CableMon: Improving the Reliability of Cable Broadband Networks via Proactive Network Maintenance

Jiyao Hu, Zhenyu Zhou, and Xiaowei Yang, *Duke University;* Jacob Malone, *CableLabs;* Jonathan W Williams, *The University of North Carolina at Chapel Hill*

https://www.usenix.org/conference/nsdi20/presentation/hu-jiyao

This paper is included in the Proceedings of the 17th USENIX Symposium on Networked Systems Design and Implementation (NSDI '20)

February 25–27, 2020 • Santa Clara, CA, USA

978-1-939133-13-7

Open access to the Proceedings of the 17th USENIX Symposium on Networked Systems Design and Implementation (NSDI '20) is sponsored by

CableMon: Improving the Reliability of Cable Broadband Networks via Proactive Network Maintenance

Jiyao Hu* *Duke University*
Zhenyu Zhou* *Duke University*
Xiaowei Yang *Duke University*
Jacob Malone *CableLabs*
Jonathan W Williams *The University of North Carolina at Chapel Hill*

Abstract

Cable broadband networks are one of the few "last-mile" broadband technologies widely available in the U.S. Unfortunately, they have poor reliability after decades of deployment. Cable industry proposed a framework called Proactive Network Maintenance (PNM) to diagnose the cable networks. However, there is little public knowledge or systematic study on how to use these data to detect and localize cable network problems. Existing tools in the public domain have prohibitive high false-positive rates.

In this paper, we propose CableMon, the first public-domain system that applies machine learning techniques to PNM data to improve the reliability of cable broadband networks. CableMon uses statistical models to generate features from time series data and uses customer trouble tickets as hints to infer abnormal thresholds for these generated features. We use eight-month of PNM data and customer trouble tickets from an ISP to evaluate CableMon's performance. Our results show that 81.9% of the abnormal events detected by CableMon overlap with at least one customer trouble ticket. This ticket prediction accuracy is four times higher than that of the existing public-domain tools used by ISPs. The tickets predicted by CableMon constitute 23.0% of the total network-related trouble tickets, suggesting that if an ISP deploys CableMon and proactively repairs the faults detected by CableMon, it can preempt those customer calls. Our current results, while still not mature, can already tangibly reduce an ISP's operational expenses and improve customers' quality of experience. We expect future work can further improve these results.

1 Introduction

Broadband access networks play a crucial role in modern life. They help narrow digital divide, enable e-commerce, and provide opportunities for remote work, study, and entertainment. In the US, cable networks are one of the few available infrastructures that can provide broadband Internet access to US homes. In many rural areas, they are often the only broadband choice. According to a study in 2016 [6], cable broadband is available to 93% of US homes, far more than the two alternative choices: high bitrate digital subscriber line (VDLS) (43%) and Fiber-to-the-Premises (FTTP) (29%).

However, cable networks are prone to reliability problems, partly due to their Hybrid Fiber-Coaxial (HFC) architecture. This architecture uses both optical fibers and coaxial cables to deliver a mixed bundle of traffic, including video, voice, and Internet data. Unlike fiber optics, coaxial cables are vulnerable to radio frequency (RF) interference. Many parts of the cable networks are now decades old [1]. Aging can lead to problems such as cable shielding erosion, loose connectors, and broken amplifiers. All those problems can manifest themselves as poor application-layer performance, e.g., slow web responses or low-quality video streaming. Much measurement study has shown that broadband networks have poor reliability [3,12,17,22,23,28]. A recent one [3] shows that the average availability of broadband Internet access is at most two nines (99%), much less than the minimum FCC's requirement (four nines 99.99%) for the public switched telephone network (PSTN) [20]. Admittedly, if ISPs replace the last-mile coaxial cables with fiber optics, many of these problems may disappear. However, due to the prohibitive cost of FTTP, cable broadband networks are likely to remain as one of the few broadband choices in rural America for the next decade or two. Therefore, it is critically important that the cable Internet services remain robust during emergencies, especially as more and more subscribers migrate their landline phones to VoIP phones.

The cable industry has long recognized this problem and developed a platform called Proactive Network Maintenance (PNM) to improve the reliability of their networks [7]. PNM enables a cable ISP to collect a set of performance metrics from each customer's cable modem. We refer to this set of data as PNM data. One example of a PNM metric is a cable

* Jiyao Hu and Zhenyu Zhou, placed in alphabetic order, are the lead student authors and contributed equally to this work.

channel's signal-to-noise ratio. PNM aims to enable an ISP to proactively detect and fix network faults before they impact services and customers.

Although PNM has been incorporated into DOCSIS since 2005 [7], how to use PNM data to improve network reliability remains an open challenge. The best current practice recommended by CableLabs[1] [7] and the tools used by some ISPs [22] use a set of manually configured thresholds to flag a faulty network condition. The feedback from deploying ISPs is that these thresholds are often too conservative, leading to high false positives.

This work aims to improve the reliability of cable broadband networks. We speculate that the challenge of using PNM data is due to the lack of expert knowledge or ground truth on what PNM values warrant a proactive network repair. In an RF system like a cable network, network conditions may degrade gradually, making it challenging to define a static threshold that separates what is faulty from what is not. We develop a system called CableMon, which uses machine learning techniques to infer network faults that demand immediate repair. CableMon couples PNM data with customer trouble tickets to identify the ranges of PNM values that are likely to lead to a customer's trouble call. Our hypothesis is that if a network fault impacts an ISP's customers, then some customer is likely to call to report the problem. Therefore, we can use customer trouble tickets as hints to learn what network conditions are likely to lead to customer trouble calls. It is desirable for an ISP to prioritize its effort to repair those problems, because if they persist, they are likely to reduce customer satisfaction and increase the cost for customer support.

A key technical challenge we face is that both customer tickets data and PNM data contain much noise. Customer tickets are not reliable indicators of network faults. A customer may or may not call when there is an underlying network problem, and vice versa. PNM data, by its nature, describe cable channels' conditions as well as environmental noises. Therefore, if we use customer tickets to label PNM data as normal or faulty, and apply an off-the-shelf machine learning technique to detect network faults, we may not have good detection results. In addition, manual labeling is not practical in this context, because there lacks expert knowledge and the dataset is too large.

In CableMon's design, we use three techniques to address the above challenges (§ 4). First, to reduce noise in customer tickets, we filter customer tickets according to the ticket descriptions and only choose the tickets that suggest network problems as hints. Second, to reduce noise in PNM data, we treat a modem's PNM data as time series data and use its time series features (e.g., expected moving average or variance) for fault detection. Third, we develop a customized classification method that is robust to both noise in tickets and noise in PNM data.

[1]CableLabs is a research and development lab founded by American Cable operators in 1988 and led the development of DOCSIS and PNM.

With the support of CableLabs, we have obtained eight months of anonymized PNM data and the corresponding customer trouble tickets from a mid-size U.S. ISP. We use five months of data to train CableMon, and use the next three months data following the training set as the test set to evaluate how well CableMon detects network faults. CableMon takes the PNM data in our test set as input and detects when a network fault occurs and when it ends. Due to the lack of ground truth, we evaluate CableMon's performance using customer trouble tickets in the test set. When CableMon detects a network anomaly, if a customer who experiences the anomaly reports a ticket, we consider the detection a success. We compare CableMon with a tool currently used by our collaborating ISP, which we refer to as AnonISP, and with a tool developed by Comcast [22]. Our results show that 81.9% of the anomalies detected by CableMon lead to a customer trouble ticket. In contrast, only 10.0% of the anomalies detected by AnonISP's tool lead to a trouble ticket; and 23.5% of the anomalies detected by Comcast's tool lead to a customer ticket. In addition, CableMon predicts 23.0% of all network-related tickets, while AnonISP's tool predicts 25.3% and Comcast's tool predicts less than 3%. The trouble tickets predicted by CableMon on average last 32.5 hours (or 53.3%) longer than those predicted by other tools, suggesting that those tickets are more likely to require repair efforts. The median time from the beginning of a fault detected by CableMon to the reporting time of a ticket is 164.1 hours (or 29.3%) shorter than that of a fault detected by other tools, suggesting that the faults detected by CableMon require more immediate repair.

