Deployment of unsupervised learning in a search for new physics with ATLAS Open Data

Testing of auto encoder for semi unsupervised learning

Sakarias Frette

Spring 2022

Abstract

In this report we implemented an auto encoder using Tensorflow framework for anomaly detection on ATLAS open data. We compare the performance of several models, both using the reconstruction error against the ATLAS open data, and signal Monte Carlo samples with ROC curves. It was showed that using a more complex auto encoder, removing certain low level features, and standard scaling improved the auto encoders ability to filter out certain signals, with a max AUC score of around 0.68. Proposals are put forward to better understand the auto encoder and its use for anomaly detection, one of which is to expand the amount of hyper parameters and the range of them.

University of Oslo

CONTENTS

I. INTRODUCTION

The standard model is arguably the most successful scientific model, showing remarkable accuracy comparing with experiments. The Higgs boson was the last missing piece, and was discovered in 2012 by both ATLAS[1] and CMS[2] at CERN. There has however not been any new particle discoveries since the early 70's, and this is somewhat alarming. The standard model has several issues, some of them includes lack of an explanation of dark energy, cosmic inflation, the hierarchy problem and not including gravity.

This leads scientists to look beyond the standard model, and there are several models proposed, such as dark matter candidates, extra vector bosons and super symmetry. Attempts are made by multiple collaborations to discover evidence for these models, and there are several obstacles to overcome, such as method of analysis and sufficient amount of data. With the impressive progress of machine learning software tools such as Tensorflow[11], machine learning methods have become more and more popular as possible methods to use for searching for new physics.

In the last decade machine learning has excelled from being tedious and hard to program to become available to almost everyone. It's scale-ability and ability to discover hidden structure in large datasets makes it an intriguing candidate to use on data from the Large Hadron Collider. One possible candidate is semi unsupervised learning, which is done by an auto encoder. The idea is to train a model-independent algorithm on datasets containing only known physics, with the hope that what ever new physics that is in the data is detected, without knowing what it is. This machine learning model is the method of choice for this report.

This report is structured in a theory, implementation, results and discussion, and conclusion section. In the theory section the necessary theory is introduced, with respect to both data, method of analysis and choice of evaluation. In the implementation section, data handling and software tools are discussed, with respect to cuts, event selection, scaling and feature choices, choice of hyper parameters for hyper parameter search and the software and api's used for training and tuning. In the results section the results are displayed as well as distinct observations. In the discussion section the results are discussed and proposals are made for future work, and in the conclusion I summarize my findings.

II. THEORY

Some of theory sections about anomaly detection and machine learning algorithms are based on earlier work done in other courses, such as [6] and [5].

A. Anomaly detection

Anomaly detection is a tool with a wide range of uses, from time series data, fraud detection or anomalous sensor data. Its main purpose is to detect data which does not conform to some predetermined standard for normal behavior. The predetermined standard varies from situation to situation, both from the context it self and what is expected as an anomaly. Anomalies are typically classified in three categories[2]:

1. Point anomalies
2. Contextual anomalies
3. Collective anomalies

For the purpose of this report we will mostly consider collective anomalies. Anomalies in high energy physics have to be collective to claim anything, as noise and other factors can and often do create point or event collective anomalies. Through triggers[3], one hopes that

[1] Paper from ATLAS collaboration can be found here.
[2] Paper from CMS collaboration can be found here.
[3] ATLAS experiment website.

most of the noise and other factors are accounted for, but it is not a perfect system.

In high energy physics we can, using machine learning, separate anomaly detection into two categories, supervised and unsupervised searches. Figure 1, 2 and 3 are examples that are not based on actual predictions and are only meant as illustrations.

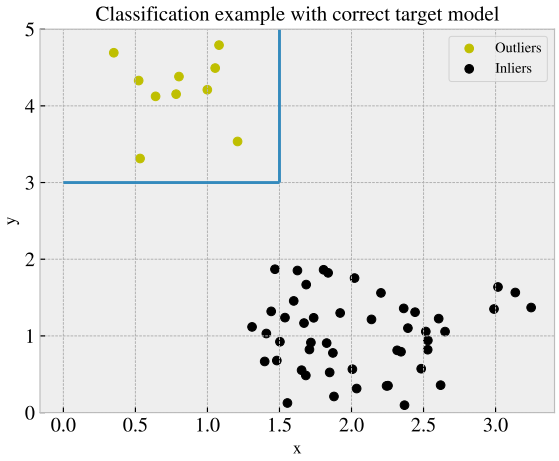

Figure 1: Supervised classification of anomaly where the target model is correct. Here the machine learning model manages to correctly identify the anomalies.

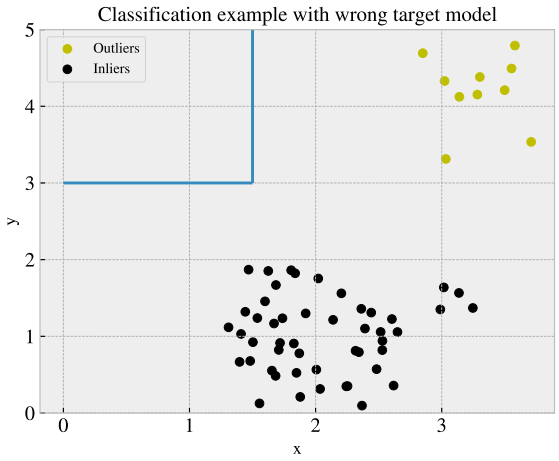

Figure 2: Supervised classification of anomaly where the target model is wrong. Here the machine learning model fails to identify the anomalies.

In figure 1 we see an example of supervised classification. Because the model trained to recognize certain anomalies that are in the data set it succeeds in classifying them. In figure 2 we do however see that the same model does not find any anomalies, as they are completely different from the ones it trained on.

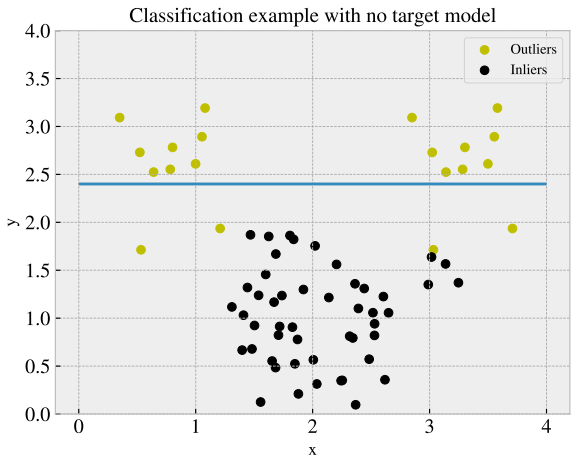

Figure 3: Unsupervised classification of anomaly with no target model. Here the machine learning model manages to with good precision identify the anomalies.

In figure 3 we have an example of unsupervised learning. Here we do not tell what are anomalies and what are expected behavior. Note that the unsupervised classifier does not classify correctly all anomalies. The grade of which it does vary from model to model and from dataset to dataset.

