CellPAD: Detecting Performance Anomalies in Cellular Networks via Regression Analysis

Jun Wu[1], Patrick P. C. Lee[2], Qi Li[1], Lujia Pan[3,4], Jianfeng Zhang[4]
[1]Tsinghua University [2]The Chinese University of Hong Kong
[3]Xi'an Jiaotong University [4]Huawei Noah's Ark Lab

***Abstract*—How to accurately detect Key Performance Indicator (KPI) anomalies is a critical issue in cellular network management. We present CELLPAD, a unified performance anomaly detection framework for KPI time-series data. CELLPAD realizes simple statistical modeling and machine-learning-based regression for anomaly detection; in particular, it specifically takes into account seasonality and trend components as well as supports automated prediction model retraining based on prior detection results. We demonstrate how CELLPAD detects two types of anomalies of practical interest, namely sudden drops and correlation changes, based on a large-scale real-world KPI dataset collected from a metropolitan LTE network. We explore various prediction algorithms and feature selection strategies, and provide insights into how regression analysis can make automated and accurate KPI anomaly detection viable.**

***Index Terms*—anomaly detection, cellular network management, measurement and analysis**

I. INTRODUCTION

The continuing advances of cellular network technologies make high-speed mobile Internet access a norm. However, cellular networks are large and complex by nature, and hence production cellular networks often suffer from performance degradations or failures due to various reasons, such as background interference, power outages, malfunctions of network elements, and cable disconnections. It is thus critical for network administrators to detect and respond to performance anomalies of cellular networks in real time, so as to maintain network dependability and improve subscriber service quality. To pinpoint performance issues in cellular networks, a common practice adopted by network administrators is to monitor a diverse set of *Key Performance Indicators (KPIs)*, which provide time-series data measurements that quantify specific performance aspects of network elements and resource usage. The main task of network administrators is to identify any *KPI anomalies*, which refer to unexpected patterns that occur at a single time instant or over a prolonged time period.

Today's network diagnosis still mostly relies on domain experts to manually configure anomaly detection rules [25]; such a practice is error-prone, labor-intensive, and inflexible. Recent studies propose to use (supervised) *machine learning* for anomaly detection in cellular networks (e.g., [3], [8], [10], [11], [13], [34]) and search engines (e.g., [25]). However, machine-learning-based anomaly detection is subject to several well-known challenges [9], [25]: (i) the issues of which machine-learning algorithms should be used and how features should be configured depend on the actual anomaly detection problems and are difficult to address; (ii) labeling which time instants are anomalies for large-scale datasets is time-consuming; (iii) differentiating between normal data and anomalies is challenging and often requires domain knowledge to resolve; and (iv) anomalies occur much more infrequently than normal data, and this imbalanced nature can degrade learning accuracy [17].

ISBN 978-3-903176-08-9 ©2018 IFIP

In the context of cellular networks, we need to address additional challenges in anomaly detection. First, Internet traffic often exhibits periodic diurnal patterns [35] and different trends after long-term usage. In addition, the performance of cellular networks depends on not only the data transmission usage as in the traditional Internet, but also the radio resource usage [30]. Their corresponding KPIs, and hence anomalies, are often correlated. Such properties need to be properly addressed in the anomaly detection design. Thus, we are motivated to look into the following issues: (i) How should we define useful KPI anomalies that correspond to practical cellular network performance degradation problems? (ii) Can we design a unified anomaly detection framework that can incorporate various anomaly detection algorithms and detect various types of anomalies for one or multiple KPIs? (iii) Can our anomaly detection framework be automated with limited manual intervention, while still achieving accurate detection?

We present CELLPAD, a unified performance anomaly detection framework for cellular networks. CELLPAD builds on *regression analysis*, which predicts the expected quantities of KPI time-series data so as to provide prediction results for anomaly detection. We consider two types of anomalies that are of practical interest to cellular network management based on our internal communication with network administrators: *sudden drops*, which indicate the unexpected degradations of a KPI, and *correlation changes*, which indicate the inconsistency between the current and historical correlations of two correlated KPIs. Using CELLPAD, we conduct trace-driven evaluation to demonstrate how regression analysis achieves automated and accurate KPI anomaly detection. To summarize, this paper makes the following contributions:

- We first present a trace-driven analysis on a large-scale KPI dataset from a real-world metropolitan LTE network. Our dataset spans six KPIs, 17 weeks of duration, and 12,463 cells. We show the presence of anomalies in the dataset and motivate the practical need of anomaly detection.
- We design CELLPAD for anomaly detection in cellular net-

works. CELLPAD supports various prediction algorithms, including simple statistical modeling, linear regression, and tree-based regression (the latter two belong to machine-learning-based regression). In particular, it takes into account both seasonality and trend components in KPI time-series data, and provides a feedback loop for retraining the prediction models using prior detection results to improve detection accuracy.

