Proactive Microwave Link Anomaly Detection in Cellular Data Networks

Lujia Pan[a,b], Jianfeng Zhang[b], Patrick P. C. Lee[c,*], Marcus Kalander[b], Junjian Ye[b], Pinghui Wang[a]

[a]*Xi'an Jiaotong University, Xi'an, China*
[b]*Noah's Ark Lab, Huawei Technologies, Shenzhen, China*
[c]*Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong*

Abstract

Microwave links are widely used in cellular networks for large-scale data transmission. From the network operators' perspective, it is critical to quickly and accurately detect microwave link failures before they actually happen, thereby maintaining the robustness of the data transmissions. We present PMADS, a machine-learning-based proactive microwave link anomaly detection system that exploits both performance data and network topological information to detect microwave link anomalies that may eventually lead to actual failures. Our key observation is that anomalous links often share similar network topological properties, thereby enabling us to further improve the detection accuracy. To this end, PMADS adopts a network-embedding-based approach to encode topological information into features. It further adopts a novel active learning algorithm, ADAL, to continuously update the detection model at low cost by first applying unsupervised learning to separate anomalies as outliers from the training set. We evaluate PMADS on a real-world dataset collected from 2,142 microwave links in a production LTE network during a one-month period, and show that PMADS achieves a precision of 94.4% and a recall of 87.1%. Furthermore, using the active learning feedback loop, only 7% of the training data is required to achieve comparable results. PMADS is currently deployed in a metropolitan LTE network that serves around four million subscribers in a Middle Eastern country.

Keywords: Anomaly detection; Network embedding; Active learning

1. Introduction

Microwave links are widely used in cellular data networks for high-speed Internet access (e.g., in 60% of mobile backhaul connections worldwide) [1, 7, 12]. Unlike

[*]Corresponding author: Patrick P. C. Lee.

Email addresses: `panlujia@huawei.com` (Lujia Pan), `zhangjianfeng3@huawei.com` (Jianfeng Zhang), `pclee@cse.cuhk.edu.hk` (Patrick P. C. Lee), `marcus.kalander@huawei.com` (Marcus Kalander), `yejunjian@huawei.com` (Junjian Ye), `wangpinghui369@gmail.com` (Pinghui Wang)

traditional fiber cable connections, microwave links can move large amounts of information at high speeds over the wireless medium. It is also easier to install microwave links than fiber cables, and the total cost of ownership is hence significantly reduced.

However, microwave link failures in cellular data networks are prevalent in practice. Various factors, such as large obstacles (e.g., buildings), background noise (e.g., electromagnetic interference), or environmental factors (e.g., heavy moisture in the atmosphere, snow, rain, and fog), can degrade the performance of microwave transmissions [1]. Such performance degradations, if severe, can further lead to microwave link failures. In current practice, network operators cannot detect microwave link failures in advance; instead, they rely on critical alarms triggered by monitoring systems or complaints filed by subscribers to identify the presence of microwave link failures. However, such failures have already disturbed the regular cellular data services and hence the perceived experience of subscribers.

Instead of dealing with microwave link failures after they occur, it is critical for network operators to *proactively* identify microwave link failures in advance by detecting any microwave links that show anomalous behavior, so as to quickly intervene and repair any imminent microwave link failures. Machine learning techniques have been widely used for anomaly detection; in particular, some studies have designed anomaly detection solutions for cellular networks [3, 5, 6, 38, 44, 49] and data centers [16, 36, 47].

There are well-known challenges for machine-learning-based anomaly detection: (i) anomalous samples and normal samples are severely imbalanced, which poses a significant challenge for achieving high precision and high recall; (ii) obtaining ground-truth labels requires major manual efforts and incurs excessive costs, thereby limiting the amounts of available labeled data (known as the "labeled data scarcity" problem); and (iii) anomaly detection may become less accurate after a long period of time due to time-varying data distribution, thereby requiring machine learning models to adapt to changes quickly and accurately. In particular, in the context of cellular data networks, many network anomaly detection or diagnosis studies rely on streaming data generated from a production network [4, 21, 30]. However, some inherent network characteristics, such as the topological information of node relationships, should also be taken into account in anomaly detection.

We design and implement PMADS, a Proactive Microwave link Anomaly Detection System, a machine-learning-based solution that aims to achieve highly accurate anomaly detection specifically for microwave link transmissions in cellular data networks. The main novelty of PMADS is that it takes into account both the offline network topological information and online performance data collected from the production network for feature engineering. Specifically, we apply a *network embedding* algorithm [8] to encode the topological information and generate network-specific features. Also, to better cope with the scarcity of labeled data as well as adapt to the time-varying data distribution, we leverage *active learning* [34] and propose a new active learning mechanism for PMADS to continuously update the existing machine learning model at low cost based on the input of new samples obtained from the interactions with experts. PMADS achieves both high precision and high recall in anomaly detection. It is now deployed in a production 4G LTE (fourth-generation long-term evolution) network that serves around four million subscribers in a Middle Eastern country. Our contributions can be summarized as follows:

- We first motivate our study by analyzing a real-world dataset collected from a production LTE network that serves around four million subscribers in a Middle Eastern country. Our dataset covers the performance data from thousands of nodes and links over a one-month period. We analyze the proximities among microwave links, and show that the anomalies tend to be clustered in a network. Our analysis provides insights into our anomaly detection design.

- We propose PMADS, a microwave link anomaly detection system. In addition to using traditional statistical features, PMADS encodes network topological information into additional meaningful features via network embedding.

- We present and motivate a novel $\underline{\text{A}}$nomaly $\underline{\text{D}}$etection by $\underline{\text{A}}$ctive $\underline{\text{L}}$earning algorithm, ADAL, as a core component of PMADS, to continuously update the model of PMADS at low cost by injecting new expert experience. To mitigate the manual efforts on labeling, our idea is to employ an outlier detection algorithm based on unsupervised learning and separate anomalies as outliers from the training set. This enables domain experts to focus on the smaller outlier set to label samples, thereby reducing labeling overhead.

- We conduct trace-driven evaluation on PMADS. We show that PMADS achieves a precision of 94.4% and a recall of 87.1% when using the topological features encoded via network embedding. Its accuracy is significantly higher than the baseline approach without network embedding. We also compare and evaluate several embedding techniques to understand their impact on the detection accuracy. Furthermore, we show that combined with active learning, PMADS achieves comparable accuracy with only 7% of training data while supporting very fast query performance (e.g., in tens of seconds).

Our dataset (whose sensitive information is anonymized) for this paper is available for download at: `http://adslab.cse.cuhk.edu.hk/software/pmads`.

The rest of the paper proceeds as follows. In Section 2, we present the background of microwave link transmissions in cellular data networks and the analysis of a real-world dataset. In Section 3, we present our design of PMADS. In Section 4, we evaluate the accuracy of PMADS and the active learning algorithm ADAL. In Section 5, we summarize our lessons learned from the deployment. Finally, we review related work in Section 6, and conclude in Section 7.

2. Preliminaries

In this section, we first give an overview of a microwave architecture in an LTE network. We then describe how we collect and preprocess data from a production network for our analysis. Finally, we present a high-level analysis on the dataset and show that microwave link anomaly detection is indeed necessary.

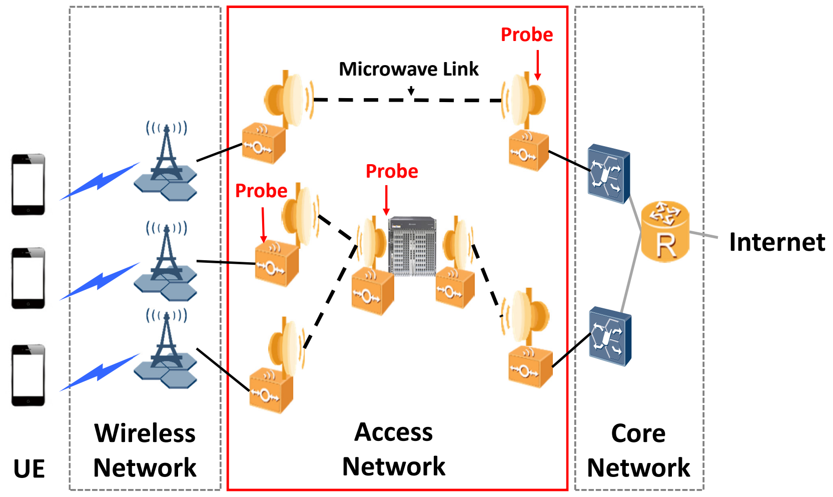


Figure 1: Microwave deployment in an LTE network architecture.