There are several ways to define events as anomalies and not expected events. One method used in time series data is to have a running average, and then if some values deviate with x standard deviations, that event is tagged. But because we are only interested in collective anomalies and not point anomalies, we need to use distributions to evaluate the output of the machine learning algorithm. A perfect binary classifier would for instance have two completely separate distributions, while a poor binary classifier would have two distributions with overlap. An additional metric to use is the so called ROC curve, which checks the output of the classifier and finds how often the classifier predicts a true and a false positive. This can only be done with labeled data, thus is not itself a metric of whether or not the algorithm detected new physics, but rather how well it can distinguish known anomalies.

B. Beyond standard model physics

In the context of high energy physics, collective anomalies would be new physics we both do and do not have a model for.
!!
rewrite this sentence to clarify
!!
Such anomalies could be super symmetric particles[10], new vector bosons or dark matter particles, as well as many other candidates. These types of searches require enormous amounts of data, and both supervised and

unsupervised methods are used continuously to see if there are "hidden" relations in the data that can separate the known standard model from the unknown new physics.

The methods have their strengths and weaknesses, and should be used with this in mind. A supervised model will always be better to classify and separate distributions of background[4] and signal[5] events. and if the signal actually exists in the data from the detector, it will give better accuracy than any (semi-) unsupervised methods.

In the event that some signal actually exist in the data from the detector that we do not have a model for, a supervised model will not be able to recognize it purely as a signal. The unsupervised model might claim that there is something other than sm background. The certainty of the results will vary with different models, but in the case of an auto encoder, this is due to the fact that it will only recognize background events that it has trained on, and not any new physics. The drawback with using unsupervised methods in high energy physics, or other fields with multiple anomaly candidates, is that you cannot know what the signal is, only that is it not background.

C. Stacked auto encoder

One method to attack the anomaly detection problem is the so called (stacked) auto encoder. The idea is based on reconstruction, and has been implemented for denoising of images, image compression, and anomaly detection. An auto encoder is a subgroup of feed forward neural networks[6]. The goal with that model is to compress the information into fewer variables, called the latent space, which then can explain much of the data through decoding that information. This allows the algorithm to learn the most important components of the data. An illustrative image of an auto encoder is shown in figure 4.

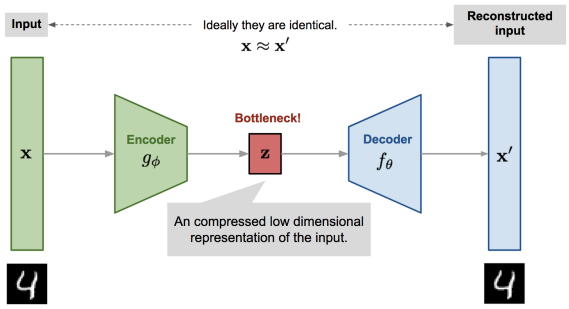

Figure 4: The architecture of an autoencoder with image reconstruction as the example use. Source, accessed 28.04.22

!!
Check if reference two figure should be changed.
!!

We see here in figure 4 that the input data is deconstructed through an encoder to a lower dimensional space called the latent space, depicted here as **z**, and then reconstructed the data with the decoder. It is important to note that the number of layers, nodes per layer and activation function per layer for the encoder does not need to match the structure of the decoder. The only requirement is that the input and output layer has the same shape. The end result is the reconstructed data **x'**. The training of the model is parameterized in the following way. We define the decoder as

$$\mathbf{z} = g_\xi(\mathbf{x}),$$

and the reconstructed information as

$$\mathbf{x'} = f_\theta(g_\xi(\mathbf{x})),$$

where the parameters (ξ, θ) are tuned to reconstruct the data as close to the original data as possible. The loss function for the model is then the simple mean squared error:

$$L_{AE}(\xi, \phi) = \frac{1}{N}\sum_{i=0}^{N-1}\left(\mathbf{x}^i - f_\theta(g_\xi(\mathbf{x}^i))\right)^2$$

III. IMPLEMENTATION

A. Handling of data

In the appendix the features used in the data set are listed, with their respective datatype and description. The datasets are separated into two categories, background & signal mc and actual data from a detector.

[4] Standard model processes created from Monte Carlo simulation will be denoted as "background mc". These simulations are carefully compared and corrected against data from detectors that we with high certainty claim have only standard model processes.

[5] New physics models are created using Monte Carlo simulation, and are based around theoretically allowed properties. They have not been discovered and cannot be considered "real" new physics until they do. In this project it is denoted as "signal mc".

The totality of the data used in this report is given as ATLAS Open Data[6]. This is a dataset that has been analyzed by ATLAS for new physics, and have thus been released to the public.

1. Cuts, pre-selection and scaling

To preserve as much information and add as little bias as possible very few cuts where made. The entire data gathering process is biased from the moment the data is collected at the collider, but even then it is good to try to minimize the bias.

The first cut I impose is to require "good leptons". This is done by requiring lep_etcone20/lep_pt $<$ 0.15 and lep_ptcone30/lep_pt $<$ 0.15. We then require only two leptons per event. The event selection was done using RDataframe[1], to speed up the selection[7]

I tested two methods of scaling in this report, Min Max Scaling[8] and Standard scaling[9][13].

2. Choice of features

The choice of features is a mixture of specification and broadness. In this context specification is defined as attributes to the main focus of the final state, in this case the leptons. Thus, most of the features are directly related to the leptons. Broadness is in this context defined as information about the rest of the final state, such as jets, photons or tau leptons. Ideally one would pick as much information as possible for each event, but there is a trade-off between amount of information and size of data, and as consequence, execution time.

The features were picked such that there are no missing values, only 0 or larger than 0. For regular machine learning problems, missing values usually gets replaced by the mean value of the feature. This is however a problem when analysing physics, as such a choice could violate the laws of physics. To avoid this, all features are designed to either have 0 if missing, or sum up the contributions of both missing and present values, given that they represent the same type of information.

One example of this is if we have two jets in one event and three in another, using the transverse momentum of the jets as features. We cannot in the first event put in the mean values of all third jet-transverse momentum in the entire dataset. But we could put 0, or sum all the transverse momentum for the jets in the given event into one feature. Another possibility is to only give the count of jets in the event. There are even more ways to solve this issue.

3. Signal samples

To test the algorithm I use the following signal sample categories. There are five supersymmetric sample categories: SUSYSlepSlep, SUSYC1C1, GG_ttn1, TT_directTT and SUSYC1N2[4], three Z' boson sample categories: Zprime_ee, Zprime_mumu and Zprime_tt[7], three Randall-Sundrum graviton sample categories: RS_G_ZZ, G_ee and G_mumu[14], and one dark matter mediator sample category: dmV_Zll.

