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

Analysis of two lepton final state from OpenData at ATLAS

Sakarias Frette

**Spring 2022**

**Abstract**

*University of Oslo*

**CONTENTS**

# I. INTRODUCTION

# II. THEORY

Some of theory sections about anomaly detection and machine learning algorithms are based on previous work done in other courses, such as [3] and [2].

## 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, and can be set by the context it self, and what is expected as an anomaly. We typically classify anomalies in three categories[1]:

1. Point anomalies

2. Contextual anomalies

3. Collective anomalies

For the purpose of this report we will mostly consider collective anomalies, as anomalies in the standard model has to be collective to claim anything due to noise etc.

In high energy physics we can using machine learning separate anomaly detection into two categories, supervised and unsupervised searches.

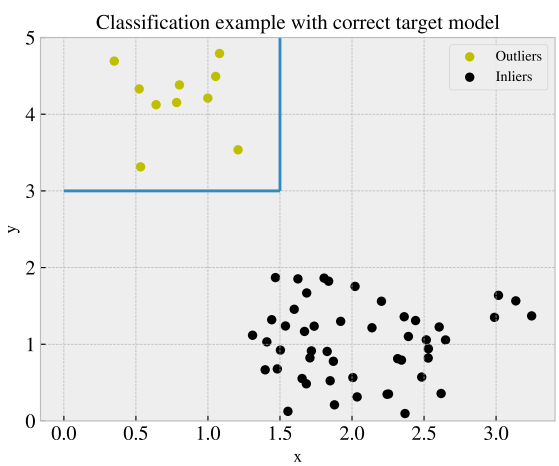


Figure 1: Example of supervised classification of anomaly where the target model is correct. Here the machine learning model manages to correctly identify the anomalies.

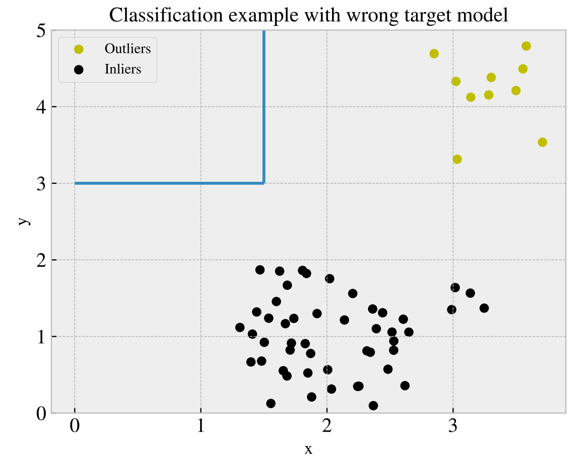


Figure 2: Example of supervised classification of anomaly where the target model is wrong. Here the machine learning model manages to wrongly identify the anomalies.

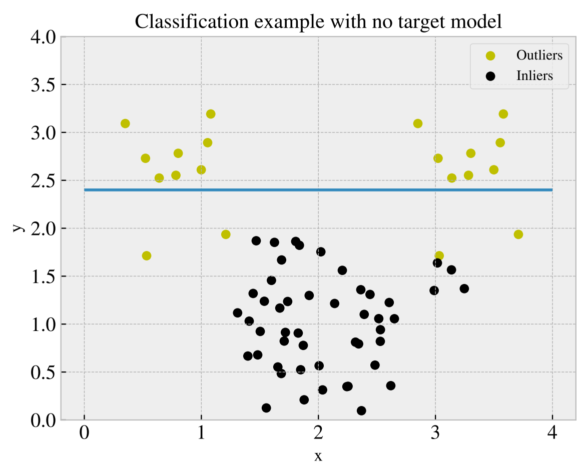


Figure 3: Example of unsupervised classification of anomaly with no target model. Here the machine learning model manages to with good precision identify the anomalies.

## B. 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, and the goal 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 below.

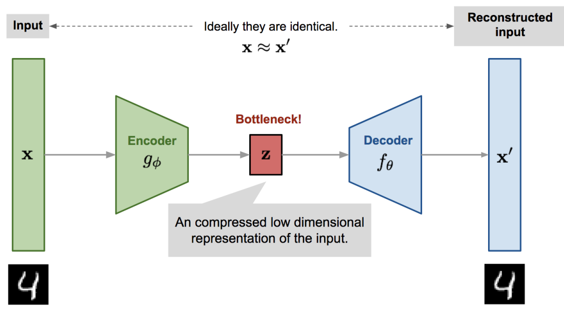


Figure 4: The architecture of an autoencoder with image reconstruction as the example use. Source, accessed 28.04.22

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 $\mathbf{z}$, and then reconstructs 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 $\mathbf{x'}$. 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 $(\xi, \theta)$ are tuned to reconstruct the data as close to the original data as possible. The model then adjusts following the simple mean squared error of the reconstruction and the actual data, given below

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

### C. Monte Carlo simulated data

### D. Beyond standard model physics

## III. IMPLEMENTATION

### A. Handling of data

In appendix **??** the features used in the data set are listed, with their respective type and description.

#### 1. *Cuts and pre-selection*

To preserve as much information and add as little bias as possible we did as few cuts as possible. 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 we 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, to speed up the selection[1]

#### 2. *Choice of features*

The choice of features is a mixture of specification and broadness. In this context specification is defined as information that specify for a given part of the physical system, in this case leptons. Thus, most of the features are directly related to the leptons, as they are the main focus of the BSM final states. Broadness is in this context defined as information about the rest part 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.

An 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 input 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, and there are even more ways to solve this issue.

#### 3. *Scaling*

- MinMaxScaling
- StandardScaling

[1] This code was written by Eirik Gramstad, and can be found here.

### B. Tuning and training

For the auto encoder to accurately distinguish background from signal, the model needs to train on the background data. However, neural networks are highly susceptible to hyperparameters, which needs tuning. In this project we used Keras-tuner[5] to tune the hyperparameters. The hyperparameters we have for our network are the learning rate, the alpha parameter for the LeakyReLU[2] activation function, the activation function for each layer and the amount of nodes per layer.

## IV. RESULTS AND DISCUSSION

### A. Semi unsupervised

- Difference between broad and narrow search
- Bias
- bias with features to choose
- difference of how well the model detects different signal models in mc data signals
- any interesting in actual data?
- difference in scaling
- tuning effect
- optimizing tensorflow code with use of autoclustering and possibly mixing floating point precision.

## V. CONCLUSION

[2] Tensorflow api for the activation function

**REFERENCES**


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

[2] 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.

[3] 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.

[4] 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.

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

[6] *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 [6]

| 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 $\chi$[4]. |
| 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 , $\eta$, for the leptons |
| lep_phi | vector<float> | Vector containing azimuthal angle, $\phi$, 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 | |
| costhstar | float | |
| 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 |