Time Series Anomaly Detection on Wind Turbines Power Curve with LSTM

Onur Poyraz
Department of Computational Science and Engineering
Boğaziçi University
onur.poyraz@boun.edu.tr

Abstract

In this project, I deal with the wind turbines power curve data. I design a model that learn the relationship between the turbine parameters and also learn sequence patterns of the turbines from the given input. This model acts as a generative model and I use that model result as a denoised ground truth. To detect the anomaly, I use mean square error between the observed data and corresponding LSTM outputs. If there is the high error for a point I label that point as an anomaly. For detect collective anomaly, I use the moving average on resulting error term.

1 Introduction

Anomaly detection (also outlier detection) is the identification of items, events or observations which do not conform to an expected pattern or other items in a dataset[1]. The main goal of this technique is to find or detect the problematic observations which indicate some kind of malfunctions or defects.

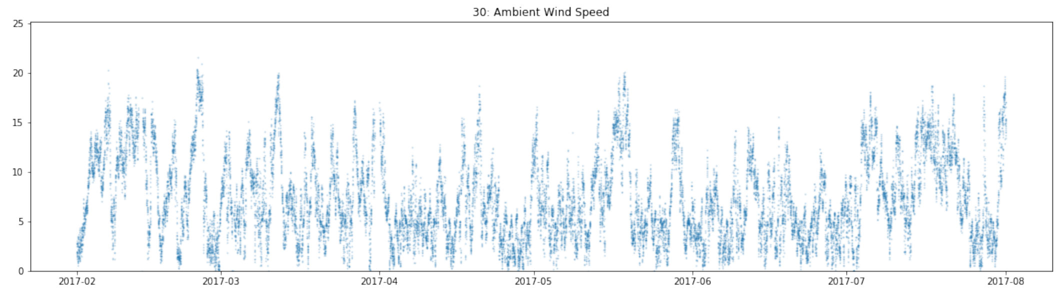
There are several types of anomaly detection. These types usually determined by the dataset and the corresponding problems. In literature, typically there are three classes of anomaly detection:

1. Supervised Anomaly Detection: Requires a labelled dataset for each observation. Therefore the main problem is becoming a classification problem. The only difference is that abnormal observations do not act as a class because of the unbalanced nature of anomalies.
2. Semisupervised Anomaly Detection: Requires the knowledge of the normal behaviour at the training set. In this technique, from the normally labelled training data normal working pattern of the system will be learned. At the test phase, abnormal observations will be detected.
3. Unsupervised Anomaly Detection: In this technique, there is dataset without labels. The main assumption of this technique is that one should suppose that there is the small amount of the abnormal observations in both train and the test dataset. The main goal of this method is to detect abnormal working pattern rather than finding abnormal observations.

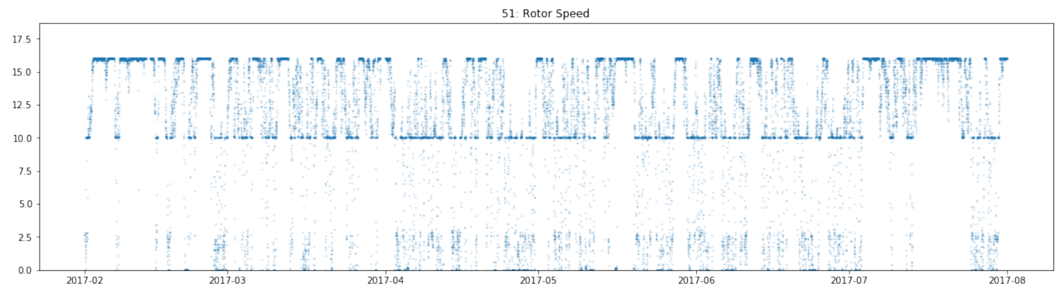
In particular, in the context of the anomaly detection the interesting objects are often not rare objects, but unexpected bursts in activity. Therefore, to detect the abnormal activities one should also consider the time-series effect of the system.

Time series is a sequence taken at successive equally spaced points in time thus it is a sequence of discrete-time data. In the context of the time series, observations will affect by previous observations. So this property is important for some kind of anomaly detection[2] problems if the objective is to detect burst in anomalies rather than single point anomaly detection[3].