B. Tuning and training

For the auto encoder to accurately reconstruct the standard model, the model needs to train on the background mc. However, neural networks are highly susceptible to hyperparameters, which needs tuning. In this project I used Keras-tuner[12] as tuning architecture for tuning. The choice of tuner for this report was Hyperband[10]. It is not well understood how the hyperparameters interact with each other to affect the result, therefor there are really no strict guidelines for approaching this problem. A result of this might be that the choice of hyperparameters seem completely random or strange. The hyperparameters used in the network are the learning rate, the alpha parameter for the LeakyReLU[10] activation function, the activation function for each layer, the amount of nodes per layer and regularization for layer's kernel and output. The regularization is added to prevent the algorithm from overfitting too much towards the background mc. The activation functions used in the hyperparameter search where

- ReLU
- LeakyReLU
- Tanh
- Linear

whilst the different learning rates that was tested was $[9 \cdot 10^{-2}, 9.5 \cdot 10^{-2}, 10^{-3}, 1.5 \cdot 10^{-3}]$, and the kernel and output regularization constant values were $[0.5, 0.1, 0.05, 0.01]$. The optimizer used for all the models is the ADAM[8] optimizer.

The background mc where split into a 80-20 split, where 80% were used for training and tuning, and 20% were mixed with signal mc for testing. The 80% training

[6] Link to ATLAS Open Data here.
[7] This code was written by Eirik Gramstad, and can be found here.
[8] More on Min max scaling here.
[9] More on Standard scaling here.

[10] In Tensorflow LeakyReLU is implemented as a layer and not an activation function, but the layer can be passed as an activation argument, which leads to creating custom objects in the Keras back-end when tuning with Keras-Tuner. More on the LeakyReLU api can be found here.

set was then again split with a 80-20 split for the hyperparameter search, called grid_train and grid_test. Due to the large amount of events in the background mc, I had to pick out a sample that had enough events to be a fair representation of the entire training set. Thus 10 million events were randomly sampled from grid_train for hyperparameter training and 2 million events sampled from grid_test for hyperparameter validation.

The auto encoder was written using the Tensorflow api[11][3], using a functional structure[11]. In practise, this model could just as well have been written as a Sequential model[12], but at a cost of flexibility and lack of potential non-linear structure in the architecture.

Certain features where removed, as they did not have a good enough overlap between mc and ATLAS data. All features where checked to ensure as good correspondence as possible, and the figures for all the features can be found in the Github repo for this project, under the "figures/implementation" folder. The features that were chosen to be excluded where lep1_tracksigd0pvunbiased and lep2_tracksigd0pvunbiased, which are shown below:

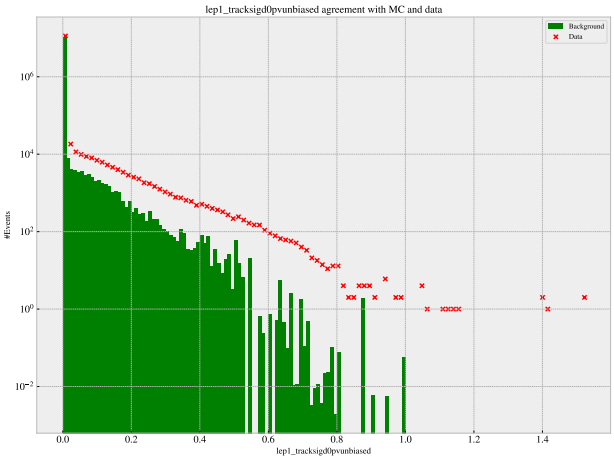

Figure 5: Lep1 tracksigd0unbiased histogram for background mc and ATLAS data.

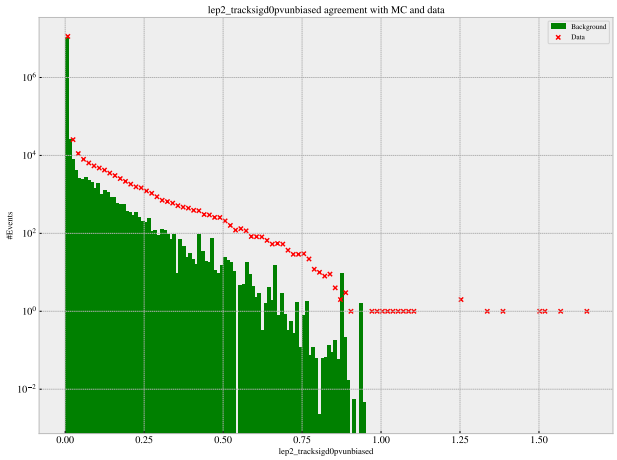

Figure 6: Lep2 tracksigd0unbiased histogram for background mc and ATLAS data.

We see in figures 5 and 6 that both have excess in the data compared to the background mc.

IV. RESULTS

A. The first model

The first auto encoder model was found with a hyper parameter search and is structured as shown in figure 7.

[11] Functional structure uses a function call for layers, i.e for layers a,b, then b(a) will connect the two layers, and equals a sequential link a $\rightarrow$ b. This allows for more flexible structures. More on the functional api can be found here.

[12] Sequential structure adds layers in sequence, i.e for layers a, b, c we have that a $\rightarrow$ b $\rightarrow$ c, with a strict structure. This allows for more organized code. More on sequential models can be found here.

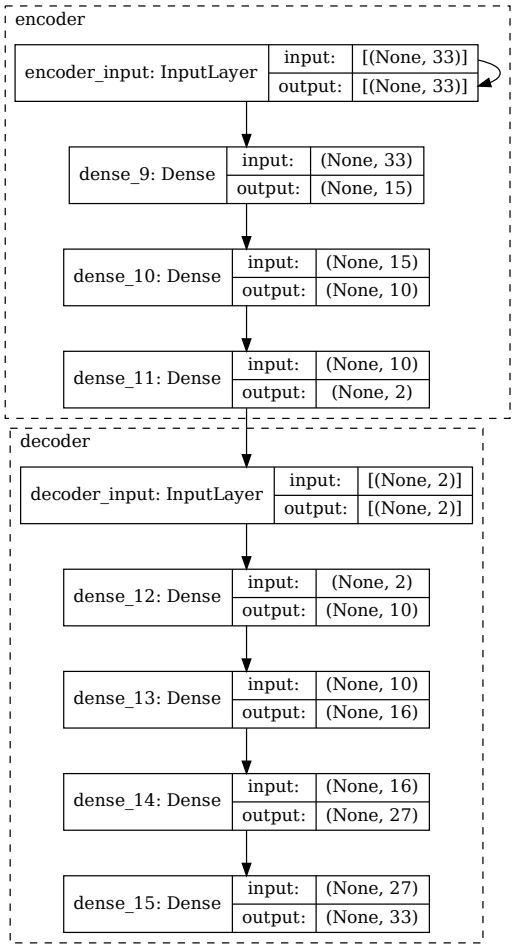

Figure 7: Auto encoder architecture after hyper parameter search.

The hyper parameter search gave the optimal hyper parameters listed below. The encoder had the following activation functions:

1. layer: tanh
2. layer: linear
3. layer: linear
4. layer: leakyReLU

the decoder has the following activation functions:

1. layer: relu
2. layer: leakyReLU
3. layer: tanh
4. layer: linear

The optimal learning rate was 0.0015, the optimal kernel regularizer was 0.05, the optimal activity regularizer was 0.5 and the optimal alpha for the leakyReLU was 1. The batch size for both training, tuning and prediction was 8192. These hyper parameters was found after around 30 minutes of searching, and then stopped, as it appeared that the validation MSE did not improve much. Thus, there are most likely a better set of hyper parameters that could get an even lower validation MSE.

