# Scaling DAA: Smart, Continuous Network Health Monitoring for vCMTS with Machine Learning

A Technical Paper prepared for SCTE by

**Ilana Weinstein**
Data Scientist
Comcast
1800 Arch St, Philadelphia, PA
ilana_weinstein@comcast.com

**Ramya Narayanaswamy**
Director, Data Science
Comcast
1800 Arch St, Philadelphia, PA
ramya_narayanaswamy@comcast.com

**Aaron Tomkins**
Data Scientist
Comcast
1800 Arch St, Philadelphia, PA
aaron_tomkins@comcast.com

**Nivedhitha Sridhar**
Data Scientist
Comcast
1800 Arch St, Philadelphia, PA
nivedhitha_sridhar@comcast.com

# Table of Contents

**Title** **Page Number**

## List of Figures

**Title** **Page Number**

## List of Tables

**Title** **Page Number**

## 1. Introduction

On Comcast's road to 10G, the virtual cable modem termination system (vCMTS) platform plays an integral part in ensuring the latest Data Over Cable Service Interface Specification (DOCSIS) technology is available to offer higher speeds and the best service to customers. The vCMTS platform is expanding rapidly, which enables enhanced flexibility, scalability, and cost-effectiveness. Therefore, ensuring the stability of this platform is critical, given its significance. However, the introduction of software upgrades often poses challenges for network operators, as it requires meticulous monitoring to detect and address any potential anomalies that may arise in the post-upgrade phase. To overcome these challenges, several systems have been put in place, such as the Automated Network Health Checks. These systems monitor the network's health immediately after deployments or software upgrades, and tools are available for alerting based on a set of key performance indicators (KPIs). While these instantaneous checks are valuable, certain KPIs and telemetry metrics may take hours to days to reveal underlying service degradation. At present, there is limited visibility into detecting these anomalous trends on Distributed Access Architecture (DAA), which could indicate a slow degradation of the platform's health. This paper proposes a comprehensive solution that leverages data science, machine learning, and big data techniques to continuously monitor and detect anomalous trends that could indicate a slow deterioration of platform health and then alert the operations team of any unusual activity.

## 2. Methodology

### 2.1. Anomaly Detection and Data Science

Identifying anomalies in data is an important task in many fields, and a variety of approaches have been developed over time to try to solve this problem. Understanding these various approaches to anomaly detection is crucial in determining the most suitable solution for the given problem. This section will briefly cover these approaches, their advantages in and out of the long-term network monitoring domain and background that contributes to the design of the anomaly detection system. Although different survey papers have categorized various machine learning approaches for anomaly detection in different ways, some of the more common approaches have been categorized as: classification-based, clustering-based, nearest-neighbor based, knowledge-based, statistical, and deep learning-based (Bhuyan et al. 2014 and Chandola et al. 2009).

The advantages and viability of a particular machine learning approach are often dependent on the problem context and format of the available data. Some problem contexts contain very high-dimensional data, while others are univariate; in some contexts, anomalies are clearly labeled in the training set, while in others they must be inferred. Classification-based approaches often work well when true anomaly labels and many features exist in the training data, while many statistical approaches can work well despite the absence of these. Furthermore, adjustments or modifications to an approach are often necessary to better fit the intricacies of a particular problem context.

In the context of detecting anomalous trends in some very different network metrics that don't have direct relationships with each other, there can be many benefits to utilizing a different approach that naturally goes with each metric. One benefit is it gives the freedom to customize each approach to that metric instead of limiting the options to modifications that only work in generality. However, as different as some metrics may be, they often can still benefit from and inform each other; thus, there is also an advantage in not keeping models siloed from each other. To that end, additional value can be achieved by building a higher-level aggregating model that is able to combine the insights from the metric-specific models and help produce an actionable summary of what is going on with the network. In this way, one can both get interpretable information at a granular, metric-specific level as well as leverage the relationships between metrics to get a higher-level summary of the network overall.

### 2.2. Data Overview

The DAA architecture offers rich telemetry, from before the headend to the device. The DAA telemetry is multidimensional; each metric monitored in the anomaly detection system is a timeseries with different properties, both in terms of frequency (ranging from 15-second telemetry to hourly polls to event-based data), the infrastructure level, and the sources of the data. See Eppes et al. 2022 for more details around the DAA infrastructure. The infrastructure levels in scope for anomaly detection encompass the cable modem (CM), remote physical device (RPD), physical pod (PPOD), etc.

The anomaly detection pipeline leverages an existing telemetry collection process (Stehman et al. 2021) that is built on top of cloud computing tools like Apache Spark, which facilitates the parallel processing needed to operate on data at this scale in an efficient manner on a distributed framework. The process collects the data from each source by polling the different data sources on an automated schedule; it then aggregates the telemetry data on the PPOD-level and the RPD-level into a standardized format for further analysis. The four major metrics that are in the current scope of this project are: 1. CM Registration States 2. Interactive Voice Response (IVR) 3. Quality of Experience (QoE) 4. Traffic/Switch Metrics.

The CM registration status data consists of 15 second polls, aggregated up to a 5-minute level indicating the registration state of the CM at each polling interval. It captures the state of the CM's attempt to pair with the CMTS, polls the CM at periodic intervals, and assigns a status based on the response and the DOCSIS specifications of the modem. The polls are aggregated such that the highest impact state in a 5-minute window is considered. This data tends to have a pattern to it – specifically, the nightly reboots that some device types undergo during the maintenance window. These would need to be accounted for as non-anomalous but would also need to factor into an algorithm such that, during that time, if more devices than expected go into an offline state or begin to waver, that would still be flagged as an anomaly.

Next metric is the IVR metric, which comprises of support calls that are made by customers. The IVR is a technology that allows a computer to interact with people who call using voice and keypad inputs. These systems are used in servicing high call volumes, reducing cost, improving the customer experience, and providing a self-service experience. The system uses a routing mechanism to determine where the call is directed based on the customer input. It also collects information on whether there is a current outage in the area and provides that information to callers. High call volumes or specific call categories can indicate regions that are experiencing increased dissatisfaction when comparing call volume before and after a deployment. By identifying these pain points, the operations teams can take measures to further investigate where the issues are or corroborate with other metrics to determine the need for a rollback.

