6
                                 corresponding to the top quark: 500
                                 (- 10 tolerance delta)

set pt_max_pdg {{ 6: 710 }}      Maximum pT cut for pdg ID=6
                                 corresponding to the top quark: 700
                                 (+ 10 tolerance delta)

done
```

The `run_card` for each production was verified to contain the generator-level cuts (all cut values are in GeV).

The following commands were used in MadGraph to generate 50,000 background events for jets coming from gluons (QCD / jj jets):

```
generate p p > j j      Generating the two jets events from proton-
                        proton collisions at 13 TeV, the considered
                        beam energy in this study.
output QCD_2jets        Creating the output directory: "QCD_2jets"
                        (background)
launch
done
set nevents 50000
set ptj 490             Minimum pT for the jets: 500 (- 10 tolerance
                        delta)
set ptjmax 710          Maximum pT for the jets: 700 (+ 10 tolerance
                        delta)
done
```

The output of MadGraph is captured in LHE files, which are then taken as input for Pythia, which is used to get all final state particles for each event.



Feynman diagram that represents the possible decay modes of the top quark pairs.

3. Clustering

Clustering jets refers to the process of grouping or clustering particles that are produced in a particle collision experiment. Along this process, it is important to mention that two approaches were taken for clustering in this study, SlowJet [5] and FastJet [6].

3.1. SlowJet

SlowJet is a component that is bundled with Pythia. It is a simple program that performs jet finding and particle clustering by applying either of the kT, anti-kT, and Cambridge/Aachen algorithms. It is a much simpler alternative to the popular library FastJet. SlowJet was initially chosen for its simplicity and immediate availability, as Pythia was already being used by us. However, it was no longer adequate due to the requirements of some of the pre-processing techniques that were later implemented, namely the need for decomposing the leading jet into sub-jets.

3.2. FastJet

FastJet is a package that, like SlowJet, implements jet finding by applying one of several algorithms, such as the ones mentioned above for SlowJet. It was employed in the study for its jet substructure analysis capabilities that some of the pre-processing techniques require.

4. Data Pre-processing

As previously discussed, data pre-processing can have a significant impact on the performance of deep learning models in classification tasks. Consequently, several pre-processing techniques have been selected after considering other research publications [1, 7 – 10].

4.1. Pre-pixelation or post-pixelation?

The question of whether the order in which different pre-processing techniques are performed matters or not is an intriguing one to consider. Since pixelation is a lossy operation, it is particularly crucial to discuss whether some pre-processing techniques should be performed before pixelation or after pixelation. For lossless transformations, such as translation and normalization, it typically doesn't matter whether they are pre pixelation or post pixelation, as the output is identical if these transformations are applied correctly. However, as will be demonstrated later, some techniques are more lossy when performed post-pixelation as opposed to pre-pixelation, such as rotation and zooming since they introduce interpolation.

4.2. Centering (Translation)

Centering involves setting a common point as the pivot for images that will be utilized for analysis. One typical example is the centering of eyes in facial recognition applications [11]. In order for the deep learning model to focus on structural differences, which are what we are most interested in, it must be made sure that it does not learn distinctions based on overall position. It is important to note that since centering requires particle translation in the $\eta - \phi$ coordinate system, the particle intensity is chosen to be the transverse momentum (pT) of the particle as it is independent of the particle's position [7].

The standard method for centering in jet image pre-processing is to select the pseudorapidity and azimuthal angle of the *leading sub-jet* as the center of the jet image [1, 7 – 9]. However, because SlowJet was initially used for jet finding, no information on the sub-jets was available. As a result, the *leading jet*'s pseudorapidity (η) and azimuthal angle (ϕ) were used as an alternative center for the jet image. Since this study focuses on the effects of data pre-processing, we were motivated to include the center in the list of study points.

Therefore, two approaches were considered in this study:

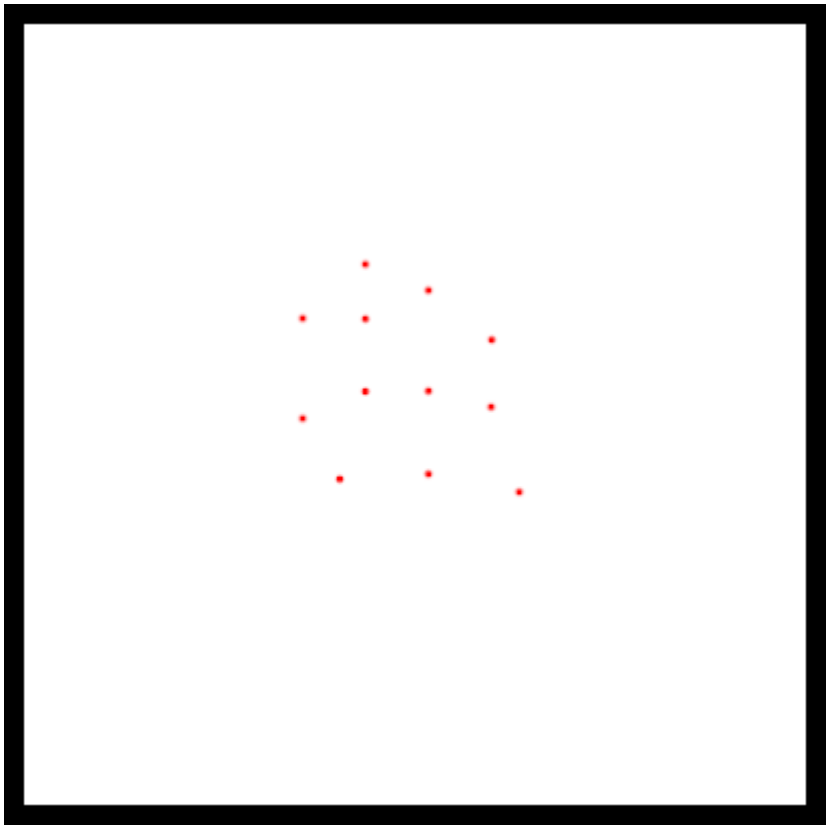
- Using SlowJet & *leading jet*'s pseudorapidity (η) and azimuthal angle (ϕ) as the center of the jet image.

