Adversarial Attacks on a Convolutional Neural Network Particle Classifier

Student Name: Suhasan Kanagasabapathy

CID: 01338402

Supervisor Name: Abigail Waldron

Assessor Name: Morgan Wascko

Word Count: 5300 words

Declaration of work undertaken:

The project was executed by three people (myself and two others: Shawn Tan and Harvey Cao). There are two parts to this project. Firstly, building a CNN classifier. Each of us built our own version of a classifier. After discussing and comparing, we settled on a final version which was mainly inspired by Shawn Tan's version. Secondly, we split the adversarial attacks to be done. I primarily investigated and implemented dead channel and space charge. Shawn Tan and I had discussions on space charge attacks. Shawn Tan implemented hot pixels, neutron radiation, and cracks in solid detectors. Harvey Cao investigated electronics noise, of which I did not talk about in the report due to constraint in page and word limit. This was agreed upon by our supervisor by email.

Abstract: Convolutional Neural Networks are being widely used in building particle classifier systems in high energy physics experiments. There are several factors in detectors that could alter the observed images of particle tracks. These detector factors are likely to be sources of adversarial examples in a CNN model which could lead to misclassifications. We built a particle classifier based on CNN using a set of simulated images provided. An accuracy of 0.96 ± 0.05 was achieved. The f1-score for each class was close to 1, indicating a good performance of the model. We considered several detector effects: dead channel, hot pixels, neutron radiation, cracks in solid detectors, and space charge. The physics of these effects was discussed. The effects were implemented on the images as adversarial attacks as accurately and as physically as possible. Multiple aspects of each attack were investigated. The response of these adversarial examples on the model built was observed. We found that many of the attacks could indeed cause misclassifications of the particle class. The misclassifications depended heavily on the nature of the detector effects.

Content Page

Summary of work undertaken:

Convolutional Neural Networks are being widely used in building particle classifier systems in high energy physics experiments. There are several factors in detectors which could alter the observed images of particle tracks. These detector factors are likely to be sources of adversarial examples in a CNN model which could lead to misclassifications. Firstly, the simulated data provided was analysed. We built a particle classifier based on CNN using the set of simulated images. The metrics used to evaluate the performance of the model were explained. Next, we considered several detector effects: dead channel, hot pixels, neutron radiation, cracks in solid detectors and space charge. The physics of these effects was discussed. The effects were implemented on the images as adversarial attacks as accurately and as physically as possible. Multiple aspects of each attack were investigated. For example, during the investigation of space charge effects, we studied the influence of its size, intensity, location and different shapes. The response of these adversarial examples on the model built was observed. As a conclusion, we found that many of the attacks could indeed cause misclassifications of the particle class. The misclassifications depended heavily on the nature of the detector effects.

1.0 Introductory Chapter

Convolutional Neural Networks (CNN) have been widely used to solve complex problems in computer vision and image recognition. With the abundance of data available in experimental particle physics, the use of CNN will be particularly beneficial in detecting and predicting the type of particles observed. However, like any other particle detectors, there is a possibility of errors occurring during the detection and analysis processes. These detector effects will likely alter the observed data in various ways. An adversarial attack is a procedure to make an image, which is initially correctly classified, to be classified wrongly in a machine learning model. We aim to investigate the detector effects by creating adversarial examples using the data given. Firstly, we construct a CNN model to classify simulated data containing the images seen by the detector. The physics and background of these detector effects are studied. The goal is to simulate these effects on the data given as accurately and as physically as possible. We study the effects of the adversarial examples produced on the CNN model built, to identify the attacks which will result in successful misclassifications.

2.0 The Data

High energy physics experiments use a variety of detectors to detect, track, and/or identify ionizing particles [1]. Our investigation focuses on the ionization tracks left behind by charged particles, particularly electrons and muons. A total of 3000 simulated images, containing the tracks of these particles were provided at the beginning of the project. The data comprises images for three classes: electrons, muons and background with no particles' signal present. Each class has 1000 examples. Each image is made up of an array of 50 by 50 pixels, with each pixel represented by a value. The values are represented by a colour bar in the plot. In this investigation, the values were arbitrary and do not correspond to the typical bit values of the pixels which is from 0 to 255. Some examples of these pictures are given in figure 1.

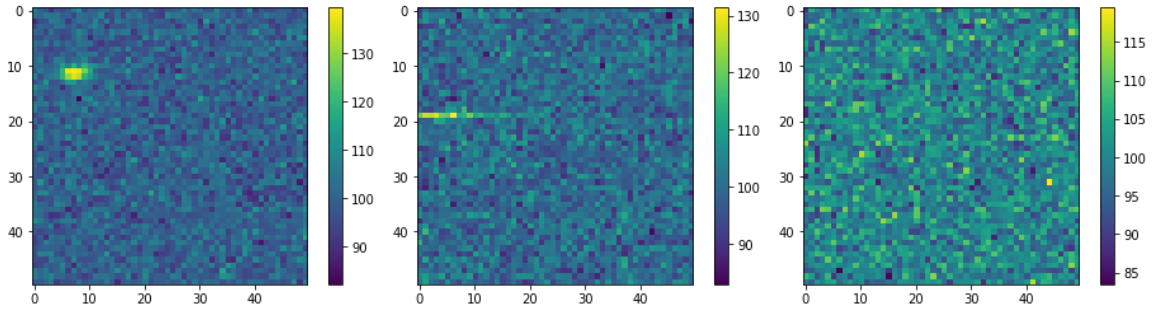

Figure 1: (Left) Simulated image of the track of an electron, which shows a circular area of high values, indicating a shorter track. (Centre) Simulated image of the track of a muon, which shows a longer and narrower track compared to an electron. (Right) Simulated image of the background. It has lower pixel values compared to electrons and muons and is uniformly distributed in the array, signifying no signal found. The mean pixel value is about 100 with a standard deviation of 5.

3.0 The Classifier

Our aim is to build a particle classifier using machine learning techniques and the simulated data given above. The features mentioned in each class will be helpful in identifying the particles. To highlight the features, we built a convolutional neural network. A python class object named "Particle Classifier" was created to ease the investigation. The summary of the procedure is shown in figure 2. Firstly, the simulated images were split into training and test sets with the proportion of test sets being 0.2 of the data given. Both sets of images were rescaled as part of preprocessing. The mean value of the images was subtracted from each image, then they were scaled to have values between -1 and 1, and the negative values were removed. Initially, we planned to apply a Median filter and a Gaussian filter to remove noise and hot channels. Though, later we decided to make the filters optional since they prevented us from doing single-pixel attacks on the images as the filters blur them. Additionally, the labels for each set were one hot encoded to ease prediction.

The CNN model itself was constructed with 3 convolutional layers with each followed by a maximum pooling layer. The convolutional layers preceded 3 hidden layers consisting of 512, 128 and 32 neurons, respectively. The activation function used in each hidden layer was 'ReLU'. The output layer consisted of 3 neurons to signify 3 expected outputs. The 'softmax' activation function was implemented at the output layer since it can be used in determining the probability of multiple classes at once. The model was then compiled and trained for 10 epochs, using 20% of the training set as the validation set. The choices for the parameters in the model were selected after several trials to achieve the optimal result for accuracy. The trained model was saved as a .h5 file so it can just be loaded without having to train again whenever an attack was executed.

To execute the adversarial attacks on the model built, a function was made to apply the transformation needed on a copy of the test set as shown in figure 2. The attacked images were then preprocessed, and the trained model predicted its outcome.