2.1. LTE Network Architecture

Figure 1 illustrates a simplified LTE network architecture that we consider in this paper. An LTE network consists of three sub-networks: *wireless network*, *access network*, and *core network*. A subscriber's mobile device, called *user equipment (UE)*, accesses the Internet through the LTE network. To do this, it must first connect over the air to a node in the wireless network, which in turn communicates with the core network through the access network. The connection between the core network devices and the Internet is usually through an optical fiber or cable. For devices in the access network, there are various connection methods, among which the most common ones are based on microwaves or optical fiber cables. In this paper, we focus on the LTE networks that deploy microwaves.

A microwave link is a communication medium that uses a beam of radio waves in the microwave frequency range to transmit video, audio, or other data between two locations. The distance between the two locations can range from just a few meters to several kilometers. The basic idea is to let the microwave antenna on the transmitting side convert data to electromagnetic wave signals and transmit the signals over the air. An antenna on the receiving side will then receive the electromagnetic wave signals and convert them back to data. We define this communication connection between two microwave devices as a *microwave link*, which is the observation object in our paper. Figure 2 shows a sketch of a microwave link (we elaborate the representation of a microwave link in Section 2.3). If a microwave link has quality deteriorations (e.g., due to bad weather, interference, or device failure), it implies that the link will likely suffer from an outage in a short period of time. Subscribers in this area would then lose access to the Internet.

Detecting link degradations is a precursor to failure predictions, since degradations are often early signs of potentially catastrophic faults. Our goal is to design a system for anomaly detection to monitor all microwave links and find those with degradations.

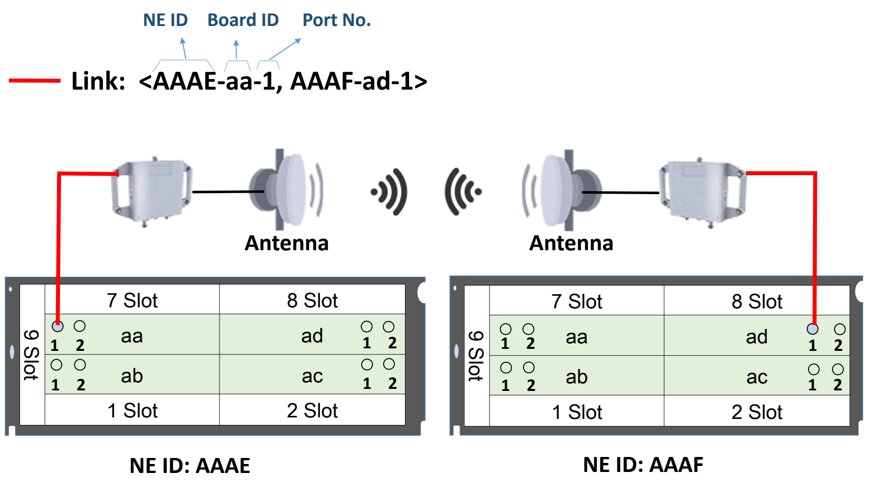


Figure 2: Sketch of the microwave link ⟨AAAE-aa-1, AAAF-ad-2⟩.

2.2. Data Collection

We use two types of data in our analysis: *link reports* and *key performance indicators (KPIs)*. In order to capture the KPIs of the microwave links, network operators deploy *probes* at each microwave device to collect data (as shown in Figure 1). Each probe collects KPI data every 15 minutes and sends the collected KPI data to a centralized network management system. Link reports and trouble tickets can also be obtained from the network management system.

We collected our dataset from a production LTE network in a Middle Eastern country. The collection period spans from September 27, 2017 to October 22, 2017. The network has 2,142 microwave links from 1,854 microwave nodes and serves around four million subscribers. We set 24 hours as an observation period for each link, and finally collected a total of 54,641 valid samples (some samples are incomplete, and inconsistent data were removed), of which 2,120 are identified as anomalies by network operators.

2.3. Data Preprocessing

We now describe how we preprocess the link reports and KPIs and how we transform them into an understandable format. We first use the link reports to determine the microwave links and infer the corresponding microwave network topology. We then introduce the KPIs used in our analysis. Finally, we describe our labeling process.

Link matching: We organize the link information and infer network topological information based on the link reports collected from the network management system. Each link is attached to two microwave devices, one at each end. There are typically multiple circuit boards on a microwave device that in turn has multiple ports. We choose six useful fields from the link reports to represent the links and uniquely identify each link based on these fields: the network element (NE) IDs, Board IDs, and the Port Numbers of both ends of microwave devices.

Table 1: Types of Key Performance Indicators (KPIs).

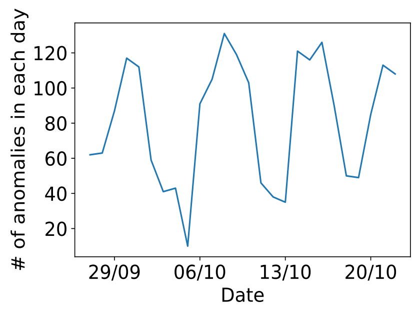
KPIs	Description
SNR_{max}	The maximum ratio of signal power to noise power of the port in one observation period.
SNR_{min}	The minimum ratio of signal power to noise power of the port in one observation period.
RSL_{max}	The maximum signal level at receiver input.
ES	The number of Errored Seconds in the observation period [20]; an errored second refers to a one-second interval that shows any type of error, either a single-bit error or a complete loss of communication.
SES	The number of Severely Errored Seconds [20]; a severely errored second refers to a one-second interval that contains no less than 30% of error blocks or at least one signal defect.
UAS	The period of unavailability, which starts with 10 consecutive SESs and ends with 10 consecutive non-SESs [20].

Figure 2 shows a sketch of a microwave link and the relevant fields. In the figure, there are two microwave devices, namely AAAE and AAAF (the names are anonymized). Each device has four boards inserted (i.e., in slots 3-6). We denote the link by the tuple $\langle$AAAE-aa-1, AAAF-ad-1$\rangle$. We append all links to a link list, which is updated based on the link reports sent from the network management system.

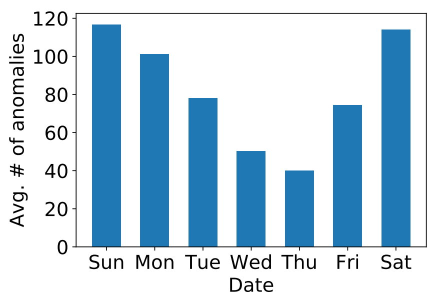
KPIs: We select and collect six types of KPIs from each port for our statistical feature construction. Table 1 summarizes the details of the six KPIs. The KPIs address microwave link performance from two aspects: (i) signaling quality (i.e., SNR_{max}, SNR_{min}, and RSL_{max}); and (ii) transmission performance (i.e., ES, SES, and UAS). For each link, we observe both of its NE KPIs. Thus, we use a total of 12 KPI values in our statistical feature analysis for each link.

Our selected KPIs are commonly available in production microwave devices. By no means do we claim that our selected KPIs exhaustively cover all performance aspects of microwave devices. Nevertheless, they provide useful information for us to characterize the basic properties of microwave link performance, so as to address the anomaly detection problem in PMADS.

Ground-truth label acquisition: We obtain ground-truth labels for the microwave links quality with the help of domain experts. Some anomalous links are reported in trouble tickets that describe the reasons behind the link failures; such links can be quickly labeled. However, some anomalous links are not present in the trouble tickets, so domain experts need to manually examine all microwave links and identify the anomalous ones by checking their corresponding KPI values. To ensure the accuracy and comprehensiveness of the samples, two domain experts have spent more than two weeks labeling all 54,641 samples as either normal or anomalous. In our analysis, we assume that all labels are correct and provide the ground truths.

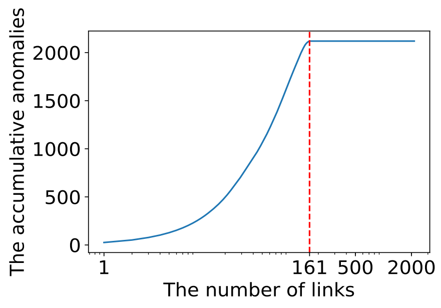


(a) Number of anomalies per day

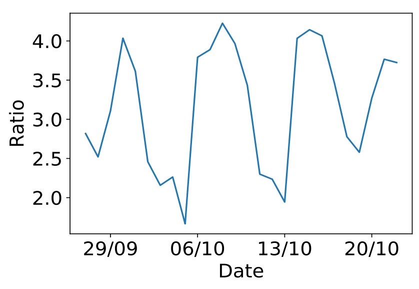


(b) Average number of anomalies each weekday

Figure 3: The distribution of anomalies over 26 days.