The reconstruction error of ATLAS data is shown in figure 8.

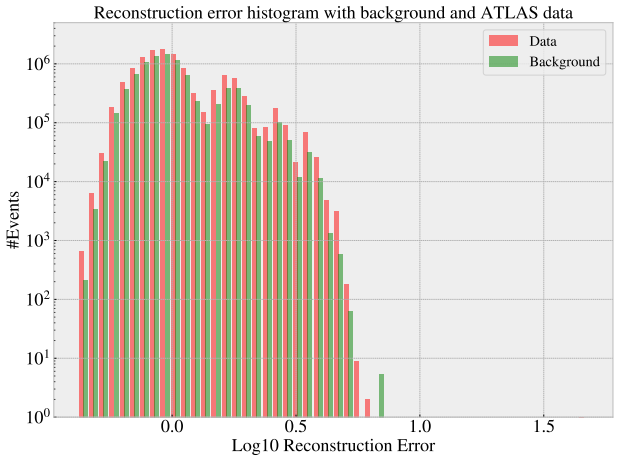

Figure 8: Log log histogram showing reconstruction error for ATLAS data and background mc for big auto encoder.

In figure 8 we see that the auto encoder gets a good overlap of reconstruction error between ATLAS data and background mc. This is in accordance with the fact that the ATLAS collaboration has with very high certainty only found standard model events in the ATLAS Open Data. We also observe here that the algorithm groups events with different error, as shown with the camel back pattern. This pattern is repeated several times in figure 8, which indicates that the algorithm learned its own categories for the standard model. Using all the signal mc samples, I tested if the classifier could distinguish new physics signal mc from sm processes by mere reconstruction. The reconstruction error is shown in figure 9.

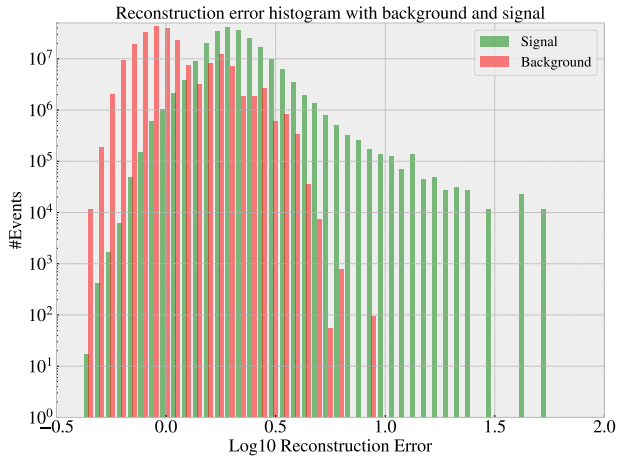

Figure 9: Log log histogram showing reconstruction error of background mc and signal mc for big auto encoder with all signal samples.

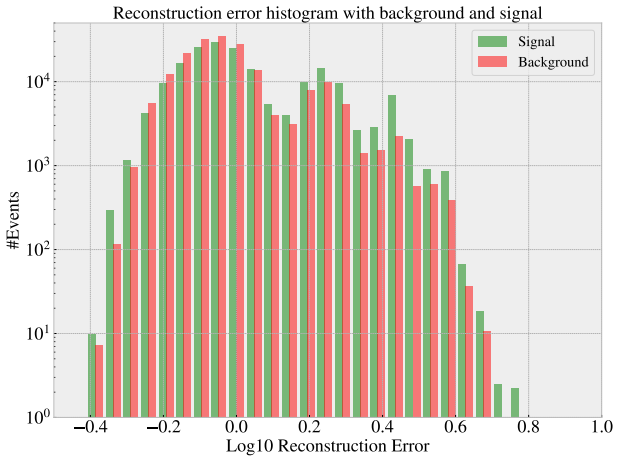

Figure 11: Log log histogram showing reconstruction error of background mc and signal mc for big auto encoder without Graviton to dimuons.

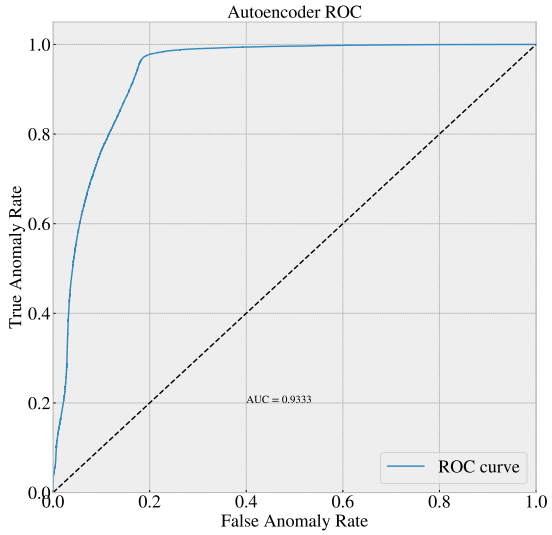

Figure 10: ROC curve signal mc and background mc for big auto encoder with all signal samples.

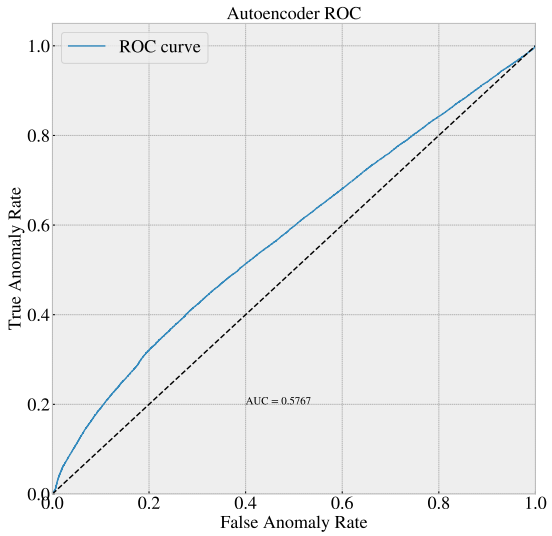

Figure 12: ROC curve signal mc and background mc for big auto encoder without Graviton to dimuons.

From the results in figure 9 and 10 we see that the auto encoder very well manages to separate signal from background, having only been shown background mc. However, there is some problems with these results. The Randall-Sundrum Graviton to dimuons samples in the signal mc were given with very high weights compared to the Graviton to dielectrons samples, which highly indicates that the data samples are corrupt somehow. Thus we have to check the reconstruction error on signal mc without the Randall-Sundrum Graviton samples. This is shown in figure 12.

In figure 11 and 12 we clearly see that the auto encoder struggles to separate background and signal mc, with a AUC score of around 0.58. This shows that for signals that are fairly similar to background mc, this model will struggle to tell them apart. Here we also observe the camel back structure, where the algorithm has grouped certain events together. There are ways to remediate this.

B. Attempts at improvement

1. *Unstacked auto encoder*

One alternative to remediate is to alter the architecture of the auto encoder. To compare this architecture I created a smaller auto encoder, shown in figure 13.