- Using FastJet & *leading sub-jet*'s pseudorapidity (η) and azimuthal angle (ϕ) as the center of the jet image.

4.3. Pixelation

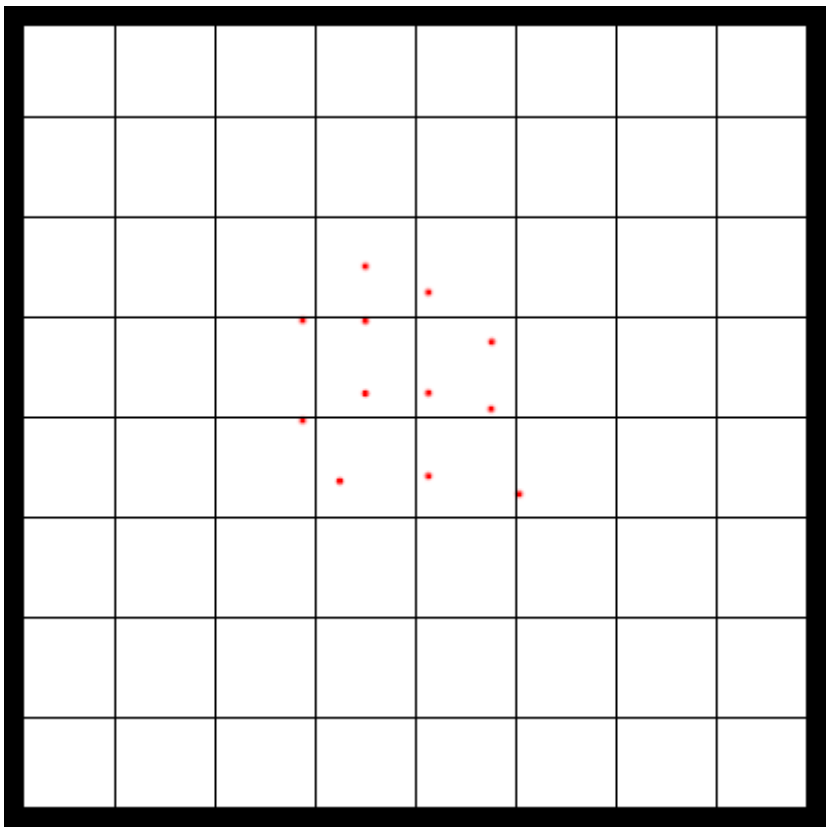
The most common approach for pixelation of the jet data into images in most sources available relating to jet images is what we call the "histogram" approach as it works by fitting the data into a histogram and then using the count in each box as its value, as will be explained below.

Consider the following image:



This image represents the distribution of particles in some space.

It’s desired to capture that distribution of particles in an 8x8 histogram:



… where each histogram cell corresponds to a pixel from an image. Only one intensity value (color) can exist for each pixel each. But as we can see from the image above, some cells may contain multiple particles. One may ask, which particle’s intensity should be used as the pixel intensity? The histogram approach solves this dilemma by setting the pixel intensity to the *sum* of intensities of only the particles that fall inside the cell.

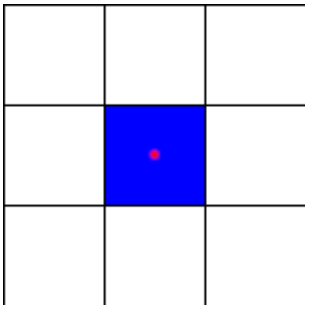
In the above example, if we assume the intensity of each particle is 1, the pixel intensities in the histogram would be as follows:

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	0
0	0	1	2	3	0	0	0
0	0	1	1	1	1	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

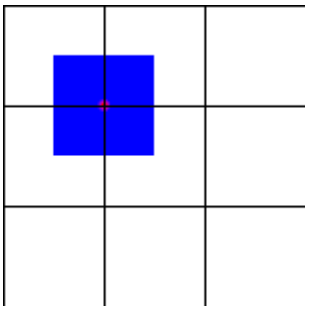
However, if this diagram is intended to give a visual representation of the jet, it seems odd that each cell contains the sum of intensities of only the particles inside it, regardless of how far these particles were from the center of that cell. What if a particle is practically shared between two cells? The histogram method states that only one cell gets the intensity of the particle, and it gets it whole. Therefore, we decided to test out a different method for pixelating the data as an image.