- We conduct trace-driven evaluation on CELLPAD based on our KPI dataset to explore a range of prediction algorithms and different feature selection strategies. We also show that CELLPAD achieves more accurate sudden drop detection than Twitter's time-series anomaly detection tool [2]. We make several observations, such as the accuracies of different prediction algorithms, the robustness against parameter choices, and the importance of prediction model retraining for accurate anomaly detection. We find that no single prediction algorithm is an absolute winner in both sudden drop and correlation change detection.

The source code of CELLPAD is available for download at **http://adslab.cse.cuhk.edu.hk/software/cellpad**.

The rest of the paper proceeds as follows. Section II presents the background details and analysis of our KPI dataset and motivates the need of anomaly detection. Section III presents our design of CELLPAD. Section IV evaluates different prediction algorithms and design choices of CELLPAD. Section V reviews related work. Finally, Section VI concludes the paper.

II. DATASET

In this section, we provide an overview of the KPI dataset that we collected from a production cellular network. We also motivate the need of detecting anomalies in such a network.

A. LTE Network Architecture

In this work, we focus on the 4G LTE cellular technologies. We first provide a high-level overview of an LTE network architecture. An LTE network comprises three main entities: User Equipments (UEs), the Radio Access Network (RAN), and the Evolved Packet Core (EPC). Each UE refers to a user's mobile device. The RAN comprises multiple base stations called Evolved NodeBs (eNodeBs), each of which manages the radio resources of UEs and provides UEs with wireless connectivity. The EPC comprises the Mobility Management Entity (MME), the Serving Gateway (SGW), and the Packet Data Network Gateway (PGW): the MME manages UEs' control-plane functions (e.g., user authentication, mobility management), while both the SGW and PGW manage UEs' data-plane functions (e.g., data routing). To send or receive data via the Internet, a UE first sets up a radio connection with an eNodeB and a signaling channel with the MME. It then sets up a data session with the EPC atop the radio connection, and uses the data session for data transmission.

Each eNodeB serves multiple geographical areas called *cells*, each of which covers a number of UEs. The size of each cell depends on the local user population and the radio coverage. A production LTE network typically covers thousands of cells.

TABLE I
DESCRIPTIONS OF SIX CELL-LEVEL KPIS.

KPIs	Descriptions
USER	It refers to the number of active users.
RRC	It refers to the number of radio resource control (RRC) connection requests between a UE and an eNodeB. Each RRC connection works at the control plane and carries signaling messages for managing the radio resources of the UE.
ERAB	It refers to the number of E-UTRAN Radio Access Bearer (ERAB) requests between a UE and the EPC. Each ERAB works at the data plane and carries the data traffic of the UE.
PRB	It refers to the number of physical resource blocks allocated. It indicates the radio resource usage.
THR	It refers to the data transmission throughput in the downlink direction.
DUR	It refers to the duration of active data transmission in the downlink direction.

B. Data Collection

Network administrators deploy probes in the EPC and every eNodeB to periodically collect KPI values, which will be sent to a centralized network management system (NMS). We call each collected input an *instance*, which specifies the time and value for a KPI. In this work, we collected per-cell KPI instances from the NMS of an operational LTE network deployed in a metropolitan city in China. Each instance is recorded on an hourly basis and describes the performance of a cell in the latest hour. We consider six types of KPIs, as summarized in Table I. The six types of KPIs address the cellular network performance in three aspects: (i) user population (i.e., USER), (ii) radio resource usage (i.e., RRC, ERAB, and PRB), and (iii) data transmission load (i.e., THR and DUR).

Our KPI dataset covers three collection periods for a total of 17 weeks: (i) November 7, 2016 to January 8, 2017, (ii) February 13, 2017 to March 12, 2017, and (iii) April 10, 2017 to May 7, 2017. We only select the cells that have the complete KPI data over the entire 17 weeks; in other words, each cell has a total of $24\times7\times17=$ 2,856 instances for each of the six KPIs. Finally, we identify 12,463 cells. To the best of our knowledge, our dataset is among the largest being studied in the literature (in terms of the collection period and the number of cells being covered) regarding KPI measurements in operational LTE networks.

C. A First Look at the Dataset

We first examine the statistical properties of our collected dataset, so as to understand the behaviors of the cellular network. Our observations are summarized as follows: (i) there exist strong seasonality and trend components in the dataset; (ii) some KPIs are strongly correlated; and (iii) there exist non-negligible variances in KPI values across the same hour of different days.

