Evaluating Statistical Models for Network Traffic Anomaly Detection

Peter Kromkowski[1¶], Shaoran Li[1¶], Wenxi Zhao[1¶], Brendan Abraham[2*], Austin Osborne[3*], Donald E. Brown[4*]
pk9t, sl4bz, wz8nx, bea3ch@virginia.edu, austin.osborne@capitalone.com, brown@virginia.edu

Abstract - **Large organizations may have hundreds or thousands of applications running simultaneously to support their operations. To maintain high levels of efficiency, they need to quickly detect outages or anomalies in order to quickly fix the problem and reduce costs. This paper describes the analytical framework for a network traffic data anomaly-detection method to reduce application downtime and the need for human involvement in detecting or reporting anomalous application behavior. We use the described framework to compare the performances of a Seasonal Autoregressive Integrated Moving Average (SARIMA) times series model and Long Short-Term Memory (LSTM) Autoencoder model at anomaly detection. We evaluated these models using false positive rates and accuracy, with a requirement of being able to give timely alerts, and saw that even though both models were accurate, their false positive rates were very high. We then improved overall detection performance by ensembling the SARIMA and LSTM autoencoder. Our results demonstrate a possible new method of anomaly detection in network traffic flow using time series and autoencoders.**

Index Terms - Anomaly Detection, Autoencoders, Network Traffic Flow, SARIMA, Time Series

Introduction

When network or applications unexpectedly fail or crash, there is a direct and far reaching impact on the downstream line and ongoing business operation. Furthermore, these anomalies are very difficult to detect and outages could result in significant financial loss. The average cost of network downtime is $100,000 per hour [1], a number expected to grow as more people become dependent on applications.

Capital One Financial Corporation is a bank holding company specializing in credit cards, auto loans, banking and savings products. Today, millions of its customers use its applications to pay their bills, keep track of their expenses, and countless other time-sensitive uses. Additionally, internal Capital One company applications are used by all of its employees to carry out necessary tasks throughout the business day. Capital One has taken many steps to make sure these applications are reliable, one of those being by their Center for Machine Learning.

Capital One's Center for Machine Learning (C4ML) currently implements many models that monitor the network traffic flow of their applications. However, these models produce a lot of false positives and noise. In partnership with the C4ML team, this paper aims to produce a model to detect anomalous activity on a Capital One application while also achieving a low false positive rate. This will enable Capital One engineers to better react to application outages, reduce the impact of application downtimes, and decrease the cost of needing to monitor these systems.

This paper uses a SARIMA model, an extension of the popular ARIMA model, that incorporates seasonal components into its time series predictions. This paper also focuses on the power of autoencoders in replicating normal network traffic flow. With successful replication of Capital One network traffic flows, the autoencoder is able to predict future traffic flow patterns and throw an alert when the actual traffic flow greatly differs from the autoencoder predictions. Lastly, this paper combines these two models in an ensemble approach to confidently detect anomalies in network traffic flow.

These methods are implemented and evaluated on a anomaly-detection framework that first monitors prediction residuals and then builds a Gaussian distribution on the residual errors, labeling predicted anomalies as the residuals that fail within the tails of the distribution.

By comparing the false positive rate and prediction accuracy for all models and comparing the influence of transaction and bytes to anomalies, the ensemble method is the suggested candidate for anomaly detection.

Related Work

I. Time Series Analysis

Anomaly detection has been an active research area in the fields of statistics and machine learning. When conducting anomaly detection, learning both normal and abnormal behaviour of the data is essential to understand the significance of each anomaly. As Toledano et al. [2] suggested, temporal methods such as Holt-Winters, classical ARIMA and seasonal ARIMA models can be used to study the temporal dynamics of normal behaviour. Several

978-1-7281-0998-5/19/$31.00©2019 IEEE

challenges are presented along with exploiting the time series algorithm, such as the robustness of the model to the anomalies in order to prevent training on unhealthy data, adjusting for seasonality including multiple patterns of seasonality in the data which requires the selected model to be sensitive to changing dynamics in the data. This issue is addressed in our methodology by replacing unhealthy data with healthy prediction data and adjusting the underlying parameters of the time series model automatically with a moving sliding window in order to adapt to various patterns of seasonality.

