Unsupervised learning (UL)

Type of ML in which the algorithm is not provided with any pre-assigned labels for the training data.
As a result, UL algo., must first self-discover any naturally occurring patterns in that training data set. Common examples include clustering, anomaly detection
Unsupervised techniques requiring a greater amount of training data and converging more slowly to acceptable performance.

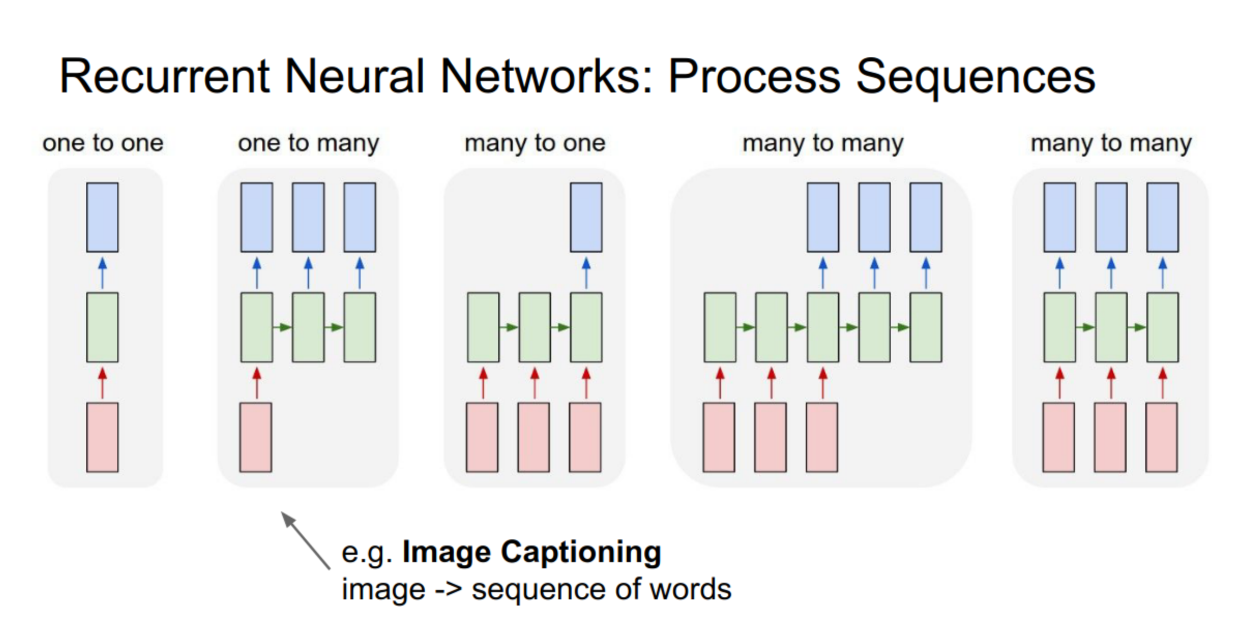
Unsupervised anomaly detection techniques detect anomalies in an unlabeled test data set under the assumption that the majority of the instances in the data set are normal by looking for instances that seem to fit least to the remainder of the data set.