The QoE measure is an aggregate score which considers a wide range of metrics to determine the device's performance. Largely Wi-Fi based, it considers network performance factors such as the data rates, signal strength, packet error rate, any noise or interference on the channel, channel utilization broadband throughput, and the topology of the network. It also factors in the device type, capabilities of the device, DOCSIS version, current and historical data usage, and other specific environmental factors.

The last major group of metrics is the traffic metrics. These metrics serve as counters, keeping track of the packets flowing between various components within the switch leaf architecture. Specifically, the focus of these metrics lies in monitoring unicast traffic, encompassing packets transmitted between the residential U-Ring router (RUR), leaf switches, vCMTS host, and other elements of the architecture.

When anomalies arise from this network connection or the RUR device, they indicate potential impairments or disruptions in the link between the RUR router and the leaf switch. Additionally, another pair of traffic metrics considered is the octet in/out counters, which quantify the total number of octets received or transmitted via the interface, including both header and frame characters. Discontinuities in the value of this counter can occur during management system re-initialization or other connection drops.

It is worth noting that traffic metrics exhibit distinct patterns over time. Generally, they demonstrate a diurnal cycle with peak values during the day and lower values during the night. Furthermore, traffic tends to increase during holidays and weekends. Hence, any effective time series anomaly detection model must possess the capability to comprehend these patterns, decompose the seasonal elements, establish a baseline, and subsequently identify data points that deviate from the expected norms. The use of these metrics will be further discussed in Section 4.2.

### 2.3. Solution Architecture

Analyzing near real-time telemetry on several KPIs for anomaly detection is a daunting task. To address this challenge, a sophisticated yet flexible workflow solution has been developed to apply an anomaly detection model to each KPI and trigger alerts to relevant stakeholders.

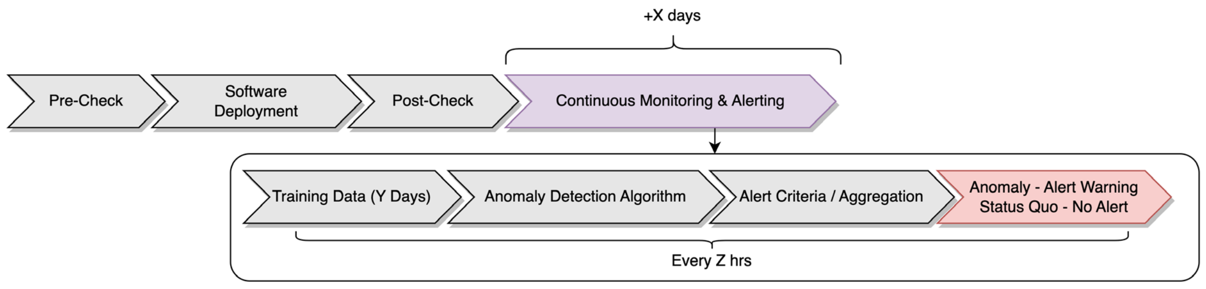

**Figure 1 – Software Deployment and Continuous Monitoring Integration**

The architecture uses a powerful analytics platform that has big data processing capabilities, machine learning tools. Aside from the platform's core data abilities, the anomaly detection solution is developed and deployed in this platform due to a few reasons. Firstly, it is on the same platform as the telemetry collection solution as mentioned in Section 2.2, providing continuity and centralization of tools. Second, it is designed for collaboration, making it easy to share and contribute to across the data sciences team. Third, the platform makes it easy to interact with the system, deep-dive and perform analyses on the models, as well troubleshoot updates. Lastly the platform has workflow and scheduling capabilities, which is essential in automating and scaling the anomaly detection system.

The workflow runs at set intervals and has the capability to extract, transform and load (ETL) data and run models at different times. The first step in the workflow is determining which KPIs are required for anomaly detection and which PPODs are of focus in that run. A PPOD is included in the workflow if it had a software deployment in the previous week. Each PPOD is monitored for an entirety of a week; see Figure 1 for a PPOD's network monitoring timeline. If the KPI is scheduled for anomaly detection, the telemetry data is ingested and prepared for anomaly detection. Next the respective anomaly detection model is applied; each KPI has an individual anomaly detection model, Section 2.4 will discuss these models in more detail. In order the keep the ML models from going stale, there is also automatic re-training based on criteria developed with the help of subject matter experts (SMEs); this includes the amount of time that has passed since the last training, if there is a new deployment to the PPOD, or if there is an extreme change in the historical data such as an RPD move, etc. If there are anomalies detected, each alert is assigned a universally unique identifier and prepared for alerting and reporting. The alert information as well as helpful metadata will go into a data storage system. An informative visualization of the anomaly, unique to each KPI, is also generated at the time of anomaly detection and saved to a storage container.

A single run can contain a handful of anomalies from multiple KPIs, so it is important that the alerts are presented in an organized manner and provide information that aids prioritization of alerts for investigation. The compiler step in the workflow handles the compiling of alerts and communicating them to stakeholders for validation; utilizing the unique alerts and information saved internally at the time of alert generation. There are two parts to the report, the first organizes the anomalies' visualizations into a document by PPOD and orders by KPI type. By presenting the anomalies in a clear and concise format, it allows easy sharing of information and enables a common understanding of the detected anomalies. The second part of the report is a spreadsheet that organizes the sheets by KPI/alert type and sorts by the percent of CMs that can be tied to the anomaly. The spreadsheet also contains additional supporting data relevant to each anomaly type as well as correlation to existing network monitoring events, providing valuable context to the anomalies. Details around the supporting data will be discussed further in Section 2.5. Presenting the alerts in this way promotes effective communication, enables informed decision-making, and ultimately contributes to maintaining a resilient and high-performing network.

In the final phase of compiling the anomaly detection alerts report, the report is automatically shared with stakeholders through a cloud-based collaboration platform for communication. In this phase of the system, the report is shared for not only investigation but also validation which will be discussed in Section 3. See Figure 2 for the full cycle of the anomaly detection workflow.