II. Sliding Window Algorithm

Sliding Window is a temporary approximation over the actual value of the time series data. H.S. Hota [3] illustrates that a sliding window algorithm is used to segment the time series data in order to minimize error approximation and to provide a robust forecast on time series data. As Figure I demonstrates, the initial window size of 5 historical data observations is used to make a prediction for the next day. The window slides after the first prediction and the observations 2-6 are used to predict day 7. The process continues until all necessary time frames are predicted or resources are exhausted. In this paper, a sliding window is combined with a time series model to forecast real-time data and auto-adjust parameters to make more accurate and timely predictions.

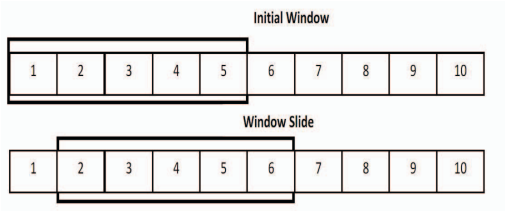

FIGURE I. Process of Sliding Window

III. Autoencoders

Long Short-Term Memory (LSTM) is an artificial recurrent neural network architecture. LSTM units include a cell that can maintain information in memory for long periods of time. It does this with a set of gates used to control when information enters the memory, when it's output, and when it's forgotten. One input gate unit is used to protect memory contents from irrelevant input and one output gate is used to protect other units from currently irrelevant stored memory. [4] However, LSTM recurrent neural networks have not always proved to be a reliable time series forecasting method. This is most likely due to their inability to understand the structure among every point in a sequence, instead trying to continuously move the model forward.

An autoencoder neural network is a unsupervised learning algorithm that tries to replicate its inputs, learning a function that can closely match, without overfitting, the identity function of the data [5]. Autoencoders help reduce noise while also decoding and constructing the complex and turbulent relationships in a signal. To model the uncertainty contained in network traffic flows, the auto-encoder is trained on a noisy dataset using unsupervised learning strategies. Then, the autoencoder is trained by a back-propagation algorithm using supervised learning strategies to capture the complex relationships over the network traffic flows. An LSTM autoencoder enhances this learning strategy for time series data by learning the importance of sequential data.

Maxim Wolpher [6] showed that an LSTM autoencoder can have a significant advantage in replicating normal network traffic flow based off of a structure trained on known non-anomalous (healthy) traffic flow. Wolpher also tested the LSTM autoencoder on anomalous malicious traffic flow and showed that the model's replication error was significantly higher than its replication error of normal network traffic.

This type of test can be used to detect future anomalous periods based of off high replication errors and is the main style of implementation of an LSTM autoencoder in this paper.

METHODOLOGY

I. Data Description

Two data sources hosted over Amazon Web Services were provided for this paper. The first, consisted of network traffic flow logs from a specific application used internally at Capital One. Due to complexities and size, the data used in this paper spans from January 15th 3:15pm to January 26th 3:30pm, with logs aggregated on fifteen minute intervals. Additionally, only the number of bytes and transactions (a request sent by a user from one IP address) per timestamp were available for this paper. The number of transactions range from 275 to 75,434 and the volume of bytes range from 164 to 4.51 billion (Figure II). Limited feature engineering was done due to the irregularity and complexity of the network traffic flow and to reduce the possibility that a feature could be affected by an unknown anomalous period.

The second datasource used in this paper provided anomalous periods for the application. These anomalous periods were represented by a rise in failed queries to the application. This dataset provided two periods that overlapped with our network traffic flow which were labeled as anomalous: Jan 23rd 6:30pm to Jan 23rd 10pm and January 24th 2pm to January 24th 9:30pm.

