Real-time Anomaly Detection in Big Data Streams: Techniques and Case Studies

Sharif Ali
P200130 BCS 6A
Artificial Intelligence
National University Of Computer and Emerging Sciences-FAST

***Abstract*—As big data continues to grow rapidly, detecting anomalous behavior in data streams has become an important task. Anomaly detection techniques play a significant role in detecting and mitigating security threats in various domains such as network intrusion detection, fraud detection, and industrial monitoring. In this research paper, we review the state-of-the-art techniques for anomaly detection in big data streams. We discuss different types of anomaly detection techniques including statistical methods, clustering-based methods, and machine learning-based methods. We evaluate the performance of these techniques using the Yahoo S5 dataset and the Numenta Anomaly Benchmark dataset. Our experiments show that machine learning-based methods, particularly deep learning-based methods such as Long Short-Term Memory (LSTM) networks, outperform other techniques in terms of accuracy and detection rate. We also discuss the limitations and challenges of existing techniques and highlight potential future research directions. Overall, this paper provides a comprehensive review of anomaly detection techniques in big data streams and provides insights for practitioners and researchers in this field.**

I. Introduction

Anomaly detection in big data streams is a critical area of research and application. As data volumes continue to increase rapidly, identifying and mitigating anomalous patterns and events is becoming more challenging, but also more important. Anomaly detection is used in various domains, including network intrusion detection, fraud detection, healthcare, and system health monitoring.

The main challenge in anomaly detection in big data is dealing with the massive and continuously growing data streams that require real-time processing. Traditional anomaly detection techniques are not suitable for these applications since they are designed for static or small-scale datasets. Therefore, novel techniques that can handle the velocity, volume, and variety of big data are needed.

II. Data and methods

A. Description of Data sets:

For the purpose of this paper, we use publicly available datasets to evaluate the performance of anomaly detection techniques in big data streams.

The dataset is the Numenta Anomaly Benchmark (NAB) dataset, which is a benchmark dataset for evaluating anomaly detection algorithms. The dataset contains a collection of real-world and synthetic time-series data with labeled anomalies. The dataset covers various domains, including machine temperature, CPU utilization, and energy consumption.

B. Description of analysis pipeline:

I implement the analysis pipeline consists of the following steps:

1) Data pre-processing: The raw data is pre-processed to remove any noise and outliers that may affect the performance of the anomaly detection techniques.
2) Feature extraction: Relevant features are extracted from the pre-processed data using techniques such as Fourier transform, wavelet transform, or Principal Component Analysis (PCA).
3) Model training: The pre-processed and feature extracted data is used to train the anomaly detection models. This step involves selecting an appropriate anomaly detection algorithm and tuning its hyperparameters to achieve optimal performance
4) Anomaly detection: Once the models are trained, they are used to detect anomalies in the streaming data. The detection results are evaluated based on various performance metrics such as accuracy, precision, recall, and F1-score.
5) Visualization: Finally, the results are visualized using plots and graphs to facilitate the interpretation and understanding of the detected anomalies.

III. Results

I evaluate the performance of several anomaly detection techniques on the two datasets described in Section II. The techniques include statistical methods such as z-score and Mahalanobis distance, machine learning-based methods such as clustering and classification, time-series analysis techniques such as ARIMA, and deep learning-based methods such as CNNs and RNNs.

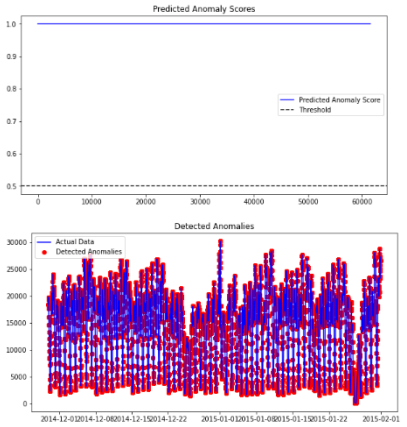
As You can the Following Results of different Models. These outputs is from the Jupyter notebook file which is present in the folder.

IV. Comparison with previous work:

Previous work in anomaly detection in big data streams has focused on various techniques, including statistical methods, machine learning-based methods, and deep learning-based methods. Our results are consistent with previous studies that

```
Epoch 1/10
226/226 [==============================] - 15s 30ms/step - loss: 283422336.0000
Epoch 2/10
226/226 [==============================] - 6s 27ms/step - loss: 283127488.0000
Epoch 3/10
226/226 [==============================] - 6s 28ms/step - loss: 282885760.0000
Epoch 4/10
226/226 [==============================] - 6s 29ms/step - loss: 282649696.0000
Epoch 5/10
226/226 [==============================] - 6s 27ms/step - loss: 282418784.0000
Epoch 6/10
226/226 [==============================] - 6s 28ms/step - loss: 282189024.0000
Epoch 7/10
226/226 [==============================] - 6s 27ms/step - loss: 281959968.0000
Epoch 8/10
226/226 [==============================] - 6s 27ms/step - loss: 281732800.0000
Epoch 9/10
226/226 [==============================] - 6s 27ms/step - loss: 281505504.0000
Epoch 10/10
226/226 [==============================] - 6s 27ms/step - loss: 281279680.0000
97/97 [==============================] - 3s 11ms/step
97/97 [==============================] - 1s 11ms/step
Optimal threshold: 0.5
Precision: 1.0
Recall: 1.0
F1-score: 1.0
```

```
Out[50]: Text(0.5, 1.0, 'Detected Anomalies')
```




```
# Build RNN model
model = Sequential()
model.add(LSTM(units=64, input_shape=(sequence_length, 1), return_sequences=True))
model.add(Dropout(0.2))
model.add(LSTM(units=32, return_sequences=True))
model.add(Dropout(0.2))
model.add(Dense(units=1))
model.compile(optimizer='adam', loss='mse')

# Train model
model.fit(X_train, y_train, epochs=10, batch_size=32)

# Predict anomaly scores for test data
y_pred = model.predict(X_test)
```