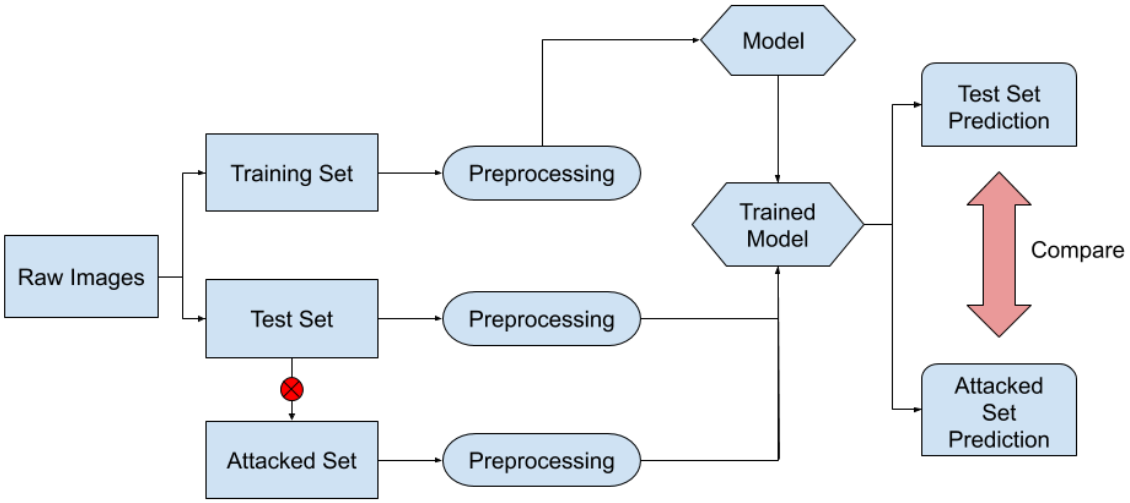

Figure 2: Summary of the procedure executed. The red circle represents the adversarial attack.

4.0 The Metrics

The classifier model built was evaluated and the classification report is shown in table 1. The first metric considered is accuracy, which is the ratio of the number of correct predictions to the total number of input samples [2]. Here, the correct prediction is compared against the original labels of the images. It works well if there are equal numbers of training samples belonging to each class, which is the case here. The accuracy achieved is 0.96 ± 0.05, meaning 96% of test samples were correctly classified. Precision is the number of correct positive results divided by the number of positive results predicted by the classifier. Recall is the number of correct positive results divided by the number of all relevant samples. F1-score is the harmonic mean of precision and recall [2]. It measures the overall performance of the model for a given class.

	Precision	Recall	F1-score
Background	0.96	0.97	0.96
Electron	0.98	0.98	0.98
Muon	0.95	0.94	0.95

Table 1: The classification report for the classifier model. The f1-score for each class is close to 1, indicating a good performance of the model.

The evaluation of the attacked images was done by comparing its prediction to its pre-attack prediction (not its original label). This allowed us to verify that the misclassification was indeed successful. To illustrate the comparison more clearly, a confusion matrix was created for each unique attack as shown in the first attack in figure 5. Each value corresponds to the ratio of the images of a particle class which is correctly identified after the attack to the correctly identified images before the attack. The closer the fraction is to 1, the brighter the box in the confusion matrix appears. The higher the diagonal values, the better the performance of the model is. We also

introduced total accuracy in the attack, which represents the fraction of correctly identified attacked images out of the originally correctly predicted ones in all classes.

5.0 The Adversarial Attacks

An adversarial attack is the process of modifying an image which was initially classified correctly by a standard machine learning model. An effective adversarial attack successfully results in a misclassification if the modified image is to be inputted into the model again [3]. The modification does not need to be visible on the image as the attack could still be successful if the change is invisible to the human eye. An example of a successful adversarial attack is shown in figure 3. The investigation focuses on non-targeted adversarial attacks as we do not want to target the outcome of the attack to be a specific class. Various detector effects that could arise in high energy physics experiments could become the sources of adversarial attack in particle classifiers. In this investigation, we consider the following sources of adversarial attacks: dead channel, hot pixels, neutron radiation, cracks in solid detectors and space charge.

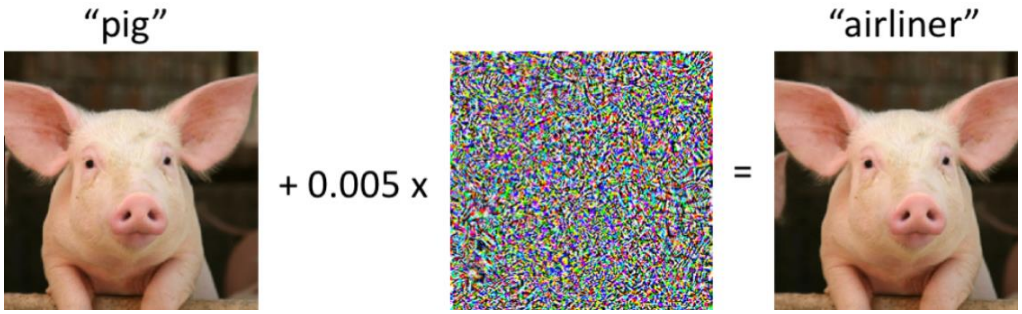

Figure 3: An example of an adversarial attack, obtained from [4]. The original picture on the left was correctly identified as a pig by a classifier. It was added to a noise which might look random, but it was constructed so it would confuse the classifier. The classifier now predicts the modified image as an airliner. Here, there is no visible change in the post-attack image from the original one.

5.1 Dead Channel

Particle detectors have basic sensors which detect photons or electrons and convert them into electrical signals [1]. These signals will then be analysed to produce the images shown previously. Each pixel value in the image corresponds to the measurement by a unique sensor. It is possible that the sensor at a particular location will be dead due to several reasons such as spoiled sensors, incomplete circuit and other physical damages to detectors.

The dead channel effect is simulated by changing the pixel value at a chosen location in an image. However, the pixel value chosen for a dead pixel could not be zero. This is because there could still be noise due to electronics even though the sensor reads ‘zero’. The solution is to use a 50 by 50 array of Gaussian noise shown in figure 4. The value at a particular location is chosen from this array, instead of setting it to zero. For now, the attack targets the pixel with maximum value in the image to effectively target the signal (note this does not work in ‘background’ images). Firstly, a single dead pixel attack was executed as shown in figure 5. The size of the dead pixels was increased into a 2 by 2, 4 by 4 and 6 by 6 arrays as shown in figure 6, 7 and 8 respectively.

Overall, the background images were unaffected while electron images were the most affected as the size of the dead pixels gets bigger.

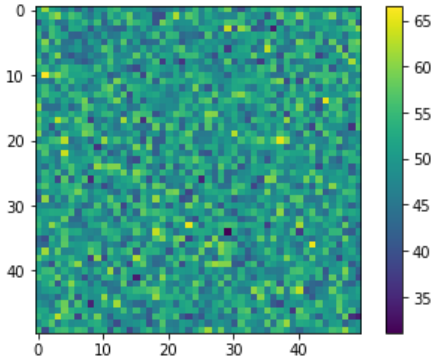

Figure 4: The electronics noise N_0 in a 50 by 50 array, obtained from random.normal method from NumPy. The mean of noise due to electronics is 50 and the standard deviation is 5. These values were estimated by assuming half of the mean in 'background' images given was due to electronics.

