Cloud Atlas

An LstmEncoder for UHECR AirShowers

G. Becuzzi L. Papalini

July 2022

Table of Contents

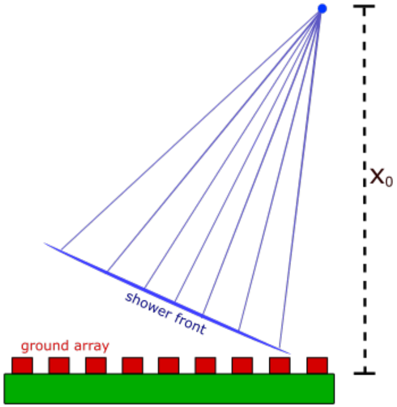
Table of Contents

UHECR Airshower

When *Ultra High Energy Cosmic Rays* (UHECR) enters the atmosphere they produce a particle cascade.

Detection: grid of water-Cherenkov ground based detectors.

Prediction: X_0 height at which the shower forms.



Dataset, first glance

The dataset is composed of 10^5 simulated events:

- 9x9 grid of detectors
- most intense detector at the center
- 80 frames of time series (40 MHz sampling rate)
- 1 frame of times of first arrival

The single record shape is then $(80 + 1, 81)$
The `pd4ml` package splits by default in 70% train 30% test.

Table of Contents

Split the dataset

Using a generator (`keras.utils.Sequence`)

- inherit multiprocessing features
- has default callbacks

The dataset is splitted *record by record* for index shuffling
The effect of the high reading time from memory ($\approx 3ms$) is mitigated by `keras` multiprocessing
For the design of the net it is convenient using `numpy` structured arrays

Split the dataset: `funky_dtype`

```python
    # custom numpy dtype
    funky_dtype = np.dtype(
        [
            ("outcome", np.float64),
            ("time_series", np.float32, (80, 81)),
            ("toa", np.float32, (9, 9, 1)),
        ])

```

Data is extracted: from a conceptually *ihomogeneous* list (activity time series together with times of arrival) to
$(80+1,81) \rightarrow [(''\textit{toa}'', (9,9,1)), (''\textit{timeseries}'', (80,9,9))]$
Data can be accessed depending on what is needed

DataFeeder class

Ensures an easy way to train the subnets separately

- shuffles data randomly
- input fields can be specified
- can be extended to more complex training strategies

DataFeeder class
Curriculum learning

Using a pre-trained network data can be “scored” in ascending order of difficulty
(work in progress) This can lead to a learning speed-up and improvements in resolution
Caveat: this training strategy is not well suited (conceptually at least) for regression tasks, since it is not clear what a “difficult” sample would look like.

Data Augmentation

Dataset has a lack of high events ($X > 850$m) so the network resolution is worse for samples corresponding to this range

Strategy

Increase the number of samples conditionally on event heigth using the symmetries of the problem

Data is augmented using

- flip up-down
- flip left right
- rotation of 90°

It must me higlighted that only a subset of the available data undergoes this procedure.
Augmenting the whole dataset would leave the sample distribution unchanged and thus would not lead to improvements.

Resolution

The reference article suggests using the resolution:

Resolution

defined as the standard deviation of the distribution given by the difference between the predictions and the actual values of X_{max}

We point out that

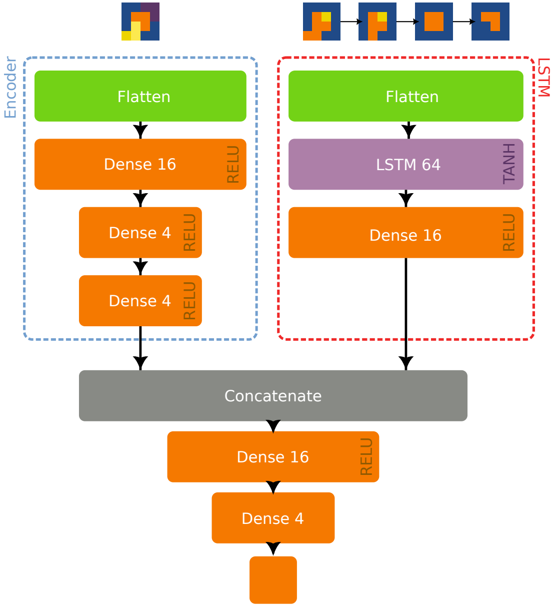
$$\sigma^2 = \frac{1}{N}\sum_i (\delta_i - \bar{\delta})^2$$

is a sensible estimator of "how much the net has gone wrong" only if $\bar{\delta} = 0$, for which the adopted resolution is equal to the *RMSE* of the distribution

$$RMSE^2 = \frac{1}{N}\sum_i (x_i - \hat{x}_i)^2$$

Since (on a typical train) $\bar{\delta} \approx 10$m we preferred the RMSE.