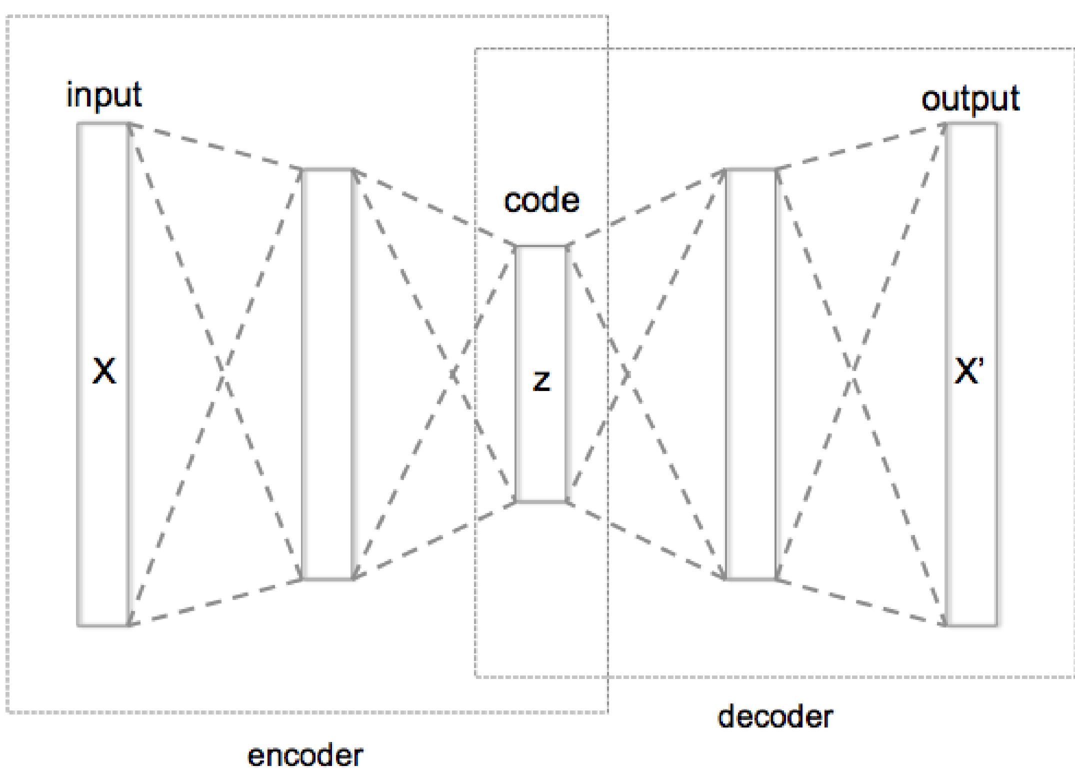
Popular techniques
----

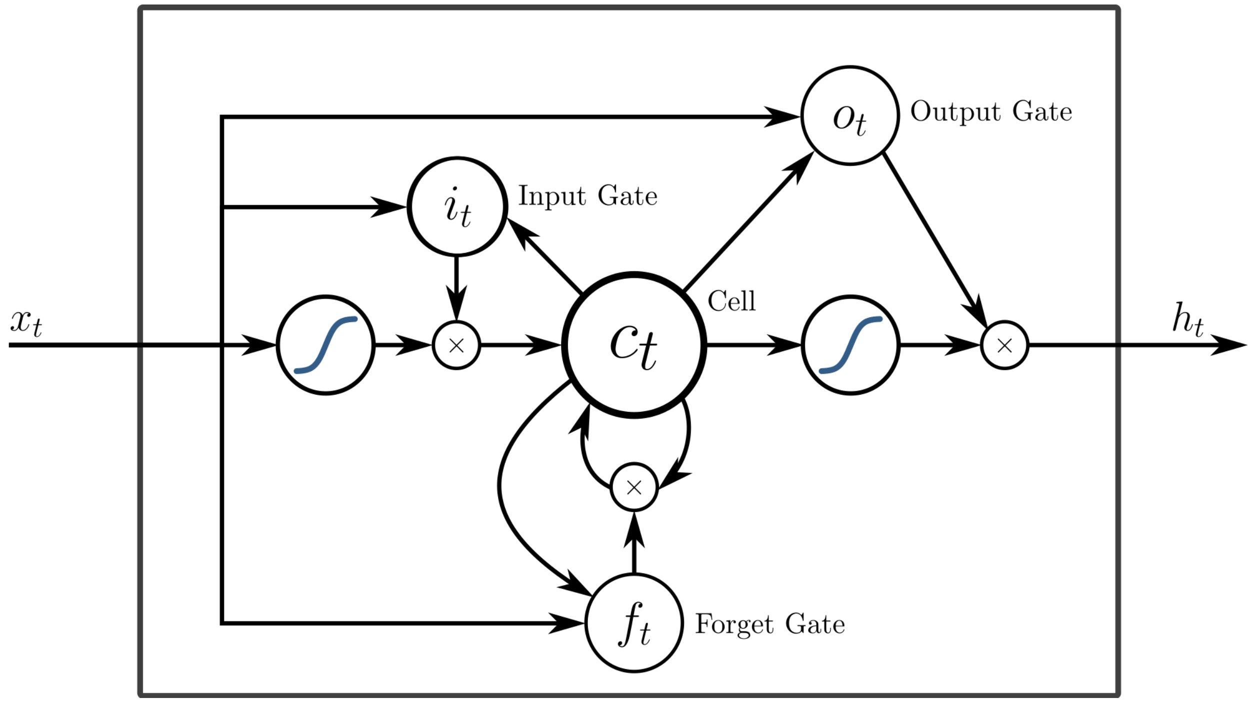
* Density-based techniques (k-nearest neighbor local outlier factor etc..)
* autoencoders, variational autoencoders, long short-term memory neural networks.
* Bayesian networks
* Hidden Markov models
* Cluster analysis-based outlier detection.