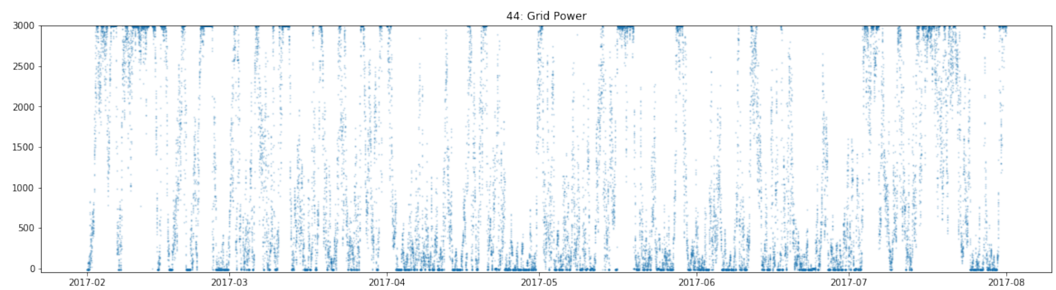
In this project, I will deal with unsupervised anomaly detection method with time series property since my dataset does not contain any labels about observations and the anomalies lie in the pattern



(a) Wind speed graph as a time-series



(b) Rotor speed graph as a time-series



(c) Generated grid power graph as a time series

Figure 1: Individual graphs for the data

of the system. You can find the implementation at the appendix.(I do not put the plot codes and the small difference like different hidden size)

2 Dataset

In this project, I use the Borusan Vestas V90-3MW wind turbines dataset. This dataset consists of 25737 observations obtained by continuous collection of wind speed, rotor speed and generated power values of 20 different types of turbines between February and July of 2017 in a sampling interval of 10 minutes. The dataset contains some Null and corrupted sensory information because of the measurement errors. These errors can occur at random observations and we assume that these errors just happens does not continuous. Therefore we know that observations do not collect totally clear and include incorrect values.

In this dataset, wind speed can be seen as an input to the system while the rotor speed and the generated power are the outputs of the system. However, it can be clearly seen from the above graphs that there is not a direct relationship between this features. The reason behind this phenomena is that previous observations and working condition of the turbine affect the current time observations. This is because of the mechanical system of the wind turbines. Because of the physical relationships, the outputs cannot be generated directly according to input data. This property of the dataset requires time series analysis. As a result of that I use HMM in my previous works and now I apply LSTM network to the dataset.

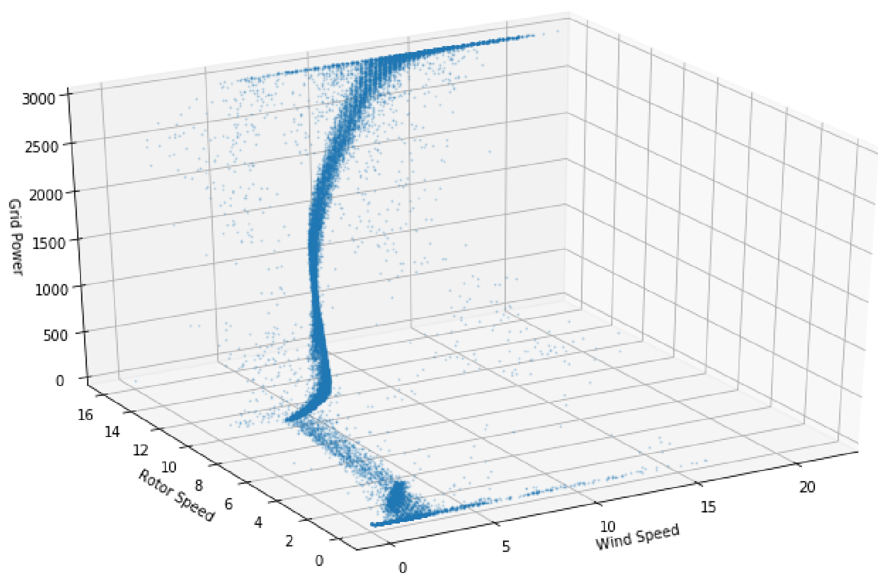


Figure 2: The relationship between the wind speed, rotor speed and grid power

3 Methods

In this project, I create models that deal my data as a time series. Therefore I use neural networks with recurrence. This is because my model should learn the data as a sequence because of its dependencies on previous observations. To learn the underlying working pattern of the turbines I use different kind of network structures with recurrency and compare the results of them. For this project I use RNN, LSTM and Bidirectional LSTM and I compare the results of them.