Consider that each particle has an area of 1x1 (in pixels) around it. The intensity of each pixel that intersects this area would be the ratio of the overlapping region's area and the particle's area (and the ratio is then multiplied by the particle's intensity).

Visual examples:



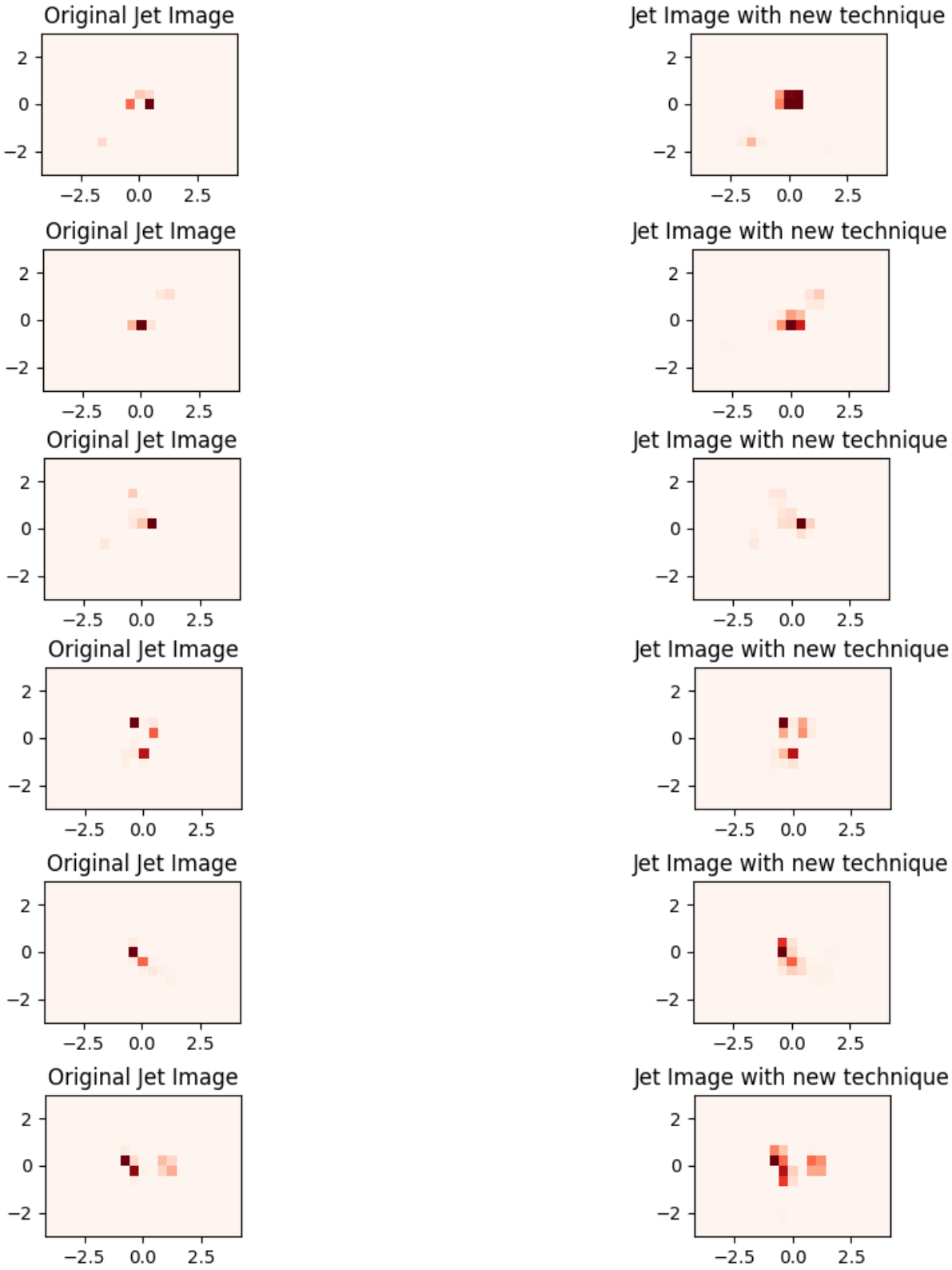
0	0	0
0	1	0
0	0	0



0.25	0.25	0
0.25	0.25	0
0	0	0

This preserves the total intensity of the entire image (the sum of the intensities of all pixels is equal to the sum of intensities of all particles in the data).

Here is a comparison on actual jet images:



Furthermore, if we print the sum of all intensities for one of them:

```
print("Sum of all pixel intensities with old technique:", np.sum(image))

print("Sum of all pixel intensities with new technique:", np.sum(new_image))
print("Sum of all jet constituents' pT:", np.sum(jet_constituents_pt))
```

… they all match:

```
Sum of all pixel intensities with old technique: 681.4094451040016
Sum of all pixel intensities with new technique: 681.4094451040016
Sum of all jet constituents' pT: 681.4094451040015
```

Old technique here refers to the original histogram technique (a single cell is the sum of all particles inside of it). New technique here refers to the ratio-of-areas technique.