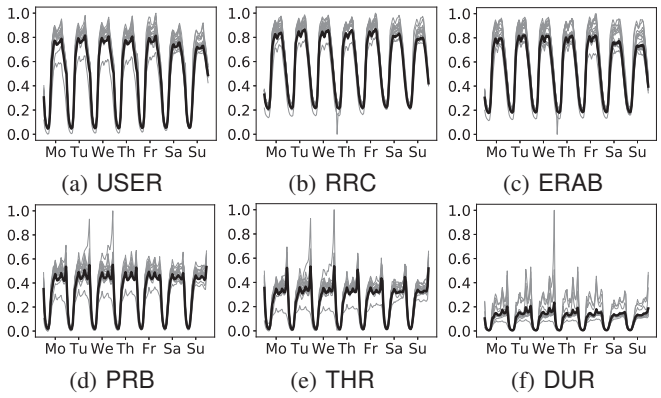


Fig. 1. Seasonality components of six KPIs. The x-axis represents the 168 hours of a week, and the y-axis represents the weekly normalized aggregated KPI value in each hour. We plot each week of KPIs separately in grey, while the black curve represents the average of 17 points in each hour.

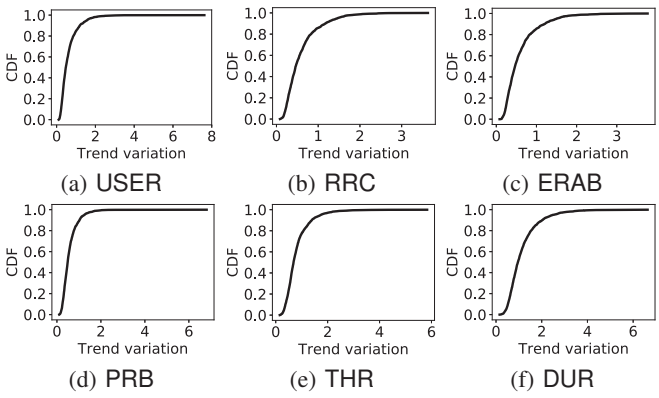


Fig. 2. Trend components of six KPIs. The x-axis represents the trend variation, and the y-axis represents the cumulative density function.

Seasonality: We first analyze the seasonality component (i.e., the recurring patterns over a time series) in all six KPIs. We first aggregate the KPI values at each hour across all cells. We then normalize each aggregate result x to the range $[0, 1]$ as $\frac{x-\min\{x\}}{\max\{x\}-\min\{x\}}$, where $\max\{x\}$ and $\min\{x\}$ represent the maximum and minimum of all 2,856 hours, respectively. Figure 1 plots the weekly normalized aggregate results for all 17 weeks. We see that all six KPIs show fairly stable diurnal patterns, albeit some abrupt increases or drops in some hours.

Trend: We next study the trend component (i.e., the increasing or decreasing patterns over a time series) in all six KPIs. We compute the trend component on a per-cell basis. Specifically, for each KPI, we compute the average KPI value of a cell at the i-th hour, denoted by y_i, over a sliding time window of 168 hours (a week) using the recent past and future KPI values, starting from the (i-84)-th hour to the (i+83)-th hour, where $i \geq 84$. We then compute the *trend variation* as $\frac{\max\{y_i\}-\min\{y_i\}}{\bar{y_i}}$, where $\max\{y_i\}$, $\min\{y_i\}$, and $\bar{y_i}$ denote the maximum, minimum, and mean of the sequence of y_i's, respectively. In our analysis, we pick the first time period from November 7, 2016 to January 8, 2017, in which we can compute 1,344 y_i's over the 9-week period. Intuitively, if the trend variation is close to zero, then the time series remains stable across any weekly cycle; otherwise, the time series has a strong trend component. For example, if the trend variation is larger than one, it means that the maximum differences between the average KPI values of any sliding windows can be larger than the overall average KPI value. Figure 2 shows the cumulative distribution of the trend variations of all cells for each KPI. We see that for each KPI, the trend variation is larger than one for a non-negligible fraction of cells.

Correlation: KPIs may be correlated; for example, if the number of active users increases, both the radio resource usage (i.e., RRC, ERAB, and PRB) and the data transmission load (i.e., THR and DUR) also increase. We compute the Pearson coefficient (PC) (a measure of linear correlation of two variables) for every pair of KPI time-series data of each cell, and obtain the average PC across all cells. If the PC is closer to 1.0, it implies that the two KPIs have high positive linear correlation. Table II shows the results. We observe all six KPIs have positive linear correlation. In particular, the pairs (RRC, ERAB), (PRB, THR), and (USER, RRC) are the top-3 pairs with the strongest correlation.

TABLE II
AVERAGE PEARSON COEFFICIENTS OF KPI PAIRS ACROSS ALL CELLS.