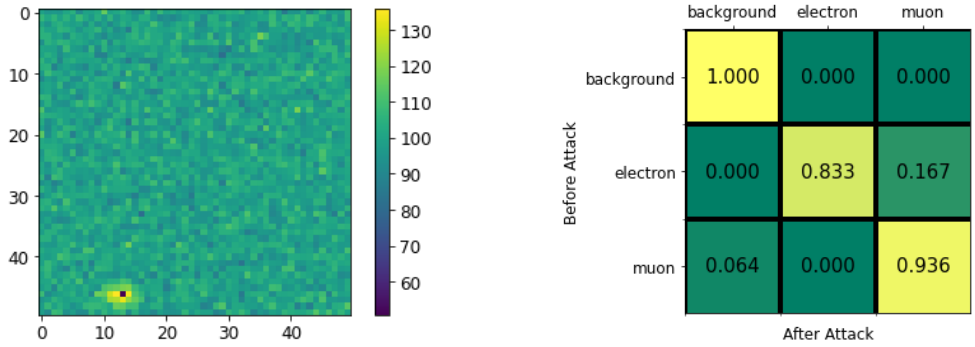

Figure 5: (Left) An example of the single-pixel attack. Originally the image contains the signal for an electron. The maximum pixel (i.e. signal) value was changed to a much lower value (not zero). (Right) The confusion matrix for the single-pixel attack. The total accuracy achieved was 0.91 ± 0.10. Most of the muons and background were unaffected. About 16% of the electrons were classified as muons.

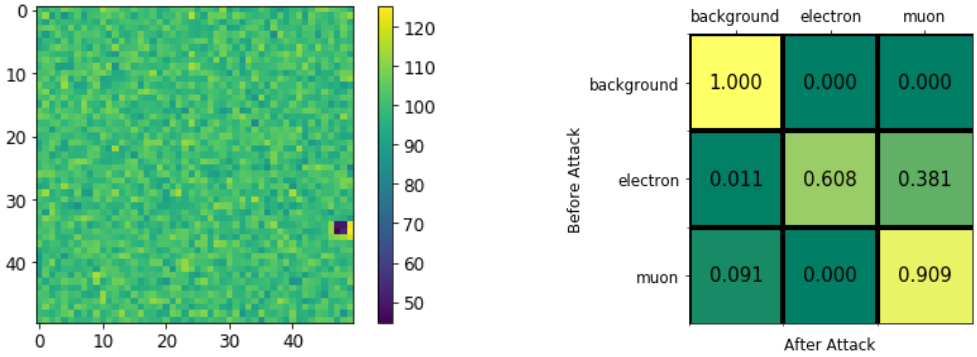

Figure 6: (Left) An example of 2 by 2 dead pixel attack on an electron image. (Right) The corresponding confusion matrix. The total accuracy achieved was 0.83 ± 0.10. As before, most of the muons and background were unaffected. 38% of the electrons were classified as muons.

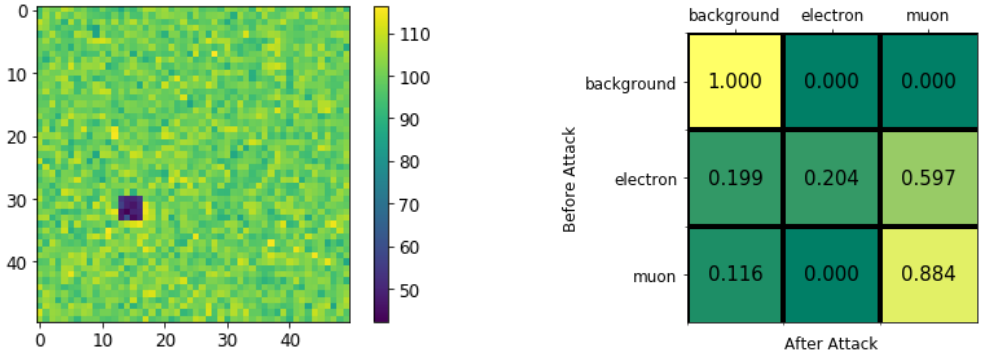

Figure 7: (Left) An example of 4 by 4 dead pixel attack on an electron image. (Right) The corresponding confusion matrix. The total accuracy was 0.70 ± 0.10. This time more electrons were classified as muons. The accuracy on muon classification dropped too. A small fraction of muons is classified as background due to a more comparable size of the attacked region to muons.

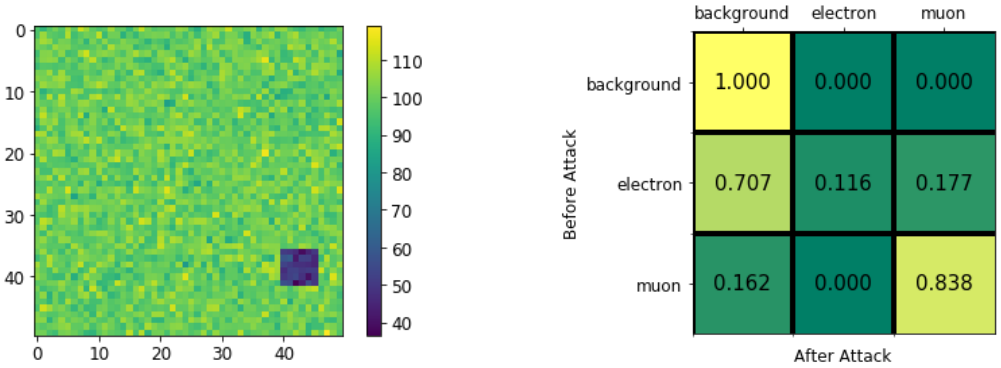

Figure 8: (Left) An example of 6 by 6 dead pixel attack on an electron image. (Right) The corresponding confusion matrix. The total accuracy was 0.65 ± 0.10. The size of the dead pixels is now bigger than most electrons, causing most of them to be classified as background.

Next, we target vertical and horizontal stripes of pixels to be dead. This might occur when all the horizontal/vertical sensors are connected in series and the circuit is broken. The attack still occurs on the maximum pixel. Figure 9 and 10 show the attack on vertical and horizontal pixels respectively. Interestingly, horizontal dead pixel attacks cause more than twice as many misclassifications as vertical pixel attacks do. This suggests the nature of simulated images given in the beginning is biased towards the horizontal orientation of the particles. Finally, we also explored the extreme case of all pixels being dead and is shown in figure 11.

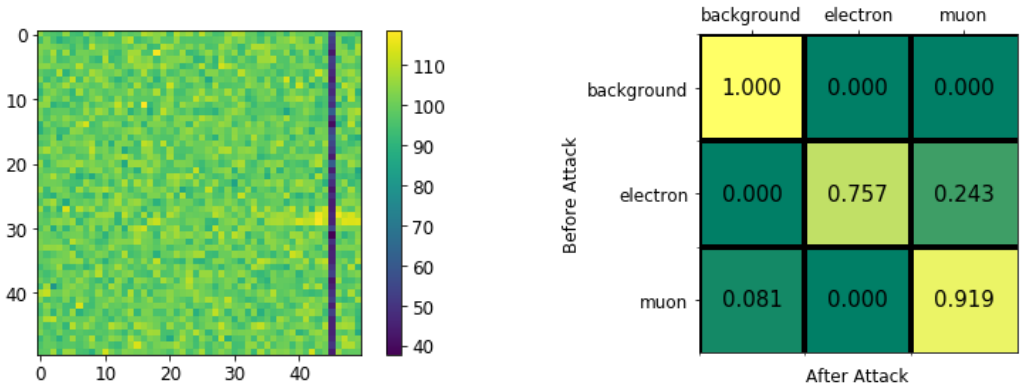

Figure 9: (Left) An example of a dead pixel attack done on the vertical stripe of the pixel with maximum value in a muon image. (Right) The confusion matrix of the corresponding attack. The total accuracy was 0.87 ± 0.10. The electrons were the most affected as about 24% were misclassified as muons. This could occur when the stripe of dead pixels on an electron produces two “streaks” that look like muons.