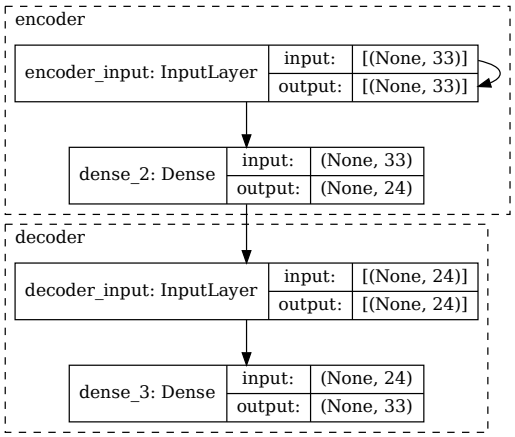

Figure 13: Smaller auto encoder architecture after hyper parameter search.

The hyper parameters used for this model was a learning rate of 0.9, 24 nodes in the latent space, the activation function for the latent space was tanh, the activation function for the output was linear, the kernel regularization was 0.05 and the activation regularization was 0.01. The batch size for both training, tuning and prediction was 8192. The model was trained for 6 epochs.

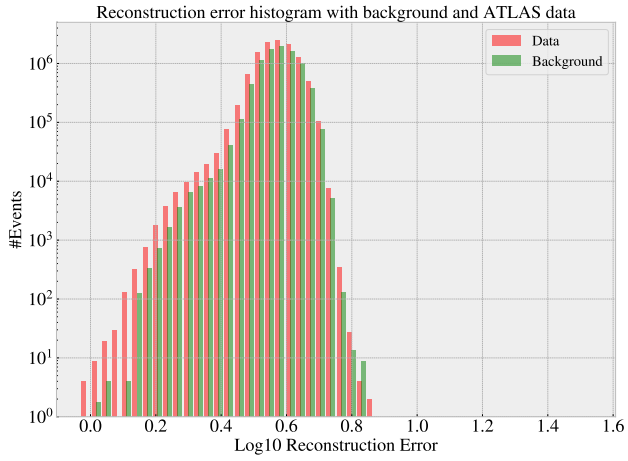

Figure 14: Log log histogram showing reconstruction error for ATLAS data and background mc for small auto encoder.

Using the architecture in figure 13 we see a good overlap of ATLAS data and background mc, with a center of about 4 MSE reconstruction error. Here we see that the algorithm creates what appears to be one group.

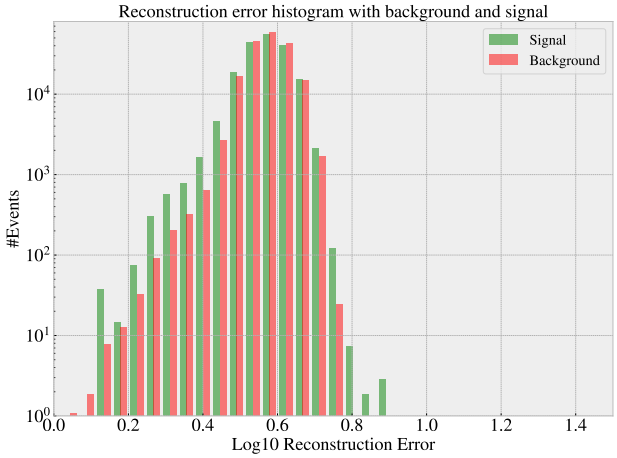

Figure 15: Log log histogram showing reconstruction error of background mc and signal mc for small auto encoder.

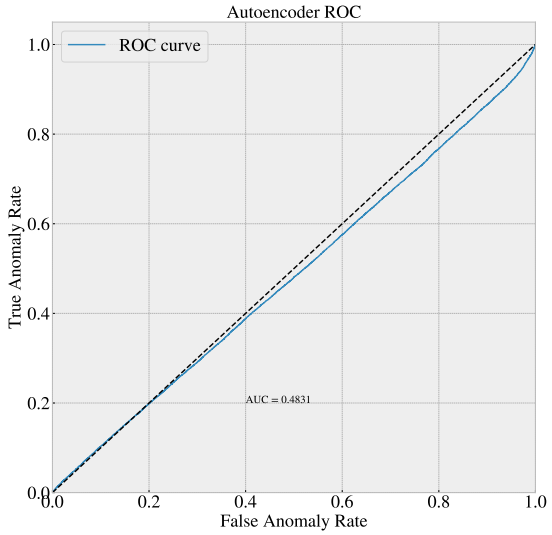

Figure 16: ROC curve signal mc and background mc for small auto encoder

As shown in figure 15 and 16 we observe that even though this model gets a good overlap, it struggles even further than the big auto encoder, having an AUC score of only 0.48 which is only slightly better than guessing. Here we observe that the algorithm created one group with all events for signal and background just as it did with ATLAS data. This could mean that such a sim-

ple model needs more complexity in terms of layers and node reduction per layer to be able to learn how to separate.

2. Reduced features in data set with stacked auto encoder

Another idea is removing low level features that are fairly similar for background and signal mc. From testing it was found that even with as low as 13 features, the output from the algorithm only improved slightly.

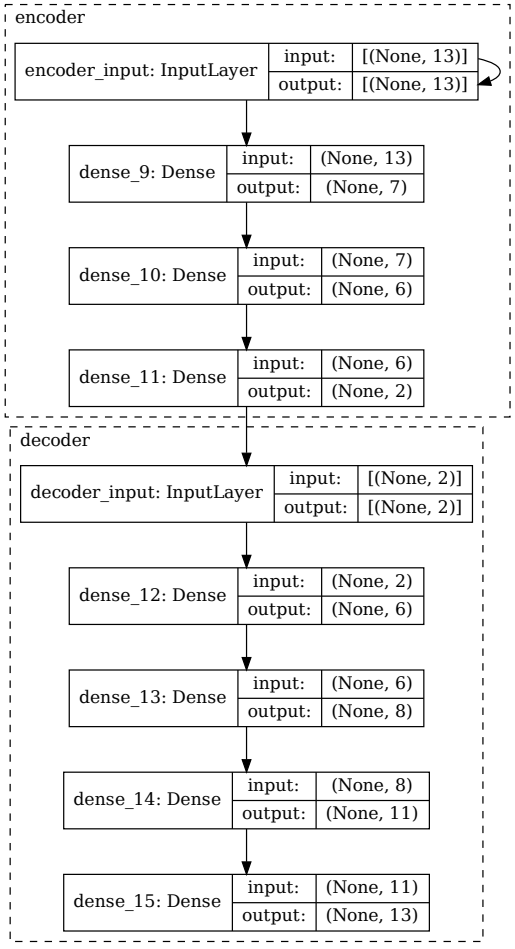

Figure 17: Auto encoder architecture found by hyper parameter search with 13 features in the input.

In figure 17 we have the tuned auto encoder architecture based on a hyper parameter search for 13 feature dataset. The hyper parameters were as listed below. The encoder had the following activation functions:

1. layer: leakyReLU
2. layer: linear
3. layer: leakyReLU
4. layer: linear

The decoder had the following activation functions

1. layer: leakyReLU
2. layer: tanh
3. layer: relu
4. layer: linear

The kernel regularization was 0.5, the activation regularization was 0.5, the leakyrelu alpha parameter was 0.1 and the learning rate was 0.0015. The batch size for both training, tuning and prediction was 8192. The model was trained with 5 epochs.