	USER	RRC	ERAB	PRB	THR	DUR
USER	1.000	0.895	0.907	0.829	0.771	0.817
RRC	-	1.000	0.961	0.709	0.602	0.654
ERAB	-	-	1.000	0.716	0.610	0.659
PRB	-	-	-	1.000	0.942	0.814
THR	-	-	-	-	1.000	0.776
DUR	-	-	-	-	-	1.000

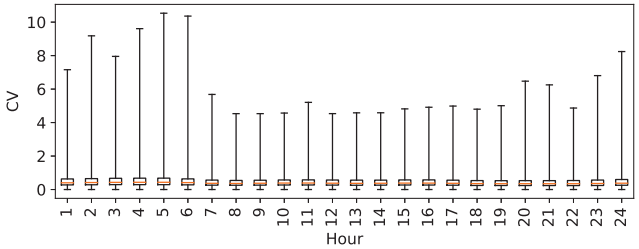
KPI variations: KPI values may fluctuate over time due to performance changes in cellular networks, thereby implying the presence of performance anomalies. To understand the frequency of such KPI variations, we calculate the coefficient of variation (CV) (i.e., the ratio of the standard deviation to the mean) of a KPI at each hour of a day for each cell. A large CV implies that the specific cell has a high deviation of the KPI. Here, we focus on USER. Figure 3(a) shows the boxplots[1] of CVs across all cells. We observe that the majority of CVs are close to zero, yet a few cells exhibit high CVs. Interestingly, we observe higher CVs during nighttime (from 23:00 to 06:00) than during daytime (from 08:00 to 18:00).

We also observe significant KPI variations in the correlations across a KPI pair in some of the cells. We calculate the PC of a KPI pair at each hour of a day for each cell. We focus on USER and RRC (which show a high PC according to Table II). Figure 3(b) shows the boxplots of PCs across all cells. While the majority of cells show a high PC (close to one), some cells show a negative PC, which is unexpected and may be anomalies.

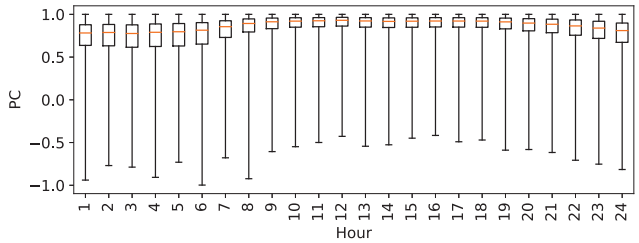
D. Definitions of Anomalies

Based on our analysis and internal communication with network administrators, we study two types of KPI anomalies,

[1]A boxplot shows the minimum, first quartile, median, third quartile, and maximum of all samples.



(a) Coefficients of variation (CVs) of USER



(b) Pearson coefficients (PCs) of USER and RRC

Fig. 3. KPI variations, in terms of boxplots at different hours of a day across all cells. Here, the x-axis represents the hour of a day (e.g., 1 means 0100).

namely *sudden drops* and *correlation changes*, that are of practical interest to cellular network management. A sudden drop refers to the sudden performance degradation of a KPI instance within a cell. For example, if there exists a sudden drop in USER, it may imply that a cell fails to provide connectivity to a significant portion of users. In general, a sudden drop happens when a KPI value is significantly less than the expected one. On the other hand, a correlation change refers to the large deviation of two correlated KPI instances within a cell. For example, a cell failure may increase the number of RRC request attempts (i.e., RRC), while the number of active users (i.e., USER) remains relatively unchanged. Thus, both sudden drops and correlation changes are complementary to each other in characterizing performance anomalies of cellular networks. In practice, if either one of the KPI anomalies persists for a prolonged period (e.g., a few hours), it may indicate the presence of network failures and requires network administrators to investigate further. In the following discussion, we propose a unified framework that can effectively detect both sudden drops and correlation changes.

Our anomaly detection focuses on a per-cell basis by inspecting the time-series instances of multiple KPIs in each cell. In this work, we do not consider the correlation across multiple cells. Also, we do not identify the root causes of the anomalies due to insufficient information in our dataset. We pose these issues as future work.

III. Design

We present CellPAD, a cellular network performance anomaly detection framework. It takes the time-series data of multiple KPIs as inputs, and detects both sudden drops and correlation changes with high accuracy by taking into account both seasonality and trend components in KPI time-series data. It also provides a feedback loop to incrementally update the prediction models based on the past detection outputs, thereby eliminating the manual efforts of specifying labeled data (i.e., ground truths) for model training.

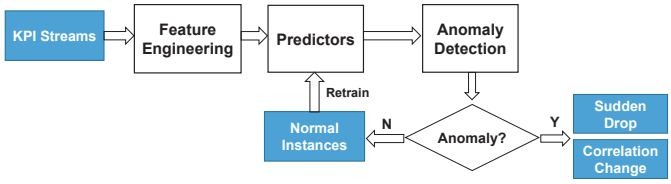


Fig. 4. CellPAD architecture.