(a) Cumulative distribution



(b) Anomaly link-cluster ratio

Figure 4: Correlation of the anomaly links.

2.4. *High-Level Analysis*

We first examine the statistical properties of the microwave link anomalies in our dataset, so as to guide our anomaly detection design. Figure 3 shows the distribution of anomalies during the 26-day span. We observe that there exists a periodic pattern in the occurrences of the anomalies, such that the number of anomalies over the weekend is much higher than that during the weekdays. The reason is that network operators regularly perform maintenance on the microwave links at the beginning of every week. Thus, the number of anomalous links forms a cumulative effect over a period of one week which peaks during the weekend.

Due to the inherent connections among network devices, we also examine if it is necessary to consider the network topological information when designing a microwave link anomaly detection solution. To investigate how the anomalies are distributed among the links, we sort the links by the number of anomalies during the whole 26-day span. Figure 4(a) shows the results. We see that all 2,120 anomalies occur at only 161 links. Also, out of the 161 anomalous links, two-thirds of the links have anomalies more than 10 times, indicating that some links are repeatedly degraded.

To study the correlation between the links with anomalies, we first define each directed connected sub-graph in the microwave network topology as a *cluster*. There are a total of 369 clusters in our network. We call a cluster an *anomaly cluster* if it contains at least one anomaly. For each day, we count the number of anomalies and the number of anomaly clusters, denoted by *al* and *ac*, respectively. We then compute the *link-cluster ratio* as al/ac. Intuitively, if for most days the link-cluster ratio is close to one, we believe that the occurrence of link degradation is independent of location; otherwise, the anomaly links have non-negligible correlation related to their physical locations. For example, if the link-cluster ratio is larger than one, it means that more than one link in a cluster has deteriorated. Figure 4(b) shows the link-cluster ratio of each day. We observe that the link-cluster ratio for all days are larger than one. This implies that there exists an aggregation effect in the anomaly occurrences, such that the anomalies tend to occur in a few clusters instead of being randomly distributed across the network. Thus, it is necessary to consider the network topological information in our design.

3. PMADS Design

PMADS is a proactive microwave link anomaly detection system. It proactively predicts microwave link failures by anomaly detection to enable network operators to intervene and deal with imminent failures. Figure 5 shows the PMADS architecture. During initialization, PMADS stores an initial detection model generated by historical training data. PMADS performs the following three main steps (as numbered in the figure):

1. PMADS takes the link reports and KPIs as inputs and performs *feature extraction* to construct a set of relevant features in two categories, namely statistical features and network features.

2. PMADS builds a detection model based on the constructed features. It then performs runtime *prediction* to identify if any microwave link is anomalous based on the current detection model. It stores the prediction results in a distributed file system and alerts network operators of any microwave links that are predicted as anomalous.

3. PMADS provides an *active learning* module to allow domain experts to update and retrain the detection model. Here, the active learning module deploys our proposed active learning algorithm, ADAL, to reduce the manual labeling cost of collecting relevant samples for training the initial detection model.

The key insight behind the design of PMADS is that the features depend on not only the general performance indicators (i.e., KPIs in our case), but also the network topological information. Also, it leverages a novel active learning algorithm to update the detection model at low cost.

In the following, we first formulate our prediction problem in Section 3.1. We then elaborate our two key design elements: network-embedding-based feature extraction in Section 3.2 and active-learning-based model training in Section 3.3.

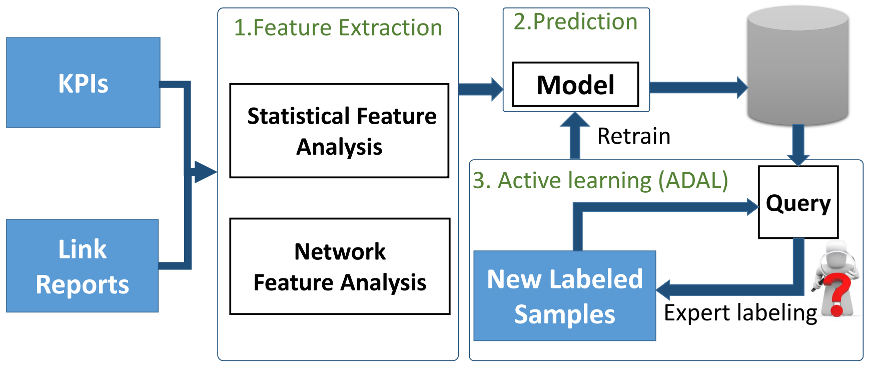


Figure 5: The PMADS architecture.

3.1. Problem Formulation

We formulate the microwave link anomaly detection problem as a binary classification problem, in which the outcome variable indicates if a link has significant degraded performance. We model this outcome as a function of the observed link records. Formally, given a set of training link records with known health type denoted by $D = \{(x_i, y_i)\}|_{i=1}^{n}$, where $x_i \in \mathbb{R}^d$ is the feature vector for link i, $y_i \in \{0, 1\}$ is the true health type (zero for a normal sample and one for an anomaly), and n is the number of observed links. We train a classification model denoted by a function f. Given an unseen link record x, we predict the health type as $y = f(x)$.

3.2. Feature Extraction

PMADS extracts two types of link features, namely statistical features and network features, as described below.

Statistical feature analysis: We derive a total of 19 statistical features from the KPI dataset for our model training. Table 2 summarizes the definitions of the 19 features, which can be categorized into three classes: (i) the distribution statistics (Features 1 to 14); (ii) the distance to a pre-specified threshold (Features 15 and 16); and (iii) the maximum values of error codes (Features 17 to 19). Note that the KPI value distributions vary across different configuration parameters. To eliminate the differences among the distributions, we normalize the data of each link by *Min-Max normalization*; that is, for a time series $\mathbf{x} = x_0, x_1, ..., x_n$, the normalized time series $\mathbf{x}'$ is calculated by:

$$x'_i = \frac{x_i - min\{\mathbf{x}\}}{max\{\mathbf{x}\} - min\{\mathbf{x}\}}.$$

To validate that the selected features are suitable for our anomaly detection problem, we calculate the correlation between each feature and the label (i.e., normal and abnormal) using the Mann-Whitney U test [29]. Here, we compute the *effect size* f (equivalent to the *area under the receiver operating characteristic curve (AUC)*) as:

$$f = \frac{U_1}{n_1 n_2},$$

Table 2: Definitions of the 19 statistical features for a microwave link attached to NE_1 and NE_2. We use the KPIs of one day in the calculations. Note that Features 13 to 19 include values from both NEs.

ID	Description	Effect Size
1	Variance of SNR_{max} at NE_1	0.777
2	Variance of SNR_{max} at NE_2	0.765
3	Variance of RSL_{max} at NE_1	0.864
4	Variance of RSL_{max} at NE_2	0.849
5	Variance of SNR_{max} at NE_1 with the top 90% of values	0.723
6	Variance of SNR_{max} at NE_2 with the top 90% of values	0.752
7	Kurtosis of SNR_{max} at NE_1	0.726
8	Kurtosis of SNR_{max} at NE_2	0.753
9	Kurtosis of RSL_{max} at NE_1	0.574
10	Kurtosis of RSL_{max} at NE_2	0.562
11	Skewness of RSL_{max} at NE_1	0.589
12	Skewness of RSL_{max} at NE_2	0.594
13	Sum of the squared differences between SNR_{max} and SNR_{min}	0.537
14	Sum of the distance between SNR_{max} and median SNR_{max}	0.530
15	Minimum distance from SNR_{max} to a threshold	0.535
16	Minimum distance from RSL_{max} to a threshold	0.532
17	Maximum of ES	0.908
18	Maximum of SES	0.887
19	Maximum of UAS	0.778

where U_1 is the Mann-Whitney U test statistics of the anomaly class, n_1 and n_2 are the sample sizes of the anomaly class and the normal class, respectively. The effect size f has a value range of $[0, 1]$. A feature with a larger effect size (e.g., larger than 0.5) is considered to have higher positive correlation with the prediction of the anomaly class. Table 2 shows that all features have effect sizes larger than 0.5. In particular, Features 17 and 18 (i.e., the features related to error codes) have the highest effect sizes, since the error codes measure the cumulative duration of errors.

Network feature analysis: Recall from Section 2.4 that link anomalies tend to be clustered. Based on our link reports, we capture the network topology as inputs to our model training. Specifically, we generate a graph based on the network structure and construct network features by using *network embedding* [8]. At a high level, network embedding is a network analysis paradigm that assigns network nodes to some low dimensional vector representations and refines the network structure effectively. While many network embedding algorithms exist (see the survey in [8]), their output formats share the property that the network structure and the node information are represented in the form of vectors. Such representations facilitate different types of analysis, such as node classification, node clustering, and anomaly detection. In this work, we adapt network embedding to microwave link anomaly detection.