To the best of our knowledge, this work is the first large-scale public study that couples PNM data with customer trouble tickets to improve the reliability of cable networks. Our main contribution is CableMon, a system that detects network faults more reliably than the existing public-domain work. It serves as a starting point to unleash the full potential of PNM data. We are working with an ISP and the CableLabs to deploy CableMon in practice and we expect the feedback from practice can further improve the performance of CableMon. One general lesson we learn is that one can use customer trouble tickets as hints to learn what values of network performance metrics indicate customer-impacting problems, despite the presence of noise in both the ticket data and the network performance data. We believe this lesson is applicable to proactive network maintenance in other types of networks, including cellular networks, WIFI access networks, and datacenter networks.

2 Background and Datasets

In this section, we briefly introduce the cable Internet architecture and describe the datasets we use in this work.

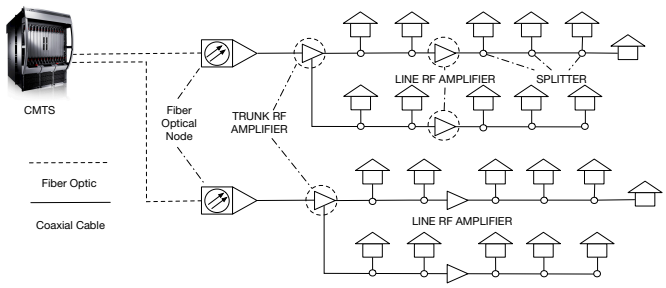


Figure 1: **An overview of the Hybrid Fiber Coaxial (HFC) architecture.**

2.1 Cable Network Architecture

Figure 1 shows a high-level overview of a cable broadband network. A cable broadband network is an access network. It provides the "last-mile" Internet connectivity to end users. A customer connects to the cable network via a cable modem residing in her home. The cable access network terminates at a device called a Cable Modem Termination System (CMTS), which is a router with one port connecting to the Internet and many other ports connecting to customers' cable modems.

At the IP level, there is only one hop between a customer's cable modem/home router and the CMTS. Underlying this single IP hop, there is a complicated link-level structure that consists of different types of physical links and devices. The "last-mile" links that connect to the customer premises are often made of copper coaxial cables. These cables terminate at a device called a fiber node (FN). A fiber node connects to the CMTS via optical fibers. It converts the incoming optically modulated signals into electrical signals and sends the signals towards the customers' homes, and vice versa. Due to signal attenuation, cable networks deploy radio frequency (RF) amplifiers between a fiber node and a residential home. Along the way to a customer's home, new branches may split from the main cable by the line splitters. All these devices could introduce signal distortion and noise.

Historically, cable TV networks divide radio frequency into multiple channels, each of 6MHz width. Cable broadband networks use a subset of these channels as data carriers. A cable ISP typically uses three or four of these channels at the lower end of the spectrum to carry data from a user's cable modem to CMTS. We refer to this direction as the *upstream* direction. An ISP may use sixteen or more of the channels at the higher end of the spectrum to carry data from a CMTS to a modem. We refer to this direction as the *downstream* direction.

2.2 Datasets

We have obtained two types of anonymized modem-level data from a U.S. cable ISP for this study. They include (1) PNM data and (2) customer trouble ticket data. We have a total of eight months of data dating from 01/06/2019 to 08/31/2019. Next, we describe each dataset in turn. [2]

PNM data: The PNM data we obtained were collected by a common standard built into DOCSIS. A CMTS can query a DOCSIS-compliant cable modem (CM) to obtain certain performance data. DOCSIS standardizes how a CM or CMTS stores these performance data in a local Management Information Base (MIB) [7]. A remote process can use the Simple Network Management Protocol (SNMP) to query the MIBs of each CM or a CMTS to obtain performance data [36].

Currently, we only have PNM data from the upstream channels. DOCSIS 3.0 gives a cable operator the ability to collect the full spectrum view of a cable modem's RF channels. It is our future work to investigate whether this type of data may further improve our detection accuracy.

A record in the PNM data we obtain has the following fields:

- *Timestamp*: The time when a PNM query is sent.
- *Anonymized MAC*: The hashed MAC address of the queried CM.
- *Anonymized Account Number*: The hashed user account number. This field is used to link a customer ticket with the corresponding PNM data from the customer's CM.
- *Channel Frequency*: This field identifies which upstream channel this record is about.
- *SNR*: The upstream signal-to-noise ratio of this channel.
- *Tx Power*: A CM's signal transmission power.
- *Rx Power*: The received signal power at the CMTS.
- *Unerrored*: The number of unerrored codewords received at the CMTS.
- *Corrected*: The number of errored but corrected codewords received at the CMTS.
- *Uncorrectable*: The number of errored but uncorrected codewords.
- *T3 Timeouts*: The number of DOCSIS T3 timeouts [5] the CM has experienced since its last reboot. A DOCSIS T3 timeout occurs when there is an error in a CM's ranging process, which we will soon explain.
- *T4 Timeouts*: The number of DOCSIS T4 timeouts [5] the CM has experienced since its last reboot. Similarly, a T4 timeout occurs when there is a ranging error.
- *Pre-Equalization Coefficients*: The set of parameters a CM uses to compute how to compensate for channel distortions during a ranging process.

A CM uses a process called *ranging* to compute a set of parameters called *pre-equalization coefficients* for mitigating channel distortions. When RF signals travel along a coaxial cable, they may be distorted as different frequencies attenuate at different speeds and noise may be added to the channel. To mitigate the channel distortions, a CM adds correction signals to the data signals it transmits. Ideally, the correction signals will cancel out the distortions when the signals arrive at the CMTS. A CM and a CMTS exchange messages periodically

[2] We note that we have discussed this work with our institute's IRB. And they consider it does not involve human subjects.

to compute the correction signals. This process is called ranging. And the set of parameters used to compute the correction signals are called pre-equalization coefficients.

The PNM data we obtain are collected every four hours from several of an ISP's regional markets. There are around 60K unique account numbers in our datasets.

Customer Ticket Data: We have also obtained the records of customer trouble tickets from the same ISP. The relevant fields in each record include the hashed customer's account number, the ticket creation time, the ticket close time (if it was closed), a brief description of the actions taken to resolve the ticket, and a possible diagnosis.

3 Overview

In this section, we motivate the design of CableMon by describing the limitations of existing work. We then describe the design rationale of CableMon, its design goals, and the design challenges we face.

3.1 Limitations of Existing Work

The existing PNM work in the public domain [7, 17, 22] use a set of PNM metrics and manually-set thresholds to detect network faults. If the value of a metric is below or above a threshold, it indicates a fault. This approach has several limitations. First, it is challenging to set the right thresholds. If the thresholds were set too conservatively, they might flag too many faults for an ISP to fix. In contrast, if they were set too aggressively, an ISP might miss the opportunities for proactive maintenance. There lacks a systematic study on how to set the threshold values to achieve the best tradeoff. Second, the existing work mostly uses the instantaneous values of PNM data for fault detection. However, due to the inherent noise in PNM data, using the instantaneous values may lead to instability in detection results. In addition, it may fail to detect faults that can only be captured by abnormalities in a PNM metric's statistical values, e.g., variations.

For ease of explanation, we use one threshold value recommended in the CableLabs' PNM best practice document [7] to illustrate the limitations. CableLabs' recommendation uses a variable called Main Tap Ratio (MTR) computed from a modem's pre-equalization coefficients. It specifies that when the MTR value of a modem is below a threshold ($<$18dB), there is a fault in the network that needs immediate repair.

We sample the MTR values in one of the ISP's markets. There are more than 60K modems in this market. We choose five random days' records during an eight-month period in 2019 and measure the MTR values of all modems during the sampled days. Table 1 shows the percentage of modems that have a channel whose MTR value is below the recommended threshold. If an ISP used the recommended MTR threshold, at any sampled day, there would be more than 24% of cable modems that require immediate repair. We also measure the MTR values among all PNM records during this eight-month period. In more than 26% of the records, a modem's MTR value is below 18dB.

3.2 Design Goals

CableMon aims to enable an ISP to detect network problems that demand immediate repair. Specifically, it aims to accurately detect the set of network conditions that adversely impact customer experience. We refer to these network conditions as network anomalies or faults in this work. Its design goals include the following:

- High ticket prediction accuracy, and moderate ticket coverage. Ideally, we would like to use precision (the set of true positives detected over all detected positives) and recall (the set of true positives detected over all true positives) to measure the performance of CableMon. However, because we do not know ground truth, we instead use customer tickets as indications of true positives. We define *ticket prediction accuracy* as the ratio between the number of anomalies detected by CableMon that lead to one or more customer tickets and the number of total anomalies CableMon detects. Similarly, we define *ticket coverage* as the ratio between the number of tickets CableMon predicts and the total number of network-related customer tickets. It is desirable that CableMon has high ticket prediction accuracy because an ISP is often limited by the number of technicians it has to repair network faults, avoiding false alarms is practically more important than repairing all faults proactively. What we learned from AnonISP is that even a 10% reduction in customer calls can reduce their operational costs significantly. Therefore, as a starting point, we aim for a high ticket prediction accuracy and a moderate ticket coverage.