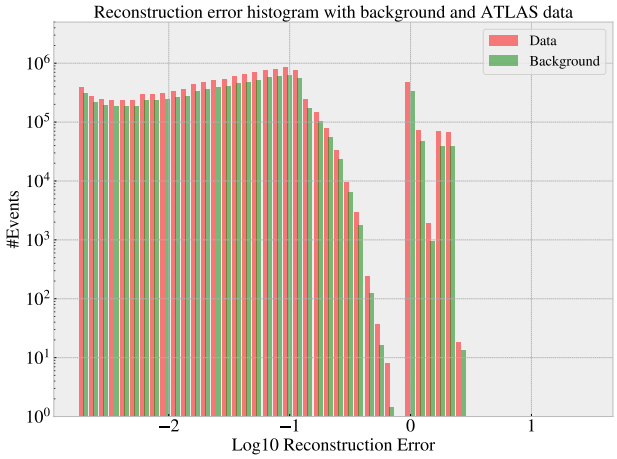

Figure 18: Log log histogram showing reconstruction error for ATLAS data and background mc for big auto encoder using only 13 features.

In figure 18 we observe that the reconstruction error of ATLAS data and background mc have fairly good overlap. As expected the reconstruction distribution is centered at a much lower value than for the larger data sets, as there are a lot fewer features to reconstruct. We also see here that the algorithm separates some of the events into at least two groups, with a clear split around 1 MSE.

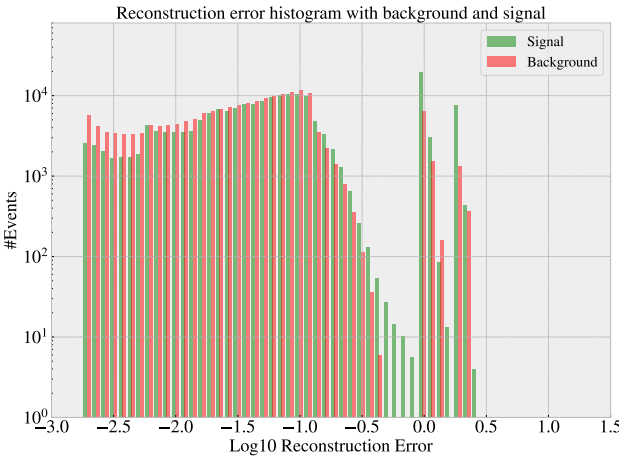

Figure 19: Log log histogram showing reconstruction error of background mc and signal mc for big auto encoder using only 13 features.

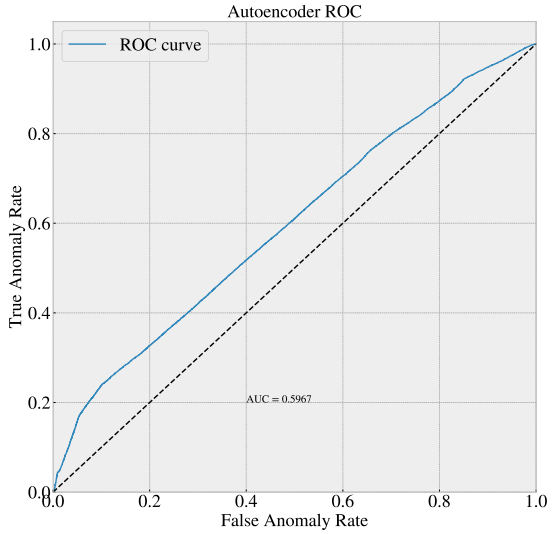

Figure 20: ROC curve signal mc and background mc for big auto encoder using only 13 features.

From figure 19 we observe that even with only 13 features, [njet60, nbjet85, isOS,isSF, mt2, met_et, , lep1_pt,lep1_E, lep1_ptcone30, lep1_etcone20, lep2_pt, lep2_E, costhstar], the auto encoder struggles to separate background and signal mc. There are about 10 events around -0.3 MSE reconstruction error that it cannot match with background mc, but the rest of the data overlaps fairly well. This also matches the reconstruction of ATLAS data, where this area has no events. The ROC score for the auto encoder was around 0.59, which only slightly better than for the dataset with all 33 features.

3. *Auto encoder with standard scaling*

A third option is to change the scaling and look at its effect on the handling of data. The previous models have all used MinMax scaling to scale the data, where as this model used standard scaling instead.

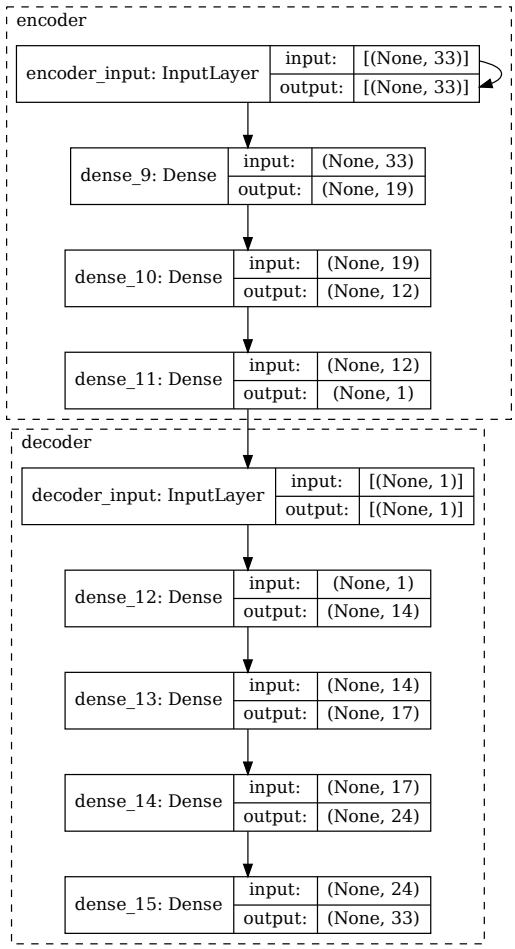

Figure 21: Auto encoder architecture after tuning of hyper parameters using standard scaling on the datasets.

In figure 21 we have the architecture for the auto encoder after hyper parameter tuning where the datasets are scaled using standard scaling. The hyperparameters were as listed below. The activation functions for the encoder were

1. layer: tanh
2. layer: tanh
3. layer: tanh
4. layer: tanh

The activation functions for the decoder were

1. layer: leakyrelu
2. layer: leakyrelu
3. layer: tanh
4. layer: tanh

The kernel regularization, activation regularization and leaky relu alpha parameter were all 0.01, and the learning rate as 0.0015. The batch size for both training, tuning and prediction was 8192. The model was trained with 15 epochs.

