Multivariate Time-series Anomaly Detection using Graph Neural Network

Team member: ***Binglin Li*** *(binglin.li@cn.ca) and* ***Van Nam Vu*** *(vannam.vu@cn.ca)*

1. INTRODUCTION

The goal of this project is to explore Graph Neural Network (GNN) on time-series data, in which we learnt and ran the model in the paper [1]. We used the subset of the public dataset Mars Science Laboratory (MSL) rover, Curiosity [2] to train and test the model in this project. The works are summarized as below.

- We explored the subset of MSL dataset by visualizing the time-series data for some sensors in the training and testing sets.
- We visualized the embedding vectors with t-SNE plot for each sensor in 2-D dimensional space after training. Sensors with similar embedding values suggest high correlations.
- The anomaly detection results in the whole system were displayed and compared with ground truth attack labels for each timestamp.
- To better understand which sensor was attacked, we also visualized the graph layout with attention scores as edge weights and show the prediction result for the sensor with highest anomaly score compared with the observed time-series data.

2. MODEL

The overall framework of the paper [1] is given in Figure 1. The input is N time-series sensor data. To capture the different relationships among sensors in a multidimensional way, an embedding vector is applied for each sensor. The parameters of the embedding vectors are learned during the training. These embeddings will be used to learn which sensors are related to one another in the graph structure on the left in the figure, and for the attention weights by concatenating the extracted time-series inputs and the sensor embeddings on the right in the figure.

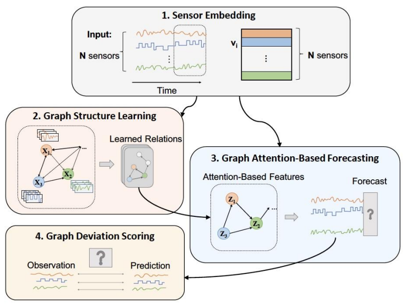


Figure 1. Overview of the framework in [1].

The inputs are defined based on a sliding window of the fixed size over the past time-series data. Features are extracted by fusing a node's information with its neighbors based on the learned graph through the Equations (5)-(8) in the paper. The extracted features are used to forecast the expected behavior of each sensor at different timestamps. At the output layer, the learned representations for all N nodes are element-wise multiplied with the corresponding embedding and followed with some full-connected layers to output the predictions for N nodes at each timestamp.

The Mean Squared Error (MSE) loss is applied between the prediction output and the observed data at each timestamp as given in Equation (10) in the paper.

3. EXPERIMENT

All the experiments can be completed by running the notebook "testrun.ipynb". It will download the folders from Github, train the model and visualize test results.

3.1. Dataset

The training data is in the ./GDN/data/train.csv file and testing data is in ./GDN/data/test.csv file with the last column "attack" to indicate the anomaly label for the whole system at each timestamp. The anomaly label for each sensor can be found in [2].

The time-series training data for sensors "M-6", "M-1", "M-2" and "S-2" are given in Figure 2.

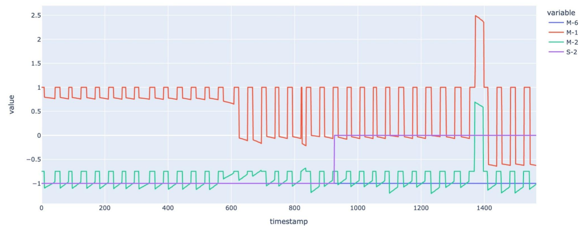


Figure 2. Time-series training data examples.

The time-series test data for sensor "M-6" and "M-2" are given in Figure 3. Sensor "M-6" contains point anomalies and "M-2" contains contextual anomalies where the context of a node needs to be considered for anomalies.

3.2. Results

The testing results are calculated with precision, recall, F1-score and AUC-score as in Figure 4. We also tried to change the number of epochs, learning rate and weight decay to improve the results. As the training set is only part of the MSL dataset, we do not aim to achieve the state-of-art results.

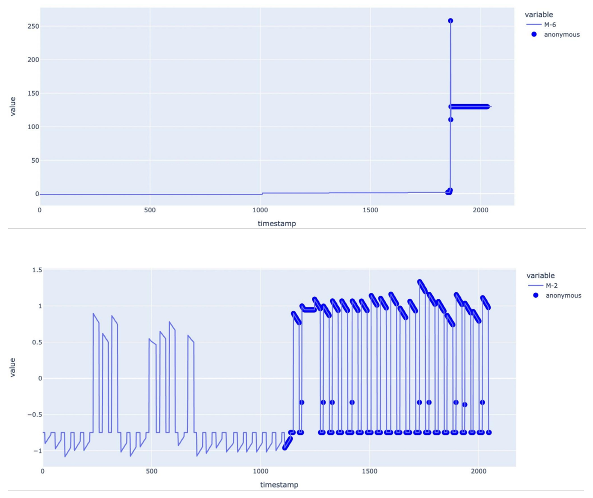
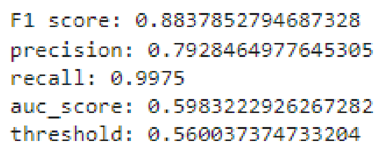


Figure 3. Time-series testing data examples for "M-6" (top) and "M-2" (bottom).



```
F1 score: 0.8837852794687328
precision: 0.7928464977645305
recall: 0.9975
auc_score: 0.5983222926267282
threshold: 0.560037374733204
```

Figure 4. Precision and recall of test results.

3.3. Analyses and Discussions

The model can compute the anomaly score for each sensor and also a single combined score for all sensors in the system for each timestamp. Figure 5 gives the anomaly detection result for all sensors and the ground truth attack label (0 for normal and 1 for attacked). Most of the attacked timestamps are captured in the predictions, so the number of false negatives is small and the recall is high. At some timestamps that are normal, the model gives attack predictions. There are some false positives and that explains why the precision score is relatively low.