- No manual labeling. One approach to detect network anomalies is to train a supervised learning classifier on labeled data. The labels tell what PNM metrics indicate network anomalies and what do not. However, we do not have such labeled data. And due to lack of ground truth and the large size of the data, manual labeling is also practically challenging. Therefore, we aim to design CableMon without requiring manual labeling.

- No extensive parameter tuning. We aim to release CableMon as an off-the-shelf-tool at cable ISPs. Therefore, we require that CableMon's fault detection methods work effectively without much parameter tuning on the ISP side.

- Efficient. We require that CableMon can detect whether there is a network fault or not in real time. This is because an ISP can deploy CableMon as a diagnosis tool in

	03/13/2019	04/09/19	06/25/19	07/15/19	08/15/19	Eight-month
MTR < 18dB	24.95 %	25.45 %	27.16 %	27.07 %	27.38 %	26.15 %

Table 1: **The percentage of cable modems that need to be repaired if an ISP were to follow one of the CableLabs' recommendations.**

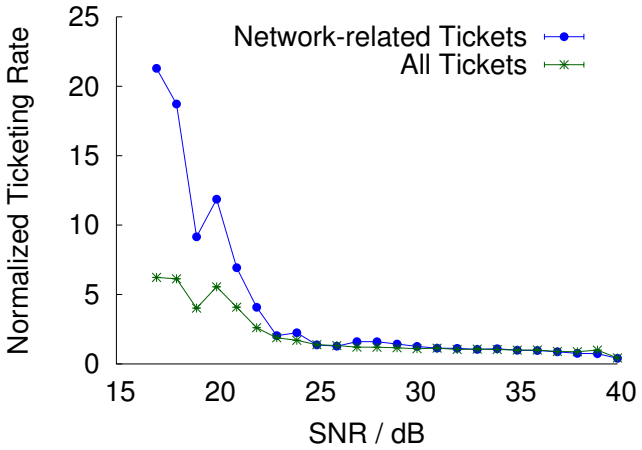


Figure 2: **This figure shows how the customer ticketing rate varies with the values of SNR. Ticketing rate tends to increase when SNR values are low.**

addition to using it for proactive network management. When an ISP receives a customer trouble call, it is often challenging to diagnose what has caused the customer's problem. An ISP can use CableMon to help diagnose whether the problem is caused by a network fault.

3.3 Design Rationale

To meet CableMon's design goals, we use customer trouble tickets as hints to train a customized classifier to detect network faults. We hypothesize that the occurrences of customer trouble tickets should correlate with those of network faults. When a customer-impacting fault occurs, some customers are likely to call the ISP to fix it. Each call creates a trouble ticket. If the values of PNM data can indicate network faults, then the values of PNM data should correlate with how frequently customer trouble tickets are created. In this paper, we define the average number of customer tickets created in a unit time as the *ticketing rate*.

To validate this hypothesis, we measure how ticketing rate changes with different values of a PNM metric. For a PNM metric m (e.g., SNR), we sort the values of m in an ascending order. Each pair of adjacent values define a bin b. For each bin b, we measure the number of tickets N_b that occur in the time periods where the value of m falls within the bin, and the total length of those time periods T_b. We then divide N_b by T_b to obtain the ticketing rate for bin b. We note that a PNM record is collected at discrete time points (roughly four hours apart in our datasets). We assume that a PNM value remains unchanged between its collection points.

As an example, we show how the ticketing rate varies with the values of SNR in Figure 2. We normalize this value by the baseline ticketing rate, which we obtain by dividing the total number of customer tickets in our dataset by the total collection period. The line marked by the legend "All Tickets" shows how the ticketing rate varies with the values of SNR if we consider all tickets; and the line marked by "Network-related Tickets" shows how the ticketing rate of network-related tickets varies with SNR. As can be seen, when the values of SNR are low, both the network-related ticketing rate and the all-ticket ticketing rate tend to increase, suggesting that low SNR values signal network faults.

In practice, customer tickets do not always indicate network faults. On the one hand, many customers may call an ISP for non-network related problems. The customer ticket data we obtain includes a ticket action field and a ticket description field, which provide the information on how an ISP deals with a ticket. We observe that nearly 25% of tickets are resolved via"Customer Education" or "Cancelled", suggesting that they are not caused by networking problems. On the other hand, customers may not report tickets when network outages indeed take place. In our ticket dataset, when an outage affects an entire region, all tickets caused by that outage are labeled as "part of primary," grouped and pointed to a primary ticket, which is a representative ticket of the outage. We manually checked an outage that affected more than 200 customers' PNM data and observed that only $\approx 6.1\%$ of the customers have a "part of primary" tickets and the rest $\approx 93.9\%$ of the customers did not report anything.

To reduce noise in tickets, we select a subset of customer tickets that are likely to be caused by network problems. We select the tickets based on both a ticket's action field and the ticket's description field. From the action field, we select tickets that lead to a "Dispatch" action. We assume that the tickets that caused an ISP to dispatch a technician are likely to be triggered by network-related problems. From the description field, we select tickets whose ticket description keywords suggest networking problems. Examples of such keywords include "Data Down", "Noisy Line" and "Slow Speed". In the rest of this paper, we refer to those selected tickets as "network-related tickets."

Figure 2 compares how the ticketing rate of network-related tickets and all tickets vary with SNR values. As can be seen, network-related tickets have higher ticketing rates when SNR is low, suggesting that the occurrences of those tickets are better indicators of network faults.

We note that according to the ISP who provided us

the datasets, network-related tickets may also contain non-networking tickets due to human errors. A human operator who fills a ticket action or description field may make a mistake. And a technician may be dispatched when there is no network fault due to an erroneous diagnosis.

Challenges: A key question we need to answer is how to use customer tickets as hints for detecting network faults. Ideally, if a customer calls only when a network fault occurs, we could label the PNM records collected around the ticket creation time as abnormal, and apply supervised learning to learn the PNM thresholds that suggest a network fault. We have tried several such machine learning algorithms when we started this project, but found that that this approach did not work well with our datasets. First, customer calls are unreliable fault indicators. A customer may or may not call when there is a fault and vice versa. Second, PNM data contain noise. During a faulty period, some PNM metrics may occasionally show normal values due to the added noise. Similarly, even when there is no fault, some PNM metrics may show instantaneous abnormal values. Thus, if we use the tickets to label PNM data, it may introduce many false positives as well as many false negatives. We found it challenging to tune a machine learning algorithm with this labeling method. It is even harder to explain the results when we change a parameter. Next, we describe how we design CableMon to use a simple and customized classifier to address these challenges.

4 Design

In this section, we describe the design of CableMon. We first describe how we reduce the noise in customer tickets and the noise in PNM data. We then describe a customized classifier that aims to robustly classify PNM values as normal and abnormal despite the presence of noise. Finally, we describe how an ISP can use the classification results to detect network faults and to help diagnose a customer's trouble call.

4.1 Reducing Noise in PNM data

PNM data measure the instantaneous state of cable's RF channels and contain noise. An added noise may make a PNM metric take an abnormally low or high instantaneous value. To address this problem, we treat PNM data as time-series data and apply statistical models to smooth the noise and generate additional features for fault detection.