Long Short Term Memory Networks for Anomaly Detection in Time Series
----

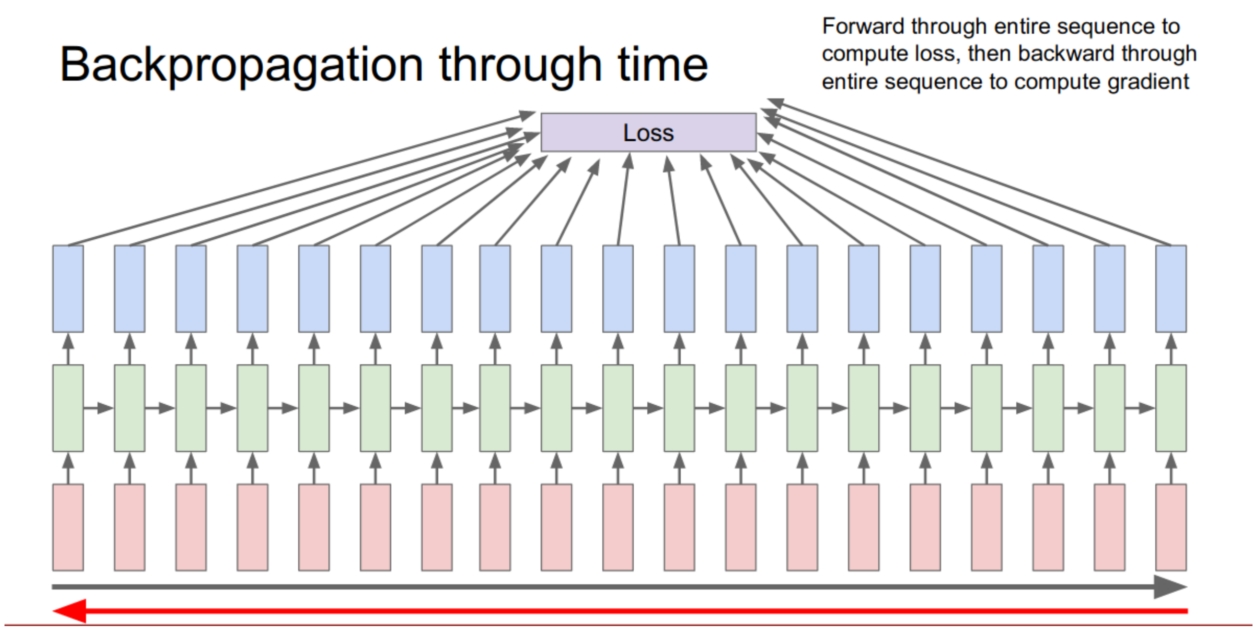
Useful for learning sequences containing longer term patterns of unknown length, due to their ability to maintain long term memory.
LSTM neural networks overcome the vanishing gradient problem experienced by recurrent neural networks.
Stacked LSTM based prediction model.
Anomaly detection using the prediction error distribution: With a prediction length of l, each of the d dimensions of $x(t) \sim X$ for $l < t <= n-l$ is predicted l times. We compute an error vector $e(t)$ for point $x(t)$, where $e(ij)(t)$ is the difference between $x(i)(t)$ and its value as predicted at time $t-j$.