A. Main Idea

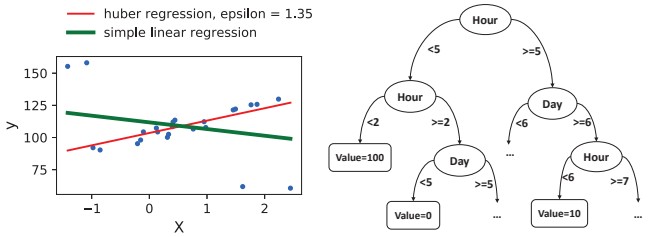
CellPAD builds on regression analysis to predict the normal values of KPI instances in order to detect anomalies. Figure 4 shows the CellPAD architecture, which provides a unified regression framework for detecting both sudden drops and correlation changes. At a high level, CellPAD takes multiple time-series streams of KPI instances at different time intervals (hours in our case) as inputs. It first performs feature engineering to extract a set of *features*, whose values are derived from the KPI instances that are observed up to the current hour. The feature values serve as inputs to different *predictors*, each of which performs a specific prediction algorithm and outputs a predicted KPI value, which is the expected value for a KPI at each hour in normal situations (i.e., without anomalies). For sudden drop detection, CellPAD returns one predicted KPI value for each KPI instance being considered, while for correlation change detection, it returns two predicted KPI values for each pair of KPI instances being considered. Finally, CellPAD performs anomaly detection based on the prediction at each hour by checking the deviations between the actual and predicted KPI values. It concludes that the current KPI instances are either anomalies (i.e., sudden drops or correlation changes) or normal instances. For the latter case, CellPAD also feeds back the normal instances to retrain the prediction models for improved detection accuracy.

One major design issue is to properly select the predictors and features. In particular, the features depend on not only what types of anomalies (sudden drops or correlation changes) being detected, but also the predictors being used. In the following, we formulate the regression framework of CellPAD in detail, in which we first state the predictors that CellPAD supports, followed by the corresponding feature engineering procedures.

B. Predictors

CellPAD supports three families of predictors: simple statistical modeling, linear regression, and tree-based regression; the latter two belong to machine-learning-based regression approaches. Each predictor returns a predicted value for each hour based on the underlying prediction algorithm. Here, we summarize the algorithms that we consider under each family.

Simple statistical modeling: CellPAD implements four algorithms:



(a) Huber regression (b) Regression tree

Fig. 5. Regression applied in CELLPAD

- *EWMA (Exponentially Weighted Moving Average)* [19]: It computes the predicted value based on the weighted values of a set of instances, such that the weights are exponentially decayed for older instances.
- *WMA (Weighted Moving Average)* [29]: Its prediction is also based on the weighted instances as in EWMA, except that the weights are linearly decayed.
- *HW (Holt-Winters)* [37]: It is a triple exponential smoothing method that extends EWMA to deal with seasonality and trend. It computes the predicted value as a function of the weighted inputs of both instances as well as the seasonality and trend components. It also estimates the seasonality and trend components from the instances using EWMA.
- *LCS (Local correlation score)* [28]: It measures the correlation of two time-series. It holds two synchronous sliding windows to compute the auto-covariance matrices continuously and then aggregates the matrices using their exponentially weighted averages. We mainly use LCS for detecting correlation changes.

Linear regression: CELLPAD implements two linear regression algorithms to model the linear relationships between features and predicted values:

- *SLR (Simple linear regression)* [5]: It computes the predicted values based on the optimal linear combination of the values of a feature that can minimize the mean square deviation.
- *HR (Huber regression)* [21]: It enhances simple linear regression to be robust against noise, by controlling whether instances are classified as outliers via an epsilon parameter (a smaller epsilon is more robust to outliers). For example, Figure 5(a) shows how Huber regression excludes outliers from modeling as opposed to simple linear regression.

Tree-based regression: To model the non-linear relationships between features and predicted values, CELLPAD also implements two tree-based regression algorithms:

- *RT (Regression tree)* [7]: It organizes the feature space into a tree structure, in which each non-leaf node is a decision-making process that splits the feature space based on a selected feature, while each leaf node holds a local predictor that averages all instances that fall into the feature partition. Figure 5(b) shows a regression tree example, in which we choose the hour and day indexes as the features (see Section III-C for details). The predicted value is 10 if the features satisfy "(Hour $==$ 5) and ($6 \leq$ Day ≤ 7)". Choosing which feature for decision making and how to split the feature space can be controlled by a set of parameters, which we omit details here.
- *RF (Random forest)* [6]: It is an ensemble learning algorithm. It samples different subsets of instances and features to form multiple regression trees and take their average prediction result. It is robust against irrelevant features and noises than a single regression tree in general.

Discussion: Simple statistical modeling is easy to implement, as it can return the predicted values based on the observed instances. However, it has the major limitation that the prediction accuracy heavily depends on the parameter settings. In contrast, both linear regression and tree-based regression are less dependent on parameters, which can be "learned" from input instances. However, they require careful feature engineering for the regression analysis, as we will explain in the next subsection.

C. Feature Engineering

We now elaborate the feature engineering process for linear regression and tree-based regression. CELLPAD extracts different features for sudden drop and correlation change detection. We also describe how we address seasonality and trend.