Table 2 summarizes all the statistical models we use to process PNM data. For each PNM metric collected at timestamp i with value V_i, we calculate its average, its weighted moving average (WMA), exponentially weighted moving average (EWMA), the difference between the current value and its WMA (WMA Diff), and its variance. We note that the average, WMA, WMA Diff, and variance values all require a window size as a hyper-parameter. Because we do not have any prior knowledge on how to set this parameter, we try a series of

Model	Equation
Average	$AVG_i = \frac{V_i+V_{i-1}+\cdots+V_{i-win+1}}{win}$
WMA	$WMA_i = \frac{win\cdot V_i+(win-1)\cdot V_{i-1}+\cdots+1\cdot V_{i-win+1}}{win\cdot(win-1)/2}$
EWMA	$EWMA_1 = V_1$ $EWMA_i = \lambda\cdot V_i+(1-\lambda)EWMA_{i-1}$
WMA Diff	$V_i - WMA_i$
Variance	$VAR_i = \frac{1}{win}\sum_{k=i-win+1}^{i}(V_k - AVG_i)^2$

Table 2: **This table summarizes the statistical models we use to generate the time-series features. (WMA: Weighted Moving Average, EWMA: Exponentially Weighted Moving Average.)**

window sizes, ranging from *1 day* to *7 days*, incrementing by 1 day at each step. For the λ parameter required by EWMA, we vary the value of λ from *0.1* to *0.9*, incrementing by 0.1 at each step. For each PNM metric, we generate 37 time-series features. We apply this approach to all nine PNM metrics and totally generate 333 time-series features. We refer to them as time-series features.

4.2 Determining A Fault Detection Threshold

After we reduce noise in both the customer tickets and the PNM data, we aim to determine a threshold for each PNM metric that indicates network faults. We note that there is no explicit definition of what a network fault is. Instead, we choose to use the network conditions that are likely to cause a trouble call to approximate a network fault. With this approximation, we may not detect minor issues that do not warrant a trouble call. We argue that this design is advantageous, because it allows an ISP to prioritize its resources to fix the customer-impacting problems.

In the case of SNR, if we choose too high a value as a fault detection threshold, an ISP may become too proactive, fixing minor problems that many customers may not care, which we refer to as false positives. If we choose too low a value, an ISP may miss opportunities to proactively repair a problem before a customer calls, which we refer to as false negatives.

We aim to design an algorithm that minimizes both false positives and false negatives. From our investigation in § 3.3, we see that different values of a PNM metric have different likelihood to concur with a trouble ticket. Inspired by this observation, we use the ticketing rate as a metric to help choose a fault detection threshold. Our intuition is that the customer ticketing rate during a faulty period should be higher than a normal period when there is no fault. Therefore, for each feature f generated from a PNM metric, we determine a threshold value thr_f such that thr_f maximizes the ratio between the ticketing rate in the time periods when a network fault exists and the time periods when there is no fault. We refer to this ratio as the ticketing rate ratio.

Specifically, we search through the range of values of a feature f in small steps. At each step s, we consider the value of the feature $f_s \in [f_{min}, f_{max}]$, as a candidate for the threshold.

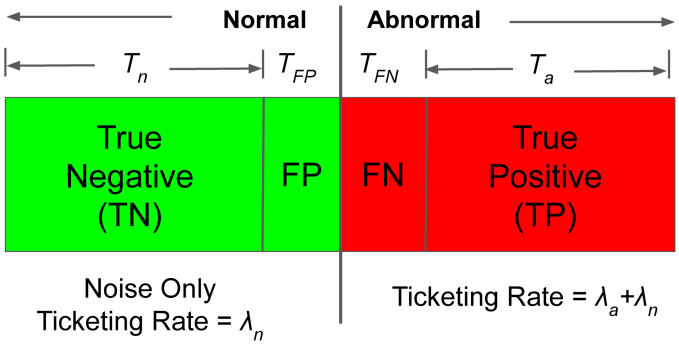


Figure 3: **Analysis of ticketing rate ratio.**

We then compare the value of f at a PNM data collection point with f_s, and label the collection time period as abnormal or normal, based on whether the value of f is below or above the candidate threshold value f_s. For some features such as the average SNR, below the threshold is abnormal. For other features, the opposite is true. After determining each collection period as normal or abnormal, we count the number of network-related tickets occurred in the normal and abnormal periods respectively and divide them by the normal and abnormal time periods determined by f_s. We then compute the ticketing rate ratio: $TRR(f_s)$. The threshold value thr_f is chosen as the value of f_s that maximizes the ticketing rate ratio $TRR(f_s)$.

We also note that for features following a normal distribution such as Rx Power, we choose to use two threshold values to determine whether a collection period is normal or abnormal. The pseudo code can be found at §A.

We now explain why choosing a threshold that maximizes the ticketing rate ratio may help minimize the false positives and false negatives. The entire time line can be divided into two subspaces: the normal (no fault) and the abnormal (with fault) periods. Ideally, the normal sub-space should not receive any trouble ticket. In practice, there is always a ticketing noise. We assume a uniformly distributed ticketing noise with the rate λ_n spreads the whole space. Similarly, we assume an additional uniformly distributed ticketing rate that occurs only in the abnormal sub-space and denote it as λ_a.

A threshold value thr_f of a feature also divides the time line into two subspaces: normal and abnormal. The first subspace includes a true negative part T_n and a false negative part T_{FN}, where an abnormal period is erroneously considered as normal. The second subspace includes a true positive part T_a and a false positive part T_{FP}, where a normal period is considered abnormal. The ticketing rate ratio determined by the threshold thr_f can be computed as follows:

$$D(T_n, T_{FP}, T_{FN}, T_a) = \frac{\frac{\lambda_n T_{FP} + (\lambda_a + \lambda_n) T_a}{T_a + T_{FP}}}{\frac{\lambda_n T_n + (\lambda_a + \lambda_n) T_{FN}}{T_n + T_{FN}}}$$

Both the numerator and denominator can be regarded as a weighted average of λ_n and $\lambda_a + \lambda_n$, with the time period lengths as the weights. Because $\lambda_a + \lambda_n > \lambda_n$ always holds, we can show that the derivatives of D over the false positives T_{FP} and the false negatives T_{FN} are non-increasing:

$$\frac{\partial D}{\partial T_{FP}} < 0 \text{ and } \frac{\partial D}{\partial T_{FN}} < 0$$

Therefore, because T_{FP} and T_{FN} are non-negative, the ticket rate ratio is maximized when both false positives and false negatives are zero:

$$D_{max} = \lim_{\substack{T_{FP} \to 0 \\ T_{FN} \to 0}} D = \frac{\lambda_a}{\lambda_n} + 1$$

Features	Ticketing Rate Ratio
snr-var-2	14.49
uncorrected-var-1	7.66
rxpower-wma-diff-4	5.31
t3timeouts-wma-diff-1	4.93
t4timeouts-var-1	4.18

Table 3: **Top 5 features and their ticketing rate ratio.**

4.3 Feature Selection

We have a total of more than three hundred time-series features and it is unlikely they are all useful indicators of network faults. To find the relevant features, we only select the features with high ticketing rate ratios from each PNM metric. Specifically, among the same type of features derived from a PNM metric with different hyperparameters, we choose the one with the highest ticketing rate ratio as the representative feature. For each representative feature derived from the same PNM value, we choose the top two with the highest ticketing rate ratios. Finally, among the remaining candidates, we choose the top N features that have the highest ticketing rate ratios. We determine the number of features N based on the desired ticketing rate ratios, ticket prediction accuracy, and ticket coverage as we soon describe in § 5.1.

Table 3 shows the top five features we used and their ticketing rate ratios calculated from our training sets (Section 5.2). The name of each feature consists of the raw PNM metric, the statistical model we apply to the metric, and the parameter. For example, the *snr-var-2* means the variance of SNR with a *2-days* window size. We note that all features have a high ticketing rate ratio and we expect them to effectively detect network faults.

4.4 Combining Different Features

Different PNM features may detect different types of network faults. Therefore, we build the final classifier by combining the detection results of all selected features. As long as one selected feature considers a PNM collection period abnormal, we classify the collection period as abnormal. For each

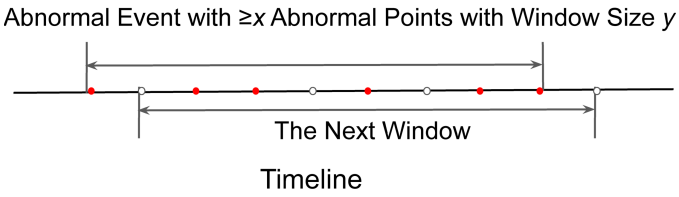


Figure 4: **This figure explains the sliding window algorithm. When the number of abnormal points within a sliding window exceeds a threshold, the window is considered to be abnormal. An abnormal event is given by merging the abnormal windows.**

selected feature, we have already chosen a threshold that maximizes the ticketing rate ratio. Therefore, we expect that combining the results of all selected features will also provide a high ticketing rate ratio. We evaluate the results of our classifier in § 5.1.