Based on network embedding, PMADS performs three steps to construct the network features to be used in microwave link anomaly detection.

- *Step (i): Graph generation.* We represent the network topology as a graph G. Each

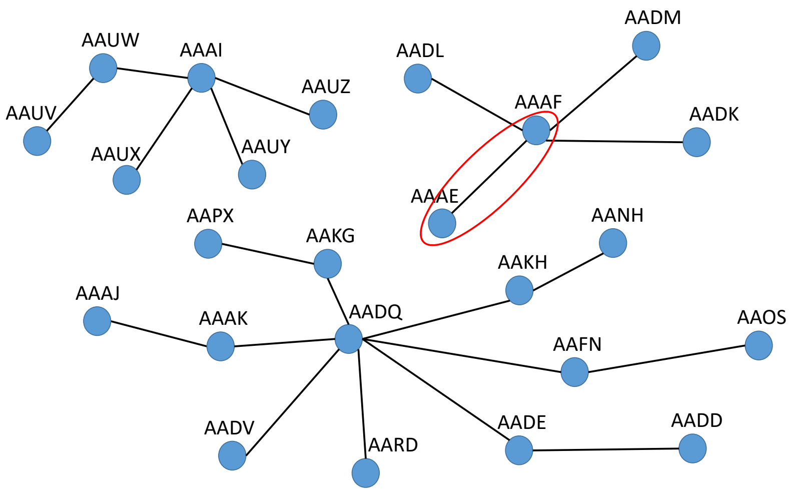


Figure 6: Subgraph of the full network topology.

node in G corresponds to an NE, with any two nodes connected by an edge if the corresponding NEs are connected by a microwave link. Figure 6 shows a network topology graph generated by a subset of the NEs (the entire graph with all NEs is too large to show here). For example, the two NEs in Figure 2, AAAE and AAAF, are connected by an edge in G.

- *Step (ii): Feature learning.* We generate vector representations of the nodes in the graph G using a network embedding algorithm. We denote the vector representations as $W = N \times L$, where N is number of nodes and L is the selected vector dimension.

- *Step (iii) Feature construction.* We extract the link features based on W. Given two nodes n_i and n_j, the corresponding feature vectors $f(n_i)$ and $f(n_j)$ are queried from W. In order to generate a representation of a microwave link (n_i, n_j), we use the following operation
$$f_n = ||f(n_i) - f(n_j)||_{ew}^1,$$
where $|| \cdot ||_{ew}^1$ is the element-wise L1-norm operator.

In particular, many structure-based embedding methods [17] can be used in feature learning (i.e., Step (ii)). In this work, we evaluate four popular embedding methods to generate network features for the microwave links (Section 4).

- DeepWalk [31]: it is the first proposed network embedding method. It treats the nodes as words, and generates sequences by using random walk. It then applies neural language models to generate vector representations.

- node2vec [15]: it generates a set of walk sequences for each node by defining a flexible notion of a node's network neighborhood, and accordingly designs a biased random walk procedure. It adopts stochastic gradient descent (SGD) to efficiently infer the vector representations.

- LINE [39]: it generates node representations based on breadth-first-search, and optimizes the first-order proximity and the second-order proximity to preserve both the local and global network structures, respectively.
- SDNE [41]: it proposes a semi-supervised model with multiple layers of nonlinear functions to capture the high non-linearity network structures.

We further consider a *basic* network feature construction approach specifically for our scenario. We represent each link with four attributes: the microwave NE ID and the (NE ID + Board ID) from the two connected nodes. In this way, we can identify the links connected to the same board and/or the same node.

3.3. Active Learning

To address the challenges of labeled data scarcity and time-varying data distribution (see Section 1), we design a novel feedback mechanism based on *active learning* [34] to continuously update the detection model of PMADS at low cost.

Background and challenges: The basic idea of active learning is to find the set of the most informative or representative samples for the learning task and use these samples to retrain the model [34]. Since the learner selects the samples, the number of samples that are necessary for learning a concept can often be much lower than what is required in normal supervised learning. In PMADS, the query component actively queries domain experts for labels and adds newly labeled samples to the labeled set for retraining.

Recall that $D = \{(x_i, y_i)\}$ is our available dataset (Section 3.1). Our goal is to label each sample x_i as either positive (i.e., anomalous or $y_i = 1$) or negative (i.e., normal or $y_i = 0$). Most existing active learning algorithms work as follows:

- *Step (i):* Train an initial model m using a small labeled set of training data $D_l \subset D$.
- *Step (ii):* Apply model m to the unlabeled data $D_u = D/D_l$.
- *Step (iii):* Select one sample $x^* \in D_u$ according to some criteria (e.g., selecting the sample closest to the decision boundary).
- *Step (iv):* Query the oracle to obtain the correct label for x^*, denoted by y^*.
- *Step (v):* Update the subsets, $D_l = D_l \cup \{(x^*, y^*)\}$, and $D_u = D_u/\{(x^*, y^*)\}$.
- *Step (vi):* Retrain model m and iterate steps 3-5 until the stop criterion is met.

There are two obstacles to directly deploying a standard active learning algorithm when applied to anomaly detection. First, many active learning algorithms need some pre-labeled samples to start the query process. However, as anomalies are rare, it is hard to directly find anomaly samples to initialize the active learning process. Second, the model needs to be retrained whenever a new labeled sample is added to the training set. For a real application, this inefficient retraining method poses a performance bottleneck to the domain experts who perform the labeling task.

Solution overview: To tackle these two problems, we propose ADAL, a novel Anomaly Detection by Active Learning algorithm. Inspired by the observation that many anomalies can be treated as outliers, we employ an unsupervised outlier detection algorithm to separate the anomalies from the dataset. Specifically, we train an outlier detection model using unsupervised learning (e.g., isolation forest [28]) on the whole dataset to separate the outliers. We select the top K outliers (denoted by the set O) to label. In this way, we can effectively find some anomalous samples. When K is big enough, we can train an accurate model directly after obtaining the labels of O. This is particularly true if we know that all anomalies are included in O, as in this case the labels of all remaining samples (denoted by the residual set D/O) are all negative.

When K is small, we can still obtain a model that can achieve a high precision for the positive class. Intuitively, we can train a classifier using O to distinguish the positive and negative samples in O. However, a model trained on O may not work properly on D/O, since the samples in O are the outliers in D. We want the model to predict *most* of the samples in D/O to be negative samples. In order to force the model to work properly on D/O, we add data from D/O to the model. We randomly select a subset of samples from D/O, denoted by S, and assign each sample in S a negative label. We then train a classifier model on the union of S and O. We set the size of S to be equal to the number of positive samples in O. A small-sized and negative-labeled S is enough to limit the model's prediction on D/O, since:

- O is a set that stores the top K outlier samples in D, implying that O and D/O should be two well-separated groups in the feature space. Therefore, adding samples from D/O can easily force the classifier's boundary to go through the margin between O and D/O.

- Most of samples in D/O are normal. According to our application, these normal samples are located in a narrow area. Hence, the randomly selected set S can well represent D/O. In addition, the small size of S can diminish the data imbalance problem during model training.

- There may be still some anomalies hidden in D/O. A small $|S|$ can reduce the risk of including some anomalies in S and avoid introducing label noise to the model.

We observe that this scenario is very similar to the *positive unlabeled problem*, to which a lot of research has been devoted [11, 26]. The difference is that most of positive unlabeled problems assume that the positive samples are completely randomly labeled. While in our scenario, the positive samples in O are selected by an outlier detection algorithm, rather than by random selection.

The classifier is designed to identify positive samples in O. In addition, anomalies in D/O that were missed by the outlier detection algorithm may be identified by the classifier. This is due to the difference in the algorithmic bias. The classifier should be able to find anomalies that have similar properties with the positive samples in O. Since normal samples are used in training the classifier, samples that are not obvious outliers could be found. Inspired by this intuition, we use the classifier to find new samples with high probability of being anomalies. We iteratively execute this process of finding new anomalies and training a classifier. We show in our evaluation in Section 4 that the

Algorithm 1 Anomaly Detection by Active Learning (ADAL)

```
Require: (1) Dataset $D$; (2) Outlier detection algorithm OTD; (3) Classifier CLF; (4) Label oracle
    ORACLE; (5) Number of samples to query in each iteration $K$; (6) Iteration times $T$; (7)
    Sampling ratio $\alpha$;
Ensure: Model $m$;
 1: Initialize $m$ =OTD.fit($D$) and $D_l = \emptyset$;
 2: for $i = 1$ to $T$ do :
 3:     Let $O$ be the top $K$ samples ranked by $m$ in $D$;
 4:     $O_{sub}$ = Random_Sample($O$, ratio = $\alpha$);
 5:     Invoke ORACLE to get the labels of $O_{sub}/D_l$;
 6:     $D_l = D_l \cup O_{sub}$;
 7:     $D_l^p = \{x|x \in D_l \wedge$ ORACLE$(x) = 1\}$;
 8:     if $|D_l^p| = 0$ then:
 9:         return to increase $K$;
10:     else:
11:         $S$ = Random_Sample($D/D_l$, size = $|D_l^p|$);
12:         Assign each sample in $S$ a negative class label;
13:         $m$ = CLF.fit($D_l \cup S$);
14:     end if
15: end for
16: return $m$;
```

number of anomalies discovered after only a couple of iterations is significantly higher than only using an outlier algorithm.