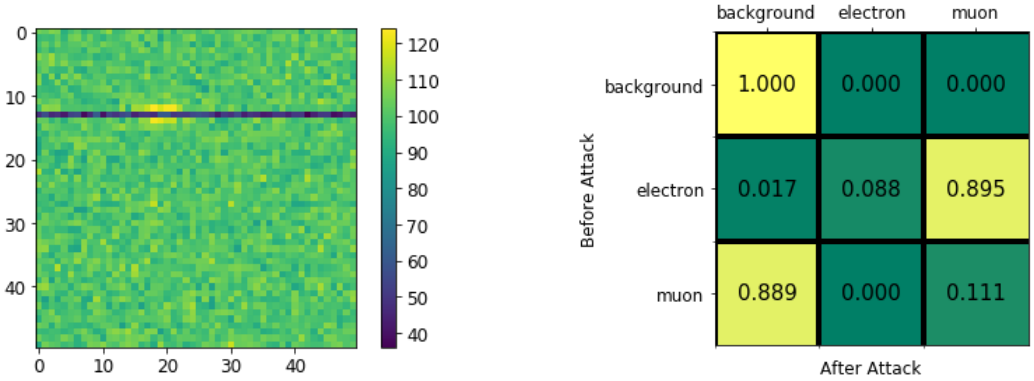

Figure 10: (Left) An example of a dead pixel attack done on the horizontal stripe of the pixel with maximum value in an electron image. (Right) The confusion matrix of the corresponding attack. The total accuracy was 0.41 ± 0.10. Now, almost 90% of all electrons are misclassified as muons. Unlike previously, about 89% of all muons are classified as background. This indicates the muons tracks are primarily horizontal and have a width of one pixel, so the dead pixels could completely remove their signals.

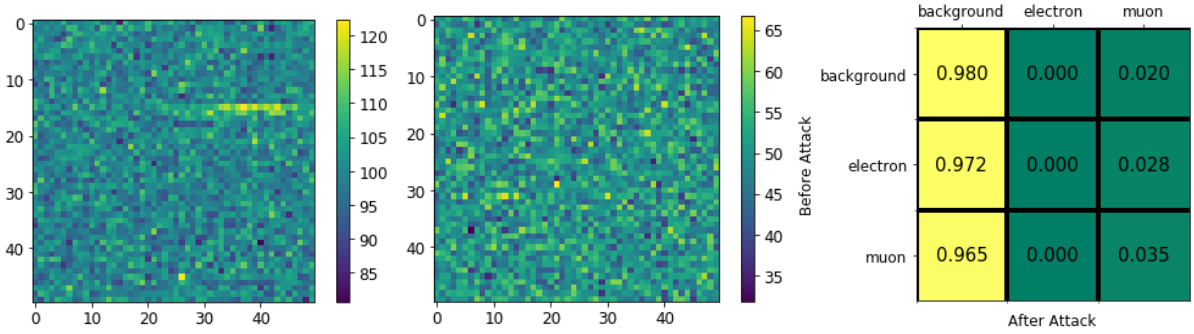

Figure 11: (Left) A pre-attack image of a muon. (Centre) A post-attack image of the muon, where all pixels are dead. As expected the attacked image is just the noise as shown in figure 4. (Right) The corresponding confusion matrix. As expected, the accuracy was low at 0.34 ± 0.10 and most electrons and muons were classified as background.

5.2 Hot Pixels

Hot pixels are individual pixels that appear brighter than they should. They form when there are electrical charges that leak into the sensor walls [5]. Since hot pixels stand out among their neighbours, it can easily be mistaken for a signal in a particle classifier, especially when the total pixel count is small. The attack was implemented by replacing a random pixel in the image with a value twice the maximum value in all the images. Figure 12 shows an example of the attack as well as the overall result. During the investigation, we also found a defence against a hot pixel attack. Using the Gaussian and median filter in the preprocessing blurs the hot pixel but also reduces the resolution. Almost all attacked images were correctly classified. However, this is not recommended as information might be lost in the process.

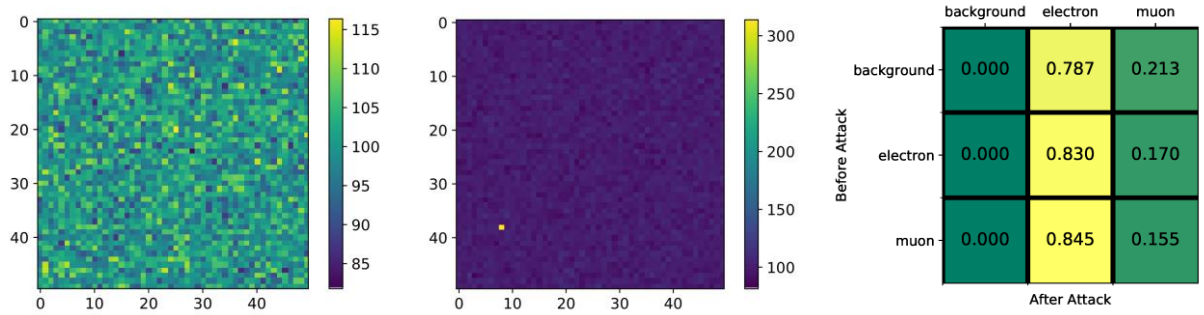

Figure 12: (Left) A pre-attack image of background. (Centre) A post-attack image of the background, where a hot pixel is placed in a random location. Visually, the attacked pixel looks like an electron signal. (Right) The confusion matrix of the attack. As expected, almost all images were classified as electrons. The accuracy was 0.33 ± 0.10. About 21% of the background was also classified as muons.

5.3 Neutron Radiation

Background radiation is a common issue faced by particle detectors since they produce similar signatures as the particles the experiment is looking for. Radiation comes in various forms such as alpha, beta particles and electromagnetic radiation. We focus on neutron radiation since neutrons are uncharged, hence they can travel a longer distance without interacting with other particles. They are also more massive than electrons, thus having a stronger ionising strength. Neutron radiation could occur due to any radioactive impurity present in the detector medium. The implementation of neutron radiation attack is similar to the hot pixel attack previously. But instead of a single pixel, we use a 2 by 2 array as the shape of a neutron track. Unlike a hot pixel, the value assigned to the pixels should correspond to a neutron's energy. Therefore, the value chosen as the average of the maximum value and the mean across all images. The location of the track is randomised. The implementation and result are shown in figure 13. Almost all background images were classified as muons, whereas electrons and muons were largely unaffected.

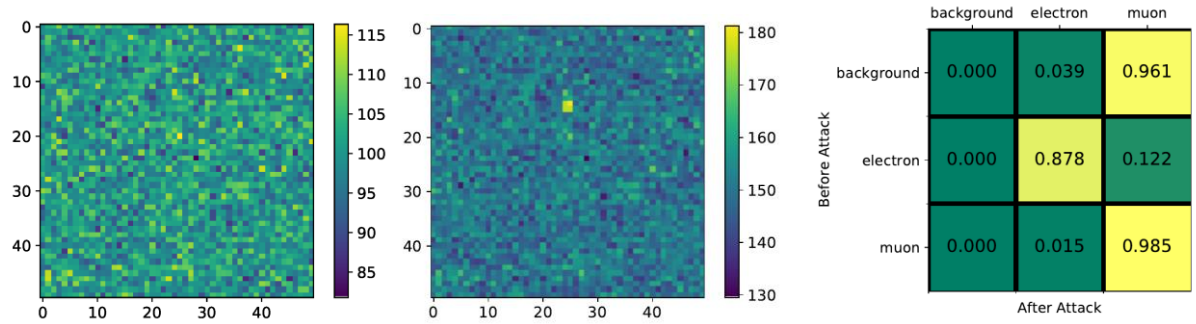

Figure 13: (Left) A pre-attack image of background. (Centre) A post-attack image of the background. The bright 2 by 2 array represents the track left by a neutron ionising the particles in the medium. (Right) The corresponding confusion matrix. The accuracy was 0.64 ± 0.10. The most affected are background images as about 96% of them were misclassified as muons. Whereas, about 12% of the electrons were classified as muons. This could be due to the extension of the electron length in one dimension, resulting in a longer track which resembles a muon.