4.5 ISP Deployment

An ISP can use CableMon in two ways: proactive network maintenance for predicted trouble tickets and diagnosing the root cause of a trouble ticket when receiving a call. In this section, we describe the algorithms for an ISP to decide when to send out a repair technician proactively and how to diagnose the root cause.

CableMon's classifier can monitor an ISP's network continuously. It can output a normal and abnormal decision when a PNM record is collected from a customer's modem. However, due to the existence of noise and the intermittent nature of some faults, if an ISP makes a dispatch decision whenever it observes an abnormal PNM data point, it may lead to many false positives. To address this problem, we design a sliding window algorithm for an ISP to make a dispatch decision. The high-level idea of this algorithm is that an ISP should only dispatch a technician after a fault persists.

Figure 4 explains this algorithm. The algorithm takes two parameters: y and x, where y is the size of the window, and x is the number of abnormal data points detected in the window. When an ISP collects a new PNM record, it looks back to a window size y of collection points. If x out of y data points are considered as abnormal, then the ISP should dispatch a technician to examine and repair the network.

An ISP can determine the parameters x and y based on the false positives and false negatives it is willing to tolerate. The ISP can estimate the values of false positives and false negatives from its historic PNM data and ticket data. Therefore, choosing those parameters only requires an ISP to train CableMon using its own PNM and ticket data and does not require tuning. In § 5.1, we use our datasets to show how an ISP can effectively choose the parameters x and y.

Similarly, an ISP can use CableMon to help diagnose the root cause of a call. When it receives a trouble call, if the customer complains about a performance problem, and the ISP sees that in the past collection window of size y, there exists x abnormal collection points, the ISP can conclude that the trouble is likely to be caused by a network problem.

5 Evaluation

In this section, we describe how we evaluate CableMon's performance.

5.1 Establishing Evaluation Metrics

Ideally, we would like to deploy CableMon on a real cable ISP and measure how it reduces the number of trouble tickets over a long term. It is our future work to conduct such a real-world experiment. In this work, we aim to estimate how many trouble tickets CableMon would reduce were it deployed on our collaborating ISP.

To do so, we emulate the sliding window algorithm described in § 4.5 using our test dataset. We start from the beginning of the test dataset. If there are x abnormal points detected in a window size of y, we mark it as the beginning of a fault. We then move the window to the next data point. When the number of abnormal points falls below x, we mark it as the end of a fault. If there is a trouble ticket occurred during a fault, we consider this fault detection as a true fault. We note that if we detect a fault simultaneously within multiple customers, as long as one customer reports a ticket, we consider it a true fault. We assume that if an ISP dispatched a repair technician when it detected the onset of the fault, it could have avoided the trouble ticket. We define *ticket prediction accuracy* as the number of true faults divided by the total number of detected faults. We define *ticket coverage* as the number of trouble tickets occurred during a detected fault divided by the total number of network-related trouble tickets.

It is not sufficient to use only ticket prediction accuracy and ticket coverage to gauge CableMon's performance. This is because if CableMon detects the entire time period that spans the test dataset as a faulty period, it will achieve 100% ticket coverage and ticket prediction accuracy. To avoid this pitfall, we also use the normalized ticketing rate, which is defined as the ticketing rate in all faculty periods normalized by the ticketing rate of the time period that spans the test dataset. If CableMon erroneously detects the entire time period as abnormal, it will achieve a low normalized ticketing rate close to 1.

How an ISP chooses the sliding window parameters: In practice, an ISP can use a training set to determine the threshold values of CableMon's classifier. It can use the ticket prediction accuracy, ticket coverage, and the normalized ticketing rate obtained from a validation set to choose the combination of the sliding window parameters.

We show an example in Figure 5. In this example, we choose a window size of 12 data points (y = 12), which is roughly two days long. We then measure the ticket prediction accuracy, ticket coverage, and the normalized ticketing rate

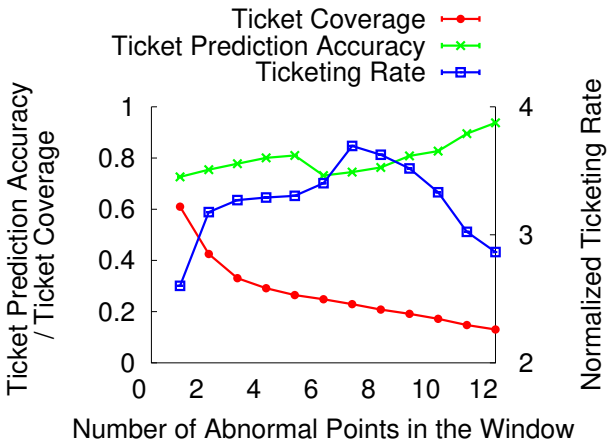


Figure 5: **This figure shows the ticket detection accuracy, the ticket coverage, and the normalized ticketing rate of the sliding window algorithm with different parameters.**

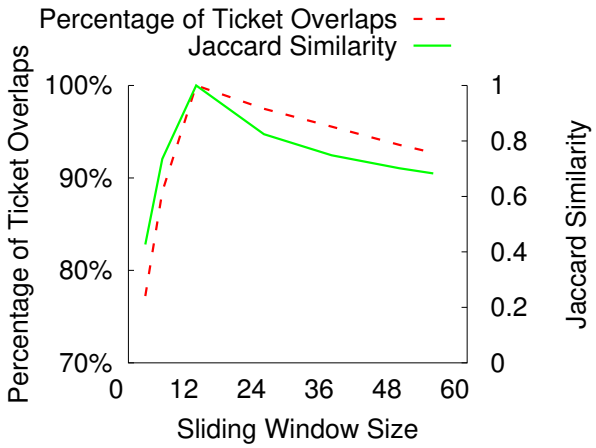


Figure 6: **This figure shows what percentage of tickets detected by different window sizes overlap with those detected by a window size of 12 and the Jaccard similarity between the faulty periods detected by different window sizes and those detected by a window size of 12.**

when the number of abnormal points x varies from 0 to 12. As can be seen, when x is around 8, the sliding window algorithm achieves a high normalized ticketing rate, a relatively high ticket prediction accuracy 80%, and a ticket coverage around 20%. Since avoiding false dispatches is more important than predicting all trouble tickets, an ISP can choose (8, 12) as its sliding window parameters for fault detection.

We have tried different sizes of the sliding window, ranging from one to 60 data points. For each window size, we use the above method to choose the parameter x such that both the ticket prediction accuracy and the normalized ticketing rate are high, and the ticket coverage is above a minimum threshold 15%. We compare the tickets and the faulty periods detected by different window parameters. We use the Jaccard similarity metric [25] to measure the overlaps of faulty periods detected by different window parameters. As can be seen in Figure 6, 90% of the tickets detected by windows larger than 12 overlap; and the faulty periods detected by them have a Jaccard similarity larger than 60%. This result suggests that different window parameters are likely to detect the same sets of faults, and the performance of CableMon is not sensitive to the window parameters.

5.2 Experiment Setup

After we establish the evaluation metric, we train and evaluate CableMon on a 50-machine Linux cluster with 40 $\sim$ 512 GB RAM and 8 $\sim$ 48 core running Ubuntu 18.04. CableMon is trained on five-month data from 01/06/2019 to 05/30/2019 and tested with three-month data from 06/01/2019 to 08/31/2019.

Comparing with the existing work: We compare CableMon's performance with two existing methods. One is from our collaborating ISP, AnonISP, which uses a visualization tool that colors different ranges of PNM values for an operator to manually monitor its networks' conditions. AnonISP's tool has two manually configured thresholds for several raw PNM values and therefore has three fault indication levels: normal (green), marginal (yellow), and abnormal (red). We compare AnonISP's tool against CableMon with these thresholds and regard both yellow and red levels as network fault, as the ISP's experts usually do.

Another tool from industry uses Comcast's scoreboard method [22]. Comcast is considered as the leading company in the area of PNM research. They developed a method that compares a PNM metric to a threshold and assigns a score to each comparison result. If the sum of the comparison scores exceeds a threshold value, then the method considers there is a fault in the network.

Since both AnonISP and Comcast's tool detects a fault using a single PNM data record, we apply the sliding window algorithm to both tools for a fair comparison.

Comparing with Machine Learning Techniques: We also compare the performance of CableMon with three classical machine learning algorithms: Decision Tree (DT) [34], Random Forest (RF) [4] and Support Vector Machine (SVM) [14]. Since these algorithms require labeled data, we label the PNM data with tickets. Each ticket has a creation time and a closed time. We label the PNM data collected between this time interval as positive samples and other data as negative samples. We generate 47,518 positive samples and the same number of abnormal samples as our training set to train the machine learning models and evaluate them with the same evaluation metrics.