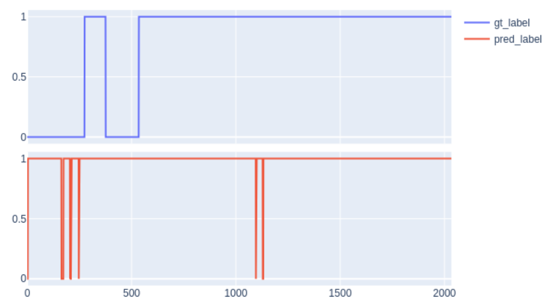


Figure 5. Prediction and ground truth anomalies.

The t-SNE plots in Figure 6 can show the similarity of sensor's behaviors reflected by the clustering in the manifold space. However, as the training set is quite small and due to lack of background knowledge of the telemetry sensors, we cannot make a conclusion for the close sensors.

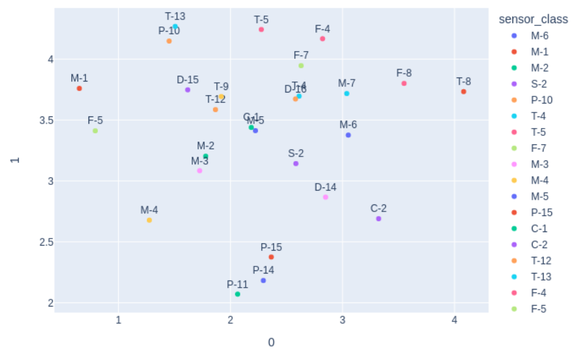


Figure 6. t-SNE plot of embeddings for all sensors.

We visualized the graph layout with attention scores as edge weights and highlighted the sensor ("D-15") which has the highest anomaly score after training in Figure 7. This helps to localize the affected sensors that result in anomalies. As we can see from embeddings in Figure 6, the sensor "D-15" is relatively close to "T-9", "T-12", "P-10". Those relationships can be shown more clearly on the graph and the graph verifies the learned embeddings.

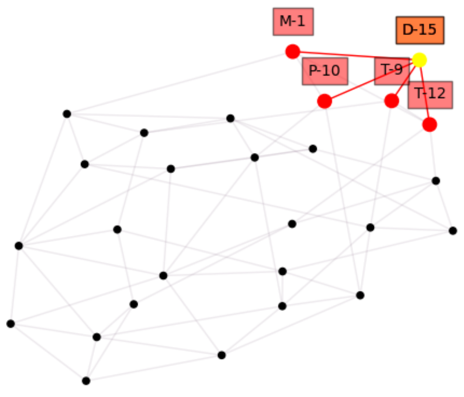
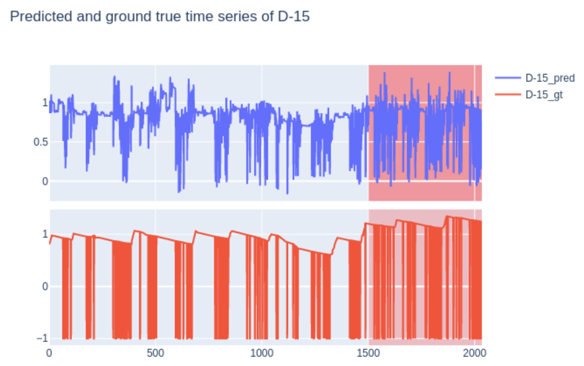


Figure 7. Graph layout with highest anomaly score (yellow node) and its neighbors (red nodes) with edge weights larger than 0.1.

The predictions for sensor "D-15" and one of its neighbors "M-1" are given in Figure 8 which shows how the prediction deviates from the expected behavior. We can see that the model gives the same pattern of time series from predicted and observed data. Time series data is more dense and frequented during the attack period. These results show us the relationship of those sensors which is helpful to find the root cause for troubleshooting.



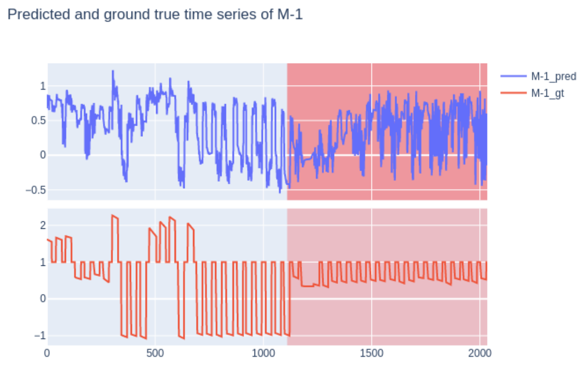


Figure 8. Comparison between the prediction and observed data for sensor "D-15" (top) and "M-1" (bottom). The attack period is shaded in red.

4. SUMMARY AND FUTURE WORKS

We trained the GNN-based anomaly detection model for multivariate time series data, and analyzed the test results in several ways, which helps us to better understand the GNN model and time-series data analyses. In the future, we will apply this model to a larger dataset in our use case. In [1], the authors used a fixed window size of 5 for all the experiments. Multiple window sizes can be fused, where the larger window size can be downsampled with the max-pooling operation.

References

[1] Deng, Ailin, and Bryan Hooi. "Graph neural network-based anomaly detection in multivariate time series." *Proceedings of the AAAI conference on artificial intelligence*. Vol. 35. No. 5. 2021.
https://github.com/d-ailin/GDN

[2] Hundman, Kyle, et al. "Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding." *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 2018.
https://github.com/khundman/telemanom/tree/master

Our repo github:
https://github.com/namvux1404/Anomalies-detection---Vector-workshop/tree/main