In this project I use wind speed as an input for the system and I use rotor speed and grid power as an output for the system. By the help of the recurrent networks the model learn the normal working patterns of the turbine according to input (wind in may case). Since I do not know about the labels of the dataset, I use trained networks as a generative model and I compare the generated outputs with the observations.

3.1 RNN

Recurrent Neural Networks are the neural networks for processing sequence data [4]. They are networks with loops in them, allowing information to persist. This chain structure makes them powerful to analyze sequences and time series problems. The mathematical definition of the RNNs are given below:

$$s(t) = f(b_s + U_{sx}x(t) + U_{ss}s(t-1)) \tag{1}$$
$$h(t) = g(b_h + U_{hs}s(t-1)) \tag{2}$$

By the help of the chain structure and recurrence, these networks can learn sequences and patterns but they have some disadvantages. One of the main problems with the RNNs is the handling with the long-term dependencies. By the increasing gap, these networks usually forget the previous inputs. This problem usually called as vanishing gradient.

3.2 LSTM

Recurrent neural networks with Long Short-Term Memory (LSTMs) have emerged as an effective and scalable model for several learning problems related to sequential data [5]. They are special kind of RNNs which are good at learning long-term dependencies. The central idea behind the LSTM architecture is a memory cell which can maintain its state over time, and non-linear gating units which regulate the information flow into and out of the cell. Like RNNs, LSTMs also have this chain-like structure, but the repeating module has a different structure [6]. The mathematical definition of the LSTMs are given below (for the simplicity I do not write the bias term in the formulas):

$$x(t) \qquad \text{Input} \tag{3}$$
$$i(t) = \sigma(U_i[x(t), h(t-1)]) \qquad \text{Input Gates} \tag{4}$$
$$f(t) = \sigma(U_f[x(t), h(t-1)]) \qquad \text{Forget Gates} \tag{5}$$
$$\tilde{s}(t) = \tanh(U_s[x(t), h(t-1)]) \qquad \text{Proposed Memory} \tag{6}$$
$$s(t) = f(t) \odot s(t-1) + i(t) \odot \tilde{s}(t) \qquad \text{Memory} \tag{7}$$
$$o(t) = \sigma(U_o[x(t), h(t-1)]) \qquad \text{Output Gates} \tag{8}$$
$$h(t) = o(t) \odot \tanh(s(t)) \qquad \text{Output} \tag{9}$$

3.3 Bidirectional LSTM

These networks consist of two same dimensional stacked LSTM networks which work in opposite directions [7]. This method brings future knowledge to the current observations. The output is then computed based on the hidden state of both LSTMs. Since there is a two directional network it is computationally expensive method rather than the others.

4 Evaluation

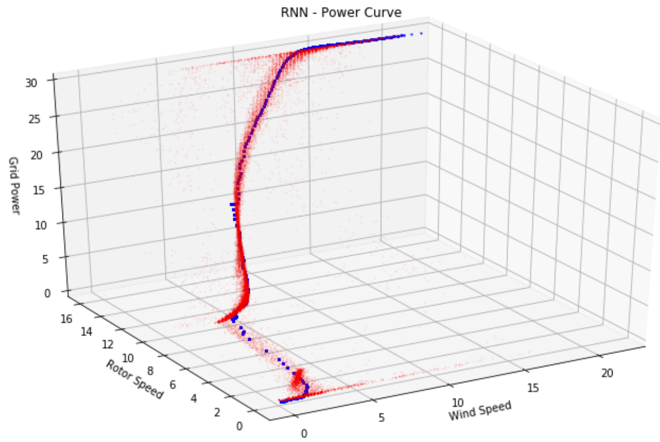
After the training of the recurrent neural networks I use them as a generative model and compare the each generated result with the corresponding observations and calculate the distance between them. After that I round that distances (because of the sensory malfunctions and null values, at some time steps there is high error distances while the system works normally) with moving averages. In the literature some of the works made this work by the network itself but I split them because it is computationally costly. At the final step I define a threshold value for the anomalies. Observations with error distance which are greater than this threshold considered as an anomaly.