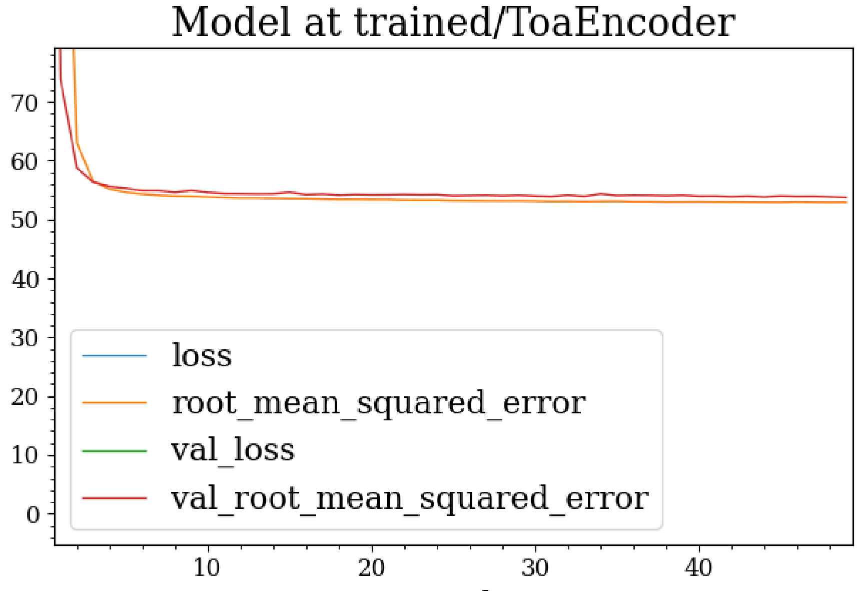
Table of Contents


Encoder
Flatten
Dense 16
RELU
Dense 4
RELU
Dense 4
RELU
LSTM
Flatten
LSTM 64
TANH
Dense 16
RELU
Concatenate
Dense 16
RELU
Dense 4

Overview on the network

The assumption that lead to this design is that from the time of arrival matrix it is possible to infer some kind of “homogeneous” shower parameters (incidence angle, spread, etc.) while the time series can be processed by a recurrent network.