5.4 Cracks in Solid Detectors

Cracks could occur due to any physical stress placed on solid detectors. These cracks can allow the flow of electric charges on the sensors, similar to the hot pixel case. In this case, the track can be longer than a single pixel. Likewise, the effect is implemented by placing a streak of bright pixels on top of the signal. The value for the track is chosen as the maximum value in the images to appear brighter than the signal present. The beginning of the track is chosen at the maximum pixel. The length of the crack corresponds to the length of the track observed. Hence, the length of the track is varied from a length of 1 pixel to 15 pixels to study its response on the model, as shown in figure 14. Overall, it is clear the length of the track due to crack has a considerable impact on the prediction of the classes.

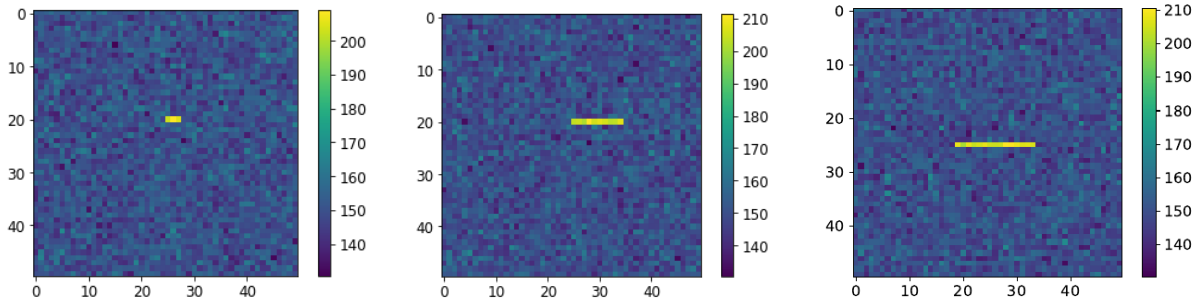

Figure 14: (Left) A post-attack image of the background with a track due to a crack of a length of 3 pixels. (Centre) A track of length of 10 pixels. (Right) A track of length of 15 pixels.

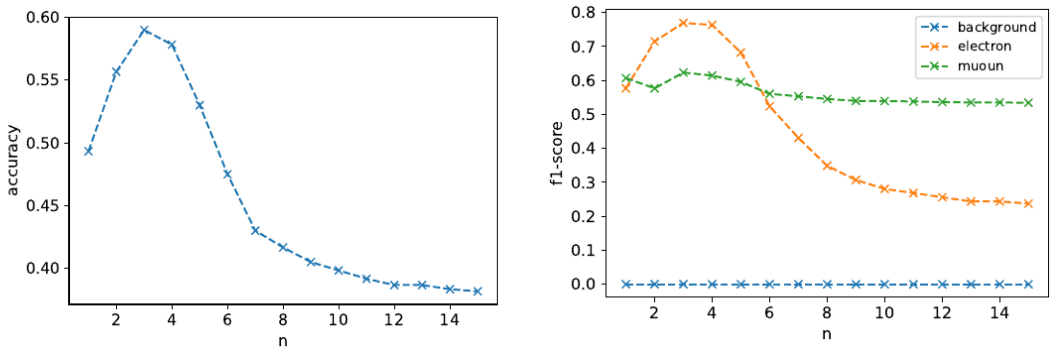

Figure 15: (Left) The post-attack total accuracy as a function of the length of the crack in number of pixels, *n*. The accuracy reached its peak at around *n* = 3, after which the accuracy dropped to around 38%. The uncertainty in each point is taken as 0.10 and we assumed this is the case in all the remaining similar plots. (Right) F1-score against *n*. The background always gets misclassified as in the case of the hot pixel attack. The performance for electron class is the best when *n* = 3. The performance for muon class stayed the same.

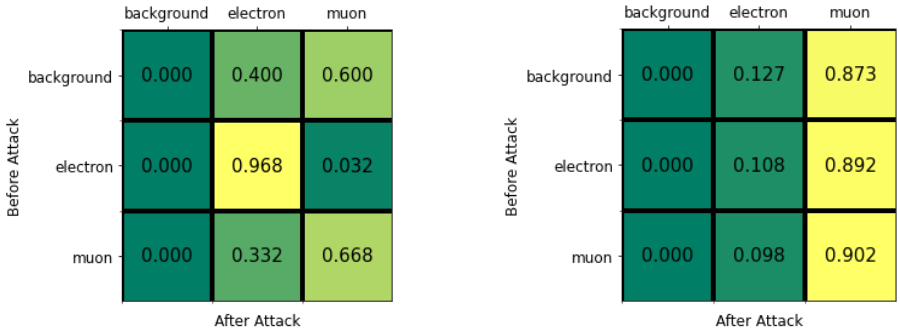

Figure 16: (Left) The confusion matrix for *n* = 3. At this length, the track resembles a short muon or an electron track, resulting in the background to be classified as such. Yet, the track is not long enough to affect the electrons. (Right) The confusion matrix for *n* = 10. Almost all images are classified as muons as the track resembles muons more.

5.5 Space Charge

Space charge is a distribution of excess electrical charges in a region of space [6]. In liquid or gas detector media, space charges could be accumulated during ionization. The charges are primarily large positive ions as they travel much slower than electrons. These accumulated space charges can interact with electrons and photons from subsequent ionisations. This reduces the strength of the signal of those ionisations measured by the sensors. Unlike a dead channel, space charge cannot be implemented by simply removing the signal at a particular pixel. As defined above, the effect needs to have a distribution with a smooth gradient across it.

We consider a kernel function, *G(x, y)* which is given by

$$G(x,y) = e^{-\frac{x_S\,|x-a|^{\alpha} + y_S\,|y-b|^{\beta}}{2\sigma^2}} \quad (1)$$

where *x* and *y*, in this case, correspond to a 2D array of dimension 50 by 50 as shown in figure 17. *a* and *b* are the coordinates for the centre of the space charge. $\boldsymbol{\sigma}$ is the standard deviation of *G*, which corresponds to its size or width. α and β are the exponents in *x* and *y* directions respectively. $\alpha = \beta = 2$ gives a standard Gaussian kernel function and we will be working with that mostly. x_s and y_s are the stretch factor in *x* and *y* directions. *G* is not normalised as it does not matter in our investigation. The space charge effect is captured by *G*', which is defined as *G' = A*G* - 1 as shown in figure 17. *A* corresponds to the intensity of the space charge. The parameters *A, a, b,* α, β, σ, x_s, y_s can be manipulated to produce any space charge effect needed for this investigation. To implement this as an attack, we perform a Hadamard product of *G'* and the image *X*, (i.e. *G' • X*) which is essentially an element-wise multiplication [7].

Similar to a dead channel attack, the sensor reading after the space charge attack cannot be zero as it is not physical. Likewise, the noise due to the electronics should be added to the attacked image. To ensure only the area affected by the space charge to be considered for the noise, we perform a Hadamard product of *G* with the same parameters and the electronics noise array, N_0 as shown in figure 4 (from dead channel attack), i.e. $G \bullet N_0$. The resultant product is shown in figure 17. The attacked image, *X'* is then given by $X' = G' \bullet X + G \bullet N_0$ where *X* is the unattacked image.