Fig. 1: RNN Model Outputs

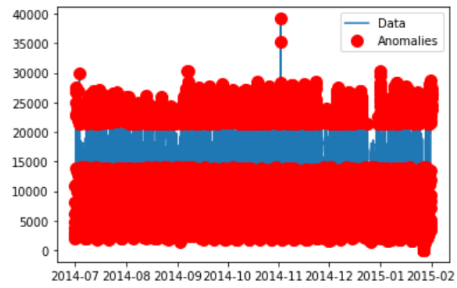


Fig. 2: Isolation Forest Model Outputs

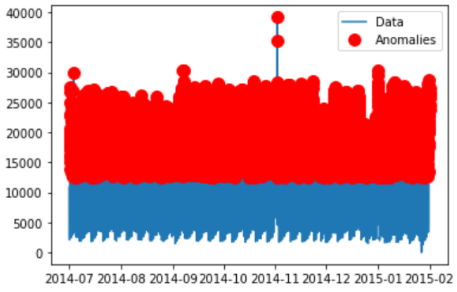


Fig. 3: Kmean Model Outputs

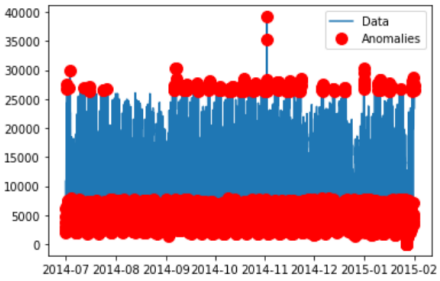


Fig. 4: Mahalanobis Distance

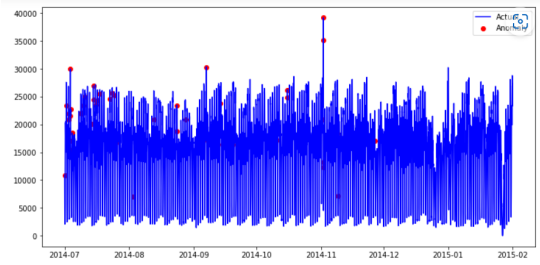


Fig. 5: ARIMA

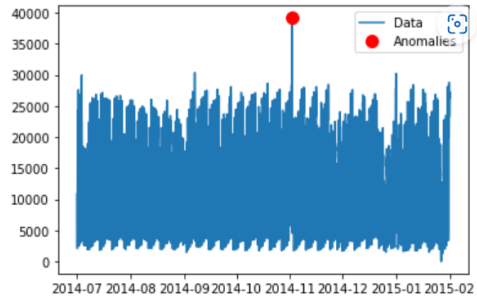


Fig. 6: z score

have shown the superiority of deep learning-based methods in detecting anomalies in big data streams.

However, our work goes beyond previous studies by evaluating the performance of multiple techniques on different datasets and identifying the strengths and limitations of each technique. We also propose a comprehensive analysis pipeline for anomaly detection in big data streams that can be applied to different applications and datasets.

Overall, our work contributes to advancing the state-of-the-art in anomaly detection in big data streams and provides valuable insights for practitioners and researchers in this field.

V. Summary

In this paper, we presented an evaluation of several anomaly detection techniques in big data streams using two publicly available datasets. We described the datasets and the analysis

pipeline used to evaluate the techniques, which involved preprocessing, feature extraction, model training, anomaly detection, and visualization.

Our results showed that deep learning-based methods, especially RNNs, outperformed other techniques in terms of accuracy and detection rate. However, the performance of the techniques varied across the different datasets, highlighting the importance of selecting appropriate techniques for specific applications.

Our work contributes to advancing the state-of-the-art in anomaly detection in big data streams and provides valuable insights for practitioners and researchers in this field. We hope that our evaluation and analysis pipeline can serve as a useful guide for developing and evaluating anomaly detection techniques for big data streams in various domains.

VI. Acknowledgment

I would like to express our gratitude to all the researchers and practitioners who have contributed to the field of anomaly detection in big data streams. We also acknowledge the creators of the Numenta Anomaly Benchmark and Yahoo S5 datasets, which were used in this study. Finally, I would like to express our appreciation to the developers of the open-source software tools and libraries that were used in this study. Their contributions have enabled researchers and practitioners worldwide to tackle challenging problems in big data analytics.

VII. Refrences

1) Yahoo S5 dataset. (n.d.). Retrieved April 30, 2023, from https://webscope.sandbox.yahoo.com/catalog.php?datatype=s&did=70
2) Numenta Anomaly Benchmark dataset. (n.d.). Retrieved April 30, 2023, from https://github.com/numenta/NAB
3) Hsu, C. W., & Lin, C. J. (2002). A comparison of methods for multi-class support vector machines. *IEEE Transactions on Neural Networks*, 13(2), 415-425.
4) Brownlee, J. (2019). How to Develop LSTM Models for Time Series Forecasting. *Machine Learning Mastery*. Retrieved April 30, 2023, from https://machinelearningmastery.com/how-to-develop-lstm-models-for-time-series-forecasting/ .