Implementation details: Algorithm 1 shows the details of our ADAL algorithm. We initialize a model from D by treating anomalies as outliers and applying an outlier detection algorithm based on unsupervised learning to D; we also initialize D_l as an empty set (line 1). We perform T iterations to refine the classification model m (lines 2-15). Specifically, in each iteration, we select the top K outliers in D to create the dataset O (line 3). Since labeling all samples in O may incur an excessive workload, we randomly select a subset of samples in O to label and denote it as O_{sub} (lines 4-5), and store the labeled samples in D_l (line 6). We examine the positive samples in D_l, denoted by D_l^p (line 7). If $|D_l^p|$ equals zero, the top K outliers do not contain any anomaly and we need to increase K to enable the outlier detection algorithm to capture anomalies (lines 8-9). Otherwise, we randomly select $|D_l^p|$ samples from D/D_l store them in S; note that $|S|$ equals $|D_l^p|$ (line 11). We assign each sample in S a negative class label (line 12), and train a classifier model on the union of D_l and S (line 13).

Model update: We can easily extend the active learning process in Algorithm 1 to support incremental learning, thereby enabling PMADS to update the model when new data arrives. Algorithm 2 shows the details of this extension. Most of the steps are kept the same as in Algorithm 1, with the exception of how the outlier detection model m and the labeled dataset D_l are initialized. In Algorithm 2, the outlier detection model is initialized using the previous model, denoted by m_{old} (line 1). D_l is initialized by the labeled data used in the previous model, rather than as an empty set (line 2). Since the size of labeled data is limited, it is not costly to maintain D_l in the system. Our

Algorithm 2 Model Update

```
Require: (1) Previous model $m_{\text{old}}$; (2) Newly arrived data $D_{\text{new}}$;
Ensure: Updated model $m_{\text{new}}$;
1: Initiate the outlier detection model by $m_{\text{old}}$;
2: Initiate $D_l$ by the labeled data used in model $m_{\text{old}}$;
3: Follow lines 2-15 in Algorithm 1 using $D_{\text{new}}$ as the input;
4: Let $m_{\text{new}}$ be the updated model;
5: return $m_{\text{new}}$;
```

evaluation (Section 4) shows that Algorithm 2 can update the model effectively.

How often ADAL should be run is highly dependent on the network and its dynamics. Real-world networks are relatively stable, so infrequent model updates (e.g., monthly) should be sufficient in practice. Also, network operators can monitor the detection accuracy. If the accuracy drops below some threshold, then the model update can be triggered immediately to adapt to any sudden changes.

4. Evaluation

In this section, we evaluate PMADS's accuracy in two aspects: (i) effectiveness of the network features and different classifiers; and (ii) effectiveness of the proposed active learning algorithm ADAL, including parameter sensitivity and model updating capability.

We use the dataset described in Section 2 for our evaluation. Following standard practice in supervised learning, we split the data set into training (80%) and testing (20%). Note that we have also verified that for other training-to-testing ratios, PMADS still achieves accuracy gain compared to the baseline that uses only statistical features without adding the topological information. The data from September 27, 2017 to October 16, 2017 is used as the training set, denoted as E_{train}. It contains 1,625 one-day link samples identified as anomalies and 40,446 nominal samples, denoted as positive and negative samples respectively. Data from October 17, 2017 to October 22, 2017 is used as the test set, denoted E_{test}. The test set contains 495 positive samples and 12,075 negative samples.

4.1. Effectiveness of Feature Construction and Classifier

Comparisons of different feature sets: We evaluate the accuracy of PMADS on different feature sets. The statistical features f_s are evaluated by themselves as a baseline for how good the classification is without adding the topological information. We then extend f_s by adding the network features obtained by the different network embedding methods presented in Section 3.2, including the network features extracted from the basic algorithm (f_{basic}) and the four network embedding methods DeepWalk, node2vec, LINE and SDNE ($f_{DeepWalk}$, $f_{node2vec}$, f_{LINE}, and f_{SDNE}, respectively), as described in Section 3.2. We fix the embedding dimension L to 128 for each method while keeping all other parameters as the default values. For each feature set, several different classifiers are evaluated. Specifically, we train our detection models on E_{train} using

Table 3: Evaluation with different feature sets and classifiers.

(a) Recall

Features	**LR**	**SVM**	**DT**	**GBDT**	**RF**
(f_s)	0.345	0.014	0.585	0.706	0.630
(f_s, f_{basic})	0.394	0.028	0.751	0.817	0.806
$(f_s, f_{DeepWalk})$	0.353	0.070	0.796	0.853	0.843
$(f_s, f_{node2vec})$	0.336	0.152	0.860	0.869	**0.871**
(f_s, f_{LINE})	0.345	0.040	0.794	0.851	0.827
(f_s, f_{SDNE})	0.351	0.051	0.787	0.859	0.849

(b) Precision

Features	**LR**	**SVM**	**DT**	**GBDT**	**RF**
(f_s)	0.771	0.75	0.689	0.774	0.853
(f_s, f_{basic})	0.783	0.764	0.793	0.910	0.917
$(f_s, f_{DeepWalk})$	0.758	0.733	0.822	0.924	0.929
$(f_s, f_{node2vec})$	0.712	0.541	0.860	0.941	**0.944**
(f_s, f_{LINE})	0.753	0.542	0.814	0.919	0.926
(f_s, f_{SDNE})	0.782	0.533	0.842	0.916	0.925

(c) PRAUC

Features	**LR**	**SVM**	**DT**	**GBDT**	**RF**
(f_s)	0.654	0.485	0.619	0.805	0.833
(f_s, f_{basic})	0.677	0.445	0.777	0.929	0.926
$(f_s, f_{DeepWalk})$	0.584	0.702	0.820	0.949	0.945
$(f_s, f_{node2vec})$	0.608	0.539	0.862	**0.960**	0.958
(f_s, f_{LINE})	0.521	0.593	0.815	0.953	0.940
(f_s, f_{SDNE})	0.657	0.591	0.826	0.952	0.946

SVM, Logistic regression (LR), Decision Tree (DT), Random Forest (RF) and Gradient Boosting Decision Tree (GBDT).

Table 3 shows the recall, precision and precision-recall area under curve (PRAUC) on E_{test} for the different classification algorithms and feature combinations. Looking at the accuracy of the classification algorithms, we observe that the tree-based methods generally achieve the best results. This could be due to the class imbalance in our dataset, with the number of positive samples being much smaller than the number of negative samples. Most of the negative samples are stable and distributed over a narrow interval, while the positive samples contain the extreme values. In this imbalanced scenario, the LR and SVM models are prone to overfitting; in contrast, the tree-based models only consider the order of the feature values rather than actual raw values, thereby enabling them to achieve a higher accuracy.

As shown, when utilizing only the statistical features, the detection model does not perform as well as when both the statistical and any set of network feature are applied together. This shows the advantage of considering the topological information when performing microwave link anomaly detection. We note that all the network embedding approaches outperform our basic approach (i.e., f_{basic}), indicating that a simple network feature extraction technique does not capture all relevant information. The network features based on node2vec [15] have the highest accuracy (i.e., 94.4% of precision and 87.1% of recall) of all considered network features, while others produce similar results. In the following analysis, we focus on *node2vec* for network feature extraction.

Parameter sensitivity: PMADS involves a large number of parameters, especially for the network embedding. To show the sensitivity of our results to parameter changes, we perform a parameter sensitivity analysis on how the results vary across parameters. We choose to adopt the classical OFAT (one-factor-at-a-time) method, in which we change a single parameter at a time while keeping the other parameters constant at their default values. The model sensitivity is measured by observing the PRAUC. We repeat the procedure for each parameter. Suppose that the node2vec algorithm [15] is used. We note that the algorithm takes the following six parameters as inputs:

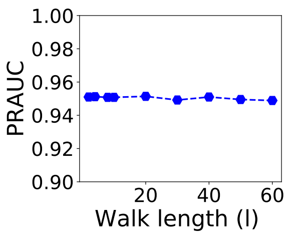
- l: the length of each walk.
- r: the number of walks per node.
- L: the embedding dimension.
- d: the window size.
- p and q: the transition probability parameters between nodes.