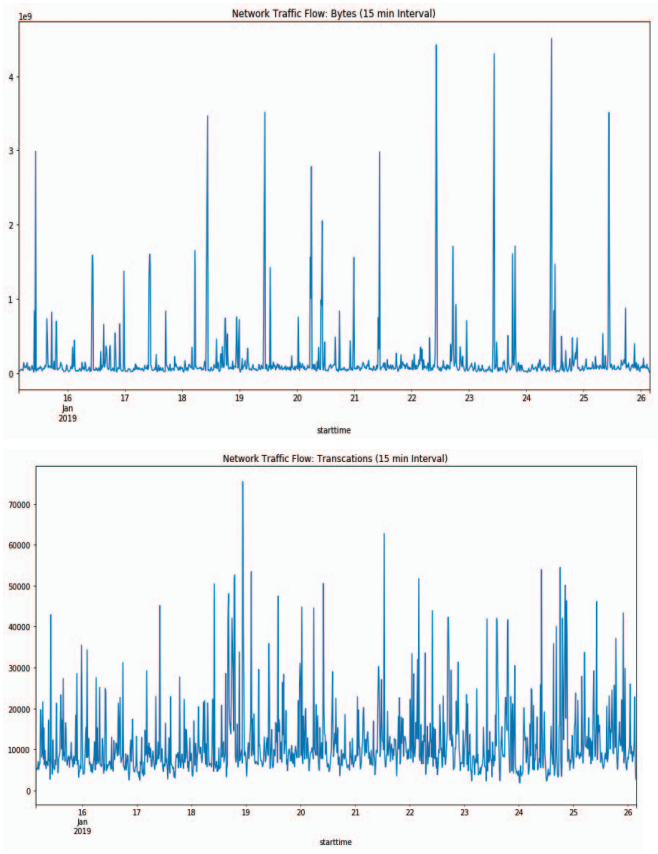

FIGURE II. Network Traffic Flow of Bytes and Transactions (15 minutes interval)

MODELING TECHNIQUES

We used two different time series techniques to predict network bytes and transaction data and measured their performance. These models were chosen since they were thought to best understand the complexity and noise of the network traffic flow. Additionally, they are both useful and widely used techniques for detecting anomalies. Two datasets were created for both the bytes and transactions and each dataset was independently used on both models. The datasets were split into a training set which contained the first week of known non-anomalous traffic, while the dates Jan 22th-25th were used in the testing set. The models were tested using the test dataset and then post-test labeled with the known anomalous periods. The residuals of the test data prediction at non-anomalous and anomalous timestamps were used to arrive at the performance metrics for model comparison.

I. Time Series -- SARIMA Model

SARIMA, or Seasonal ARIMA, is an extension of ARIMA modeling, taking into account the seasonal component of the univariate time series data. The SARIMA model contains three hyperparameters to specify autoregression, difference order, and moving average for the seasonal component of time series data. It also includes additional parameters for seasonality such as seasonal autoregressive, seasonal difference order, seasonal moving average, and the number of time steps for a single seasonal period. To find out the best model, grid search is used over all hyperparameters to test all combinations, calculating their respective Akaike Information Criteria (AIC), which provides a means of model selection; the lower the AIC value, the better the model.

Since a goal for this paper is to have the anomaly detector send alerts in a timely fashion, the SARIMA model needs to keep the prediction as close as possible to the previous trend. In order to achieve this goal, the sliding window algorithm is implemented in the SARIMA model to train data on a preceding period and then making predictions on the current period.

The first step is to make several sliding windows with one week size and each time the sliding window moves one day forward in order to predict the next day. In the SARIMA model, each window is used as a training set and one day of prediction data points is used as a test set. Since the data ranges over eleven days, the one week sliding window algorithm moves four times in total, making predictions on Jan 22nd, 23rd, 24th, and 25th.

Because anomalies can happen across days that may eventually be included in the sliding window training set, the model must be able to avoid being influenced by these unusual traffic patterns. This is done by replacing the test data, labeled as anomalous by the model, with its prediction data when it is incorporated in the sliding window to make future predictions. Because the prediction data is always based off of non-anomalous data it serves as a proxy for what should be healthy data during anomalous periods.