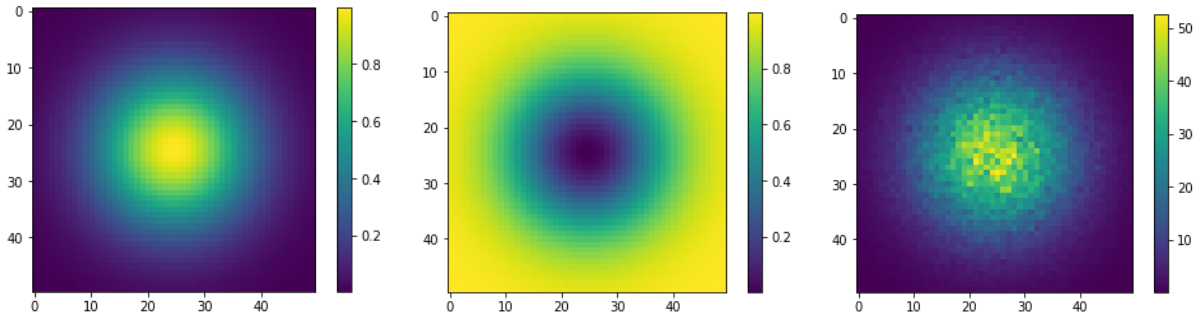

Figure 17: (Left) Kernel function *G* with parameters: $a = b = 25$, $\alpha = \beta = 2$, $\sigma = 10$, $x_s = y_s = 1$. The range of *G* is (0,1). (Centre) Space charge function *G'* which has the same parameters as *G* with $A = 1$. *G'* has a 'well' in the centre as opposed to a 'peak' in *G*. This captures the space charge behaviour well since it reduces the values where it is present and leaves the other areas unaffected. (Right) The Hadamard product of noise N_0 and *G*.

5.5.1 Varying the Size of Space Charge

The numerous parameters in G allow us to study the different attributes of space charge. Firstly, the size of the space charge is varied from $\boldsymbol{\sigma} = 1$ to 100 as shown in figure 18. The parameters used were $\alpha = \beta = 2$, $x_s = y_s = 1$. The choice of *A* is chosen as 0.9 instead of 1.0 as $A = 1.0$ might give negative values for *G'*. As before, the centre of the space charge is located in the pixel with maximum value. The metrics measured for the attack is shown in figure 19. The accuracy generally drops as the size of space charge increases. Yet, all three classes behave differently to an increasing size of space charge.

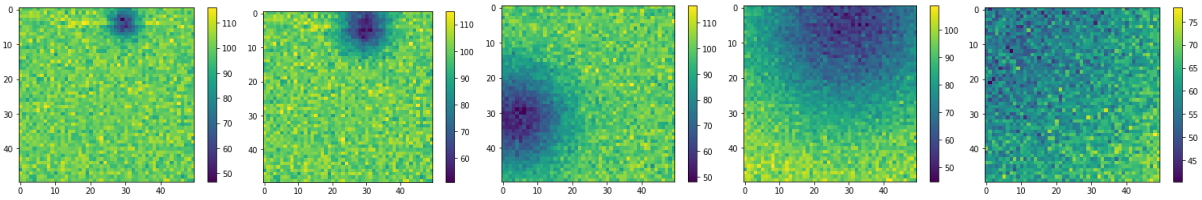

Figure 18: Varying the size of space charge; σ = 3, 5, 15, 25 and 70 respectively. When σ = 70, the space charge effectively blocks most of the image. The minimum values observed correspond to electronics noise as opposed to values close to zero.

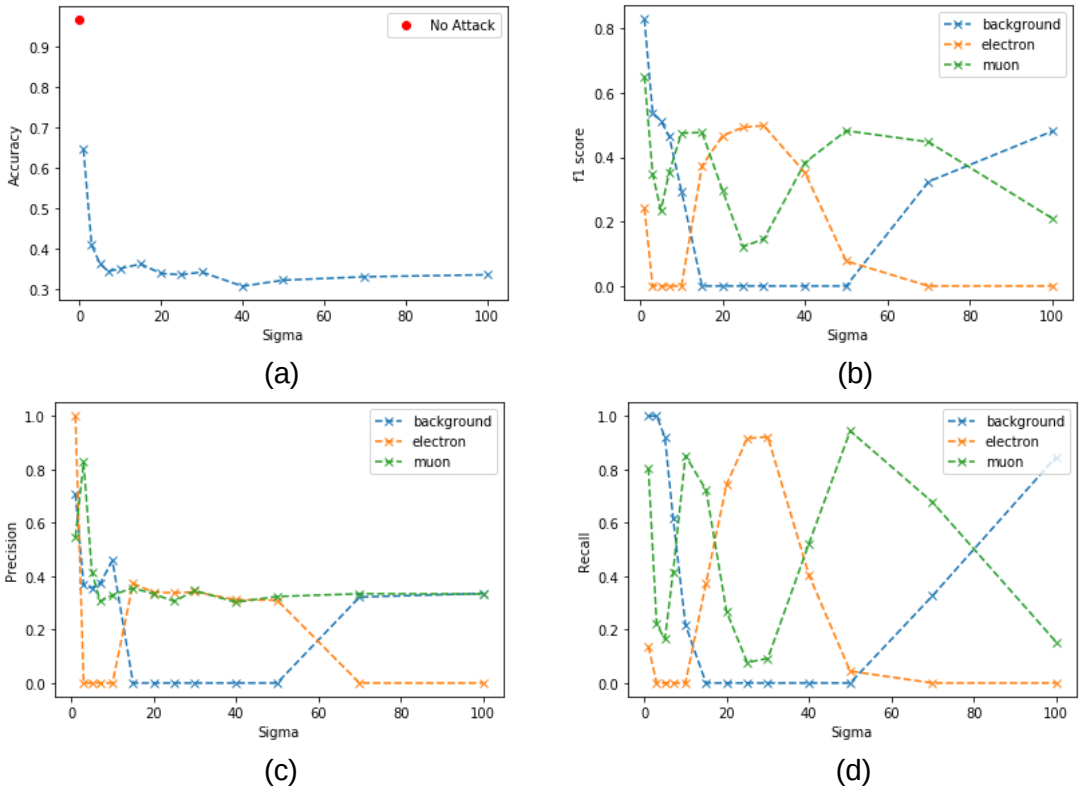

Figure 19: (a) Accuracy against σ (called sigma here). The accuracy decreases rapidly until it levels off to an accuracy of about 0.33 when σ = 7. (b) F1-score as a function of σ for the three classes. For the electron, the performance dropped to zero, rose and peaked when σ = 30 before dropping to zero gradually. For the muon, the performance seemed to fluctuate wildly. Whereas the f1-score of background dropped and rose when σ = 50. This suggests the space charge gets big enough to cover almost all the image, leading the electric noise to be classified as background. (c) Precision against σ. Unlike its recall, the precision of muon stayed constant at about 0.30 after σ = 10. (d) Recall against σ. The trends look similar to the f1-score.

5.5.2 Varying the Intensity of Space Charge

The intensity of the space charge is also varied from A = 0.1 to 0.9 as shown in figure 20. The parameters used were $\alpha = \beta = 2$, $x_s = y_s = 1$. $\sigma = 10$ so that the space charge is in intermediate size. The results are plotted in figure 21. Overall, the results signify that the performance of the classifier worsens as the intensity of the space charge effect increases. The electron class is the most affected for the given size of space charge as this is also witnessed in figure 19 previously.