5 Experimental Results

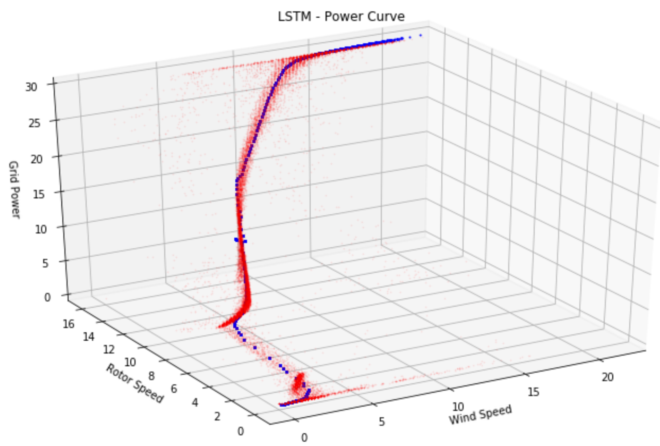
In the experiments I compare three different network with the different hidden sizes. I use learning $rate = 0.01$ for all of the setups. For both of the models I use $epoch = 10$. Again in all of the test setup I use same batch size(1000). For the experiments I use different number of hidden layers and hidden layer size. Since the results does not differs to much I do not put all of the graphs to this report. In this part I just compare results with different model with same parameters. ($numberofhiddenlayer = 2, hiddensize = 256$). The detailed results are shown in the figures.

6 Conclusion

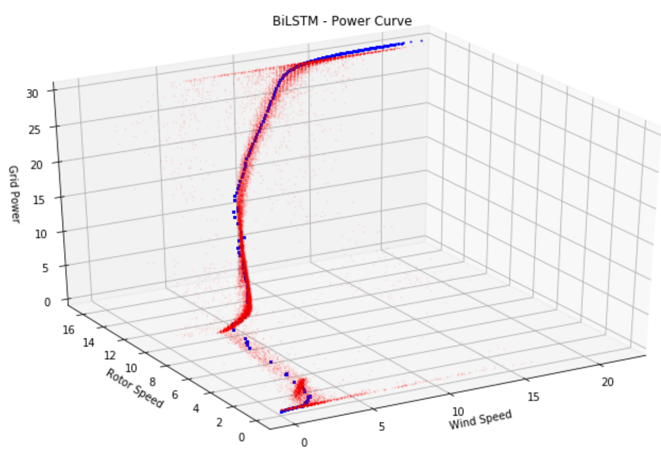
In this project I implement three different variants of the recurrent networks and I obtain almost same results. This is because my dataset does not contains long-term dependencies and to obtain good estimation we do not need to know information from future. These networks almost totally correctly estimate the generated power and rotor speed values from the wind speed.



(a)

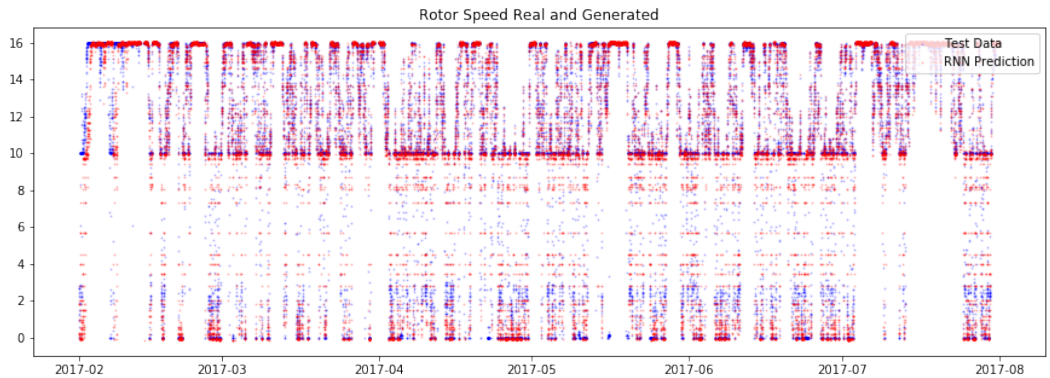


(b)