Figure 7 shows the results of the sensitivity analysis. We choose to use f_s and $f_{node2vec}$ as input features to a random forest classifier with 100 trees in our testing scenario. We observe that increasing the embedding dimension L improves the detection accuracy, but any changes in the other parameters only have a negligible effect. That is, the network features in PMADS are insensitive to changes in the routing parameters (p and q) and the neighborhood parameters (the number of walks r, walk length l, and window size d).

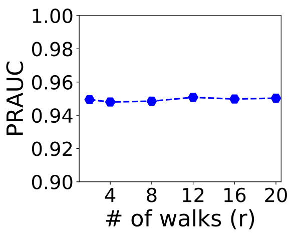
4.2. Effectiveness of ADAL

Comparison with other active learning algorithms: We compare ADAL with several existing active learning algorithms. Since ADAL uses a pool-based sampling strategy, in order to make a fair comparison, we only include pool-based active learning algorithms in our experiments. We consider the following baselines:

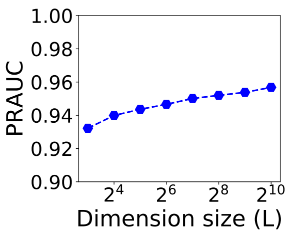
(1) Random Sampling (RS): randomly selects samples to query in each iteration.



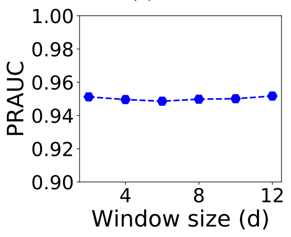
(a) l



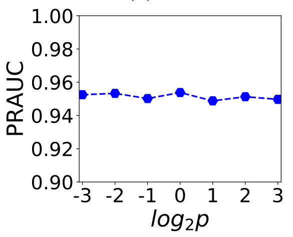
(b) r



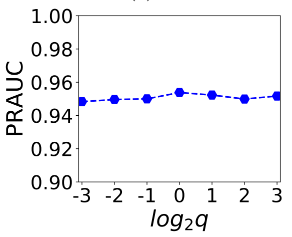
(c) L



(d) d



(e) $\log_2 p$



(f) $\log_2 q$

Figure 7: Parameter sensitivity.

(2) Uncertainty Sampling (US): queries the sample with the most uncertainty [25].

(3) Query by Committee (QBC): queries the sample with the most disagreement among the committee members [35].

(4) Learning Active Learning (LAL-iter): uses a regression model to select the sample which has the maximum potential error reduction [23].

We conduct the experiment on the combination of the statistical features f_s and the network features obtained by the node2vec embedding, $f_{node2vec}$. We train all models on the training set E_{train} and evaluate with the test set E_{test} after each query iteration. A random forest classifier with 100 trees is used as the classifier. We implement the RS, US, and QBC experiments using the libact package [45]. For US, we use the least confidence as the criterion to select samples. For QBC, we include one logistic regression model, one random forest model, and one naive Bayesian model in the committee. For the LAL-iter algorithm, we follow the implementation in [23]. We initialize all these baseline models with the top 10 outliers found by the isolation forest algorithm in E_{train}.

For ADAL, we set the parameter K to 3,000 and the sampling ratio α to 0.05. We set the number of iterations T as 40 and use the isolation forest algorithm as the initialization model. For the classifier, we use a Random Forest with 100 trees. All algorithms are run five times to reduce the random variance, and we use the average PRAUC for method comparison. For ADAL, since the number of queries sampled in the five experiments may be different, we use the average queried samples in each iteration instead.

Figure 8 shows the results of these models on the test set E_{test}. We observe that active learning, regardless of the algorithm, achieves higher PRAUC than random sam-

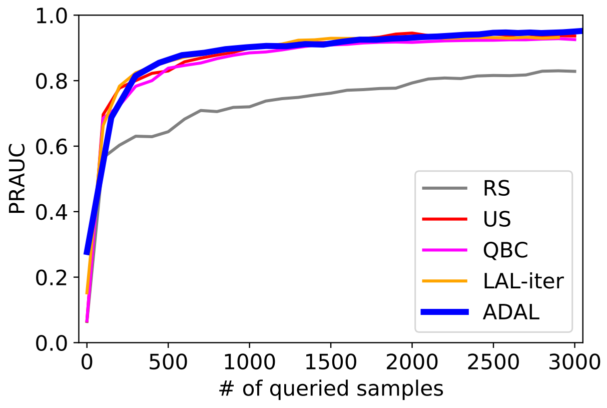


Figure 8: Active learning algorithm comparison.

Table 4: The running time (seconds) of each algorithm.

RS	**US**	**QBC**	**LAL-iter**	**ADAL**
2,616	4,280	6,685	6,992	**68**

pling. Compared to the model trained with the full training set E_{train} using the same features and classifier, the active learning algorithms significantly reduce the labeling cost while bringing only limited loss of accuracy. A PRAUC of 0.9 is achieved by only using around 1,000 samples. After 3,000 training samples have been queried (i.e., 7% of the full training set), the active learning algorithms achieve approximately the same PRAUC compared to the model trained on the full data.

Although ADAL has a comparable PRAUC to other four active learning algorithms, it is far more computationally effective. The other four algorithms operate on a *per-query* basis. In each iteration, they need to query one unlabeled sample and then update the model by adding the new labeled sample into the training data. This implies that in order to query 3,000 samples, these algorithms need to train the model 3,000 times and hence will incur high computational overhead. In contrast, ADAL operate on a *batched-query* basis. To query 3,000 samples, the number of iterations is typically far less than 3,000, as each iteration of ADAL will select the top $K \times \alpha$ samples to query. Also, similar to other four algorithms, in each iteration, ADAL just needs to update the model by adding the new labeled samples to the training data. As a result, the number of iterations of ADAL (which is about 30 iterations based on our experiments) is far less than other four algorithms. Table 4 shows the average running time of each algorithm under the same number of queries. ADAL only takes a few tens of seconds to run, which is two orders of magnitude less than the other active learning algorithms.

Proportion of anomalies in O_{sub}: We evaluate the proportion of anomalies found in O_{sub}. Figure 9 shows the results. Here, we set K as 3,000, which is considerably more than the number of positive samples in the E_{train}. This means that it is possible for all anomalies in E_{train} to be included in O_{sub}. From the figure, we find that the proportion of anomalies in O_{sub} increases along with the increasing number of iterations and after a couple of iterations, almost all anomaly samples are included in O_{sub}.

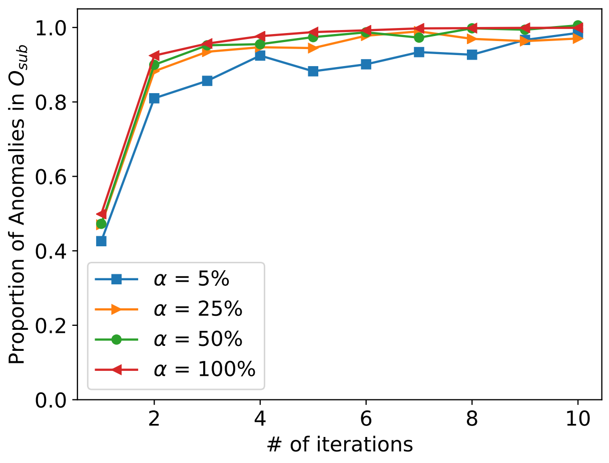


Figure 9: The proportion of anomalies discovered by ADAL in O_{sub} when increasing number of iterations. The proportion of anomalies in $O_{sub} = \frac{true\ anomaly\ samples\ in\ O_{sub}}{(anomalies\ in\ E_{train})\cdot\alpha}$.

Parameter sensitivity: ADAL has two parameters to set, the number of samples to query in each iteration K and the sampling ratio α. We analyze the model sensitivity for these two parameters by testing different values for K and α. Figure 10 shows the results. When K is smaller than the number of anomalies in the training data, the final accuracy is sub-optimal (K = 1,000 in the figure). This is because K is too small to cover all potential anomalies, resulting in some anomaly samples being missed by the classifier. When K is big enough, we find that the model achieves a comparable result to the model trained on the full dataset. The α parameter control the sampling ratio in the top K window. Using different α will converge to the same result, however, a smaller α can help the model achieve high accuracy earlier.