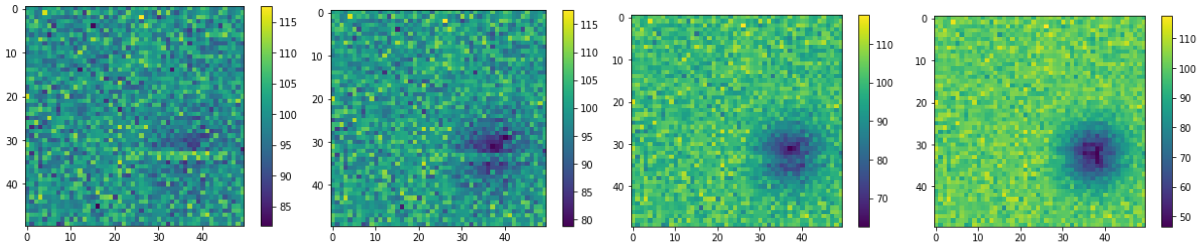

Figure 20: Varying the intensity of space charge; A = 0.1, 0.3, 0.6, and 0.9 respectively on an image of muon track. The higher the intensity of space charge, the more strongly the space charge blocks the signal of the muon track.

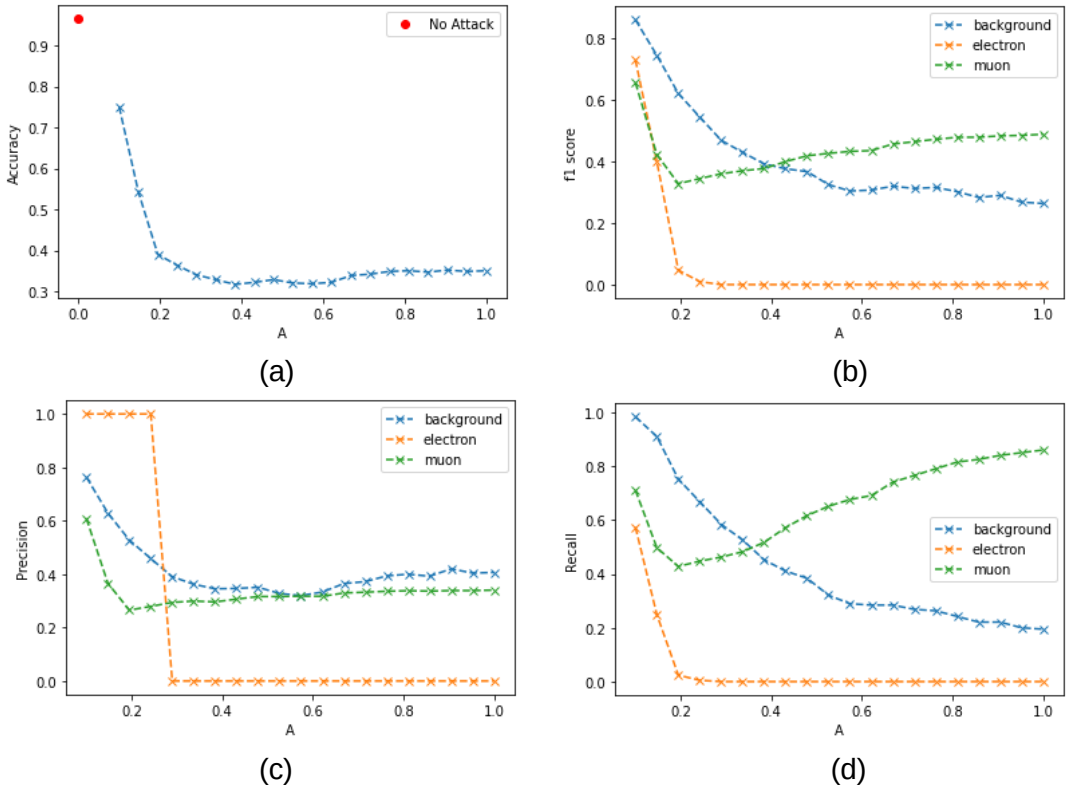

Figure 21: (a) Accuracy as the function of A. The accuracy drops rapidly as A increases, until it levels off at an accuracy of about 0.33 when A is around 0.3. (b) F1-score against A. The electron class has the worst performance as the space charge has a comparable size to an electron track. (c) Precision against A. The precisions for muons and background level off after A = 0.2 and A = 0.5 respectively. (d) Recall against A. After A = 0.2, the recall of muon gradually increases to about 0.80.

5.5.3 Varying the Shapes of Space Charge

Different shapes of space charge were also investigated using different parameters of *G*. Figure 22 and 23 shows the linear and elliptical space charges respectively along with the results. Interestingly, both shapes produce similar results. Compared to the linear shape, the elliptical shape causes more of the background to be classified as electrons than as muons. More than half of muons were misclassified as electrons in both cases. This suggests the model recognises the smooth variation in values across an image as a feature of an electron track. Whereas muon tracks often have an abrupt change at its boundaries. Nevertheless, both the shapes result in the same misclassifications.

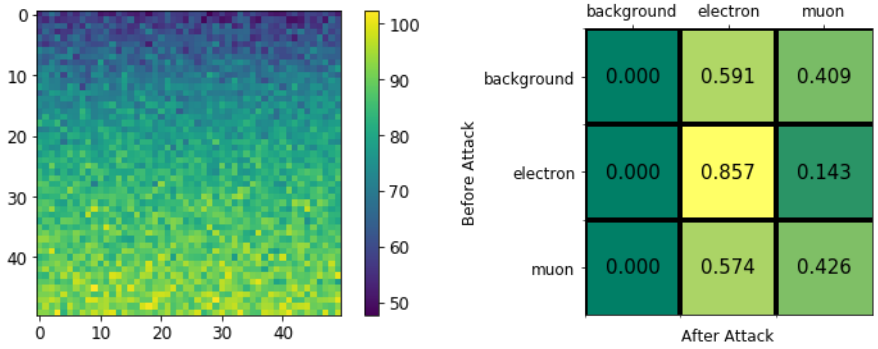

Figure 22: (Left) The linear-shaped space charge. The parameters chosen: A = 0.9, a = b = 0, σ = 4, α = 1.1, β = 0, x_s = 1.1, y_s = 1. (Right) The resulting confusion matrix. The total accuracy was 0.43 ± 0.10.

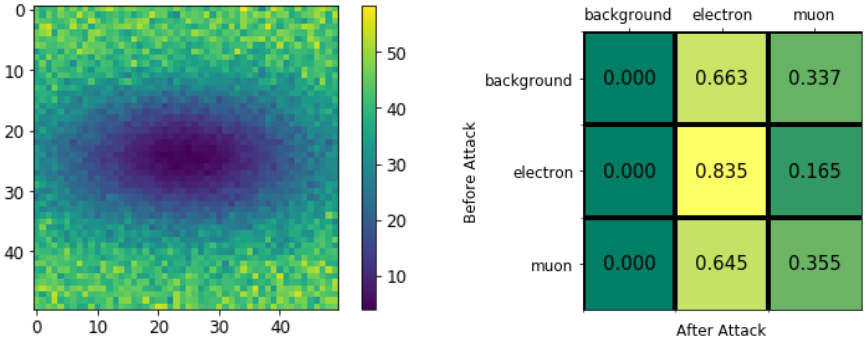

Figure 23: (Left) The elliptical space charge. The parameters chosen: A = 0.9, a = b = 25, σ = 50, α = β = 2, x_s = 30, y_s = 2. (Right) The resulting confusion matrix. The total accuracy was 0.42 ± 0.10.

5.5.4 Varying the Location of Centre of Space Charge

So far, the centre of space charge is fixed on the signal. A space charge with parameters of A = 0.9, σ = 10, α = β = 2, x_s = y_s = 1 was created. The location of the centre of the space charge is randomised within the image. Figure 24 shows the resulting confusion matrix. Next, we investigate the effect of varying the distance *r* of the centre from the signal (i.e. the maximum pixel) as shown in the same figure. From the plots in figure 25, the accuracy generally increases as the space charge is placed further away from the signal. Here, the electron class seems to be the least affected.

