ATLAS autoencoders

Victor R. Russo

victor.rodrigues.russo@gmail.com

The Dataset

Our dataset is a .csv where every line is a triggered event in the ATLAS experiment .

The problem is that the in this kind of data can be very large to store, so we will investigate some method to compress it and decompress later.

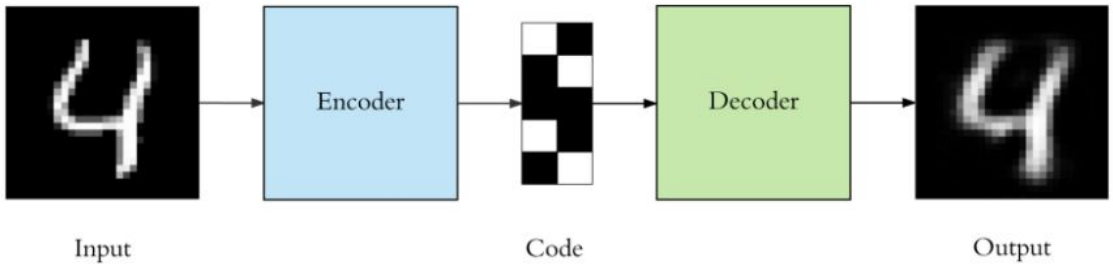
```
event ID; process ID; event weight; MET; METphi; obj1, E1, pt1, eta1, phi1; obj2,
                                    E2, pt2, eta2, phi2;  ...
```

This is the format of our data.

Deep Autoencoders

Deep autoencoders are a type of artificial neural networks in which the output is the same of the input.

So in the hidden layers of the network, there is a layer with less dimensions than the input that represents the data well in a more abstract way.



A high level schematic of a deep autoencoder network. source:https://towardsdatascience.com/applied-deep-learning-part-3-autoencoders-1c083af4d798

Deep Autoencoders applied on our problem

We will try to compress the data from 4 dimension to 3 dimension using a deep autoencoder network.

To reduce the scope for now, we should only use jet particles.

E	pt	eta	phi
258722.0	243675.0	0.328962	2.25014
520092.0	108590.0	-2.247300	-1.85679
383024.0	88405.6	2.145160	-1.95635
39507.6	35365.1	0.470460	-1.16445
225430.0	26878.2	-2.816080	-2.25938
1069460.0	751597.0	0.858186	-1.84217
457647.0	110079.0	2.102870	2.17501

A little sample of the dataset, already with only jet particles selected.

The network

For the encoding half we'll have 4 input nodes(the 4 kinect features) that are going to pass through 3 fully connected layers of different node sizes.

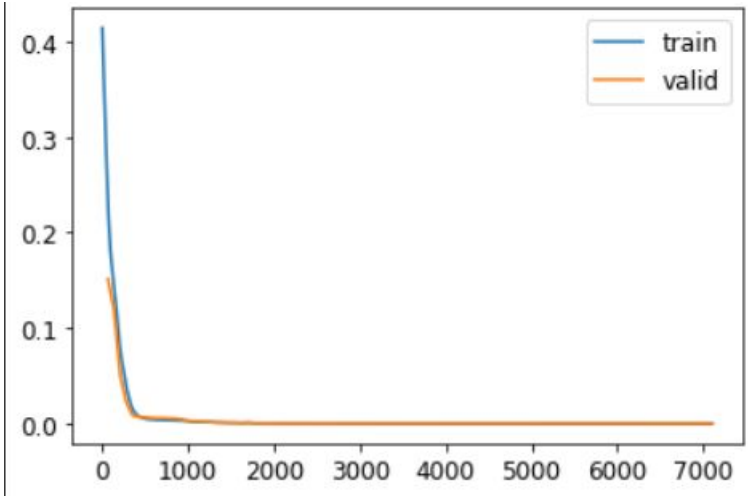
After this process, our data will have only 3 dimension, and it won't be the 4 features of input, but some more abstract(and compressed) representation of our data.

Now, to see if our network is working properly, we have to decode this 3 dimension into the 4 original features, and compare with the original input. This process of decoding consists of a mirror of encoding in this network.

