Real-Time Anomaly Detection from Edge to HPC-Cloud

Judy Qiu[1], Bo Peng[1], Ravi Teja[2], Sahil Tyagi[1], Chathura Widanage[1], Jon Koskey[3]

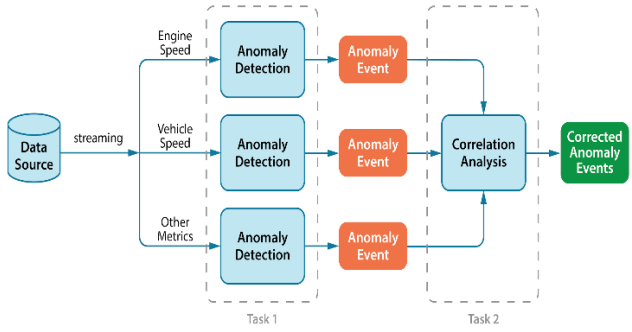
[1]Indiana University
[2]India Institute of Technology
[3]IndyCar

Abstract

Telemetry data plays an important role in many areas such as motor racing, meteorology, agriculture, transportation, manufacturing processes and energy monitoring to name a few. There lies a direct intersection between embedded computing and big data analysis. In the domain of motor racing, telemetry data is very widely used for analyzing or improving the performance of the racing cars and monitoring the effects of racing towards the physical status of the race car drivers. A large number of sensors, in the number of 100s, are fixed on the racing car. There are on-board and off-board electronic systems that transmit the sensor readings to teams on the pit and electromechanical systems that control the fuel utilization, throttling, etc. The generated sensor readings are then analyzed using several big data analysis methods, with detecting anomaly events at edge and conducting correlation analysis for high dimensional data on HPC and Cloud.

Keywords Time Series, Anomaly Detection, Online AI/ML

Challenges and Impact of Anomaly Detection on Time Series Datasets



Anomaly detection is a heavily studied area of data science and machine learning. It refers to the problem of finding patterns in data that do not conform to expected behavior [1]. Detection of anomalies, especially temporally in real-time streaming data, has significant importance to a wide variety of application domains, as it can give actionable information in critical scenarios. In this streaming application, data are observed sequentially, and the processing must be done in an online fashion, i.e., the algorithm cannot rely on any look-ahead procedures.

Traditional time-series modeling and forecasting models can be utilized to detect temporal anomalies. Approaches based on ARIMA are capable and effective for data seasonal patterns [2]. Techniques based on relative entropy, graph [3, 4] are also utilized to detect temporal anomalies. Another mainstream approach is to build simulation model with explicit domain knowledge for domain-specific applications. However, model-based approaches are limited for lack of generalizability. A novel approach was proposed in [5] to use Hierarchical Temporal Memory (HTM) [6, 7] networks to robustly detect anomalies on real-time data streams. HTM is a state-of-the-art online machine learning technology that aims to capture the structural and algorithmic properties of the neocortex. Figure 1 outlines the steps to create a complete anomaly detection system. The input time series x_t are fed to the HTM component. It models temporal patterns in $a(x_t)$ and output a prediction in $\pi(x_t)$. Then by building a statistical model on the prediction error, $\pi(x_t)$ - $a(x_{t-1})$, anomaly likelihood score can be calculated on x_t.

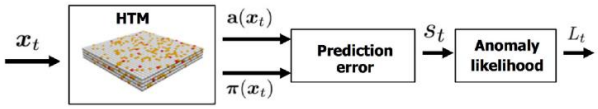
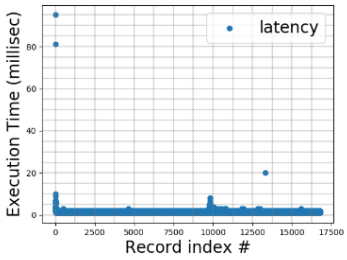


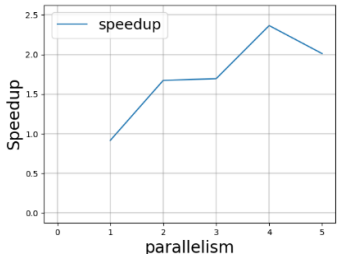
Figure 1 Anomaly Detection System Based on HTM