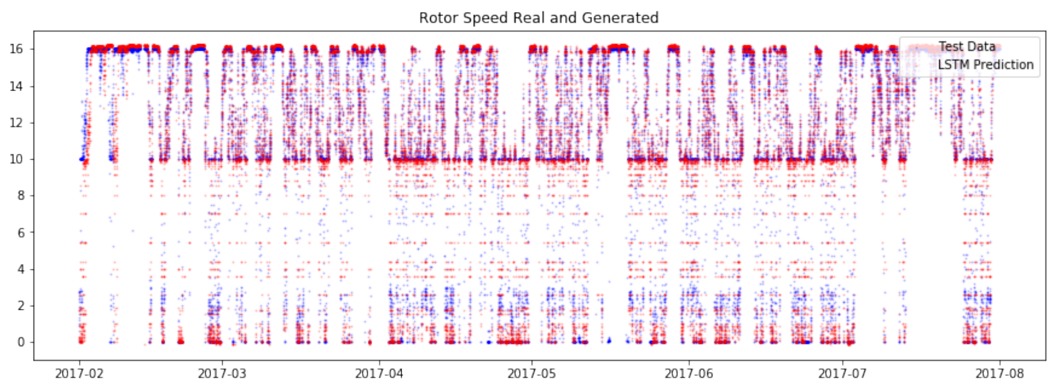


(c)

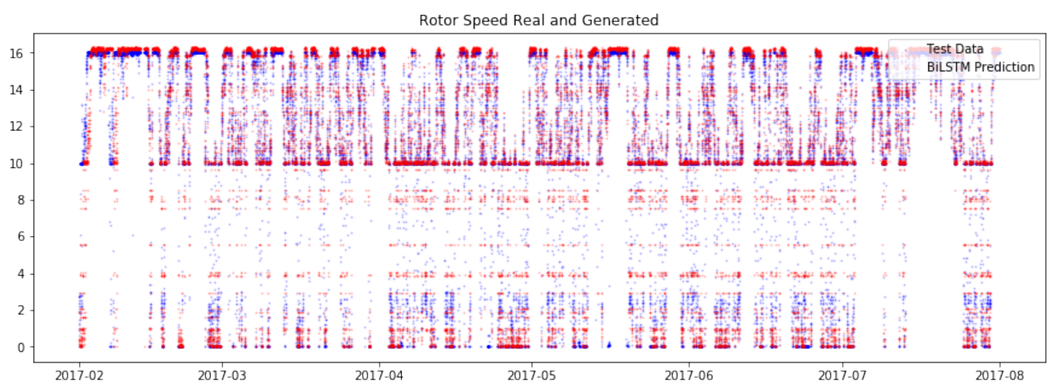
Figure 3: The generated and observed power curves. Both models succesfully creates the power curve with a very low variance. There is just a little difference in the model results. Therefore I can use the result of both models as a baseline to detect anomalies. The red dots represents the observations while blue dots shows the predictions



(a)

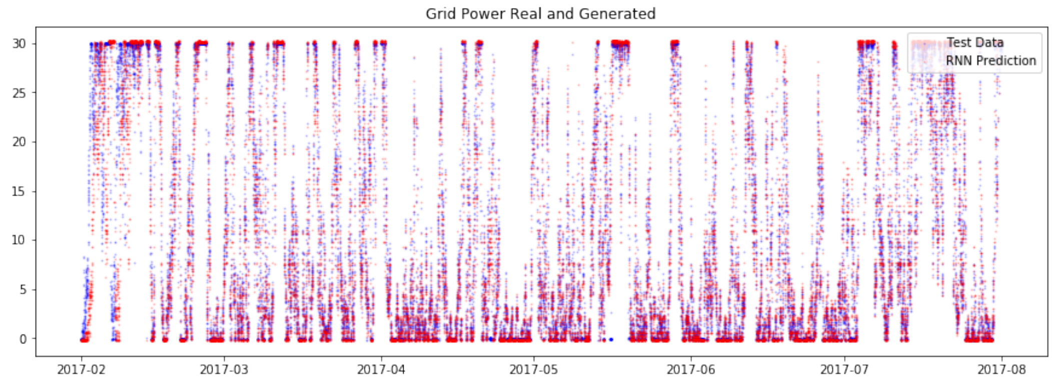


(b)