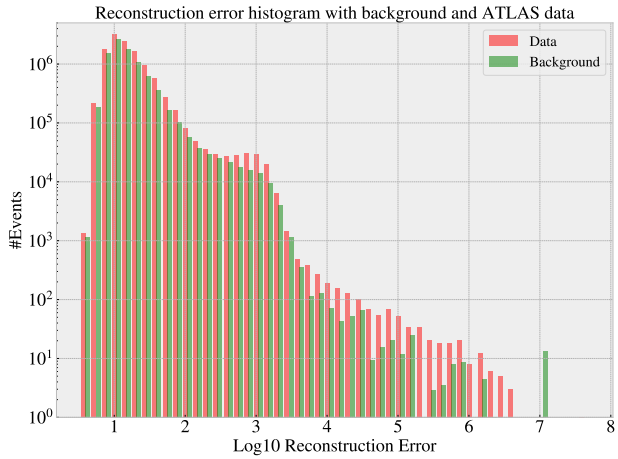

Figure 22: Log log histogram showing reconstruction error for ATLAS data and background mc for big auto encoder, using standard scaling.

In figure 22 we see that even though the reconstruction error for ATLAS data and background mc has a wide spread, it overlaps well, which indicates that the autoencoder learned to reproduce the standard model, even though some events where much harder to reproduce than others. We also note here that only a few 10's to 100's of events have an error above $10^{3.5}$.

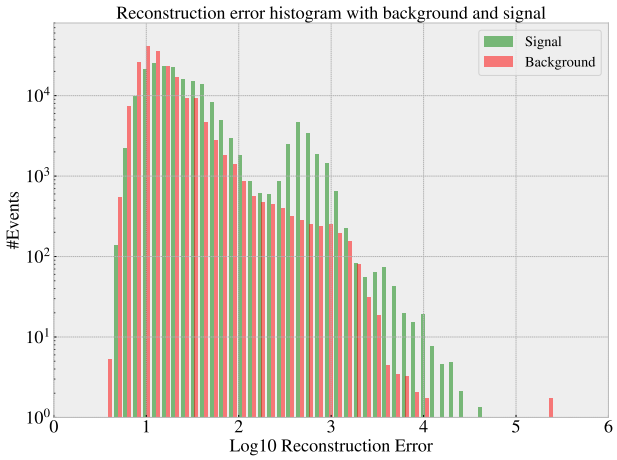

Figure 23: Log log histogram showing reconstruction error of background mc and signal mc for big auto encoder, using standard scaling.

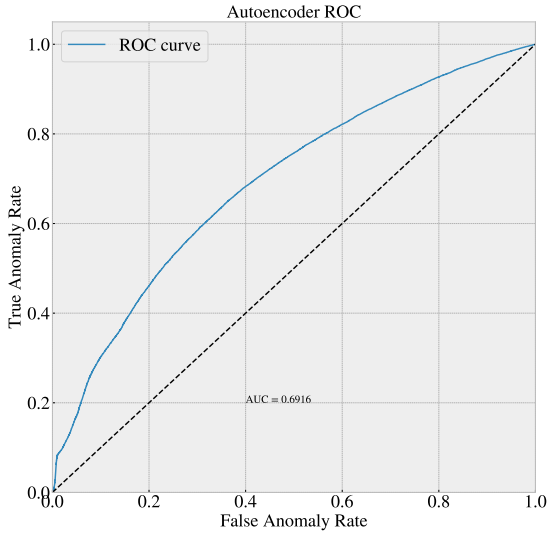

Figure 24: ROC curve signal mc and background mc for big auto encoder, using standard scaling.

In figures 23 and 24 we observe that the model somewhat overlaps the background and signal mc, which the exception of a few thousand samples in the second peak. It appears here that the model manages to separate some signal events from the background mc. This model preformed better than the models using Min Max scaling, with a AUC score of about 0.69.

V. DISCUSSION AND POSSIBLE WAYS FORWARD

There were several issues with training and tuning the model, much of which had to do with hardware. The background and signal mc data set alone was around 17 Gb of data in hdf5 format. To do training, prediction and hyper parameter searches especially, GPU usage was needed. With Tensorflow's XLA[13] (Accelerated Linear Algebra), the algorithm got a speedup from 1 hour and 30 minutes to around 1 minute per epoch when training on 80% of the background mc. Still, because of the enormous amount of data, the hyper parameter search had to be done using a smaller sample size. Thus each epoch only took 10-30 seconds, which allowed for many iterations of hyperparameter tuning. There is however the issue that the random sampling does not guarantee accurately representing the sm background sample distribution, and thus the hyperparameters could very well have been even better tuned.

It is also worth noting that the ROC curves and reconstruction error figure using the signal mc only in part shows how the auto encoder performs. The signal mc samples does not represent all possible new physics, and thus we can only conclude from our results that the auto encoder struggles to separate those signal mc samples. The auto encoder has one metric, and that is minimizing reconstruction error. If you then have a signal that looks very similar to background, then the reconstruction error will be similar to the background. This does not mean that the auto encoder has not learned some hidden relations in the data, only that those relations are not enough for those kinds of signals. In the event that other signals appears in detector data, the auto encoder could still find them.

Another crucial point to make is that in the realm of machine learning, there are few guidelines for how the architecture should look. In this case with the auto encoder, the only requirement is that in the encoder the previous layers must have at least one more node than the one after, and in the decoder the layer after must have at least one more node than the one before it. With respect to the amount of layers, regularization of layers towards bias or output, implementation of dropout layers, initialization of weights, or symmetry with regards to encoder and decoder architecture, an enormous hyper parameter search is the only way to find the ideal model. This bias that the designer of the given auto encoder brings is very hard to quantify, but is indeed crucial, and generates a gigantic pool of possibilities which, given enough time and resources could yield valuable insight.

Scaling also had an interesting impact on the output, as it appears that standard scaling allows for better detection of anomalies. This choice of scaling improved the AUC score from around 0.58 to around 0.69, which is a significant improvement. It is also important to note here that these results are all done using all signal samples, except for Graviton to dimuons. Ideally one should have had individual predictions on all the signal mc that we have to see if there are differences in the prediction, thus allowing us to see what kind of signals the auto encoder struggles to filter out and vice versa.

The choice of features also contributes to the effect of the auto encoder. The removal of the d_0 significance for both leptons was done by looking at the distribution of data and background mc by checking the difference between them, and was done by eye sight. This could certainly have been done better, for instance by requiring a threshold for how large the ratio between data and background mc can be.

From the results above it is not clear that the auto encoder is not a viable candidate for new physics search, as there are still many aspects that are not well understood. A good start would be to further study how the model learns to merge and reconstruct each feature, and if it quickly converges towards only focusing on one or a few main features.

A good start would be to further study how the model learns to merge and reconstruct each feature, and also to see if certain features dominate the predictions. If this is better understood, then maybe the construction of certain features in the preparation of the background mc and ATLAS data could be made to handle this better. It could very well be that certain features or combinations of certain features are two similar for the algorithm to tell the events apart.

Another point of interest is the relation between number of features and complexity of the architecture. From the models tested above, it would appear that a stacked and more complex auto encoder is needed for the algorithm to learn to reproduce the dataset without copying it. It is also not clear if a symmetric architecture is more ideal than a non symmetric architecture, and a hyper parameter search with this included could yield interesting results. Another important, but computationally heavy, task is to do an extensive hyper parameter search, using a lot more background mc, to check ideal structure in terms (non)-symmetry or amount of layers in architecture, number of nodes, activation functions etc, as mentioned earlier.