Between all the layers, there is a tanh activation function. This forces us to normalize the data to [-1;1] range.We can describe our network in terms of number of nodes as: in-200-200-20-3-20-200-200-out

Training the network and looking for overfitting

After 100 iteration over the dataset we can plot the loss function:



This plot is a good sign, as the loss function quickly converges and do not shows any sign of overfitting(memorizing the training data in despite of the test)

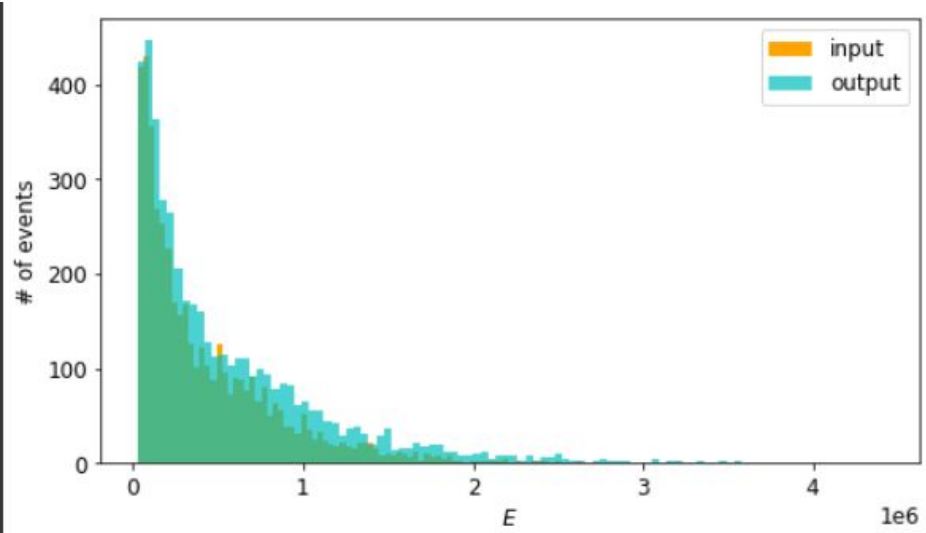
Results

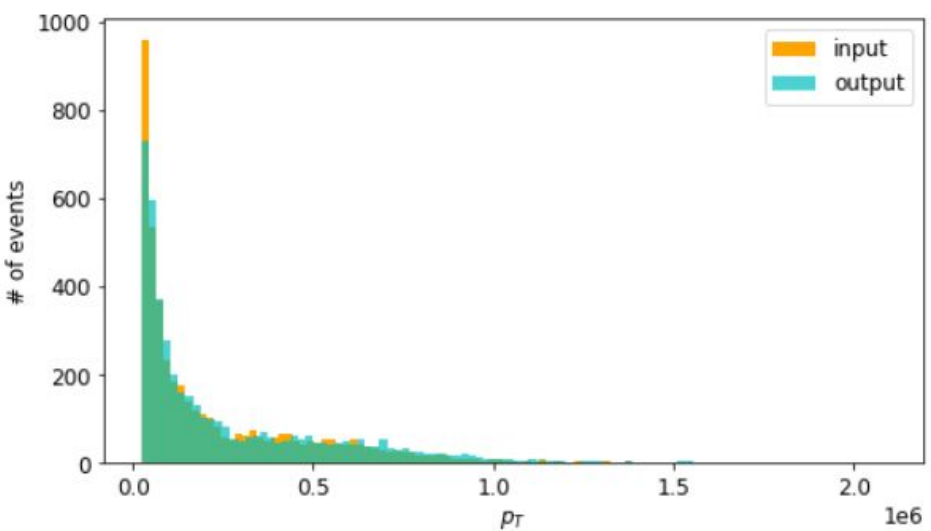
In the next slide there are plots of the four input variables overlapped with the output, and as it is possible to see the data remains pretty similar.