Recurrent Neural Networks: Process Sequences
one to one
one to many
many to one
many to many
many to many
e.g. **Image Captioning**
image -> sequence of words


input
output
code
X
z
X'
decoder
encoder

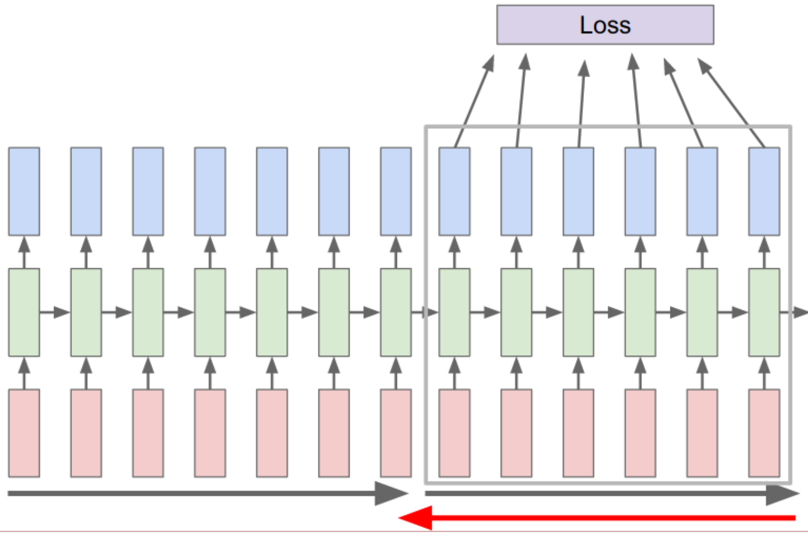


x_t
i_t
Input Gate
o_t
Output Gate
c_t
Cell
h_t
f_t
Forget Gate



Backpropagation through time
Forward through entire sequence to compute loss, then backward through entire sequence to compute gradient
Loss

Truncated Backpropagation through time

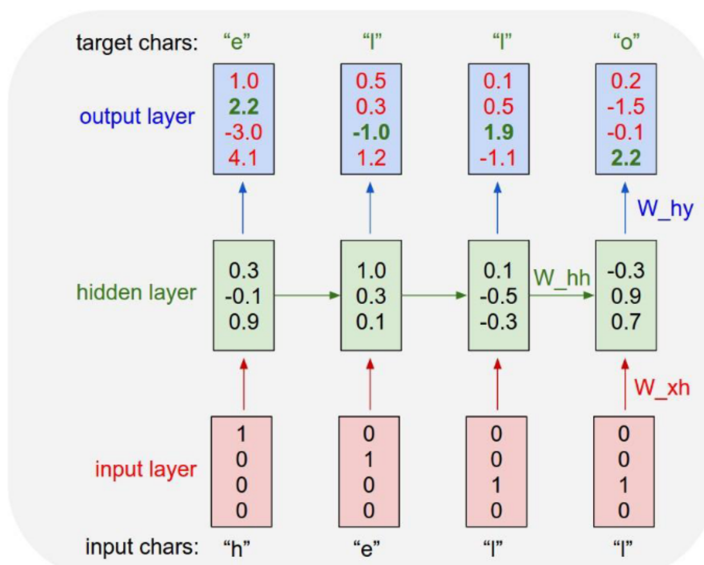


Carry hidden states forward in time forever, but only backpropagate for some smaller number of steps

Example:
Character-level Language Model

Vocabulary:
[h,e,l,o]