As can be seen, this new method allows for the creation of images with greater detail at low resolutions while maintaining the total sum of intensities (i.e., no new colors were introduced, they were simply better distributed). However, it is a fact that there will be fewer visual differences between these techniques the more the image size / resolution increases. This makes sense, as a higher image resolution means less likelihood of several particles being present in a single pixel.

4.4. Normalization

Normalization is one of the standard pre-processing techniques in Computer Vision. It is a process that transforms pixel intensities from one range to another.

4.4.1. L^2 Normalization

L^2 Normalization is the normalization method typically used in Jet Image applications [1, 7 – 9], such that it is desired that the sum of squared pixel intensities is equal to one. It is implemented by dividing each pixel or particle intensity by the L^2 Norm [12]. L^2 Norm is computed as the square root of the sum of the squares of all pixel or particle intensities.

$$x_{normalized} = \frac{x}{\sqrt{\sum_{i=1}^{n} {x_i}^2}}$$

Where x refers to some pixel or particle's intensity, and n is the number of pixels in the image or particles in a jet.

4.4.2. Min-max Normalization

Another common method of normalization is the min-max normalization [13]. The result is that all values are transformed into the range [0, 1]. It is implemented as follows:

$$x_{normalized} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

Where x_{min} is the minimum pixel or particle intensity (zero) and the x_{max} is the maximum pixel intensity in the image or particle intensity in the jet.

4.5. Rotation

Rotation can be useful in removing the random aspect of the decay angle relative to the $\eta - \phi$ coordinate system [1]. Typically, rotation is performed such that the direction connecting the axes of the leading two sub-jets is rotated until the leading sub-jet is directly above the sub-leading sub-jet [1, 7 – 9]. However, in the case in which the jet finder (e.g., FastJet) is unable to find at least two sub-jets, rotation is done instead by determining the principal axis of the original image and rotating the image such that the principal axis is always vertical [1, 7 – 10].

In practice, two implementations of this strategy have been identified, and we were interested in comparing their performance. The first implementation identified comes from the jet-generation project [14], a framework used for building jet images used in many research projects. The second implementation is based on the deepjets project [15].

4.6. Reflection

Reflection is the final pre-processing stage relevant to this study. In simple terms, the image is mirrored over the vertical axis such that the right side of the image always has the greater transverse momentum [1, 7 – 10]. By ensuring that the strongest radiations always appear in consistent locations, the deep learning models may benefit during training.

5. Analysis

5.1. Models

Three deep learning models of varying complexities, common for classification tasks, were employed for the analysis conducted in this study: Logistic regression, Multilayer perceptron, and Convolutional Neural Network. (Based on the work in Ref. 16)

5.1.1. Logistic regression

A linear approach known as logistic regression is widely used for binary and multiclass classifications. It models the probability that a given input belongs to a specific class as log-odds and converts it to a probability score between 0 and 1 using logistic functions such as the sigmoid function. This model has a single layer with a linear transformation, followed by a logistic activation function.

5.1.2. Multilayer perceptron

The Multilayer perceptron (MLP) is a type of feedforward artificial neural network that is capable of approximating complex nonlinear functions. It consists of multiple layers of interconnected artificial neurons with a nonlinear kind of activation function, such as tanh or relu. In MLPs, one or more hidden layers are often present, together with dropout layers for regularization to prevent overfitting.

5.1.3. Convolutional Neural Network (CNN)

Another type of feedforward artificial neural network is the Convolutional Neural Network (CNN). Characterized by its use of convolutional layers to extract features in a hierarchical structure, it reduces the need for handcrafted feature engineering. (This implies that *less* pre-processing of the data is required, as will be demonstrated later.)

5.2. Nature and Scheme of Analysis

The purpose of the analysis in this study is to examine the effects of various combinations of pre-processing procedures on the performances of the three selected models.

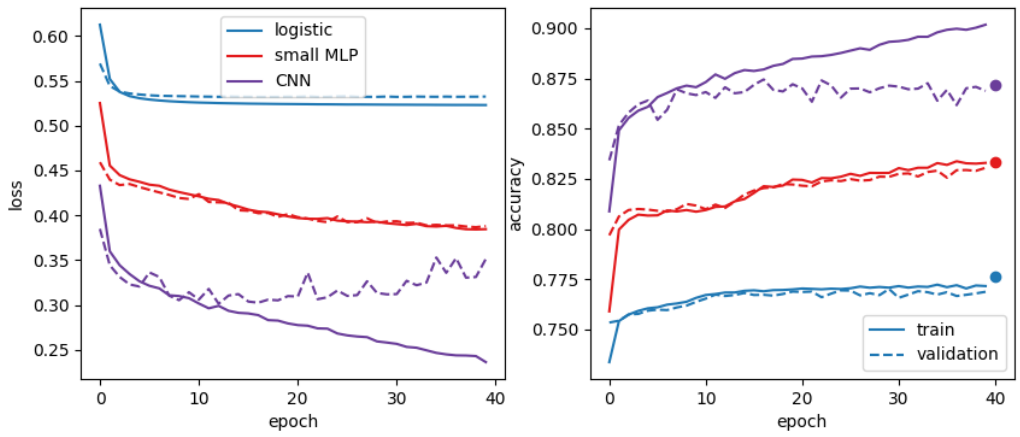
5.3. Performed Tests

The following are the different tests that have been performed:

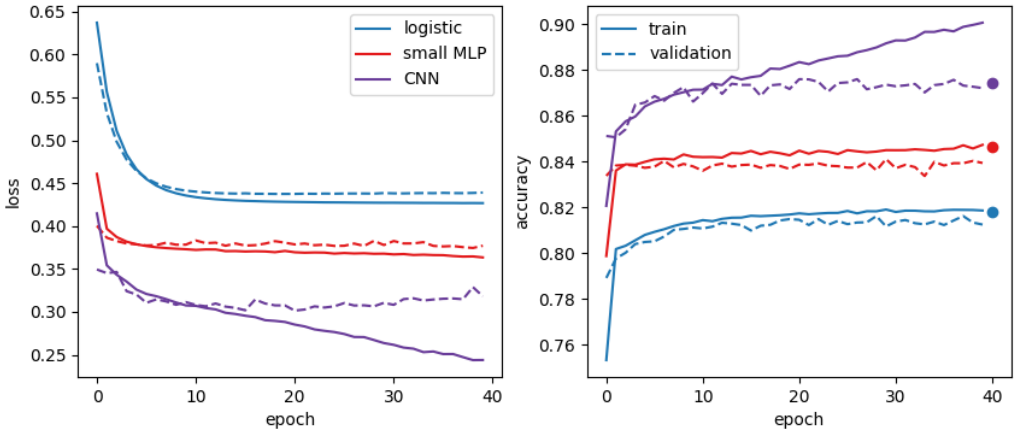
1. SlowJet & Centering on leading jet & histogram pixelation.
2. FastJet & Centering on leading sub-jet & histogram pixelation.
3. FastJet & Centering on leading sub-jet & ratio-of-areas custom pixelation. (All following tests use this as a base.)
4. Rotation post-pixelation (jet-generation method).
5. Rotation post-pixelation (deepjets method).
6. Rotation pre-pixelation (jet-generation method).
7. Rotation pre-pixelation (jet-generation method) & Reflection.

All tests have been performed with min-max normalization, with effects of normalization being shown in tests 2 & 3. The dataset used includes 30,000+ QCD jets and top jets, with an 80:20 split for training and validation purposes.

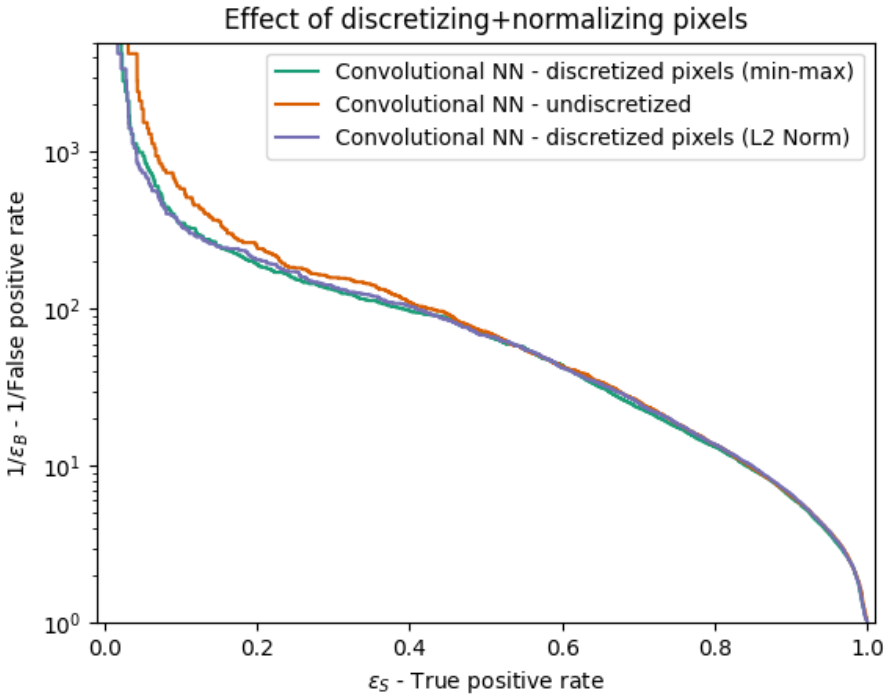
5.3.1. SlowJet & Centering on leading jet & histogram pixelation



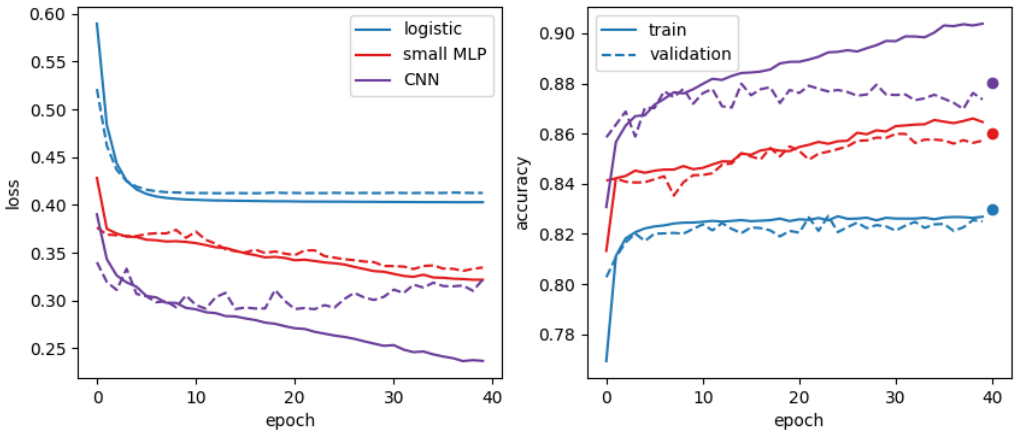
5.3.2. FastJet & Centering on leading sub-jet & histogram pixelation



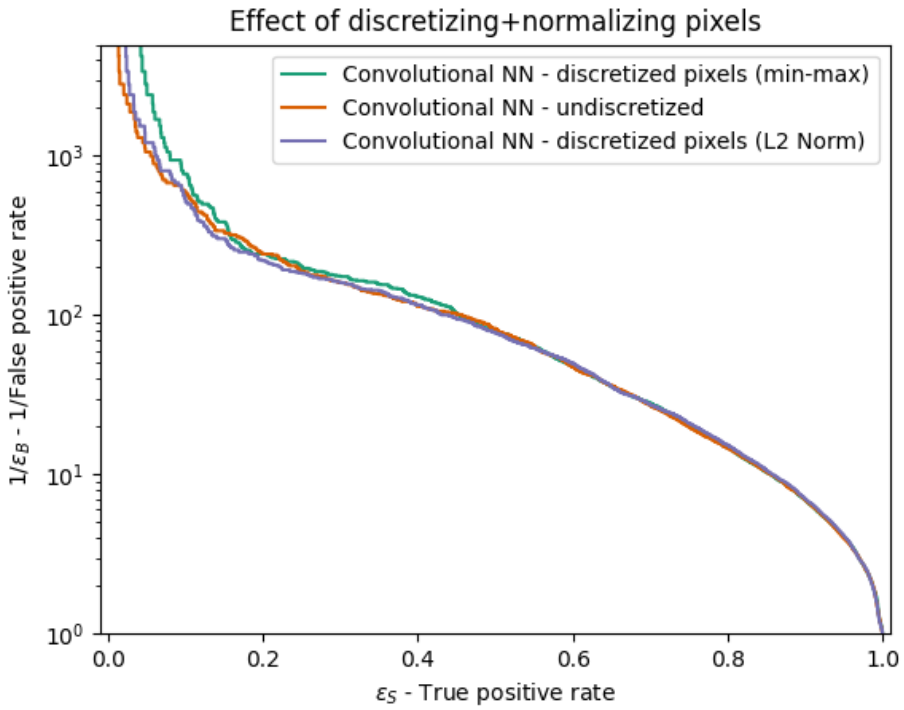
5.3.2.1. Effects of normalization



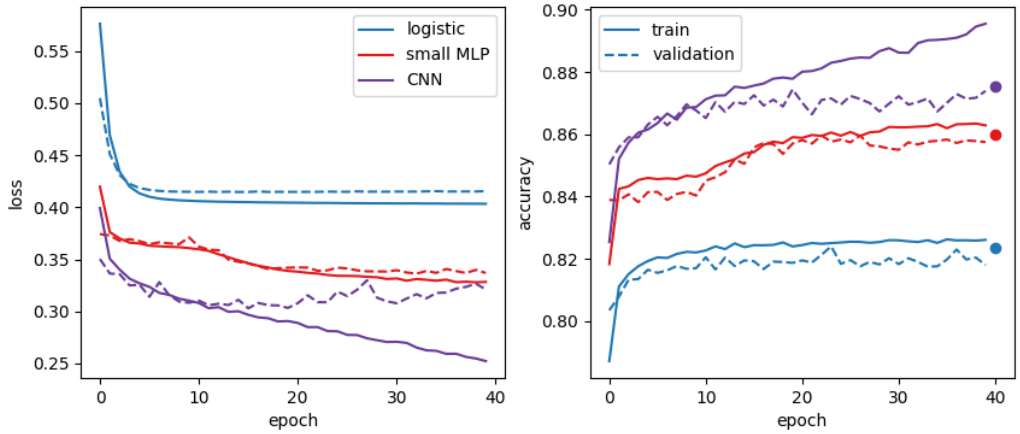