Sudden drops: CELLPAD uses two types of features for sudden drop detection. The first type is called *indexical features*, in which we use the time indexes of each KPI instance as features. To take into account seasonality, we index the hour and day from 0 to 23 and from 0 to 6, respectively, and use the hour and day indexes as the features (called Hour and Day, respectively). Intuitively, if we group the instances by the same Hour only, we capture daily seasonality; if we group the instances by both the same Hour and Day, we capture weekly seasonality. In this work, we mainly focus on weekly seasonality. The indexical features are mainly used by tree-based regression (see Figure 5(b)).

The second type is called *numerical features*, in which we apply some numerical operations to KPI instances to extract features. We define each numerical feature as $\langle win, oper \rangle$, in which we run $oper$ on the KPI instances in the past win weeks. For instance, $\langle 5, mean \rangle$ means that we take the mean of KPI values in the past five weeks. By sampling different values of win and types of $oper$, we can generate a number of numerical features. To account for weekly seasonality, we only pick the KPI instances with the same Hour and Day. The numerical features can be used by both linear regression and tree-based regression.

Correlation changes: For correlation change detection (say, for KPI1 and KPI2), CELLPAD trains two predictors, one for KPI1 and one for KPI2. The predictor for KPI1 (resp. KPI2) takes the value of the current instance of KPI2 (resp. KPI1) as a feature. The rationale is that if the two KPIs are correlated, each KPI instance is dependent on another KPI instance at any given time. Linear regression uses this feature for prediction, while tree-based regression additionally takes Hour and Day as features for prediction.

Trend removal: As the changes in the KPI values caused by the trend component affect anomaly detection accuracy, we provide an option of removing the trend component from the raw KPI time-series. CELLPAD removes the trend component before extracting the features based on the idea of time-series decomposition [22]. Specifically, for a given KPI instance at some hour, CELLPAD computes the average KPI value of over a sliding window of 168 hours using the recent past and future KPI values as in Section II-C (note that we do not start anomaly detection until we collect enough past KPI instances for trend removal). To remove the trend component, CELLPAD divides the raw KPI value by the computed average value and feeds the result to feature engineering. Note that we can treat the trend component as additive or multiplicative, yet we choose the latter as it achieves better detection accuracy after trend removal based on our experience. We study the effect of trend removal in Section IV.

D. Anomaly Detection

To perform anomaly detection, we first calculate the degree of deviation. For sudden drop detection, CELLPAD computes the *drop ratio* $D = \frac{\text{KPI}_a - \text{KPI}_p}{\text{KPI}_p}$, where KPI_a and KPI_p denote the actual and predicted KPI values, respectively. If D is much less than 0, it likely implies a sudden drop. To detect correlation changes of two KPIs (say, KPI1 and KPI2), CELLPAD computes the *change ratio* for KPI1 by $C_1 = \frac{\text{KPI1}_a - \text{KPI1}_p}{\text{KPI1}_p}$, and that for KPI2 by $C_2 = \frac{\text{KPI2}_a - \text{KPI2}_p}{\text{KPI2}_p}$, where KPI1_a and KPI1_p (resp. KPI2_a and KPI2_p) denote the actual and predicted KPI values of KPI1 (resp. KPI2), respectively.

CELLPAD uses the "N-sigma rule" for anomaly detection, in which an anomaly is expected to deviate from the mean by a significant number N of standard deviations. At each hour, we calculate the mean μ and standard deviation σ for the drop ratios or change ratios in the last 168 hours. We call a KPI instance a sudden drop if $D < \mu - N\sigma$, and call two KPI instances a correlation change if $C_1 \notin [\mu - N\sigma, \mu + N\sigma]$ or $C_2 \notin [\mu - N\sigma, \mu + N\sigma]$. By default, we set $N = 3$, yet we also consider different values of N for the threshold selection.

Finally, CELLPAD outputs the anomalies, or feeds back the remaining normal instances to retrain the prediction model (see Figure 4), which extracts features from the normal instances for prediction.

IV. EVALUATION

We have implemented a CELLPAD prototype in Python. For EWMA, WMA, and LCS, we implement their algorithms directly; for HW, we use the open-source code [26], which selects the optimized weights that minimize a loss function; for SLR, HR, RT, and RF, we implement them using scikit-learn [1].

We evaluate the anomaly detection accuracy of CELLPAD, and compare CELLPAD with Twitter's open-source time-series anomaly detector [2] (called TWITTER for short). We address the following questions: (i) What is the accuracy of different predictors in sudden drop and correlation change detection? (ii) How do seasonality and trend affect detection accuracy? (iii) How is CELLPAD compared with TWITTER?