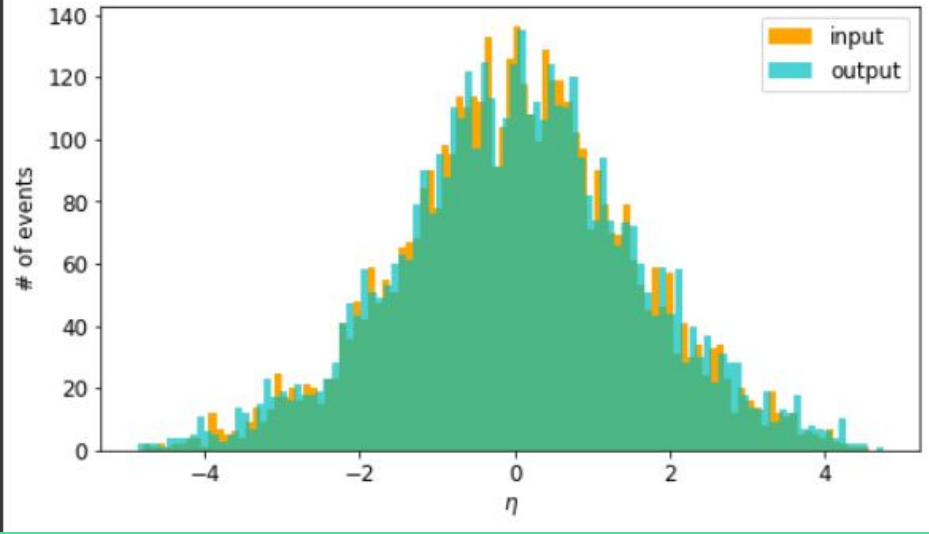
We should also observe the accuracy of the network.This metric is computed with Mean Squared Error that basically measures the difference between the input and output.

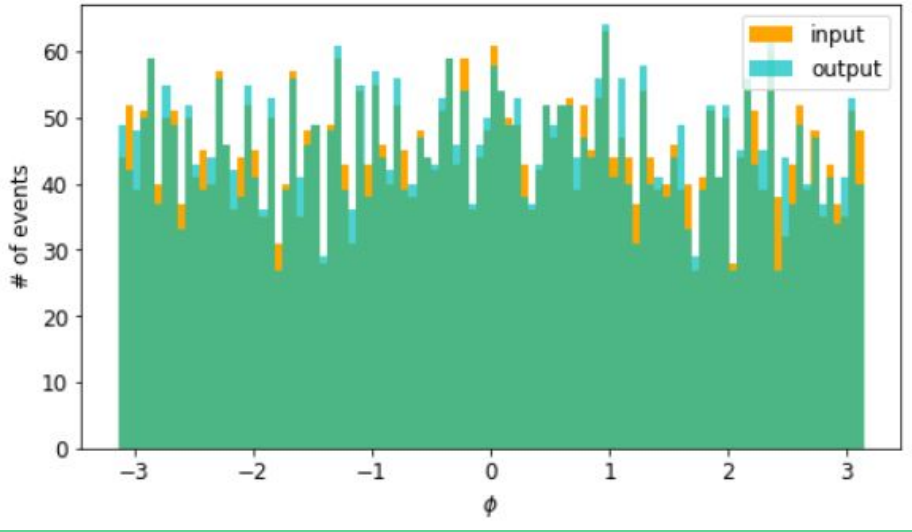
In all the tests the error of the network did not surpass the order of 0.001.

With this two observations, it is pretty plausible to enunciate that this network is doing a pretty good job of encoding and decoding the data


input
output
# of events
0
100
200
300
400
0
1
2
3
4
E
1e6


input
output
# of events
0
200
400
600
800
1000
0.0
0.5
1.0
1.5
2.0
p_T
1e6


input
output
# of events
0
20
40
60
80
100
120
140
−4
−2
0
2
4
η


input
output
# of events
0
10
20
30
40
50
60
−3
−2
−1
0
1
2
3
ϕ

Thank you!

For more details on the implementation and the neural network check the notebook on :

https://github.com/vrrusso/atlas-autoencoders-evaluation

Resources

https://github.com/ATLAS-Autoencoders-GSoC-2021/Evaluation-Exercise/blob/main/TLA_4D_Example.ipynb

Applied Deep Learning - Part 3: Autoencoders | by Arden Dertat

https://zenodo.org/record/3961917/files/dataset.pdf

Understanding Fastai's fit_one_cycle method

This thing called Weight Decay. Learn how to use weight decay to train... | by Dipam Vasani

Dive into Deep Learning — Dive into Deep Learning 0.16.1 documentation