Model update: We conduct an experiment to demonstrate the model updating capability of ADAL. We split the data into several separate parts with regards to time. An initial model $M0$ is trained using the first 40% of the samples and the following 40% are used to update the model. These are further split into four equal parts to mimic four model updates (models $M1$ to $M4$). The final 20% of the samples are used as test data on which all the five models are evaluated. We use PRAUC as the evaluation metric and the experiment is run five times. Figure 11 shows the mean and standard deviation of the runs. It shows that the model update process in Algorithm 2 can continuously improve the model accuracy as new data arrives. Although it cannot get the same accuracy as achieved using the full dataset (PRAUC 94.39% in update mode vs. 95.8% in full data training mode using random forest), the online update is important for PMADS during real deployment.

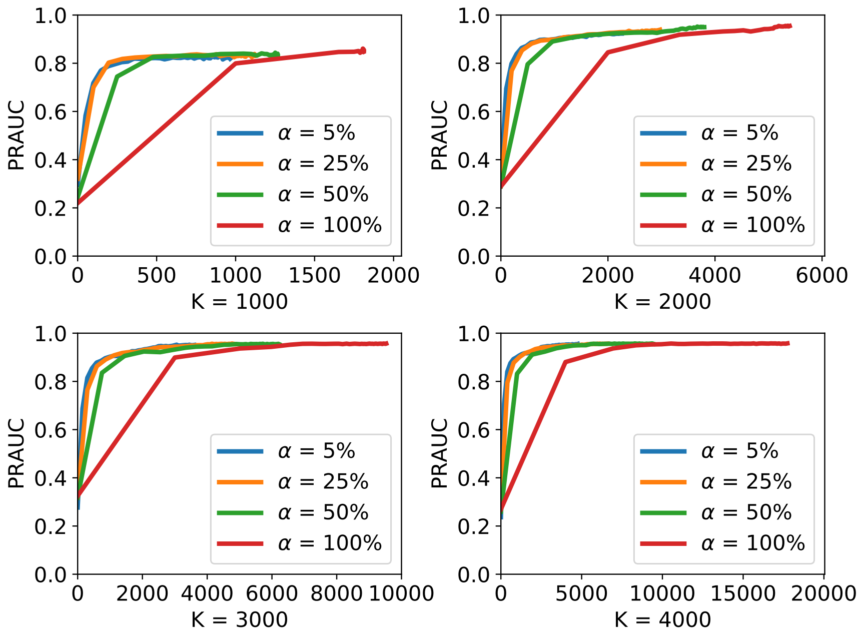


Figure 10: The effect of different K and α on the model. The x-axis is the number of queries and the y-axis is PRAUC. With higher α, the number of queries per iteration is higher.

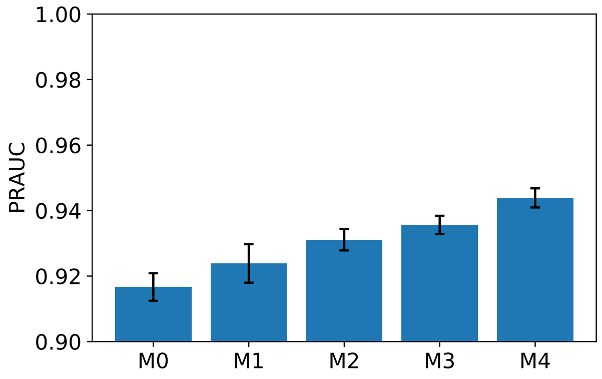


Figure 11: Online model update imitating a real scenario, $M0$ is the initial model and $M1$ to $M4$ are four incrementally updated models.

5. Lessons Learned

This section presents our experiences of designing and implementing the microwave link anomaly detection system PMADS in a cellular data network. We summarize our lessons learned from the deployment of the system.

Feature construction: We know from domain experience that the network topology includes rich information which is helpful for fault diagnosis in cellular networks.

However, it has been ignored by many previous studies. In this work, we encode the network structure to a series of vectors and add them to the model, resulting in a significantly improvement in detection accuracy. This is a meaningful result and suggests that we can leverage abundant tools from other fields, such as social network analysis and graph theory, for the cellular network fault diagnosis problem.

Active learning: Many studies have been devoted to the area of active learning, but most of them only focus on how to find informative or representative samples to improve the accuracy, rather than the algorithmic efficiency or user experience. However, time efficiency is an important concern in real applications, since the labeling task is often manually done by domain experts. We have noticed that active learning can lose its advantage if it takes too long time to receive the labeling result of a query.

6. Related Work

Proactive maintenance: Some studies pay special attention to proactive maintenance in operational networks [33, 43]. Kimura et al. [22] propose a log analysis system for proactive detection of failures and show that the abnormality depends not only on the keywords in messages but also generation patterns. Opprentice [27] is a novel anomaly detection approach based on supervised machine learning by collecting performance data in social networks. Hora [32] employs architectural knowledge to predict how a failure can propagate and affect other components. Our work addresses failure prediction in cellular networks, with specific focus on the degradation of microwave links.

Analysis of cellular networks: Failure prediction and anomaly detection have been widely applied in cellular networks. Some studies analyze the logs of network devices [13, 46], while others focus on using performance data (e.g., KPIs) for failure detection or prediction purposes [3, 6, 38, 44, 49]. Our work specifically considers the topological information (i.e., the dependencies among microwave links) in failure prediction.

Active learning: Most active learning studies focus on selecting a single sample to query in each iteration [10, 19, 23, 24, 35]. Uncertainty sampling finds the unlabeled sample with the most uncertainty to the model [24]. Query by Committee queries the sample that has maximal disagreement among the classifiers in the committee [35]. Learning Active Learning chooses the sample that maximizes the decrease in loss, using a regressor trained from other representative data [23]. Batch mode active learning aims to find a subset of unlabeled samples rather than a single unlabeled sample [18, 42]. Besides these pool-based algorithms, there are also stream-based active learning methods, in which the learner is presented with a stream of unlabeled instances. In this setting, the samples are presented one by one and the algorithms have to decide whether to query the label of the sample or ignore it [14, 48, 50].

Some studies have applied active learning in anomaly detection. Active-Outlier [2] reduces the anomaly detection problem to a binary classification problem by adding a synthetic positive dataset. A selective sampling process is then invoked to train a classifier to solve the classification problem. AI^2 [40] merges the analyst's knowledge with machine learning algorithms using an analyst-in-the-loop security system. AAD

[9, 37] maximizes the number of true anomalies presented to experts through an interactive data exploration loop.

7. Conclusions and Future Work

PMADS is a proactive microwave link anomaly detection system that is currently deployed in a production cellular data network. It extracts statistical features from KPIs and learns network features from network topological information. It additionally applies a novel active learning algorithm ADAL, which selects samples for model updates and keeps the model adaptable at low cost. We show via evaluation that the features related to the network topology are crucial for anomaly detection and enable PMADS to achieve both high precision and high recall. We also show that ADAL obtains comparable results to popular active learning algorithms, while having significantly lower running time. Furthermore, when combined with active learning, PMADS only requires 7% of the training data to achieve similar accuracy as using the entire dataset. Our findings shed light on how PMADS provides proactive fault tolerance for microwave links in cellular data networks.

References

[1] Microwave Link. `http://www.microwave-link.com`.

[2] N. Abe, B. Zadrozny, and J. Langford. Outlier detection by active learning. In *Proceedings of the 12th ACM SIGKDD International Cconference on Knowledge Discovery and Data Mining*, pages 504–509. ACM, 2006.

[3] F. Ahmed, J. Erman, Z. Ge, A. X. Liu, J. Wang, and H. Yan. Detecting and localizing end-to-end performance degradation for cellular data services. In *Proceedings of the 35th Annual IEEE International Conference on Computer Communications (INFOCOM)*, pages 1–9. IEEE, 2016.

[4] P. Casas, P. Fiadino, and A. D'Alconzo. Machine-learning based approaches for anomaly detection and classification in cellular networks. In *Proceedings of IFIP Traffic Monitoring and Analysis workshop (TMA)*, 2016.

[5] B. Cheung, G. Kumar, and S. A. Rao. Statistical algorithms in fault detection and prediction: Toward a healthier network. *Bell Labs Technical Journal*, 9(4):171–185, 2005.

[6] G. F. Ciocarlie, U. Lindqvist, S. Nováczki, and H. Sanneck. Detecting anomalies in cellular networks using an ensemble method. In *Proceedings of the 9th international conference on network and service management (CNSM)*, pages 171–174. IEEE, 2013.

[7] Cisco. Microwave Adaptive Bandwidth Feature: Make Better Use of Available Bandwidth . `https://www.cisco.com/c/en/us/solutions/collateral/service-provider/unified-ran-backhaul/white_paper_c11-728355.pdf`.