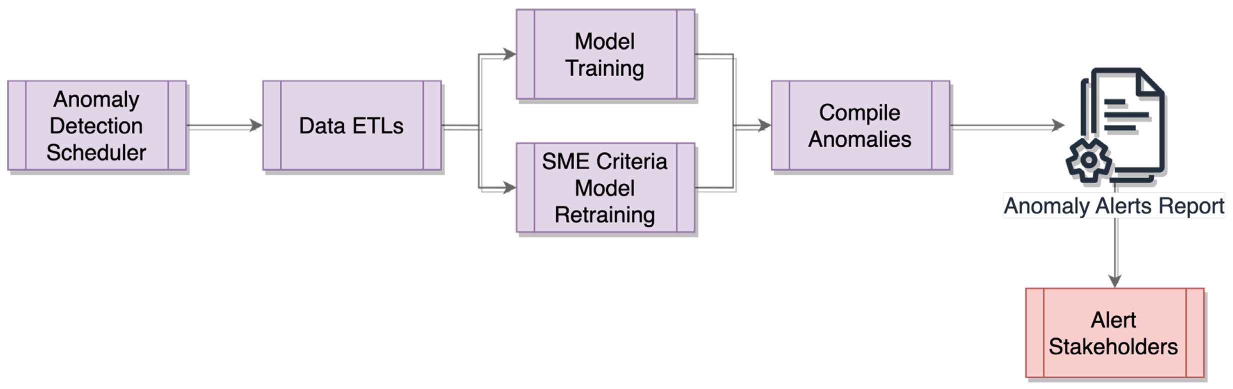

**Figure 2 – Anomaly Detection Architecture**

### 2.4. Anomaly Detection Algorithms

This section will discuss the details of each anomaly detection model to the KPIs introduced in Section 2.2. The models range from statistics-based to unsupervised machine learning models and all come with nuances that are important in building an accurate and reliable anomaly detection system.

For post-deployment long term monitoring, all models follow similar criteria for validating and alerting an anomaly. Since deployments can result in different behavior immediately after, the first is omitting anomalies up to 12 hours after a deployment. Any behavior in those hours is caught in the instantaneous network health checks or is expected, so they do not warrant an alert. The second is incorporating threshold criteria for alerting, requiring the anomaly to reach a severity threshold in time and/or maximum customer impact (these data points also play into ranking later). Section 4 will mention the future work of incorporating causality, further improving the accuracy in distributing anomalies in front of the right stakeholder.

#### *2.4.1. Cable Modem Online Signal*

The initial anomaly detection algorithm was developed for the CM Online signal. Instead of relying solely on thresholds, this algorithm considers historical patterns and the severity of fluctuations in the CM signal. As mentioned in Section 2.2, when an RPD is re-booted, all CMs will go offline momentarily, but since it is instant and expected it should not be considered an anomaly. Similarly, during maintenance windows, it is possible to see the same pattern for a given RPD each day which is also not anomalous. The CM algorithm overcomes this by utilizing statistical methods for anomaly detection, providing a lightweight yet powerful model with results that can be easily interpreted.

As mentioned in Section 2.1, standard deviation in time-series anomaly detection provides a lot of power and is the core of the CM anomaly detection model. To reach that predictive power, pre-processing is the first step which includes removing extreme outliers, normalizing, and differencing the data. After pre-processing, the following method is applied to each timestamp. Initially, a fixed number of hours is looked back on, and a rolling window of approximately one hour is applied. Next, the standard deviation of each window is calculated. If a large percentage of the windows exceeds the standard deviation threshold, the timestamp is considered an anomaly. By adjusting the window size and applying appropriate thresholds, the model becomes less sensitive to sudden or small-scale anomalies, thereby increasing accuracy in identifying anomalies that warrant alerts in this domain.

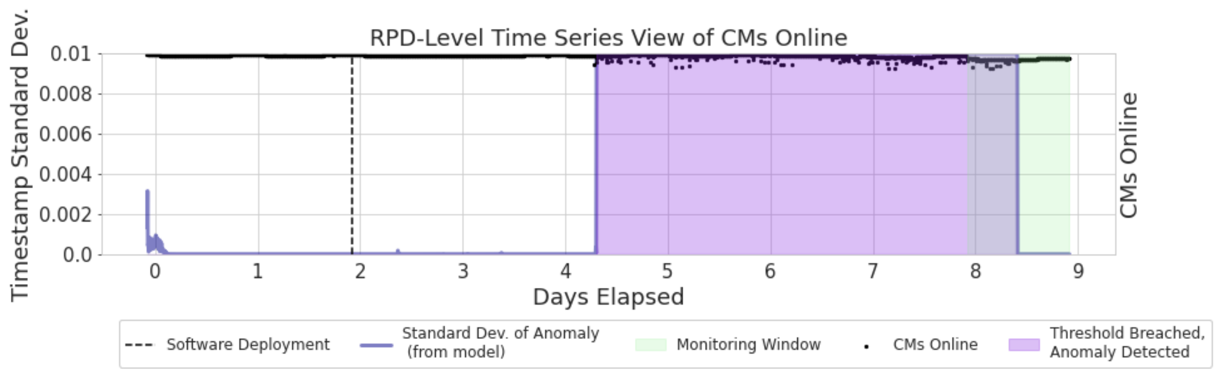

**Figure 3 – CM Signal Oscillating Post-Deployment Anomaly Example**

Figure 3 exemplifies a historical CM anomaly that occurred several days after the software deployment. The anomaly persisted for several days and is caught when applying the model, alerting during the monitoring window. The model's ability to catch this known anomaly highlights its potential to alert stakeholder from the initial day of occurrence. It is worth noting that examples of this scale are now rare as they are actively flagged and addressed immediately with the CM anomaly detection model in production.

#### *2.4.2. Customer Calls (Interactive Voice Response)*

As mentioned in Section 2.2, IVR works as a comprehensive metric when monitoring the network post upgrade. Customers can call in at any time of the day for a multitude of reasons, either network platform related or not (billing questions, in-home support, etc.). Detecting anomalies on the PPOD in and out of peak hours is crucial as a spike in calls can be very telling of the state of the network. Since customers for each PPOD exemplify different historical patterns, that is not related to the software deployment, a simple threshold-based model does not apply. The anomaly detection system utilizes a generalized extreme studentized deviate (ESD) detector for IVR at a PPOD level. The ESD is a statistical method used for detecting outliers or anomalies in a univariate dataset. The IVR model takes a statistical approach for similar reasons as CM Online Signal; the model is easy to understand, implement, and flexible due to significance level and maximum number of outliers thresholds within the model. To incorporate severity of the customer calls, the algorithm is applied to the rolling percentage of IVR to total number of CMs in the PPOD. The model accurately detects impactful customer call anomalies that then can be correlated back to a deployment.