II. LSTM Autoencoder

Individual sequences of data were created for each datapoint in the training and test set that included the datapoint and six hours worth of prior timestamps, to give an array of 25 timestamps. This ultimately created a very large, complex, and overlapping state space. A stateful LSTM autoencoder model was used so the cell states remained persistent while the model was carried forward across the dataset. The stateful LSTM autoencoder parameters, such as number of layers and hidden units, were modified until the training loss was less than one percent. Ultimately, the autoencoder relied on one 100 unit hidden layer with another time-distributed dense layer.

EVALUATION

Both models were evaluated on both bytes and transaction datasets. However, analysis showed that both models evaluated on the transaction data were better suited for anomaly detection than their bytes counterparts. Therefore,

the analysis for anomaly detection using the transaction dataset is highlighted in this paper.

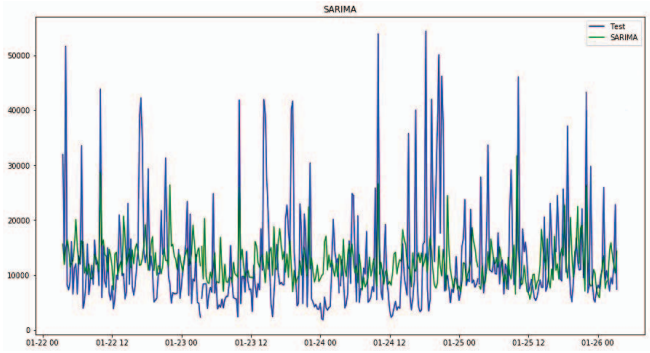

Figure III. Combined SARIMA predictions on the test transaction data

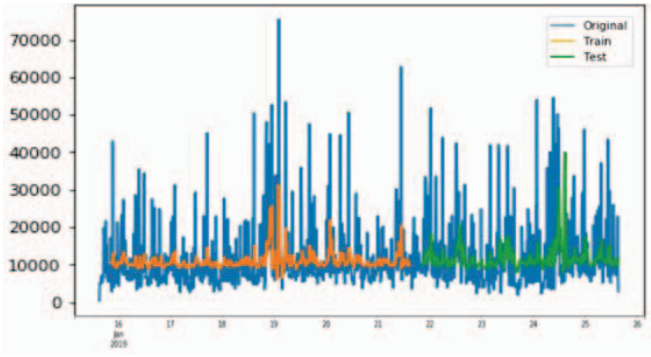

Figure IV. LSTM Autoencoder replication of the test transaction data based on the training replication

Figure III shows the combined SARIMA predictions of every sliding window for the transaction data test set. Figure IV displays the autoencoders interpretation and replication of the transaction testing set based off of the structure of the training set. Both models discovered the seasonality and trends of the underlying movement of the data.

The models were evaluated using a distribution method on both the bytes and transaction datasets. For each model, all of the residual errors of the testing set before the first anomaly period (Jan 22nd - Jan 23rd 6:30pm), which is a period that only contains healthy data, were scaled and used to build a standard normal Gaussian distribution (Figure V).