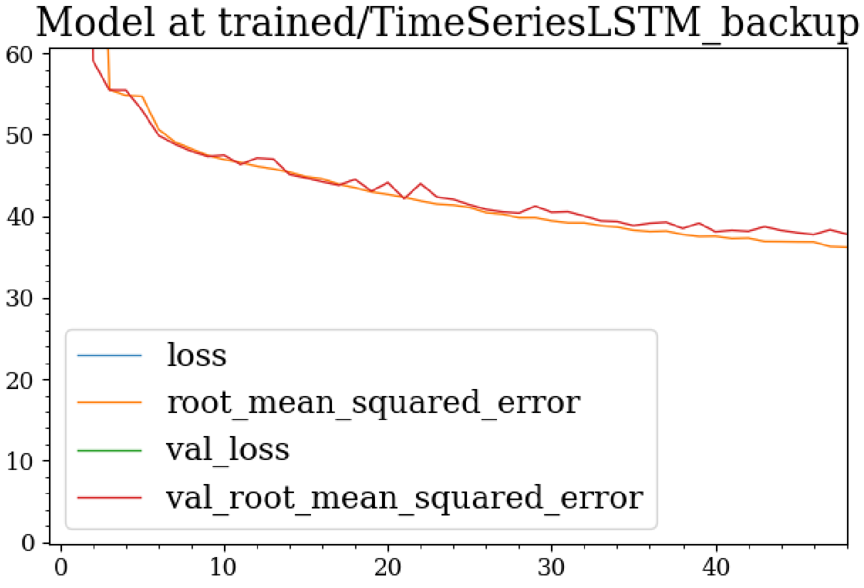
Encoder for time of arrivals



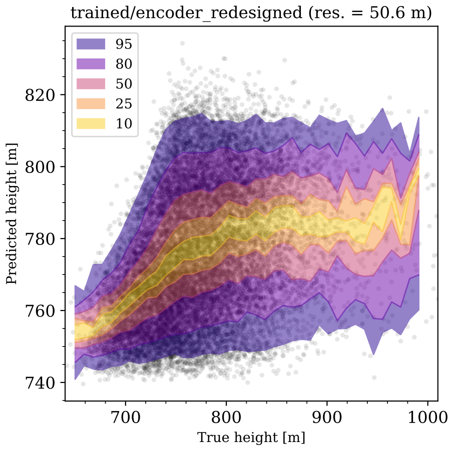
LSTM

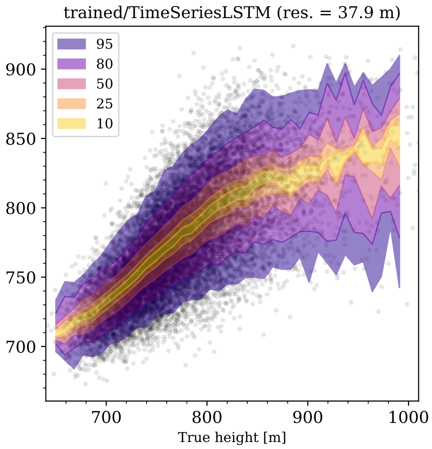
si spiega che cos'è

LSTM for the time series



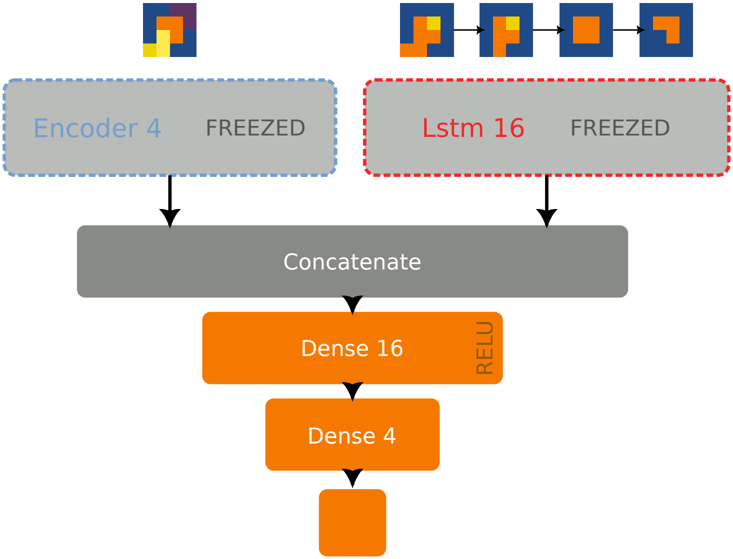
Subnets performance



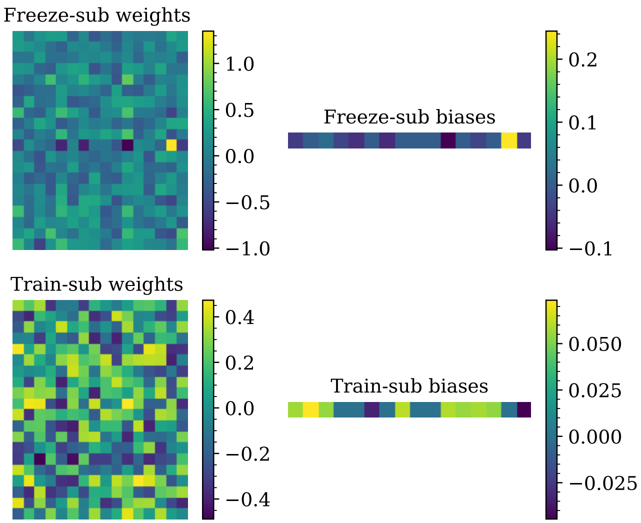


Concatente + dense layers

Subnets train freezing



Subnets train freezing



Network's output

```
    import numpy as np

    def incmatrix(genl1,genl2):
        m = len(genl1)
        n = len(genl2)
        M = None #to become the incidence matrix
        string = "ciao"


```

Hyperparameters tuning

Whole Network performance

Test setup on CircleCI

Danke e bibliography

Danke Schon