Figure 3 illustrates an instance of an anomaly (red dot) occurring after a software deployment (dashed vertical line). The green shaded area represents the monitoring window within which the anomaly detection system analyzes the metrics, if there is an anomaly within the window then it is alerted on. It is evident that the anomaly detected is valid in comparison to historical trends prior to the deployment, but the question still lingers if this anomaly is directly tied to the deployment. This example will be further discussed in Section 2.5.

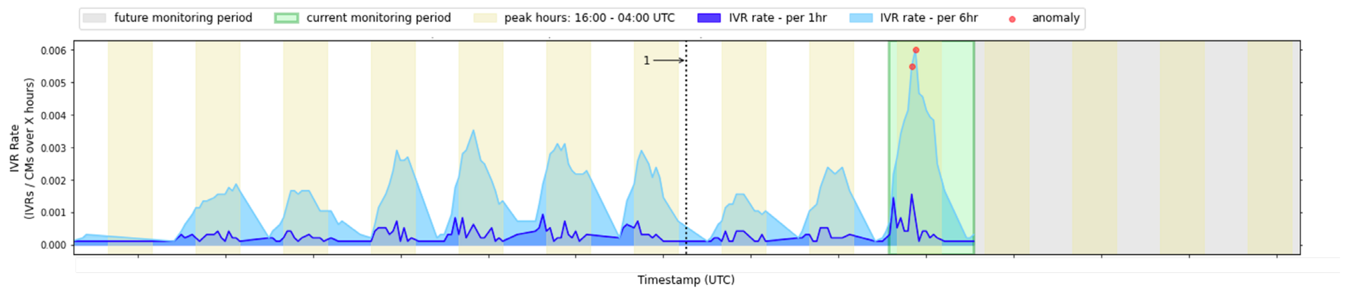

**Figure 4 – Spike in IVR Post-Deployment Anomaly Example**

### *2.4.3. Quality of Experience*

Quality of experience is a metric derived from multiple network performance statistics (latency being one) that represents the experience of residential high-speed data (HSD) customers. When certain thresholds are crossed, a QoE issue occurs at the cable modem level. When traffic gets congested or some event adversely affects network health, the number of CMs with QoE issues will grow. However, the QoE metric tends to follow regular patterns depending on the hour of the day, day of the week, holiday schedule, etc. The challenge is being able to capture an adverse effect on the network at any time and not just during peak hours when some upper bound is pushed too far. To that end, one goal is to be able to accurately model what is typical at every hour and every day of the week. Furthermore, since the network is constantly evolving, another goal is to capture this evolution while still catching any slow deterioration in network health. While capturing cyclic patterns in the data is typically handled by the model itself, capturing the evolution of the network can be jointly tackled by both the model and various heuristics in the application layer. Some logic in the application layer we use to trigger model retraining includes data shifts caused by RPD movements related to the PPODs, irregular historical patterns that cause exploding forecast uncertainty, and redeployments of PPODs.

Several univariate modeling approaches that don't require covariates were considered. One approach that was considered is a mixture model (e.g., Gaussian mixtures). However, these models have natural limitations when it comes to capturing dependencies between timestamps (sequential dependency); much of their advantage also disappears when we limit them to a single metric. One way to convert a mixture model to capture sequential dependency is to form a hidden Markov or attempt to add more flexibility with a long short-term memory (LSTM) or other deep learning architecture. Although these approaches have the benefit of using a single model on a variety of different time series, one downside is the lack of interpretability when trying to understand model predictions and align them with domain expertise. Deep learning approaches can also run into issues with their lack of consistency in their uncertainty estimates. Frequentist autoregressive approaches (e.g., ARIMA) are another option, and they provide a better uncertainty framework. However, such models can be tricky to adapt when multiple cyclic patterns are present; they also can run into similar interpretability issues since their data generative process is very statistical and often obscured by various transformations. After further understanding the need for a QoE anomaly detection model and the advantages and disadvantages of some approaches, it led to the development of the model currently in production.

A dynamic linear (state space) model was developed to capture the multiple cyclic patterns in the data and get reasonable model interpretability. To acquire uncertainty estimates in these patterns, the model was made Bayesian. In the Bayesian framework, such a model can suggest interpretable latent variables that evolve according to a pre-specified pattern (e.g., hourly repetitions) that generalizes well and is controlled by simple parameters (such as regression coefficients). See Figure 5 for these detected patterns in the QoE data with the model. The Bayesian variation then considers the observed data to be composed additively by the latent variables and computes the posterior distribution of all parameters simultaneously; this computation of the posterior takes into account both the likelihood of the observed data (based on our model specification) and priors (having mainly a regularization role here). Some key components that can help with anomaly detection are ‘dynamic’ (latent components can evolve), ‘linear’ (model specification that can generalize) and ‘Bayesian’ (priors that can regularize and model that can quantify uncertainty). Another benefit of this approach is it can be tuned to specify the sequential evolution of the latent states to explicitly get a stable level suitable for anomaly detection.