The camel back and box pattern shown in the results are also of interest. In the figures background mc was plotted as one dataset, but it would be very interesting to see which of the samples that were easier and harder to reproduce. This could also give more insight into which classes of signal models that would be easier to find using auto encoders.

Finally, to search for new physics using an auto encoder, there should be better insight as to what "classes" of new physics final states that the algorithm could better filter out. Suppose you compare two candidates for new physics and they are somewhat different, it might

[13] XLA architecture can be found here, and XLA api can be found here.

be that certain auto encoder could have the architecture altered to better filter out certain signal classes, and not a specific model, thus preserving the semi unsupervised part.

VI. CONCLUSION

In this project an attempt was made to use an auto encoder for anomaly detection on ATLAS open data. 33 features were used to describe the 2 lepton final state. All machine learning was run using the Tensorflow api, with functional structure. Several models where tuned using Keras-Tuner with a hyperband searcher to get the semi-optimal hyperparameters. Even with hyperband, we found that a large reducing of training and validation datasets were needed for fast searching. All though all of the models successfully managed to have good overlap between background mc and ATLAS Open data, only the autoencoder trained on standard scaled datasets managed to separate out signals. The results showed that the ROC score increased, when using a multi layer auto encoder, removing certain low level features and by using this standard scaling. Several proposals for improvement are given for better performance of the model, most of which have to do with expanding the amount of and range of hyper parameters. Such a search could have a large impact on the understand and use of auto encoders in anomaly detection, but are extremely time consuming. Based on the findings in this report, the auto encoder cannot be completely disregarded as a useful method for anomaly detection in high energy physics, as there are still a lot about the algorithm that is not well understood.

REFERENCES

[1] Rene Brun et al. "root-project/root: v6.18/02". In: (Aug. 2019). DOI: 10.5281/zenodo.3895860.

[2] Varun Chandola, Arindam Banerjee, and Vipin Kumar. "Anomaly Detection: A Survey". In: *ACM Comput. Surv.* 41 (July 2009). DOI: 10.1145/1541880.1541882.

[3] François Chollet et al. *Keras*. https://keras.io. 2015.

[4] Antonella De Santo. "Electroweak SUSY Searches At ATLAS". In: (Oct. 2018). URL: https://cds.cern.ch/record/2644352.

[5] Sakarias Frette and William Hirst. "Seaching for Higgs through supervised and unsupervised machine learning algorithms". In: (2022). URL: https://github.com/WilliamHirst/Advanced-Machine-Learning/blob/main/article/Project_1.pdf.

[6] Sakarias Frette, William Hirst, and Mikkel Metzch Jensen. "A computational analysis of a dense feed forward neural network for regression and classification type problems in comparison to regression methods". In: (2022), p. 5. URL: https://github.com/Gadangadang/Fys-Stk4155/blob/main/Project%202/article/Project_2_current.pdf.

[7] Daniel Hayden, Raymond Brock, and Christopher Willis. *Z Prime: A Story*. 2013. DOI: 10.48550/ARXIV.1308.5874. URL: https://arxiv.org/abs/1308.5874.

[8] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2014. DOI: 10.48550/ARXIV.1412.6980. URL: https://arxiv.org/abs/1412.6980.

[9] Christopher G. Lester. "The stransverse mass, M T2, in special cases". In: *Journal of High Energy Physics* 2011.5 (May 2011). DOI: 10.1007/jhep05(2011)076. URL: https://doi.org/10.1007%2Fjhep05%282011%29076.

[10] Lisha Li et al. "Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization". In: *Journal of Machine Learning Research* 18.185 (2018), pp. 1–52. URL: http://jmlr.org/papers/v18/16-558.html.

[11] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: https://www.tensorflow.org/.

[12] Tom O'Malley et al. *KerasTuner*. https://github.com/keras-team/keras-tuner. 2019.

[13] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[14] Lisa Randall and Raman Sundrum. "Large Mass Hierarchy from a Small Extra Dimension". In: *Physical Review Letters* 83.17 (Oct. 1999), pp. 3370–3373. DOI: 10.1103/physrevlett.83.3370. URL: https://doi.org/10.1103%2Fphysrevlett.83.3370.

[15] *Review of the 13 TeV ATLAS Open Data release*. Tech. rep. All figures including auxiliary figures are available at https://atlas.web.cern.ch/Atlas/GROUPS/PHYSICS/PUBNOTES/ATL-OREACH-PUB-2020-001. Geneva: CERN, Jan. 2020. URL: https://cds.cern.ch/record/2707171.

Appendix A: Features

Table I: Features and their description [15]

Feature	Type	Description
njet20	int	Number of jets with $p_T > 20$ GeV
njet60	int	Number of jets with $p_T > 60$ GeV
nbjet60	int	Number of b-jets with $p_T > 60$ GeV
nbjet70	int	Number of b-jets with
nbjet77	int	Number of b-jets with
nbjet85	int	Number of b-jets with
isOS	int	1 if leptons have opposite charge, 0 if leptons have same charge
isSF	int	1 if leptons are of same flavor, 0 is leptons are of different flavor, flavor code 11 is electron, flavor code 13 is muon
mll	float	Invariant mass of the two leptons
mt2	float	The maximal lower bound on the mass of each member of a pair of identical parent particles which, if pairproduced at a hadron collider, could have each undergone a two-body decay into (i) a visible particle (or collection of particles) and (ii) an invisible object of hypothesised mass χ[9].
met_et	float	Transverse energy of the missing momentum vector
met_phi	float	Azimuthal angle of the missing momentum vector
lep_flav	vector<int>	Flavor of the lepton, 11 for electron and 13 for muon
lep_pt	vector<float>	Vector containing transverse momentum for the leptons
lep_eta	vector<float>	Vector containing pseudo-rapidity , η, for the leptons
lep_phi	vector<float>	Vector containing azimuthal angle, ϕ, for the leptons
lep_E	vector<float>	Vector containing the energy for the leptons
lep_ptcone30	vector<float>	Vector containing scalar sum of track p_T in a cone of $R = 0.3$ around lepton, used for tracking isolation
lep_etcone20	vector<float>	Vector containing scalar sum of track E_T in a cone of $R = 0.2$ around lepton, used for calorimeter isolation
lep_trackd0pvunbiased	vector<float>	d_0 of track associated to lepton at point of closest approach (p.c.a.)
lep_tracksigd0pvunbiased	vector<float>	d_0 significance of the track associated to lepton at the p.c.a.
lep_isTightID	vector<bool>	Vector containing boolean indicating whether leptons satisfies tight ID reconstruction criteria
lep_z0	vector<float>	Vector containing z-coordinate of the track associated for the leptons wrt. primary vertex
channelNumber	int	Data sample ID
costhstar	float	Cosine of dilepton decay angle
weight	float	MC sample weight
category	string	SM or BSM category
physdescr	string	MC process name
isSignal	int	1 if category is a BSM signal, 0 if the category is SM background