A. Methodology

It is a labor-intensive task for network administrators to identify real anomalies (i.e., labels) from our dataset, which is large and complex by nature; the same problem is also reported by previous work [3], [8], [24], [36]. Thus, we resort to injecting synthetic anomalies into the raw data of our dataset for evaluation. Specifically, we randomly select 80 cells from our dataset for evaluation. We aggregate the three collection periods into a continuous 17-week period (see Section II-B). In each cell, we randomly pick 1.5% of hours and three continuous segments with a uniformly distributed length of 3 to 24 hours each to inject anomalies. For sudden drops, we decrement the KPI values of each anomaly hour by a percentage uniformly distributed from 30% to 100%. For correlation changes, we pick one of the two KPIs of each anomaly hour, and either increments or decrements its value by a percentage uniformly distributed from 30% to 100%.

We also apply a simple rule-based method to label the obvious anomalies from the dataset based on the raw values. For sudden drops, we treat a KPI instance whose raw value is 75% smaller than either one of the KPI values at the same hour and day in the past two weeks as a sudden drop. For correlation changes, we compute and rank the ratios of the values of all KPI instance pairs, and treat the top 0.5% and lowest 0.5% of pairs as correlation changes. Finally, we have roughly 3-4% of anomalies in the whole 17-week dataset in each cell, and this percentage is consistent with the real-world scenarios based on our internal discussion with network administrators.

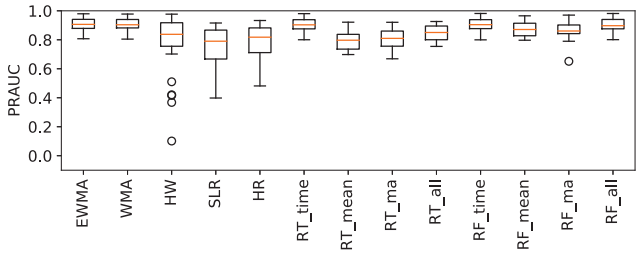
We use the first two weeks of KPI instances, including both normal instances and synthetic anomalies, to bootstrap our predictors. We then start our evaluation from the third week onwards. We do not exclude the synthetic anomalies in our bootstrapping process; instead, we rely on prediction model retraining to improve the robustness of our prediction.

B. Sudden Drop Detection

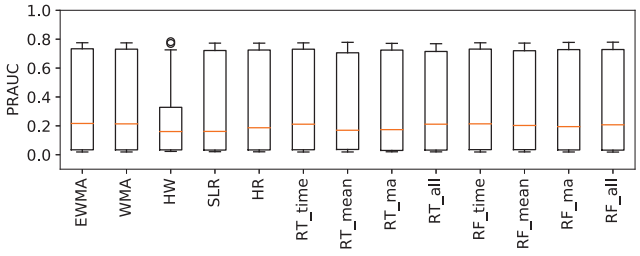
We first evaluate CELLPAD in sudden drop detection. We consider the metric *PRAUC (Area Under Precision-Recall Curve)*, which is shown to be robust when the distributions of normal instances and anomalies are highly imbalanced [15]. Here, we use the drop ratio (see Section III-D) as the prediction input to PRAUC, which computes various precision and recall pairs against different thresholds to obtain an accuracy measure between 0 and 1 (higher means more accurate). We only present the results for the KPI USER.

We consider the following predictors:

- *EWMA, WMA, and HW*: We compute the average using the values with the same hour and day indexes from the first week to the previous week. For EWMA, we set the weight to 0.8; for WMA, the weights are set based on the number of previous weeks; for HW, we set the seasonal period as 168 weeks and use it to compute the optimized weights [26].



(a) With trend removal



(b) Without trend removal

Fig. 6. PRAUC of different predictors in sudden drop detection.

- *SLR and HR*: The features are the mean and median of the values with the same hour and day indexes in the past w weeks, where w is sampled at $w = 3, 5, 7, 10, 13$; for HR, we set $\epsilon = 1.35$.
- *RT and RF*: We consider four variants for each of RT and RF. (i) RT_time and RF_time, which use the hour and day indexes as indexical features; (ii) RT_mean and RF_mean, which use the mean and median features as in SLR; (iii) RT_ma and RF_ma, which use the moving averages of both EMWA and WMA as features; and (iv) RT_all and RF_all, which use all features as described in (i), (ii), and (iii). For RF, we set the number of trees as 100.

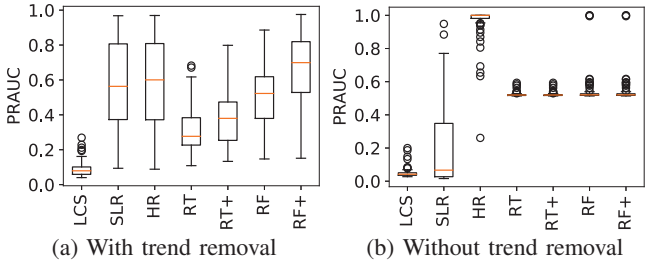
Figure 6 shows the boxplots of PRAUC for different predictors. Figure 6(a) first considers the case in which we remove the trend components. Simple statistical modeling and tree-based regression generally achieve good accuracy; for example, EWMA, WMA, RT_time, RF_time, and RF_all have an average PRAUC of more than 0.9. On the other hand, HW, SLR, and HR have low accuracy, with an average PRAUC of below 0.8. We note that RF maintains high accuracy using different features (with a mean of at least 0.86).