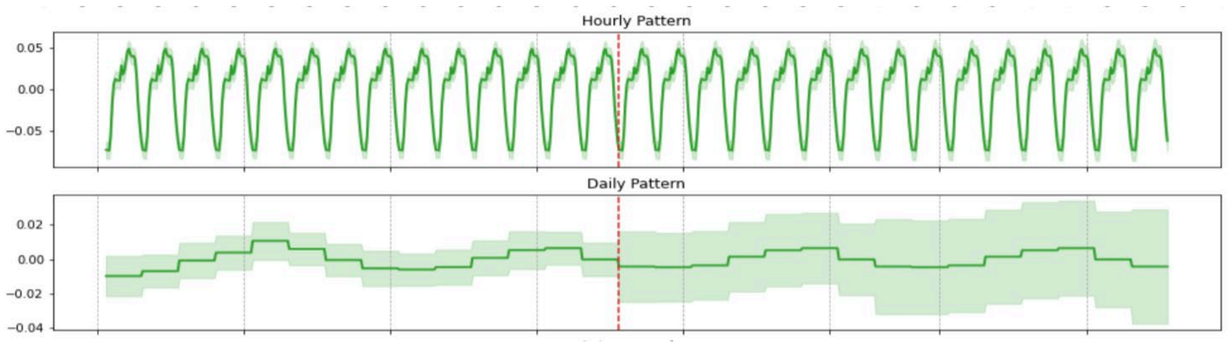

**Figure 5 – QoE Hourly and Daily Patern Detected by Bayesian Framework**

Since this is a statistical model without true anomaly labels, one of the challenges is identifying and accounting for anomalies that occur in the training data when learning the normal behavior of the system. Two common model-native solutions include setting the prior variance of latent or observation noise dynamically based on certain heuristics and using more flexible noise distributions to prevent outliers from having too much influence on fit. Another general solution, which takes advantage of workflows that retrain models at regular intervals, is to impute anomalies in the new training data using the prediction from the model that existed prior to this. Of course, this solution can have a cold start effect when there is no pre-existing model, and in that situation the model-native solutions mentioned previously can help with this.

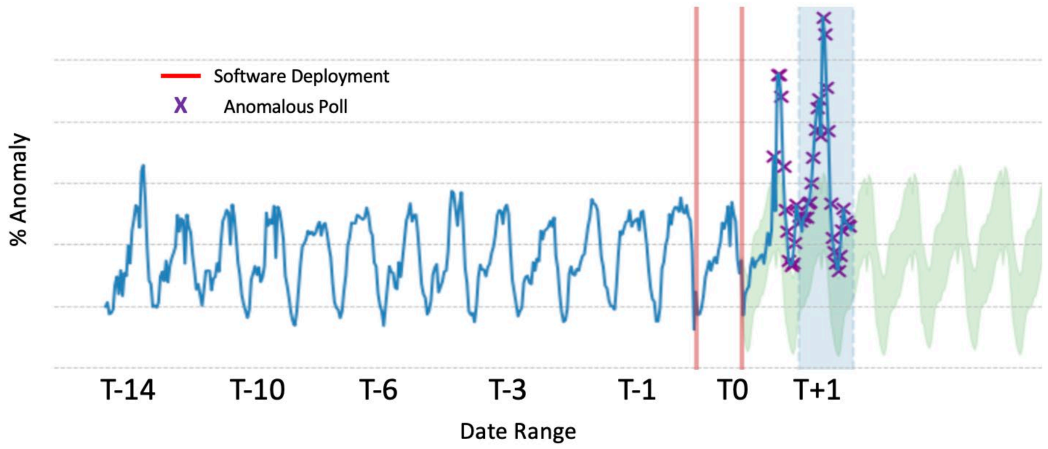

**Figure 6 – QoE Display Slow Degredation Post-Deployment Example**

Figure 6 is a QoE anomaly example, the blue represents the ratio of QoE events to CMs in the PPOD, the red vertical line indicates a software deployment, and the purple marks are anomalies detected. Timestamps are labeled as anomalies if the actual QoE ratio falls outside of the confidence bands (green, generated by forecast). The training data relies on empirical data prior to the deployment and retrains based on a handful of SME criteria (re-training step seen in Figure 2). These model characteristics result in a confident and telling QoE anomaly alert.

## 2.5. Understanding Anomalies

When constructing an anomaly detection system, it is crucial to instill confidence in individual alerts when presenting them to stakeholders. As the number of alerts scale with deployments, it is crucial to provide additional supporting data to help prioritize and understand the scenario. Therefore, the alert system provides useful metadata within the final report.

To leverage the existing alerting systems, each anomaly is accompanied by a list of events that occurred within a few hours in its vicinity. It is important to have a wide correlation window since it is possible for events to be correlated but not occur at the same time. An example use case would be if there is a customer power outage (CPO) due to a storm event, it is possible for the power supply (PS) and node reside on a different part of the power grid than some customers that they service. If the storm event affects the grid with the PS (reaches end of discharge) but not the customers, then those customers would call in causing an IVR anomaly. Knowing this information, the anomaly alerted on is most likely not due to the network deployment being monitored. By using systems that track these events, the end-user would be able to make their own assumptions based on the events provided.

Table 1 – Example Supporting Data for Anomaly (Corresponds to Figure 4)

| alertId | PPOD | Alert Type | Deployment Group Priority | Existing Events | CM Info |
|---|---|---|---|---|---|
| example_uuid | example_PPOD | IVR | LOW | Storm Outage Event #A, RPD Unreachable #B | X% offline DOCSIS 1.0, Y% offline DOCSIS 3.0, Z% offline DOCSIS 3.1 |

Another indicator of whether or not the anomaly is a side effect of the deployment, is the breakdown of CM behavior by DOCSIS version. The supporting data includes the count of offline CMs around the time of the anomaly, which can be especially valuable when an entire group experiences connectivity issues. This information guides subject matter experts toward a better understanding of the underlying cause behind the anomaly.

Lastly, it is important to note that the deployments being monitored are scheduled in groups of PPODs. The supporting data also includes the anomaly priority for each PPOD in the report. If multiple PPODs within a group exhibit an anomaly compared to a single PPOD, it can be argued that the latter represents an isolated event, while the former group is more likely to display anomalies related to the deployment. Therefore, a group priority alert is assigned to each PPOD, with a high priority indicating multiple PPODs experiencing alerts, medium priority signifying a threshold crossing, and low priority assigned otherwise. By incorporating these measures, the end-users can maintain awareness of anomalies without prematurely drawing conclusions. This approach enables them to proactively respond to alerts in a manner aligned with the available information and take appropriate actions accordingly.

Looking at Figure 3 again from Section 2.4.2, it is important to note that while these anomalies are valid, they can be influenced by various causal events unrelated to the deployment itself. In this example, the supporting data presented in Table 1 guides the operations team in identifying potential causes. The data includes information on existing events, cable modem (CM) details, and the priority of the deployment group. In this example, the existing events prove to be the most valuable in assisting the team to pinpoint a potential root cause (storm outage) and go from there.