Convergence of HPC, Big Data and Machine Learning

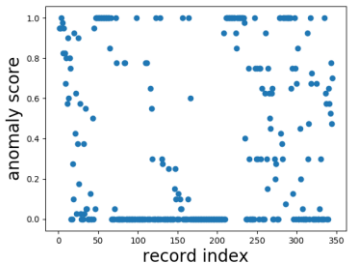
We apply anomaly detection algorithms on the dataset of Indycar race held on May 28, 2017. The raw log file contains around 670 megabytes of data, amounting to roughly 368,000 individual records logged throughout the race from 33 racing cars with an average speed between 200 and 250 miles per hour. We run HTM detection at various parallelism on Apache Storm and measure the speedup with respect to the batch HTM job. We also match batch HTM with Storm on a parallelism of 1 to determine the overhead incurred in a distributed framework. The batch and stream experiments run on a single Linux node with 48 CPU cores of Intel(R) Xeon(R) CPU E5-2670 v3 @ 2.30GHz. The total available memory is 125 gigabytes. The minimum Java heap size (-Xmn) set for the batch mode is 16GB.



(a) Execution latency in batch mode



(b) Speedup in HTM detection using Apache Storm



(c) Anomaly Scores of 345 injected anomalous datapoints

Figure 2 Performance and Validation of HTM over Streaming Dataset

The experiments run on the eRP (enahnced results protocol) data for car #9. This is due to the car (by driver Scott Dixon) suffered a car crash during the race and provides ample potential anomalies for us to detect. For the sake of simplicity, we predict anomalies based on the two parameters: ***time_of_day*** and ***vehicle_speed*** from Dixon's car, which has a dataset containing around 17,263 records. Fig 2(a) shows the execution latency (time taken to predict anomaly on a single input record) to process each record of car #9 on Apache Storm framework with parallelism of 1. An average execution time to predict an anomaly is 1.43 milliseconds. Fig 2(b) shows the speedup, the ratio of time to predict anomalies on a serial process to the time taken by running Storm at various parallelism levels. To establish a sense of ground truth on labels and validate our approach, we deliberately inject anomalies at 2% data fraction (~ 345 data points) at known indexes. We inject speeds of 0.00 mph (absolute vehicle halt) at various indexes. Anomaly score of 0.0 implies a normal event. The anomaly scores of each of the 345 data points is shown in Fig 2(c).

Conclusion

It is an important research topic on the convergence of HPC and Big Data to help people select appropriate computing hardware and software architectures based on the characteristics of different AI algorithms and applications. HTM is a special type of neural network on sparse data representation and operations for anomaly detection. HPC is needed to harness high-speed data with increasing complexity of predictive analytics, e.g. from all 33 cars, all 150 car sensors, all 10 timing sensors laid on the track with 1/10 millisecond accuracy, and 36 cameras streaming to 6 video feed servers, plus the possibility of sophisticated anomalies in issues like the way drivers take curves and strategies to deliver an exciting data-driven experience.

References

[1] V. Chandola, A. Banerjee, and V. Kumar. Anomaly Detection: A Survey. ACM Comput. Surv., 41(3):15:1–15:58, July 2009.

[2] A. M. Bianco, M. G. Ben, E. J. Martnez, and V. J. Yohai. Outlier Detection in Regression Models with ARIMA Errors using Robust Estimates. Journal of Forecasting, 20(8):565–579.

[3] L. Akoglu, H. Tong, and D. Koutra. Graph based anomaly detection and description: a survey. Data Mining and Knowledge Discovery, 29(3):626–688, 2015.

[4] S. Guha, N. Mishra, G. Roy, and O. Schrijvers. Robust random cut forest-based anomaly detection on streams. In International Conference on Machine Learning, pages 2712–2721, 2016.

[5] S. Ahmad, A. Lavin, S. Purdy, and Z. Agha. Unsupervised real-time anomaly detection for streaming data. Neurocomputing, 262:134–147, Nov. 2017.

[6] J. Hawkins and S. Ahmad. Why neurons have thousands of synapses, a theory of sequence memory in neocortex. Frontiers in neural circuits, 10:23, 2016.

[7] Y. Cui, S. Ahmad, and J. Hawkins. Continuous online sequence learning with an unsupervised neural network model. Neural computation, 28(11):2474–2504, 2016.