Table 4 shows the ticket prediction accuracy, the ticket coverage, and the normalized ticketing rate of different methods. As can be seen, CableMon achieves the highest ticket prediction accuracy and the highest normalized ticketing rate

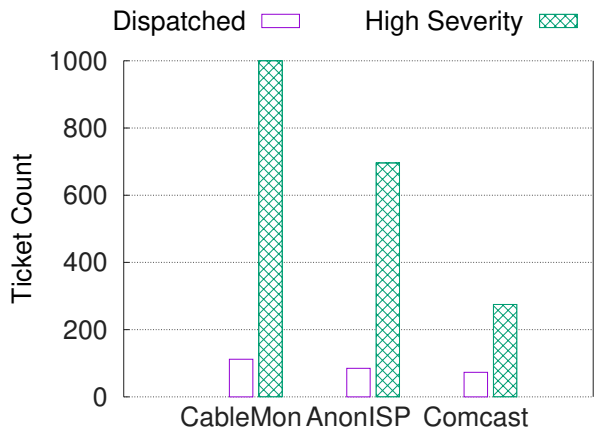


Figure 7: **This figure shows the number of different types of tickets detected by different methods.**

among all methods. Its ticket coverage is lower than that of AnonISP. However, this is because AnonISP detects too many false faults, as shown by its low ticket prediction accuracy. We note that all three machine learning algorithms require a long training time, as each has multiple parameters to tune. The results we present here are the best ones after many rounds of tuning. When we started this project, we started with those algorithms, but abandoned them due to the challenges to tune them and to explain the results when certain parameters are changed.

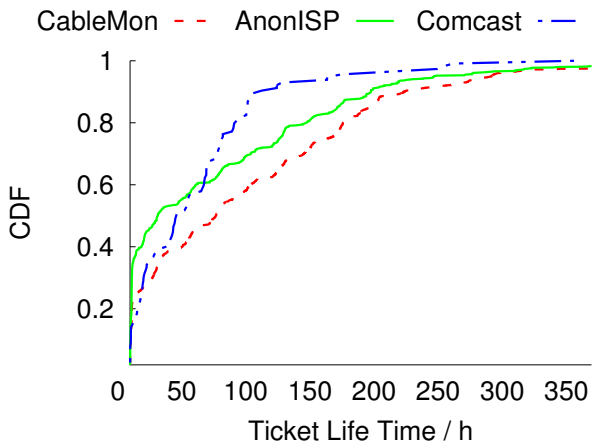
Methods	Ticket Prediction Accuracy	Ticket Coverage	Normalized Ticketing Rate
CableMon	81.92%	22.99%	3.55
Decision Tree	68.93%	15.53%	2.52
SVM	75.64%	12.54%	2.02
Random Forest	73.14%	14.21%	2.24
Comcast	23.48%	2.21%	1.18
AnonISP's tool	10.04%	25.13%	0.98

Table 4: **Performance of different methods**

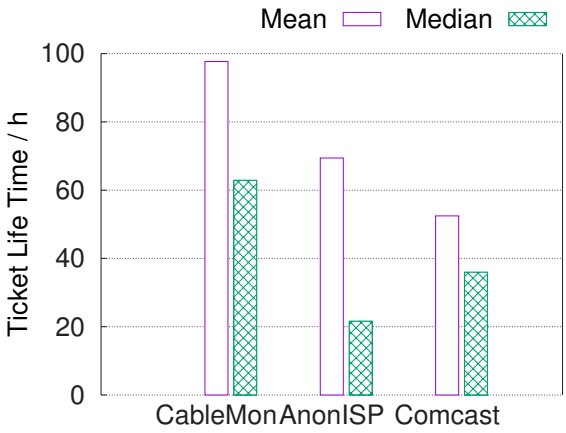
5.3 Detected Tickets Statistics

To further analyze the detected tickets, we examine the tickets detected by CableMon and existing ISP tools according to the ticket action and description fields. We omit the results of the machine learning algorithms for clarity. The characteristics of the tickets detected by those algorithms are similar to those of CableMon, but they have lower ticket detection accuracy and coverage. Figure 7 shows the number of different types of tickets detected by different methods. As can be seen, CableMon can detect a significantly more number of dispatched and high severity tickets than the two existing ISP tools.

Figure 8(a) shows the distribution of a detected ticket's life time, and figure 8(b) shows the average and median life time



(a) CDF



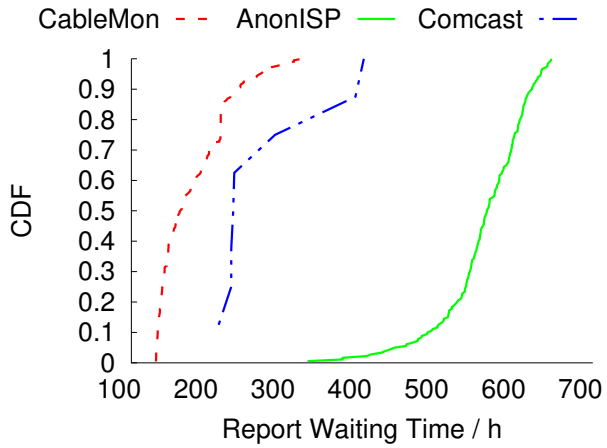
(b) Mean and Median

Figure 8: **The figures show the CDF, mean, and median of the life time of tickets predicted by different methods. A longer ticket life time indicates that the problem that triggered the ticket takes a longer time to fix.**

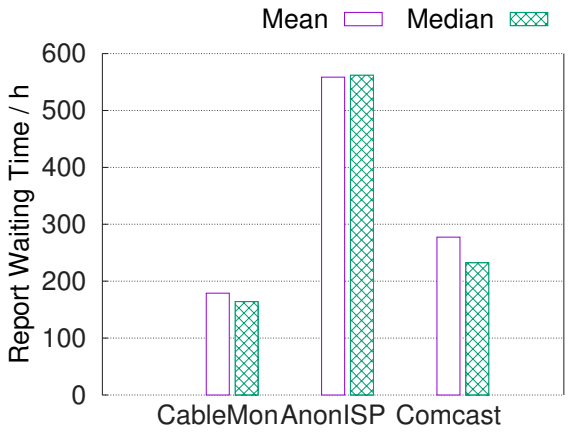
of a detected ticket. A ticket's life time is defined as the time between a ticket is created to the time a ticket is closed. As can be seen, the tickets detected by CableMon have longer life times, suggesting that CableMon detects the problems that take longer to resolve.

We also measure the time elapsed from when a fault is detected to when a ticket is created. We refer to this time as "Report Waiting Time." Figure 9(a) shows the cumulative distribution of the report waiting time of different methods, and figure 9(b) shows the average and median report waiting time of different methods. As can be seen, CableMon's report waiting time is also significantly shorter that that of other methods, indicating that its detected faults lead to customer trouble tickets faster than those detected by other methods.

Finally, we measure the distribution of a fault detected by different methods. Figure 10 shows the PDF of the length of a fault detected by different methods. As can be seen,



(a) CDF



(b) Mean and Median

Figure 9: **The figures show the CDF, mean, and median of the report waiting time of tickets predicted by different methods. A shorter report waiting time indicates that the problem triggered by the ticket is more urgent.**

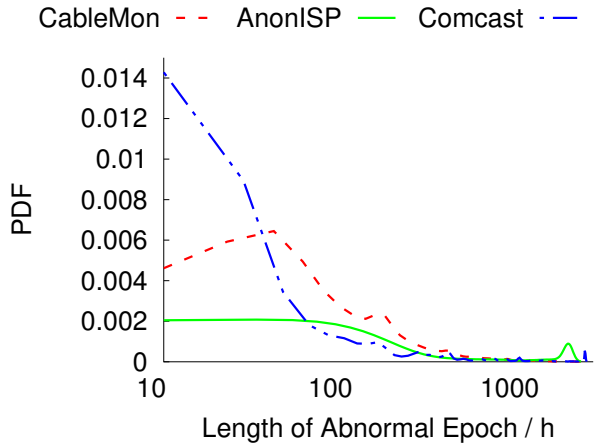


Figure 10: **This figure shows the PDF of the length of a detected fault.**

CableMon detected faults tend to last a moderate period of time. The highest probability density is slightly less than 100 hours (roughly four days). Comcast's tool detects many faults that last less than one day, shorter than what a typical network repair action takes. This result suggests that many of the detected faults could be false positives. The faults detected by AnonISP's tool have a wide range of life span, from very short faults to very long faults (>500 hours), which are outside the normal range of repair actions. Again, this result suggests that many of the detected faults could be false positives.