## 3. Performance and Impact Findings

Detecting anomalies and evaluating the effectiveness of anomaly detection algorithms are pivotal steps in the development and deployment of models. However, this process presents challenges due to the absence of labeled anomalies, which can manifest as diverse patterns and trends such as contextual, point-based, seasonal, cyclical, or collective anomalies. Consequently, conventional evaluation metrics like precision, recall, F1-score, or AUC-ROC (area under receiver operating characteristic curve) may not be directly applicable in this context.

To overcome this challenge of producing highly confident models without labeled data, the anomaly detection approach relies on empirical data and trend-based training to produce high-confidence results leading to increased performance. As mentioned in Section 2.4, this is done by either predicting future values and creating upper and lower confidence bounds (QoE model) or incorporating thresholds based on the empirical data (CM and IVR model).

As described in 2.4.3, to further validate the identification of an anomaly, there is criteria to determine whether the empirical data for a specific period preceding the anomaly occurrence meets the requirements for exclusion, training or re-training. The network is ever changing and producing anomaly alerts from high-certainty training data is crucial.

By considering these exclusion and re-training criteria, we ensure that the anomalies identified by the algorithm are reliable and meaningful. This approach combines empirical data analysis, trend-based predictions, and criteria-based validation to enhance the accuracy and effectiveness of anomaly detection in real-world applications. The QoE example mentioned previously encompasses all of these high-performance efforts.

In addition to the above steps, measuring the effectiveness of anomaly detection algorithms involves correlating them with other events (Section 2.5) captured through existing systems and tools. Correlating with these events helps build confidence in the detected anomalies and prevents disregarding an anomaly that lacks corresponding events.

When assessing the impact of modeling and detecting anomalies, the focus returns to the original objective outlined in Section 1. The goal is to identify service degradation over a period of time that may not be detected in instantaneous checks immediately after a deployment. Some degradations are gradual and may go unnoticed by existing tools. These degradations have a negative impact on customer experience and, if undetected, can be masked by other changes being rolled out. Figure 6 is a great example of a gradual degradation; as noted, the empirical data did not display a similar behavior as post deployment behavior, and the percent of QoE events slowly increases resulting in an impactful anomaly detection alert.

To assess a subset of anomalies, network operations experts validated ~250 alerts over a span of two months. To date, these identified anomalies have demonstrated a 98% accuracy and low recall, implying that the algorithm is sensitive in capturing subtle changes based on empirical data. By incorporated expert feedback, various enhancements have been implemented. These include the grouping of anomalies based on deployment date and type, assigning severity based on the number of periods of degraded service, and considering the frequency of anomaly occurrences.

Another important aspect is making the detected anomalies actionable by identifying their root causes and establishing correlations with other anomalies. This effort will be further discussed in a subsequent section.

## 4. Future Work

In the preceding sections, we provided an overview of the existing long-term monitoring system. Now, we turn our attention towards future endeavors that aim to propel the system closer to its ultimate objective of effectively suggesting rollbacks for DAA software deployments and offering correlation and causality for detected anomalies. One of these future endeavors is the incorporation of additional metrics and attempting supervised learning.

### 4.1. Labeling Tools

The goal for future models and metrics is to incorporate supervised learning for anomaly detection, which can significantly increase the confidence and accuracy of generated alerts. The process of supervised learning necessitates labeled data. As mentioned in Section 2.1, labeled data plays a crucial role in training an effective anomaly detection model. It provides ground truth information about which events are normal and which events are anomalous. It also enables the training of supervised learning models, especially for classification algorithms. By using these labeled examples, the model can establish a baseline, and learn the patterns, features and examples of which deviations beyond the baseline patterns are expected and which ones are unexpected. The availability of labeled data also helps in performance evaluation – performance metrics like accuracy, precision, recall, F1-score, etc. can be used to understand and compare different models.

However, the collection of labeled data comes with its set of challenges, mainly revolving around the requirement of human involvement, i.e., the need for subject matter experts to flag the anomalies. This challenge gets larger when factoring in scale, because as the size of the dataset increases, labeling becomes more challenging and time consuming. It is also a challenge to represent the different types of patterns, trends, and anomalies that exist in real-world data.

To address some of these issues, two user interfaces (UIs) were internally designed to make event labeling easier. The first is a larger tool that operates on the cloud and facilitates collaboration and a shared directory to store the labels in. This is the tool that is intended to be the source of truth for future labeling efforts across multiple projects in data sciences. In the process of development for anomaly detection efforts, there is a second internal tool based purely in Python, HTML and JavaScript that incorporates various features to increase the simplicity of the labeling process. The process creates a standalone UI that displays time series views of a selected set of metrics of interest and provides features such as zooming, time range highlighting, event labeling, providing additional annotation, and the ability to export the changes into a CSV file.

In order to standardize training set creation across future metrics, criteria are put in place to select time series samples with at least 28 continuous days of data and represent a wide range of samples – samples with no anomalies, samples that only had anomalies in the latter two weeks, and time series samples with anomalies throughout the time range.

The next section will discuss the on-going efforts to expand the metrics monitored and discusses the first metric to utilize the labeling efforts for model development. By utilizing the developed labeling tools, there is a substantial labelled data set covering multiple scenarios across the platform footprint. Since the occurrence of anomalies is rare, the labeled data results in an imbalanced target variable. Thus, future approaches need to take this into account by either leveraging existing labels to take a semi-supervised approach or creating synthetic samples through up-sampling/down-sampling techniques, potentially improving future models' performance.

### 4.2. Expanding Metrics Monitored

Having a set of labeled anomalies to experiment with, this section will introduce an analysis on traffic metrics (previously mentioned in 2.2) and demonstrate how to use these labels to confidently select a model for production. This analysis compares a handful of models, both a set of naïve baseline models and forecasting models. Since this work is on-going and experimental, understanding each model evaluated is out of scope for this paper. The baseline models are light since they rely on simple heuristics (similar to CM Signal and IVR); they are interquartile range (IQR), delta-based, and Auto-Regression. The baseline captures regular patterns, traffic volumes, and expected behaviors. The set of forecasting models (similar to QoE) include ARIMA, exponential smoothing (Holt 2004), Facebook Prophet (Taylor and Letham 2017), Theta (Assimakopoulos and Nikolopoulos 2000), and Season-trend Decomposition (Hyndman and Athanasopoulos 2021*)*. These methods involve forecasting the time series, computing residuals or error from the true values, and then setting thresholds on the residuals to identify where the data points are much further from where they would be expected to be.