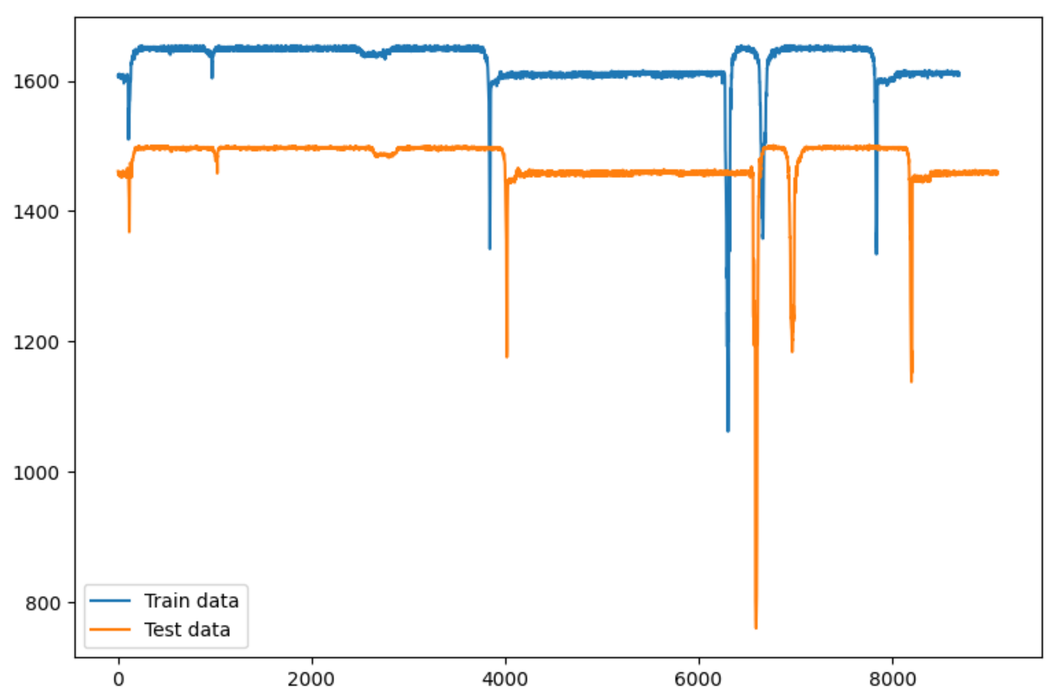
Example training sequence:
“hello”



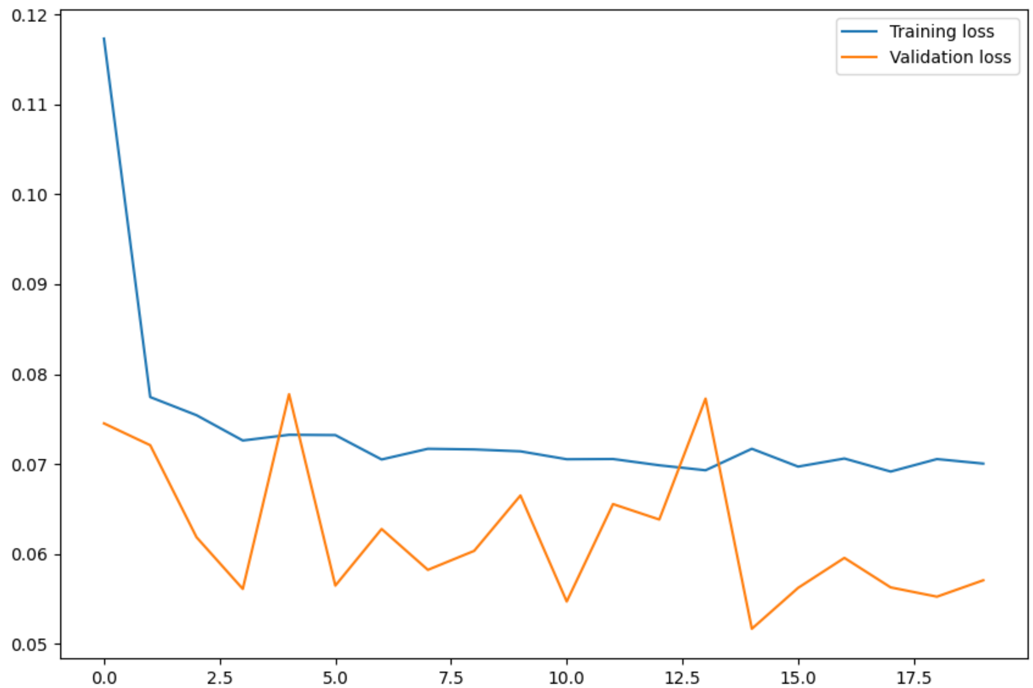
Layer (type)	Output Shape	Param #
Input	(None, 5, 1)	0
lstm (LSTM)	(None, 128)	66560
repeat_vector (RepeatVector)	(None, 5, 128)	0
lstm_1 (LSTM)	(None, 5, 128)	131584
time_distributed (TimeDistri	(None, 5, 1)	129