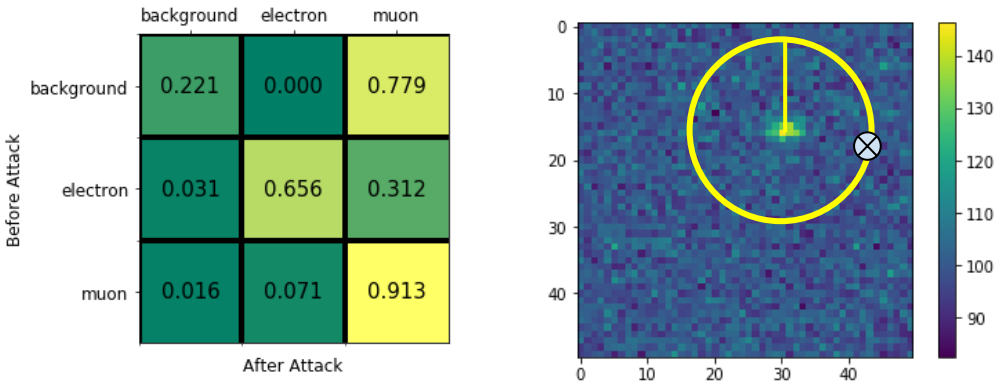

Figure 24: (Left) The confusion matrix for choosing a random location as the centre of space charge. The total accuracy was 0.57 ± 0.10. Most of the background were classified as muons. (Right) The illustration of the attack which varies *r* on an image of an electron for example. The cross represents the centre of space charge which will be randomly placed in that circumference of radius *r*.

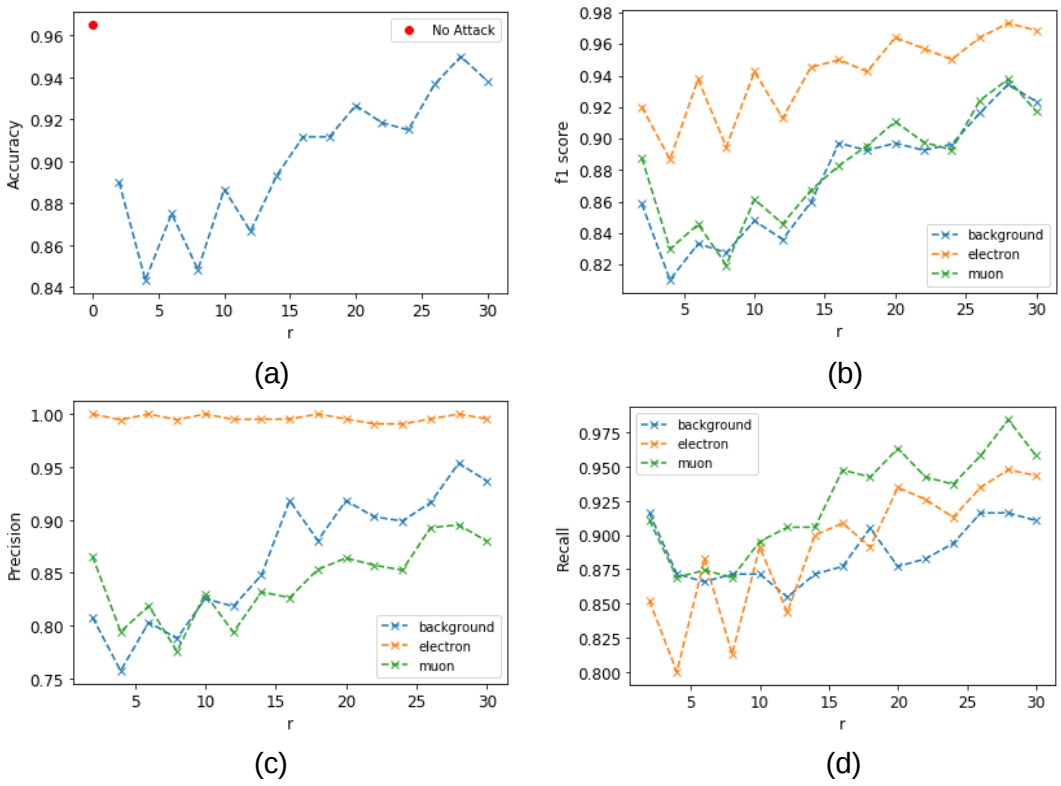

Figure 25: (a) Accuracy against *r*. The accuracy improves as the space charges get further away from the signal. The fluctuations in the values can be attributed to the random nature in choosing the centre in the circumference. (b) F1-score as a function of *r* for the three classes. The electron is the least affected as its track is small. Hence, once the space charge is just a few pixels away from the electron track, the model's performance improves. (c) Precision against *r*. (d) Recall against *r*. Recall of electrons seems lower than the other two classes for small *r*.

6.0 Conclusions

Convolutional Neural Networks are being widely used in building particle classifier systems in high energy physics experiments. There are several factors in detectors which could transform the observed images of particle tracks. These detector factors are likely to be sources of adversarial examples in a CNN model which could lead to misclassifications. We built a particle classifier based on CNN using a set of simulated images provided. An accuracy of 0.96 ± 0.05 was achieved. The f1-score for each class was close to 1, indicating good performance of the model. We considered several detector effects: dead channel, hot pixels, neutron radiation, cracks in solid detectors and space charge. The physics and background of each of these effects were discussed. The effects were implemented on the initially correctly classified images as adversarial attacks as accurately and as physically as possible. Multiple aspects of each attack were investigated. The response of these adversarial examples on the model built was observed. We found that many of the attacks could indeed cause misclassifications of the particle class. The misclassifications depended heavily on the nature of the detector effects. For example, the hot pixels almost always caused anything to be classified as electrons.

In the future, more detector effects could be studied. For example, in a liquid detector, we could investigate the convective and turbulent flow as well as bubbles as possible sources of adversarial attacks. Besides, one could also develop defences against these attacks. We saw using median and Gaussian filters in the preprocessing was effective against single-pixel attacks. One common defence against adversarial attacks is to incorporate adversarial examples in the training set.

Acknowledgements

Thank you to my supervisor, Ms Abigail Waldron for her consistent support and guidance throughout the project. I would also like to thank my project partners for their enthusiasm and corporation in making this project fruitful and full of lessons.

References

[1] M. Tanabashi *et al.* (Particle Data Group), *Particle Detectors at Accelerators.* Phys. Rev. D **98**, 030001 (2018).

[2] Olson, David L.; and Delen, Dursun; *Advanced Data Mining Techniques*, Springer, 1st edition (February 1, 2008), page 138, (2008).

[3] Goodfellow, Ian J., Jonathon Shlens, and Christian Szegedy. “Explaining and harnessing adversarial examples.” arXiv preprint arXiv:1412.6572 (2014).

[4] Design News. *Yes, AI Can Be Tricked, And it's a Serious Problem.* Available from: https://www.designnews.com/electronics-test/yes-ai-can-be-tricked-and-its-serious-problem/161652909959780 [Accessed 14th April 2020].

[5] Lewis Mcgregor. *What is a Hot Pixel and How Can You Remove One?*. Weblog. Available from: https://www.premiumbeat.com/blog/what-is-a-hot-pixel-and-how-can-you-remove-one/ [Accessed 14th April 2020].

[6] Encyclopædia Britannica, inc.. *Space Charge.* Available from: https://www.britannica.com/science/space-charge [Accessed 14th April 2020].

[7] Horn, Roger A.; Johnson, Charles R. *Matrix analysis*. Cambridge University Press. (2012)