Evaluating event-based predictions presents another set of challenges, such as differences in temporal alignment or event durations. Imbalanced data is another challenge, since the occurrence of events is rare. Additionally, since the values being compared are pairs of true and predicted events, there is no concept of a true negative, which necessitates some specific metrics to provide a more meaningful assessment of model performance.

To this end, the event matching criterion utilized a margin of 20 minutes, where predicted events were considered a match to true events if the predicted event occurred within 20 minutes of the true event. These matched events were then evaluated using precision, recall, F1-Score, and Jaccard index. All evaluation scores fall within a range of zero to one.

**Table 2 – Evaluation Score by Model for Traffic Metric (Leaf-MAGG) – Leading Methods Based on F1-score Highlighted**

| metricGroup | eventMethod | Precision | Recall | Jaccard Index | F1 Score |
|---|---|---|---|---|---|
| Leaf-MAGG | IQR | 0.0594 | 0.0507 | 0.0281 | 0.0547 |
| | ARIMA | 0.1192 | 0.1305 | 0.0665 | 0.1246 |
| | Facebook Prophet | 0.9438 | 0.1065 | 0.1058 | 0.1913 |
| | **Exponential Smoothing** | **0.573** | **0.3384** | **0.2702** | **0.4255** |
| | **Season-Trend Decomposition (Moving Average)** | **0.556** | **0.327** | **0.2593** | **0.4118** |
| | Auto-Regression | 0.482 | 0.237 | 0.1889 | 0.3178 |
| | Delta from Previous Poll (%) | 0.7976 | 0.2548 | 0.2393 | 0.3862 |

See Table 2 for the evaluations of the models for each facet of traffic metrics. Given that the data is imbalanced, the models are primarily evaluated on the F1-score as it is considered best practice in evaluating imbalanced data sets. It is evident that exponential smoothing is the best model across the board, although the score is low that can be attributed to the size and nature of the sample data set. See Figure 7 for the experiment's top two models' results on a sample traffic metric. The top graph is the labeled event, the traffic metric experiences anomalous behavior after a software deployment. The middle and bottom graph are the results of the exponential smoothing and season-trend decomposition models respectively. Most anomalous timestamps are detected (green), some missed (red) and there aren't any false positives which is beneficial in avoiding false alarms. The model for this metric is currently being developed, but by having labeled data the experimental models can be evaluated for selection and ultimately used for supervised ML.

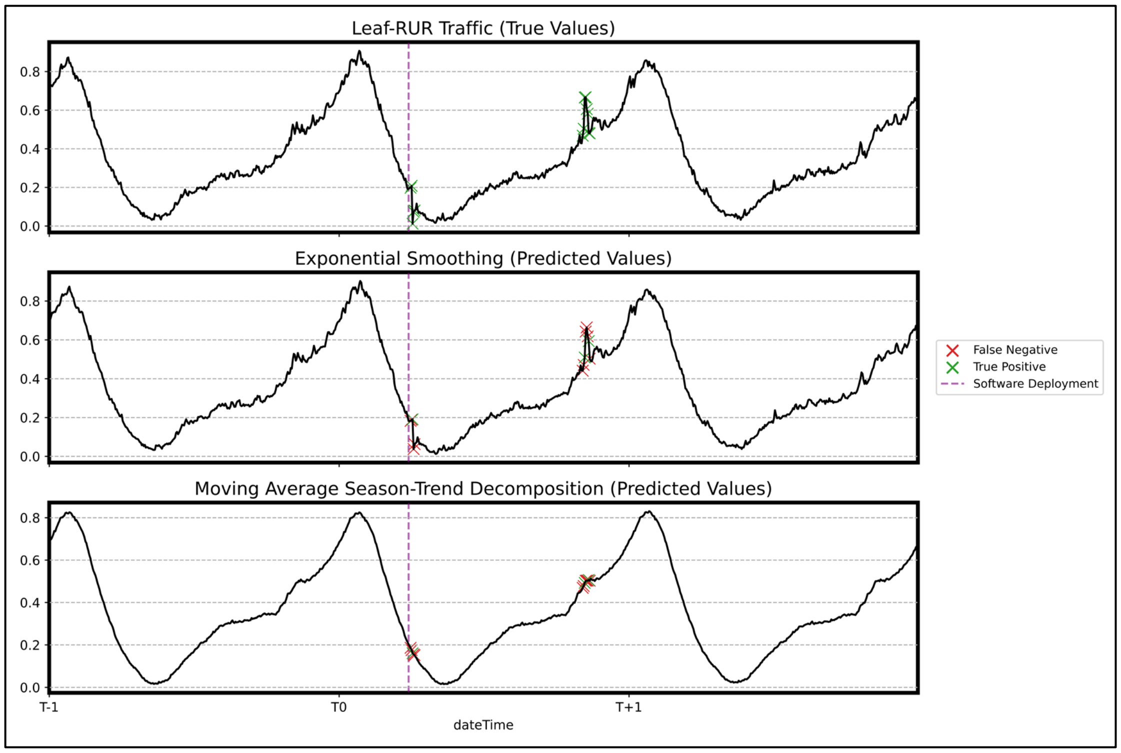

**Figure 7 – Traffic/Switch Anomaly Detection Development – Top Two Models on Leaf-RUR Traffic Metric**

### 4.3. Alarm Correlation and Causation

There are a handful of existing efforts and ideas on enhancing anomaly detection capabilities within the network infrastructure at Comcast, one is expanding existing KPIs discussed in sections 2.4 and 4.2 for measuring platform health. To accomplish the goal of autonomous anomaly detection holistically around different systems - i.e., platform health at home or radio frequency plant, the detection system will monitor additional metrics such as network latency or host utilization which will allow a full analysis of network performance capabilities in a more comprehensive manner.