6 Related Works

Previous work measured the reliability of broadband networks. The Federal Communications Commission launched the Measuring Broadband America (MBA) project [10] since 2010. Bischof et al. [3] showed that poor reliability will heavily affect user traffic demand. Padmanabhan et al. [28] demonstrated that the outages of broadband networks tend to happen under bad weather conditions. Baltrunas et al. [2] also measured the reliability of mobile broadband networks.

Network fault diagnosis has attracted much attention from the community for a long time. Many approaches from industry, especially the cable industry, set manual thresholds for certain measured metrics to detect network outages. Amazon [11] used a fixed threshold to monitor the condition for its cloud services. Zhuo et al. [41] treated packet loss as a fault indicator and showed the correlation between Tx/Rx Power and packet loss rate. They again use manually set thresholds to detect network faults. Lakhina et al. [21] proposed the first framework that applied Principal Component Analysis (PCA) to reduce the dimension of network traffic metrics. Huang et al. [16] showed that Lakhina's framework works well with a limited number of network devices, but has performance issues on larger datasets. Moreover, Ringberg et al. [35] pointed out that using PCA for traffic anomaly detection is much more tricky than it appears. Besides PCA, many other statistical models are applied to network anomaly detection. Gu et al. [13] measured the relative entropy of the metrics and compared them to the baseline. Subspace [26] is introduced to deal with high-dimensional and noisy network monitoring data. Kai et al. [19] used Expectation–Maximization (EM) algorithm to estimate the parameters of their model and obtain the upper or lower bound of the common metric values. Independent Component Analysis [30], Markov Modulated Process [32], and Recurrence Quantification Analysis [29] are also introduced to find the anomaly points in time series data. These methods aim to detect sudden changes in data. Differently, CableMon uses customer ticket as hints to label the input data and uses the ticketing rate ratio to select relevant features.

Recently, machine learning has been used for network anomaly detection. Leung et al. [24] designed a networking anomaly detection system using a density-based clustering

algorithm, which obtained the accuracy as 97.3%. Dean et al. [9] presented an Unsupervised Behavior Learning framework based on the clustering algorithm. However, cluster-based approaches do not work well with sparse data, which is the case of our PNM data where abnormal events are rare. Sung et al. [37] deployed Support Vector Machines (SVMs) to estimate the actual crucial features. According to our evaluation, SVMs do not perform as well as CableMon. Liu et al. [27] adopted more than twenty statistics models to obtain more features from the original data. They used all generated features in Random Forest and achieved high accuracy and effectiveness. However, they still require manual labeling to train the Random Forest model. PreFix [39] predicts switch failures with high precision and recall. However, it also requires significant manual efforts for labeling, while our work does not. Pan et al. [31] also used the tickets as the hints to select potential network faults. However, they still asked experts to manually label network faults and use this labelled data to train a Decision Tree model. In contrast, CableMon does not use any manual label.

Previous researches have also focused on processing customer report tickets. LOTUS [38] deploys Natural Language Processing (NLP) techniques to understand the tickets. Potharaju et al. [33] built a system that automatically processes the raw text of tickets to infer the networking faults and find out the resolution actions. Jin et al. [18] studied the tickets in cellular networks and categorized the types of customer trouble tickets. Chen et al. [8] and Hsu et al. [15] use both customer trouble tickets and social media postings to determine network outages. This work combines an ISP's customer trouble tickets and PNM data to infer network faults.

7 Discussion

CableMon uses customer trouble tickets as network fault indicators to build a classifier without manual labeling. We plan to focus on the following directions to improve the performance of CableMon:

- When there lacks a large set of labeled data, semi-supervised learning [40] combines a small set of labeled data and a large set of unlabeled data to improve classification accuracy. We plan to investigate whether semi-supervised learning approach as well as other machine learning methods such as deep learning can improve the performance of CableMon.

- Presently, we use network-related tickets to train the classifier. We have discovered that customers tend to report tickets at weekdays rather than on weekends and during the day rather than at night. From this pattern, one may infer that if a customer reports a ticket at an "atypical" time, it is more likely to indicate a customer-impacting problem. If we place a higher weight for such "outlier" tickets in a classification algorithm, we may increase both the ticket prediction accuracy and coverage.

- ISPs desire to differentiate failures that affect a group of customers from those that affect a single customer. We refer to faults that affect multiple customers as "maintenance issues." If there is a maintenance issue, it is also desirable to locate the place where this issue has happened. It is possible to infer maintenance issues by clustering customers' PNM data, and to infer the location of a maintenance issue by combining the geographical location of each modem with the topology of HFC network. It is our future work to study these problems.

- When detecting network faults, CableMon outputs whether there is an abnormal event and how long it exists. It is desirable to rank the severity of abnormal events so that an ISP can prioritize its repair actions. It is our future work to explore such ranking algorithms.

8 Conclusion

Cable broadband networks are widely deployed all around U.S. and serve millions of U.S. households. However, cable networks have poor reliability. Although the cable industry has developed a proactive network maintenance (PNM) platform to address this issue, cable ISPs have not fully utilized the collected data to proactively detect and fix network faults. Existing approaches rely on instantaneous PNM metrics with manually set thresholds for fault detection and can introduce an unacceptably high false positive rate. We design CableMon, a system that learns the fault detection criteria from customer trouble tickets. Our design overcomes the noise from both PNM data and customer trouble tickets and achieves a nearly four times higher ticket prediction accuracy than the existing tools in the public domain. This is a first step toward enabling an ISP to use PNM data to proactively repair a failure before a customer calls and to diagnose whether a customer trouble call is caused by a network fault.

Acknowledgment

The authors would like to thank the NSDI Shepherd Vyas Sekar and anonymous reviewers for their valuable feedback. This work is supported in part by an NSF award CNS-1910867.

References

[1] History of Cable. https://www.calcable.org/learn/history-of-cable/, 2018.

[2] Dziugas Baltrunas, Ahmed Elmokashfi, and Amund Kvalbein. Measuring the Reliability of Mobile Broadband Networks. In *ACM IMC*, 2014.

[3] Zachary S Bischof, Fabian E Bustamante, and Nick Feamster. Characterizing and Improving the Reliability of Broadband Internet Access. In *46th Research Conference on Communication, Information and Internet Policy (TPRC)*, 2018.

[4] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[5] CableLabs. Data Over Cable Service Interface Specifications DOCSIS 3.0 - Operations Support System Interface Specification. 2007.

[6] CableLabs. Cable Broadband Technology Gigabit Evolution. https://www.cablelabs.com/insights/cable-broadband-technology-gigabit-evolution/, 2016.

[7] DOCSIS CableLabs. Best Practices and Guidelines, PNM Best Practices: HFC Networks (DOCSIS 3.0). Technical report, CM-GL-PNMP-V03-160725, 2016.

[8] Yi-Chao Chen, Gene Moo Lee, Nick Duffield, Lili Qiu, and Jia Wang. Event Detection Using Customer Care Calls. In *IEEE INFOCOM*, 2013.

[9] Daniel Joseph Dean, Hiep Nguyen, and Xiaohui Gu. UBL: Unsupervised Behavior Learning for Predicting Performance Anomalies in Virtualized Cloud Systems. In *ACM International Conference on Autonomic Computing*, 2012.

[10] Federal Communications Commission (FCC). In the Matter of Reliability and Continuity of Communication Networks. PS Docket 11-60, 2011.

[11] Filippo Lorenzo Ferraris, Davide Franceschelli, Mario Pio Gioiosa, Donato Lucia, Danilo Ardagna, Elisabetta Di Nitto, and Tabassum Sharif. Evaluating the Auto Scaling Performance of Flexiscale and Amazon EC2 Clouds. In *IEEE International Symposium on Symbolic and Numeric Algorithms for Scientific Computing*, 2012.

[12] Sarthak Grover, Mi Seon Park, Srikanth Sundaresan, Sam Burnett, Hyojoon Kim, Bharath Ravi, and Nick Feamster. Peeking Behind the NAT: An Empirical Study of Home Networks. In *ACM IMC*, 2013.