Total params: 198,273
Trainable params: 198,273
Non-trainable params: 0

Training data and Test data Graph.



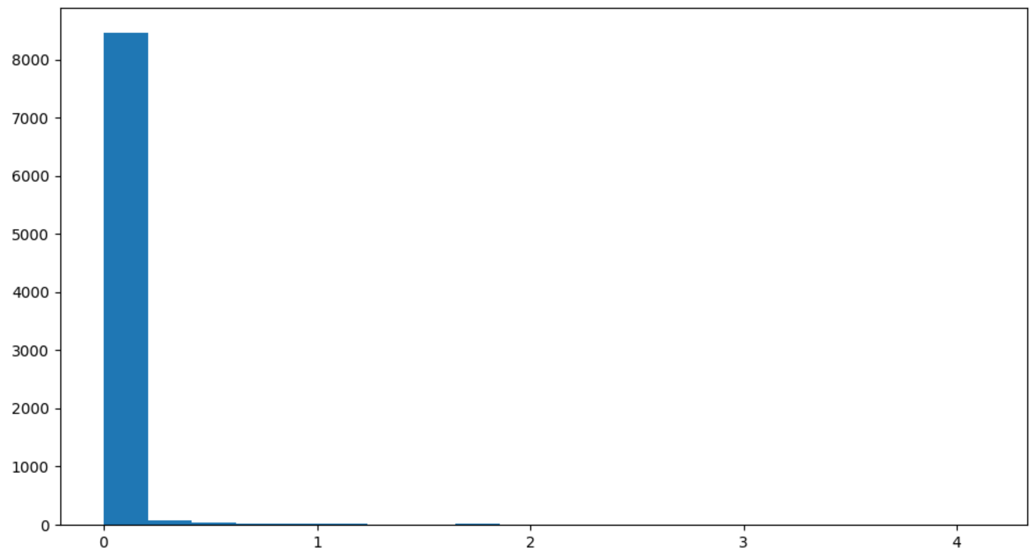
Training loss and Validation loss Graph.