With this mindset towards proactive problem-solving extending beyond just end-user consumption but also analyzing correlation between different metrics and root-causes, the causality analysis should lead not only towards improvements regarding service reliability but also reduced customer downtime whenever hardware issues seemingly appear randomly.

Essentially through extensive correlative investigation between detected anomalies along with pattern identification/detection throughout event occurrences, the anomaly detection system can actively prevent future instances of the same pattern tends to be on the horizon. Gaining insight into the causes behind different types of anomalies within the complex vCMTS platform architecture requires an approach where relationships are analyzed between various network metrics like latency spikes and host utilization rates. Also of note is establishing any existing correlations which would assist in identifying corresponding contributory factors towards such anomalous trends seen. Once common trends exist among different types of anomalies within our system environment via this data-driven approach, proactive measures could be employed through enhanced capacity planning/resource allocation strategies which optimize future performance.

In better distinguishing between isolated incidents and systemic issues awaiting triage, causal analysis can be conducted on the occurrences of these anomalous events. This helps in pinpointing specific changes to the network like software/hardware updates or hardware failures responsible as anomaly root causes. Once identified, it becomes easier to deliver tailored-fit solutions such as targeted software updates or hardware maintenance procedures geared towards ensuring comprehensive performance optimization. Producing the root cause can also minimize investigation time and speed up recovery.

When these analytical methods are deployed across the network environment for all components, this will help identify recurring themes or common factors across multiple anomalies that occur when specific software updates are implemented. If discovered early enough, improvements like enhanced testing protocols could ensure future anomalies are prevented from occurring again.

Lastly, implementing daily full-footprint anomaly detection offers precise insights into our entire platform through its simultaneous evaluation of all relevant metrics and KPIs across subsystems. Doing this across the entire platform will lead to a much clearer view of system behavior while providing improved precision towards preventing future problems from arising. Full-footprint analysis has potential to provide deep insights pertaining to complex interdependencies among various components resulting in unexpected behaviors showcasing potential issues throughout platforms as opposed to isolated incidents.

Incorporating daily full-footprint (across all PPODs) analyses ensures superior accuracy and efficiency in detecting these anomalous occurrences since they enable establishing base norms and benchmarking them against new telemetry. Adopting this methodology empowers the organization with holistic understanding aligning all integral components, reinforcing fault-tolerance capabilities by extensive cross-functional validation rather than just viewing each KPI and anomaly as an individual siloed outpost.

This approach also enables the identification of anomalies associated with cascading or correlated effects across diversified workplace components which could result in unexpected behavior. By expanding the scope of the analysis, the anomaly detection system will not only identify sporadic or intermittent occurrences but frequently mitigate associated impediments to maintain the relevant networks' overall operational stability. Hence it can be summarized that full-footprint analysis proves imperative in detecting irregularities facilitated through intricate mechanics within the system, providing better user experiences while ensuring platform stability by capturing otherwise-infrequent aberrations of behavior affecting overall performance.

## 5. Conclusion

This paper presented a comprehensive analysis of anomaly detection post software deployment to detect slow deterioration in system performance. The sections covered big data architecture, ML techniques used to detect anomalies, challenges associated with detecting anomalies, particularly in the absence of specific patterns and the existence of various anomaly types. To address these challenges, the system relies on empirical data, trend-based training, and the establishment of validation criteria to ensure the accuracy and effectiveness of the anomaly detection algorithm.

However, the work does not end here. As part of future endeavors, the application of the anomaly detection algorithm is planned to be expanded to encompass the entire virtualized platform daily. This expansion will provide a more comprehensive understanding of system behavior and enable the identification of anomalies that may not be captured in smaller subsets of data. Furthermore, efforts will be made to broaden the range of monitored KPIs incorporating additional metrics to enhance the overall analysis of system performance and correlate between different anomalies and identify root cause for the anomalies detected.

By addressing these future research directions, this work aims to contribute to the ongoing improvement of network anomaly detection techniques and the overall performance of virtualized platforms.

## Abbreviations

| | |
|---|---|
| ARIMA | auto-regressive integrated moving average |
| AUC-ROC | area under receiver operating characteristic curve |
| CM | cable modem |
| CPE | customer premise equipment |
| CPO | commercial power outage |
| DAAS | distributed access aggregate switching |
| DAA | Distributed Access Architecture |
| DOCSIS | Data-over-cable Service Interface Specification |
| ESD | extreme studentized deviate |
| ETL | extract, transform and load |
| HAGG | hub aggregator |
| HSD | high speed data |
| IQR | inter-quartile range |
| IVR | interactive voice response |
| KPI | key performance indicator |
| LSTM | long short-term memory |
| MAGG | mother/master aggregator |
| ML | machine learning |
| PPOD | physical pod |
| PS | power supply |
| QoE | quality of experience |
| RPD | remote physical device |
| RUR | residential U-Ring router |
| SME | subject matter expert |
| UI | user interface |
| vCMTS | Virtual Cable Modem Termination System |

## Bibliography & References

*Assimakopoulos, V., & Nikolopoulos, K. (2000). The theta model: a decomposition approach to forecasting. International Journal of Forecasting, 16(4), 521-530.*

*Bhuyan, M. H., Bhattacharyya, D. K., & Kalita, J. K. (2014). Network Anomaly Detection: Methods, Systems and Tools. IEEE Communications Surveys & Tutorials, 16(1), 303-336. doi: 10.1109/SURV.2013.052213.00046.*

*Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly Detection: A Survey. ACM Computing Surveys, 41. doi: 10.1145/1541880.1541882.*

*Eppes, M., et al. (2022). Scaling DAA: Automated Network Health Check for vCMTS Platform. Paper presented at SCTE Expo 2022.*

*Holt, C. (2004). Forecasting Seasonals and Trends by Exponential Weighted Moving Averages. International Journal of Forecasting, 20, 5-10.*

*Hyndman, R.J., & Athanasopoulos, G. (2021). Forecasting: Principles and Practice, 3rd edition, OTexts: Melbourne, Australia. OTexts.com/fpp3.*

*Stehman, M., et al. (2021). Solving The Mysteries of the Distributed Access Architecture. Paper presented at SCTE Expo 2021.*

*Taylor, S. J., & Letham, B. (2017). Prophet: Forecasting at scale. American Statistician, 72(1), 37-45.*