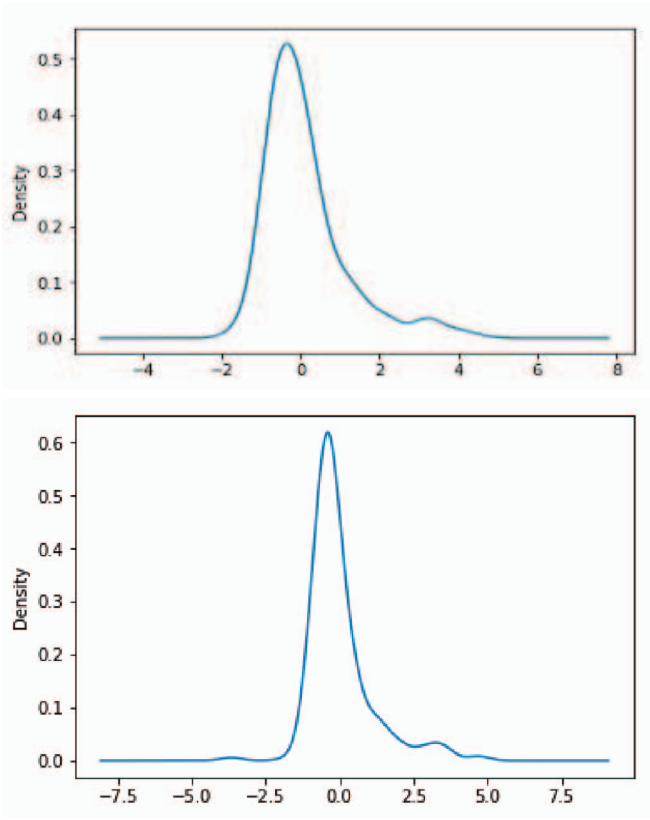

FIGURE V. Gaussian plots for SARIMA (top) and Autoencoder (bottom). Both are the distributions for the transaction residuals.

These distributions validify the assumption of standard normal distribution of residuals during a non-anomalous period. However, both plots are slightly skewed to the right which indicates that the mean is greater than the median and the distribution may contain outliers.

After the distributions were created the rest of the testing residuals were fit into the distribution. Residuals were labeled as anomalous if they exceeded a certain standard deviation away from the mean.

Detecting anomalous behaviour in network traffic flow requires accurate labeling of the data in real time. Additionally, for Capital One, a conservative model with little noise and low false positive rates is essential for implementation. Models with a high false positive rate can cause alert fatigue and expensive monitoring costs. Therefore, though accuracy and true positive rates are take into consideration during model evaluation, a useful model with a low false positive rate must be emphasized.

The appropriate standard deviation for each model was chosen with this in mind by using a ROC curve to analyze the tradeoff between specificity and sensitivity of the known non-anomalous and anomalous periods. With low false positive rates given priority, a standard deviation of 2.2 was chosen for the SARIMA model and a standard deviation of 2.8 was chosen for the autoencoder (Figure VI).

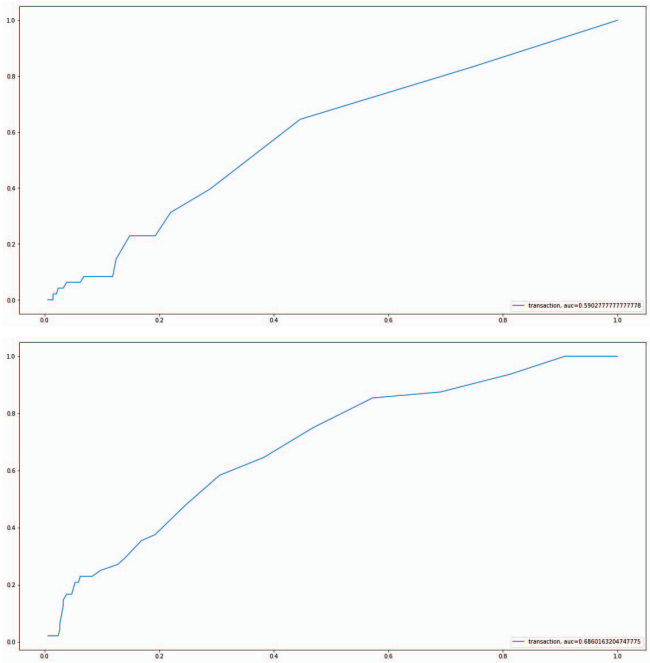

FIGURE VI. ROC curves for Transaction SARIMA (Top) and LSTM Autoencoder (Bottom)

RESULTS

The resulting detection rates from the SARIMA model and LSTM Autoencoder are shown in Table 1. In both models the transaction dataset was better suited for modeling normal traffic behavior, and thus detect anomalies based on prediction error. The true positive rates for the transaction based models are significantly higher than those seen in the bytes based models . However, every model has a very high false positive rate relative to the how much this application is used on a daily basis. Such a high false positive rate would render a model useless for real life implementation.