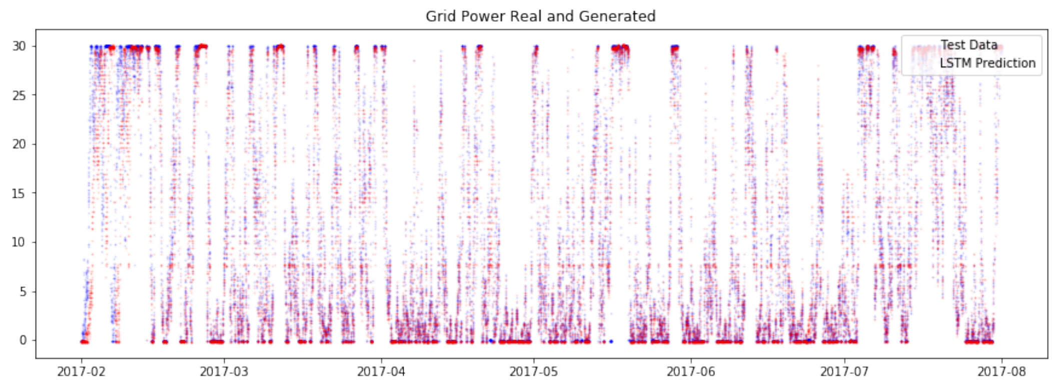


(c)

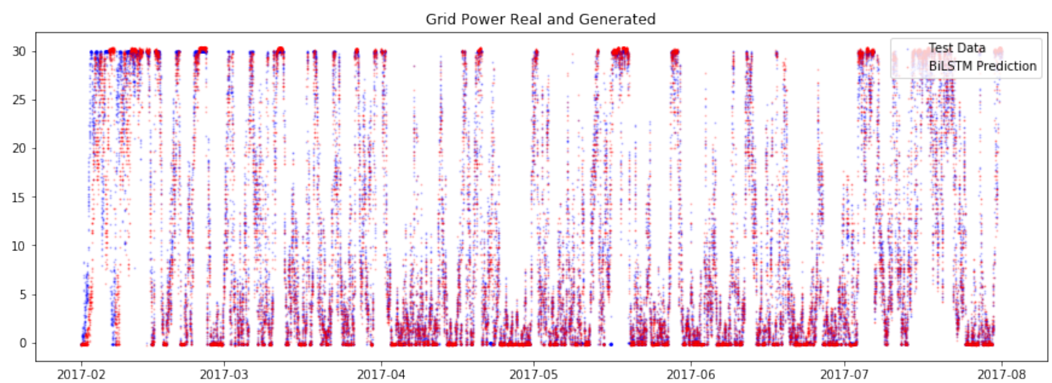
Figure 4: Predicted rotor speed data for different model. Except an small difference both model creates almost same predictions. The blue dots shows the observations while red dots shows predictions



(a)

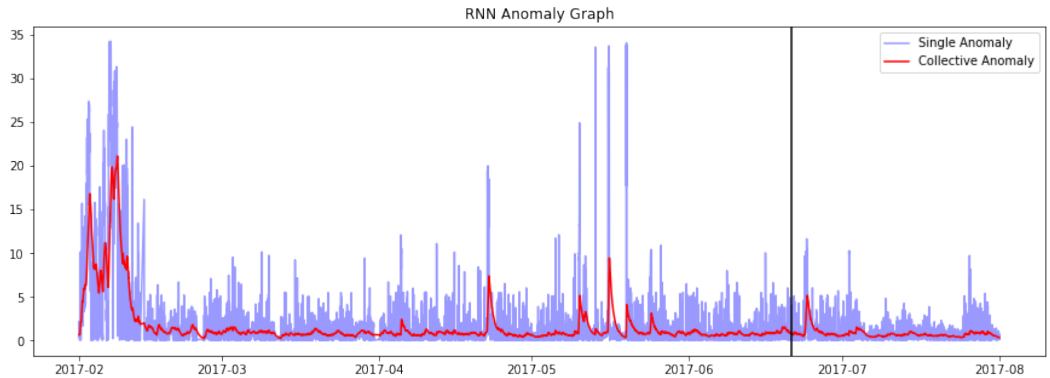


(b)