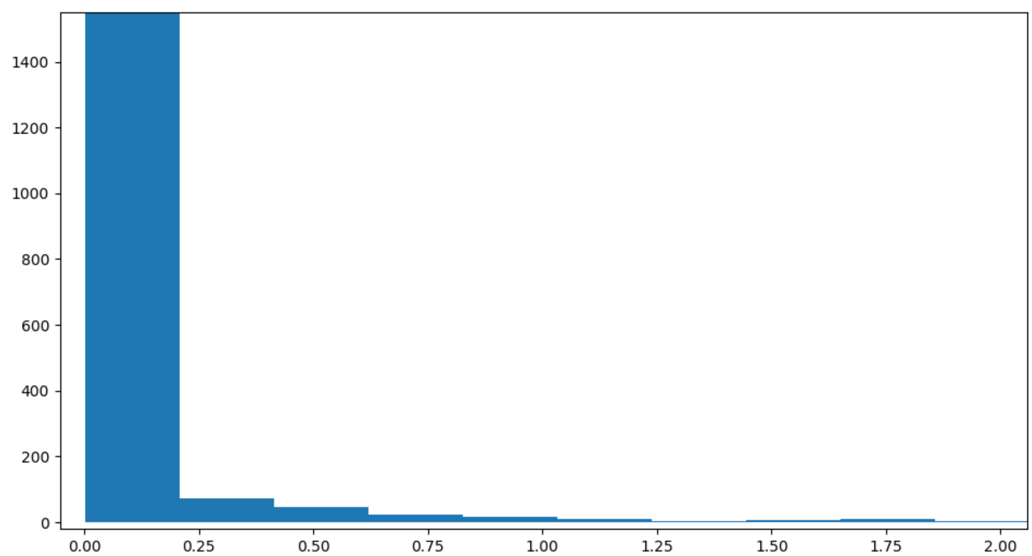


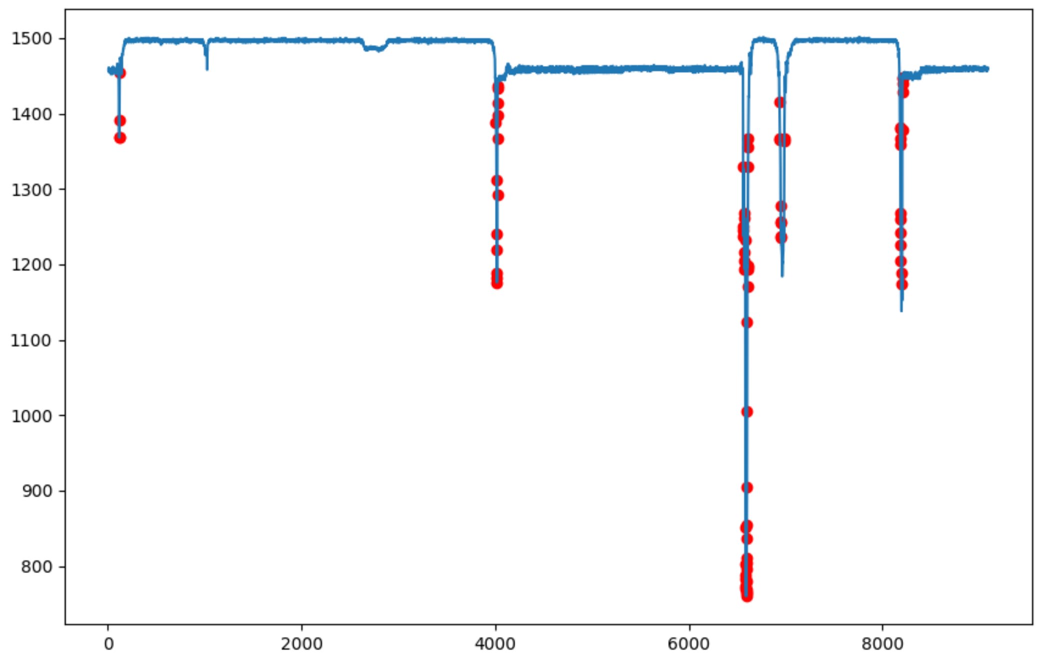
Anomaly is where reconstruction error is large. We can define this value beyond what we call anomaly. Let's look at Mean Average Error in training prediction.