TABLE I

	SARIMA		**Autoencoder**	
	Bytes	Transactions	Bytes	Transactions
True Positive Rate	4.17%	25.00%	2.07%	14.58%
False Positive Rate	2.97%	3.86%	2.08%	3.26%

III. Ensemble Model

This papers final approach is an ensemble model that combines the predictions of both the SARIMA and Autoencoder model in order to reduce the false positive rate. The ensemble model classifies a data point as an anomaly if both models do. As shown in Table II, this ensemble method reduces the the false positive rate of the autoencoder by nearly 50%

TABLE II

	Ensemble Model	
	Bytes	Transactions
True Positive Rate	2.08%	14.58%
False Positive Rate	1.48%	1.78%

CONCLUSION AND FUTURE WORK

This paper set out to create a model that could ingest real time network traffic flow and detect if it was anomalous. Based off of transaction data for network traffic flow time stamps, this paper was able to create an ensemble model, comprised of a SARIMA model and an autoencoder that had a true positive rate of 14.58% and a false positive rate of 1.78%. Implemented with other models, this type of model could prove useful in confident anomalous behavior detection.

Most importantly, this paper supported the usefulness of LSTM autoencoders and SARIMA models in network traffic flow prediction. It also provided a methodology of describing healthy network traffic as a Gaussian distribution with anomalous network traffic flow falling outside of a certain standard deviation.

However, this paper encountered many obstacles, such as limited features and very specific and scarce data that limit the significance of the particular findings of this paper. Future work should expand on the concepts and methodologies outlined in this paper. More data should be analyzed to learn the true distribution of healthy data and where anomalous activity falls inside of that distribution for each type of model. Additional future work also includes improving class imbalance and testing other distributions.

References

1. Andrus, Kolton. “Why CTOs And CIOs Should Care More About The Cost Of Downtime.” Forbes, Forbes Magazine, 26 Apr. 2018, www.forbes.com/sites/forbestechcouncil/2018/04/26/why-ctos-and-cios-should-care-more-about-the-cost-of-downtime/#9d3bde9131c1.
2. Toledano, Meir, Cohen, Ira et al. “Real-Time Anomaly Detection System for Time Series at Scale.” PMLR,
3. Hota, H.S, et al. “Time Series Data Prediction Using Sliding Window Based RBF Neural Network .” International Journal of Computational Intelligence Research , vol. 13, no. 5, 2017.
4. Namin, SIMA SIAMI, and AKBAR SIAMI Namin. “Forecasting Economics and Financial Time Series: ARIMA vs. LSTM.” *ArXiv.org*, 16 Mar. 2018, arxiv.org/abs/1803.06386.
5. Unsupervised Feature Learning and Deep Learning Tutorial, Stanford, ufldl.stanford.edu/tutorial/unsupervised/Autoencoders/.
6. Wolpher, Maxim. “Anomaly Detection in Unstructured Time Series Data Using an LSTM Autoencoder.” KTH ROYAL INSTITUTE OF TECHNOLOGY SCHOOL OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE, vol. 2, 2018.

Author Information

[1] **Peter Kromkowski,** M.S. Data Science, Data Science Institute, University of Virginia.

[1] **Shaoran Li,** M.S. Data Science, Data Science Institute, University of Virginia.

[1] **Wenxi Zhao** M.S. Data Science, Data Science Institute, University of Virginia.

[2] **Brendan Abraham,** Co-Advisor, School of Systems and Information Engineering, University of Virginia.

[3] **Austin Osborne,** Co-Sponsor, Center For Machine Learning, Capital One.

[4] **Donald E. Brown,** Co-Sponsor, School of Systems and Information Engineering, University of Virginia.

¶ These authors contributed equally to this work.

* Corresponding author