5.3.3. FastJet & Centering on leading sub-jet & ratio-of-areas custom pixelation



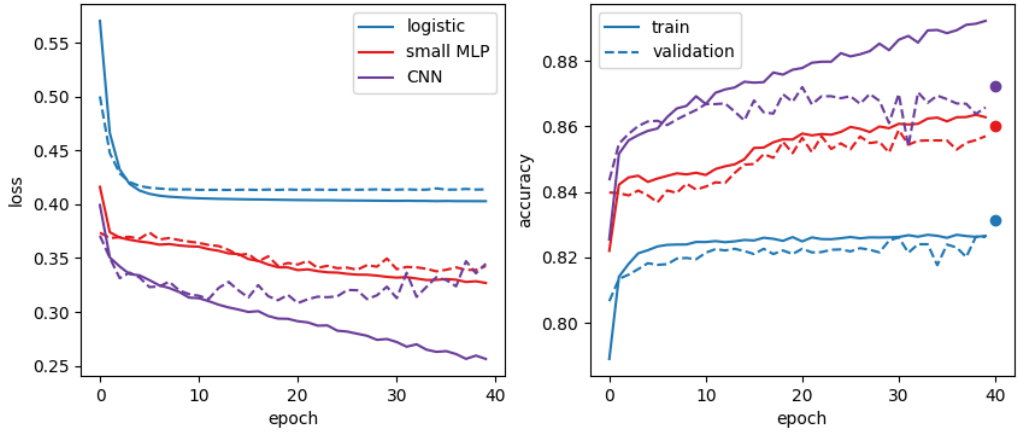
5.3.3.1. Effects of normalization



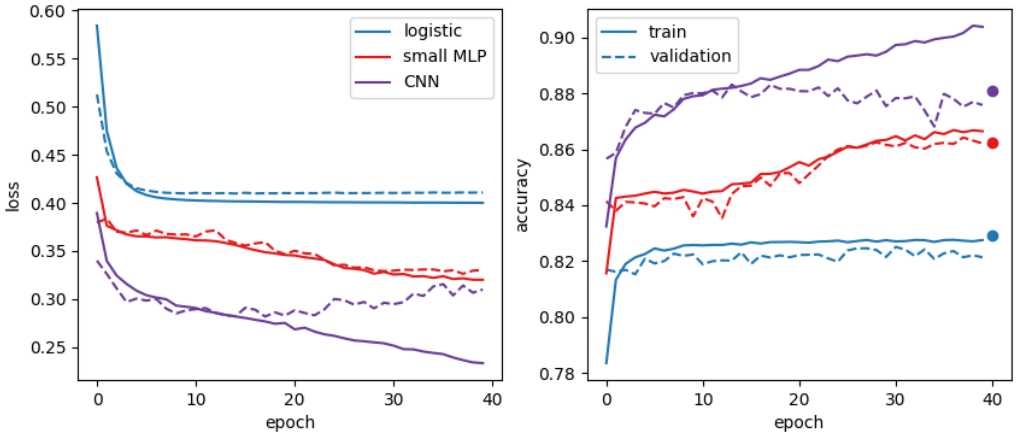
5.3.4. Rotation post-pixelation (jet-generation method)



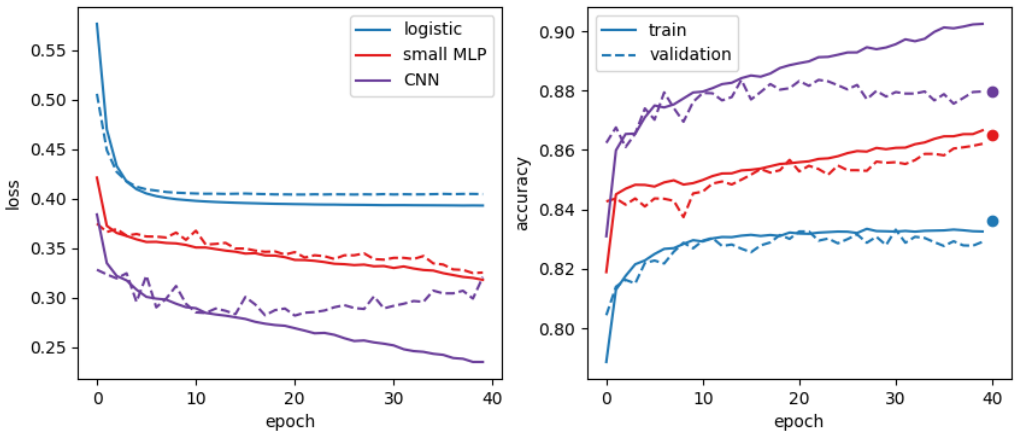
5.3.5. Rotation post-pixelation (deepjets method)



5.3.6. Rotation pre-pixelation (jet-generation method)



5.3.7. Rotation pre-pixelation (jet-generation method) & Reflection



6. Discussion

Many interesting things can be seen from the results. For starters, a trend that can be seen in the logistic and MLP models is that the accuracy was almost always increasing between epochs in all tests, ***and*** between each test overall, whereas the loss was almost always decreasing between epochs in all tests, ***and*** between each test overall. (The exception being the deepjets post-pixelation rotation method, which would either perform the same as the jet-generation pre-pixelation rotation method, or worse ever so slightly) The relationship between accuracy and loss is as follows:

	High Accuracy	Low Accuracy
Small Loss	Small errors on few data (best case)	Small errors on a lot of data
Big Loss	Big errors on few data	Big errors on a lot of data (worst case)

However, a trend that can be seen in the CNN model is that the loss starts as decreases, but eventually starts increasing again after some epochs, while the accuracy remains roughly the same. This is usually a sign of overfitting, and it can perhaps be fixed by early stopping.

In the logistic model, the biggest improvement comes from switching from SlowJet and centering on the leading jet to FastJet and centering on the leading sub-jet. In the MLP model, switching from the histogram pixelation method to the ratio-of-areas custom pixelation method is what resulted in the biggest improvement. As for the CNN, the combination of rotation and reflection is what resulted in the biggest improvement, although it was minor.

In general, it appears that the CNN is the least affected by pre-processing. This is to be expected as that's one of the key points of CNN, which is that it has a lesser need for handcrafted feature engineering, as mentioned previously.