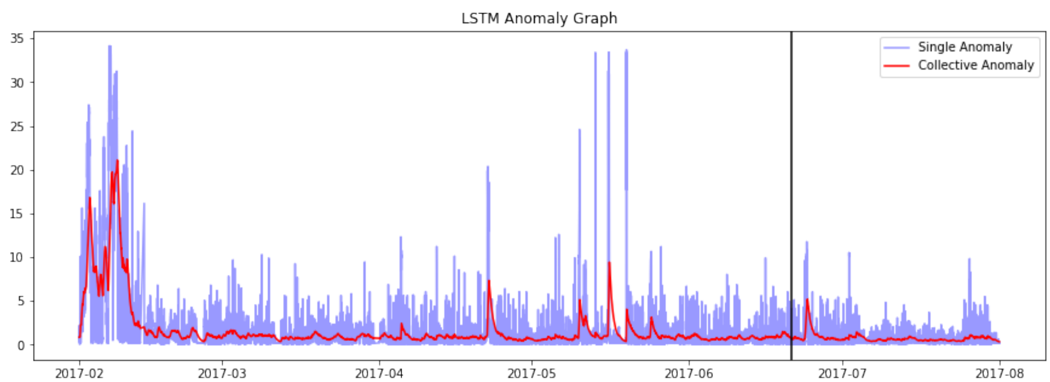


(c)

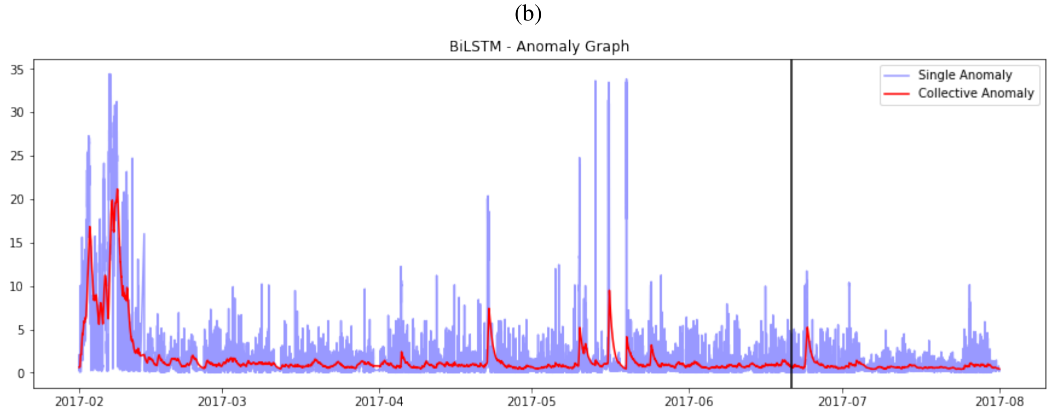
Figure 5: Predicted generated power data for different model. Except an small difference both model creates almost same predictions. The blue dots shows the observations while red dots shows predictions



(a)



(b)



(c)

Figure 6: Anomaly scores of the each model for the test data. Except from small variations each model creates desired results. Black vertical line represents the malfunction in turbines and the system generates warning before the malfunctions. Additionally we know that at the February there is iciness on the turbines. Models succesfully detects this anomalities.

References

[1] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):15, 2009.

[2] Pankaj Malhotra, Lovekesh Vig, Gautam Shroff, and Puneet Agarwal. Long short term memory networks for anomaly detection in time series. In *Proceedings*, page 89. Presses universitaires de Louvain, 2015.

[3] Loïc Bontemps, James McDermott, Nhien-An Le-Khac, et al. Collective anomaly detection based on long short-term memory recurrent neural networks. In *International Conference on Future Data and Security Engineering*, pages 141–152. Springer, 2016.

[4] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.

[5] Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. Lstm: A search space odyssey. *IEEE transactions on neural networks and learning systems*, 28(10):2222–2232, 2017.

[6] Christopher Olah. Understanding lstm networks, 2015. *URL https://colah. github. io/posts/2015-08-Understanding-LSTMs*, 2015.

[7] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997.