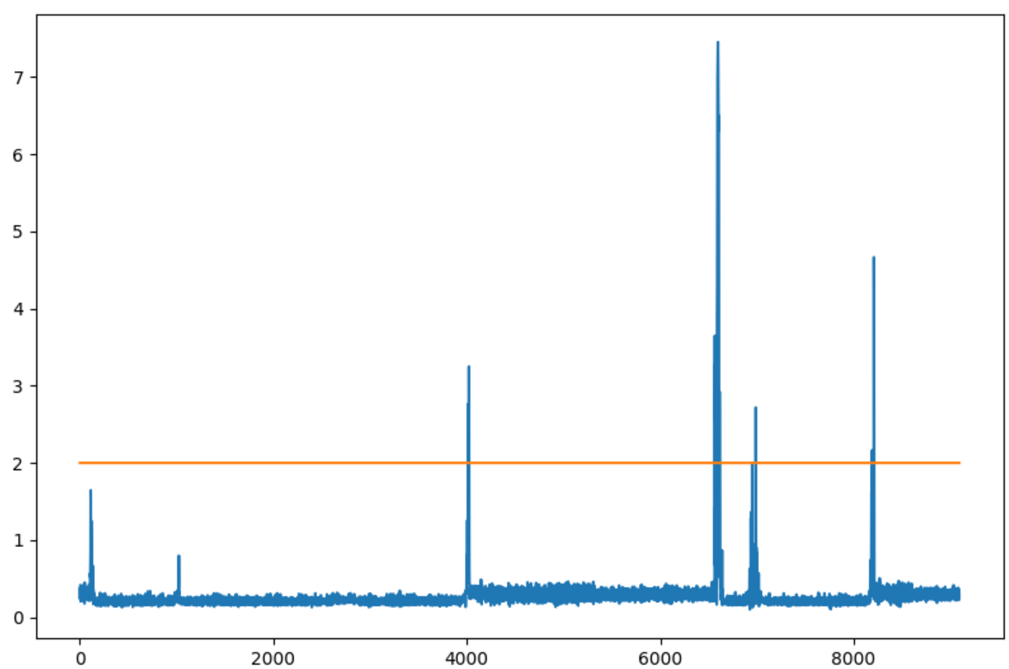
```
trainPredict = model.predict(trainX)
trainMAE = np.mean(np.abs(trainPredict - trainX), axis=1)
plt.hist(trainMAE, bins=20)
plt.show()
```

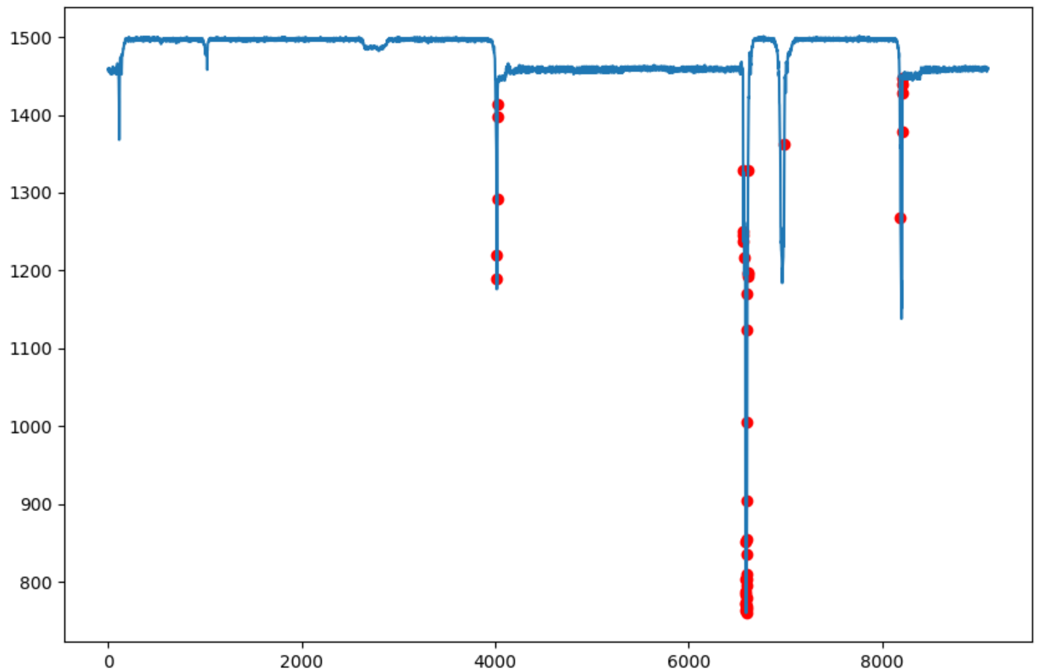
Now let's see the zoomed view and decide error greater than x, I'll consider it as the dip.


8000
7000
6000
5000
4000
3000
2000
1000
0
0
1
2
3
4


1400
1200
1000
800
600
400
200
0
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00


1500
1400
1300
1200
1100
1000
900
800
0
2000
4000
6000
8000


7
6
5
4
3
2
1
0
0
2000
4000
6000
8000


1500
1400
1300
1200
1100
1000
900
800
0
2000
4000
6000
8000