While the CNN was slightly affected by the switch from SlowJet and centering on the leading jet to FastJet and centering on the leading sub-jet, the custom pixelation did not have that much effect in comparison to the histogram pixelation. However, even though post-pixelation rotations improved performance in comparison to no rotation whatsoever in the logistic and MLP models, post-pixelation rotation caused a relative decrease in performance of the CNN. In all models, pre-pixelation rotation performed better than post-pixelation rotation. For the CNN model, the impact of applying pre-pixelation rotation and reflection in comparison to not applying them is very minimal that rotation and reflection may be ignored for CNN models for the time being.

Overall, the best performing pre-processing combination for each model was:

- Logistic: FastJet & Centering on leading sub-jet & ratio-of-areas custom pixelation & Rotation pre-pixelation (jet-generation method) & Reflection. (Test 7, 83% accuracy)
- MLP: FastJet & Centering on leading sub-jet & ratio-of-areas custom pixelation & Rotation pre-pixelation (jet-generation method), with or without Reflection. (Tests 6 & 7, 86.1% accuracy)
- CNN: FastJet & Centering on leading sub-jet & ratio-of-areas custom pixelation & Rotation pre-pixelation (jet-generation method) & Reflection. (Test 7, 88% accuracy)

7. Conclusion

In conclusion, this study has provided valuable insights into the impact of data pre-processing techniques on the classification performance of deep leaning models for jet images with reconstructed data in High Energy Physics (HEP). Three different models were examined: Logistic regression, Multilayer perceptron, and Convolutional Neural Network (CNN).

Initially, it was observed that all three models exhibited reasonably good performance even without any data pre-processing, underscoring the potential of these models for jet image classification tasks. However, the investigation into the effects of various data pre-processing techniques gave interesting results.

For the Logistic regression and Multilayer perceptron models, the application of pre-processing techniques led to some improvements in classification accuracy. In contrast, the CNN model displayed a resilience to the effects of pre-processing. Applying certain pre-processing steps, such as post-pixelation rotation, resulted in a slight degradation in its classification performance compared to the unprocessed data. This outcome aligns with the expected behavior of CNN models, which are engineered to handle data efficiently and automatically learn relevant features from the raw input.

Overall, the findings of this study highlight the nuanced relationship between pre-processing and deep learning models in the context of jet images classification. While pre-processing techniques can offer performance improvements for some models, such as Logistic regression and Multilayer perceptron, they may not necessarily benefit models like CNN, which are optimized for handling data automatically. Therefore, researchers should carefully consider the choice of pre-processing techniques based on the specific deep learning model being employed and the nature of the data.

8. Acknowledgements

I would like to extend my deepest gratitude to my supervisor, Dr. Rachik Soulah, as well as Prof. Abbes Amira, the Dean of the Computing & Informatics College at the University of Sharjah, for their support and encouragement. They were constantly giving me feedback and diligently monitoring my progress, which I highly appreciated. Furthermore, I would also like to show my deep appreciation of CERN for providing such great and extensive opportunities through their Summer Student program. Before I started working on this project, I must admit that my personal enthusiasm for the field of machine learning was somewhat limited. However, thanks to this program, which gave me first-hand experience on the subject, my opinion on it changed a bit. Therefore, I'm genuinely grateful that I was able to participate in this program. Finally, I would like to extend my gratitude to the Professors of University of Sharjah that have given valuable lectures throughout my time as an undergraduate, and I would not have made it here without them.

9. References

[1]
Josh Cogan, Michael Kagan, Emanuel Strauss and Ariel Schwartzman, Jet-Images: Computer Vision Inspired Techniques for Jet Tagging [arXiv:1407.5675]

[2]
Pierre Baldi, Kevin Bauer, Clara Eng, Peter Sadowski and Daniel Whiteson, Jet Substructure Classification in High-Energy Physics with Deep Neural Networks [arXiv:1603.09349]

[3]
PYTHIA

[4]
MadGraph5_aMC@NLO

[5]
SlowJet

[6]
FastJet

[7]
Luke de Oliveira, Michael Kagan, Lester Mackey, Benjamin Nachman and Ariel Schwartzman, Jet-Images — Deep Learning Edition [arXiv:1511.05190]

[8]
James Barnard, Edmund Noel Dawe, Matthew J. Dolan and Nina Rajcic, Parton Shower Uncertainties in Jet Substructure Analyses with Deep Neural Networks [arXiv:1609.00607]

[9]
Michael Kagan, Image-Based Jet Analysis [arXiv:2012.09719]

[10]
CMS Collaboration, Identification of heavy, energetic, hadronically decaying particles using machine-learning techniques [arXiv:2004.08262]

[11]
W. Zhao, R. Chellappa, P. J. Phillips and A. Rosenfeld, Face recognition: A literature survey, ACM Comput. Surv. 35 (2003) 399.

[12]
L^2 Norm – from Wolfram MathWorld.

[13]
Normalization | Machine Learning | Google for Developers

[14]
GitHub - lukedeo/jet-generation: Dockerized Jet Image generation

[15]
GitHub - deepjets/deepjets: Main repository for image generation and CNN training https://arxiv.org/abs/1609.00607

[16]
Jet tagging in one hour with convolutional neural networks - Excursions in data (ilmonteux.github.io)