[13] Yu Gu, Andrew McCallum, and Don Towsley. Detecting Anomalies in Network Traffic Using Maximum Entropy Estimation. In *ACM IMC*, 2005.

[14] Marti A. Hearst, Susan T Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf. Support Vector Machines. *IEEE Intelligent Systems and their applications*, 13(4), 1998.

[15] Wenling Hsu, Guy Jacobsen, Yu Jin, and Ann Skudlark. Using Social Media Data to Understand Mobile Customer Experience and Behavior. In *European Regional Conference of the International Telecommunications Society*, 2011.

[16] Ling Huang, XuanLong Nguyen, Minos Garofalakis, Michael I Jordan, Anthony Joseph, and Nina Taft. In-Network PCA and Anomaly Detection. In *Advances in Neural Information Processing Systems*, 2007.

[17] David Hunter and Tom Williams. Improved Customer Service Through Intermittent Detection. In *SCTE Cable-Tec Expo*, 2015.

[18] Yu Jin, Nick Duffield, Alexandre Gerber, Patrick Haffner, Wen-Ling Hsu, Guy Jacobson, Subhabrata Sen, Shobha Venkataraman, and Zhi-Li Zhang. Making Sense of Customer Tickets in Cellular Networks. In *IEEE INFOCOM*, 2011.

[19] Huang Kai, Qi Zhengwei, and Liu Bo. Network Anomaly Detection Based on Statistical Approach and Time Series Analysis. In *IEEE International Conference on Advanced Information Networking and Applications Workshops*, 2009.

[20] D Richard Kuhn. Sources of Failure in the Public Switched Telephone Network. *Computer*, 30(4):31–36, 1997.

[21] Anukool Lakhina, Mark Crovella, and Christophe Diot. Diagnosing Network-Wide Traffic Anomalies. In *ACM SIGCOMM Computer Communication Review*, 2004.

[22] Bryan Thomas Larry Wolcott, John Heslip and Robert Gonsalves. A Comprehensive Case Study of Proactive Network Maintenance. In *SCTE Cable-Tec Expo*, 2016.

[23] William Lehr, Mikko Heikkinen, David Clark, and Steven Bauer. Assessing Broadband Reliability: Measurement and Policy Challenges. *Research Conference on Communication, Information and Internet Policy (TPRC)*, 2011.

[24] Kingsly Leung and Christopher Leckie. Unsupervised Anomaly Detection in Network Intrusion Detection Using Clusters. In *Twenty-Eighth Australasian Computer Science Conference*, 2005.

[25] Michael Levandowsky and David Winter. Distance Between Sets. *Nature*, 234(5323):34, 1971.

[26] Xin Li, Fang Bian, Mark Crovella, Christophe Diot, Ramesh Govindan, Gianluca Iannaccone, and Anukool Lakhina. Detection and Identification of Network Anomalies Using Sketch Subspaces. In *ACM IMC*, 2006.

[27] Dapeng Liu, Youjian Zhao, Haowen Xu, Yongqian Sun, Dan Pei, Jiao Luo, Xiaowei Jing, and Mei Feng. Opprentice: Towards Practical and Automatic Anomaly Detection through Machine Learning. In *ACM IMC*, 2015.

[28] Ramakrishna Padmanabhan, Aaron Schulman, Dave Levin, and Neil Spring. Residential links under the weather. In *ACM SIGCOMM*. 2019.

[29] Francesco Palmieri and Ugo Fiore. Network Anomaly Detection Through Nonlinear Analysis. *Computers & Security*, 29(7):737–755, 2010.

[30] Francesco Palmieri, Ugo Fiore, and Aniello Castiglione. A Distributed Approach to Network Anomaly Detection Based on Independent Component Analysis. *Concurrency and Computation: Practice and Experience*, 26(5):1113–1129, 2014.

[31] Lujia Pan, Jianfeng Zhang, Patrick PC Lee, Hong Cheng, Cheng He, Caifeng He, and Keli Zhang. An Intelligent Customer Care Assistant System for Large-Scale Cellular Network Diagnosis. In *ACM International Conference on Knowledge Discovery and Data Mining*, 2017.

[32] Ioannis Ch Paschalidis and Georgios Smaragdakis. Spatio-Temporal Network Anomaly Detection by Assessing Deviations of Empirical Measures. *IEEE/ACM Transactions on Networking (TON)*, 17(3):685–697, 2009.

[33] Rahul Potharaju, Navendu Jain, and Cristina Nita-Rotaru. Juggling the Jigsaw: Towards Automated Problem Inference from Network Trouble Tickets. In *USENIX/ACM NSDI*, 2013.

[34] J Ross Quinlan. *C4. 5: Programs for Machine Learning*. Elsevier, 2014.

[35] Haakon Ringberg, Augustin Soule, Jennifer Rexford, and Christophe Diot. Sensitivity of PCA for Traffic Anomaly Detection. In *ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, 2007.

[36] W Sawyer. Management Information Base for Data Over Cable Service Interface Specification (DOCSIS) Cable Modem Termination Systems for Subscriber Management. *RFC 4036*, 2005.

[37] Andrew H Sung and Srinivas Mukkamala. Identifying Important Features for Intrusion Detection using Support Vector Machines and Neural Networks. In *IEEE Symposium on Applications and the Internet.*, 2003.

[38] Shobha Venkataraman and Jia Wang. Assessing the Impact of Network Events with User Feedback. In *Proceedings of the 2018 Workshop on Network Meets AI & ML*, 2018.

[39] Shenglin Zhang, Ying Liu, Weibin Meng, Zhiling Luo, Jiahao Bu, Sen Yang, Peixian Liang, Dan Pei, Jun Xu, Yuzhi Zhang, Yu Chen, Hui Dong, Xianping Qu, and Lei Song. Prefix: Switch Failure Prediction in Datacenter Networks. In *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 2018.

[40] Xiaojin Zhu and Andrew B Goldberg. Introduction to semi-supervised learning. *Synthesis lectures on artificial intelligence and machine learning*, 3(1):1–130, 2009.

[41] Danyang Zhuo, Monia Ghobadi, Ratul Mahajan, Klaus-Tycho Förster, Arvind Krishnamurthy, and Thomas Anderson. Understanding and mitigating packet corruption in data center networks. In *ACM SIGCOMM*, 2017.

A Pseudo-code of Determining Thresholds

Algorithm 1 Threshold Determining

$\mathcal{P}_{label} \leftarrow$ Set of all data points associated with tickets
$\mathcal{P} \leftarrow$ Set of all data points
if one threshold **then**
    $\mathcal{V} \leftarrow$ Set of all distinct values
    for each $v \in \mathcal{V}$ **do**
        $N_l \leftarrow \{p | p \in \mathcal{P}_{label}, p.value \leq v\}$
        $T_l \leftarrow \sum_{p \in \mathcal{P}, p.value \leq v} p.time$
        $N_r \leftarrow \{p | p \in \mathcal{P}_{label}, p.value > v\}$
        $T_r \leftarrow \sum_{p \in \mathcal{P}, p.value > v} p.time$
        $TRR_v \leftarrow \max(\frac{|N_l|/T_l}{|N_r|/T_r}, \frac{|N_r|/T_r}{|N_l|/T_l})$
        $thr_m \leftarrow \arg\max TRR_v$
    return thr_m
else
    $\mathcal{A} \leftarrow$ Sorted array of all data points
    $\mathcal{B} \leftarrow$ Binning $\mathcal{A}$
    $\mathcal{V} \leftarrow$ Set of maximum value in each bin $b \in \mathcal{B}$
    for each $v_l \in \mathcal{V}$ **do**
        for each $v_r \in \mathcal{V}$ **do**
            $N_n \leftarrow \{p | p \in \mathcal{P}_{label}, v_l \leq p.value \leq v_r\}$
            $T_n \leftarrow \sum_{p \in \mathcal{P}, v_l \leq p.value \leq v_r} p.time$
            $N_a \leftarrow \{p | p \in \mathcal{P}_{label}, p \notin N_n\}$
            $T_a \leftarrow \sum_{p \in \mathcal{P}, p.value > v_r | p.value < v_l} p.time$
            $TRR_{(v_l, v_r)} \leftarrow \frac{|N_a|/T_a}{|N_n|/T_n}$
            $thr_l, thr_r \leftarrow \arg\max TRR_{(v_l, v_r)}$
    return thr_l, thr_r