[8] P. Cui, X. Wang, J. Pei, and W. Zhu. A survey on network embedding. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 31(5):833–852, May 2019.

[9] S. Das, W.-K. Wong, T. Dietterich, A. Fern, and A. Emmott. Incorporating expert feedback into active anomaly discovery. In *Proceedings of the 16th International Conference on Data Mining (ICDM)*, pages 853–858. IEEE, 2016.

[10] S. Dasgupta and D. Hsu. Hierarchical sampling for active learning. In *Proceedings of the 25th International Conference on Machine learning (ICML)*, pages 208–215. ACM, 2008.

[11] C. Elkan and K. Noto. Learning classifiers from only positive and unlabeled data. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 213–220. ACM, 2008.

[12] Ericsson. Microwave Towards 2020. `https://www.ericsson.com/assets/local/microwave-outlook/documents/microwave-towards-2020-report-2014.pdf`.

[13] Y.-Y. Fanjiang and C.-P. Wu. Automatic data logging and quality analysis system for mobile devices. *Mobile Information Systems*, 2017, 2017.

[14] K. Fujii and H. Kashima. Budgeted stream-based active learning via adaptive submodular maximization. In *Advances in Neural Information Processing Systems*, pages 514–522, 2016.

[15] A. Grover and J. Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 855–864. ACM, 2016.

[16] Q. Guan, Z. Zhang, and S. Fu. Ensemble of bayesian predictors and decision trees for proactive failure management in cloud computing systems. *Journal of Communications*, 7(1):52–61, 2012.

[17] C. Haochen, P. Bryan, R. Al-Rfou, and S. Steven. A tutorial on network embeddings. *CoRR*, abs/1808.02590, 2018.

[18] S. C. Hoi, R. Jin, J. Zhu, and M. R. Lyu. Batch mode active learning and its application to medical image classification. In *Proceedings of the 23rd International Conference on Machine Learning (ICML)*, pages 417–424. ACM, 2006.

[19] S.-J. Huang, R. Jin, and Z.-H. Zhou. Active learning by querying informative and representative examples. In *Advances in neural information processing systems*, pages 892–900, 2010.

[20] ITU. ITU-T Recommendation G.826-Error performance parameters and objectives for international, constant bit rate digital paths at or above the primary rate. `https://www.itu.int/rec/T-REC-G.826`.

[21] R. M. Khanafer, B. Solana, J. Triola, R. Barco, L. Moltsen, Z. Altman, and P. Lazaro. Automated diagnosis for umts networks using bayesian network approach. *IEEE Transactions on vehicular technology*, 57(4):2451–2461, 2008.

[22] T. Kimura, A. Watanabe, T. Toyono, and K. Ishibashi. Proactive failure detection learning generation patterns of large-scale network logs. *IEICE Transactions on Communications*, 2018.

[23] K. Konyushkova, R. Sznitman, and P. Fua. Learning active learning from data. In *Advances in Neural Information Processing Systems*, pages 4225–4235, 2017.

[24] D. D. Lewis and J. Catlett. Heterogenous uncertainty sampling for supervised learning. In *Machine learning proceedings*, pages 148–156. Elsevier, 1994.

[25] D. D. Lewis and W. A. Gale. A sequential algorithm for training text classifiers. In *Proceedings of the 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 3–12. Springer, 1994.

[26] B. Liu, Y. Dai, X. Li, W. S. Lee, and P. S. Yu. Building text classifiers using positive and unlabeled examples. In *Proceedings of the 3th International Conference on Data Mining (ICDM)*, pages 179–186. IEEE, 2003.

[27] D. Liu, Y. Zhao, H. Xu, Y. Sun, D. Pei, J. Luo, X. Jing, and M. Feng. Opprentice: Towards practical and automatic anomaly detection through machine learning. In *Proceedings of the 2015 Internet Measurement Conference (IMC)*, pages 211–224. ACM, 2015.

[28] F. T. Liu, K. M. Ting, and Z.-H. Zhou. Isolation forest. In *Proceedings of the 8th International Conference on Data Mining (ICDM)*, pages 413–422. IEEE, 2008.

[29] H. B. Mann and D. R. Whitney. On a test of whether one of two random variables is stochastically larger than the other. *The Annals of Mathematical Statistics*, 18(1):50–60, 1947.

[30] L. Pan, J. Zhang, P. P. Lee, H. Cheng, C. He, C. He, and K. Zhang. An intelligent customer care assistant system for large-scale cellular network diagnosis. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge discovery and data mining*, pages 1951–1959. ACM, 2017.

[31] B. Perozzi, R. Al-Rfou, and S. Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 701–710. ACM, 2014.

[32] T. Pitakrat, D. Okanović, A. van Hoorn, and L. Grunske. Hora: Architecture-aware online failure prediction. *Journal of Systems and Software*, 137:669–685, 2018.

[33] F. Salfner, M. Lenk, and M. Malek. A survey of online failure prediction methods. *ACM Computing Surveys (CSUR)*, 42(3):10, 2010.

[34] B. Settles. Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 6(1):1–114, 2012.

[35] H. S. Seung, M. Opper, and H. Sompolinsky. Query by committee. In *Proceedings of the 5th Annual Workshop on Computational Learning Theory*, pages 287–294. ACM, 1992.

[36] L. H. Shuan, T. Y. Fei, S. W. King, G. Xiaoning, and L. Z. Mein. Network equipment failure prediction with big data analytics. *International Journal of Advances in Soft Computing & Its Applications*, 8(3):59–69, 2016.

[37] M. A. Siddiqui, A. Fern, T. G. Dietterich, R. Wright, A. Theriault, and D. W. Archer. Feedback-guided anomaly discovery via online optimization. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge discovery and data mining*, pages 2200–2209. ACM, 2018.

[38] P. Szilágyi and S. Nováczki. An automatic detection and diagnosis framework for mobile communication systems. *IEEE Transactions on Network and Service Management*, 9(2):184–197, 2012.

[39] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web (WWW)*, pages 1067–1077. International World Wide Web Conferences Steering Committee, 2015.

[40] K. Veeramachaneni, I. Arnaldo, V. Korrapati, C. Bassias, and K. Li. AI2: Training a big data machine to defend. In *Proceedings of the 2nd International Conference on Big Data Security on Cloud (BigDataSecurity)*, pages 49–54. IEEE, 2016.

[41] D. Wang, P. Cui, and W. Zhu. Structural deep network embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1225–1234. ACM, 2016.

[42] Z. Wang and J. Ye. Querying discriminative and representative samples for batch mode active learning. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 9(3):17, 2015.

[43] Z. Wang, M. Zhang, D. Wang, C. Song, M. Liu, J. Li, L. Lou, and Z. Liu. Failure prediction using machine learning and time series in optical network. *Optics Express*, 25(16):18553–18565, 2017.

[44] J. Wu, P. P. Lee, Q. Li, L. Pan, and J. Zhang. CellPAD: Detecting performance anomalies in cellular networks via regression analysis. In *Proceedings of the IFIP Networking Conference (IFIP Networking)*, 2018.

[45] Y.-Y. Yang, S.-C. Lee, Y.-A. Chung, T.-E. Wu, S.-A. Chen, and H.-T. Lin. libact: Pool-based active learning in python. Technical report, National Taiwan University, Oct. 2017. available as arXiv preprint `https://arxiv.org/abs/1710.00379`.

[46] Z. Yuan, Y. Li, C. Peng, S. Lu, H. Deng, Z. Tan, and T. Raza. A machine learning based approach to mobile network analysis. In *Proceedings of the 27th International Conference on Computer Communication and Networks (ICCCN)*, pages 1–9. IEEE, 2018.

[47] S. Zhang, Y. Liu, W. Meng, Z. Luo, J. Bu, S. Yang, P. Liang, D. Pei, J. Xu, Y. Zhang, Y. Chen, H. Dong, X. Qu, and L. Song. PreFix: Switch failure prediction in datacenter networks. *Proceedings of the ACM on Measurement and Analysis of Computing Systems (POMACS)*, 2(1):2, 2018.

[48] Y. Zhang, P. Zhao, J. Cao, W. Ma, J. Huang, Q. Wu, and M. Tan. Online adaptive asymmetric active learning for budgeted imbalanced data. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 2768–2777. ACM, 2018.

[49] X. Zhao, X. Luo, T. Wu, and D. Xiao. The prediction mathematical model for HO performance in LTE networks. In *Proceedings of 8th International Conference on Networking, Architecture and Storage (NAS)*, pages 191–197. IEEE, 2013.

[50] I. Žliobaitė, A. Bifet, B. Pfahringer, and G. Holmes. Active learning with drifting streaming data. *IEEE Transactions on Neural Networks and Learning Systems*, 25(1):27–39, 2013.