Figure 6(b) shows the results when we do not remove trend components. We see that the accuracy of all predictors drops significantly. This justifies the necessity of removing trend components in sudden drop detection.

C. Correlation Change Detection

We now study correlation change detection, in which we consider the following predictor implementations in CellPAD:

- *LCS*: We set the sliding window size as 20 hours and the smoothing constant as 0.8.
- *SLR and HR*: For each of the predictors of a KPI, we set the value of another KPI as the only feature.



(a) With trend removal (b) Without trend removal

Fig. 7. PRAUC of different predictors in correlation change detection.

- *RT and RF*: We consider two variants for each of RT and RF. (i) RT and RF, which use the value of another KPI as the only feature as in SLR and HR; and (ii) RT+ and RF+, which use the value of another KPI as a feature as well as the hour and day indexes as the indexical features. The rationale of using indexical features in RT+ and RF+ is to take into account weekly seasonality.

We use PRAUC as the accuracy metric. We use the average of two absolute change ratios $\frac{1}{2}(|C_1|+|C_2|)$ (see Section III-D) as the input to PRAUC. Here, we focus on the KPI pairs (USER, RRC).

Figure 7 shows the boxplots of PRAUC for different predictors. Depending on the predictors, the accuracy may be improved or degraded with trend removal. As opposed to sudden drop detection, RF does not achieve high accuracy here, even though using different features. Overall, HR without trend removal (i.e., using the raw KPI data for anomaly detection) achieves the highest PRAUC (with a mean 0.93).

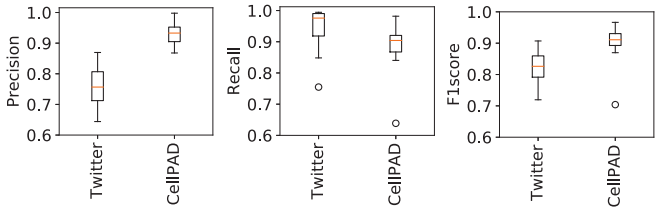
D. Comparisons with Twitter

We now compare CellPAD with Twitter [2] in sudden drop detection. Twitter is an open-source anomaly detection system that also takes into account the seasonality and trend components in the anomaly detection of time-series data. Since Twitter is designed for anomaly detection in a single time-series (as opposed to two time-series in correlation change detection), we only focus on sudden drop detection. Also, Twitter only tells if a time point is an anomaly, but does not return an anomaly measure for us to compute PRAUC for different thresholds. Thus, we consider the following accuracy metrics instead: (i) precision, (ii) recall, and (iii) F1-score (i.e., 2×Precision×Recall / (Precision + Recall)). For CellPAD, we pick RF_all (with trend removal) as the predictor.

Figure 8 compares CellPAD and Twitter in sudden drop detection for the KPI USER. CellPAD has much higher precision than Twitter, but with slightly lower recall. Overall, CellPAD achieves higher F1-score than Twitter (with means 0.90 and 0.82, respectively). One possible reason is that Twitter builds on statistical modeling, while CellPAD uses random forest regression here to achieve high accuracy; we pose further investigations as future work.

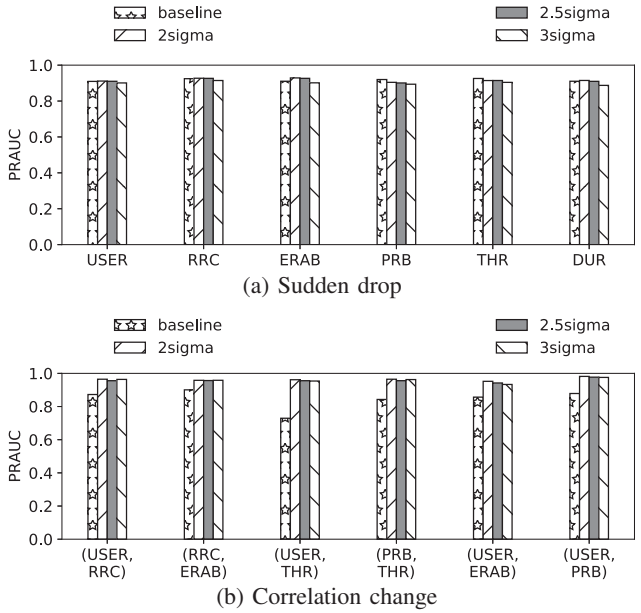
E. Effects of Model Retraining

Finally, we study the effect of retraining the predictor by feeding back the prior detection results. Here, we consider



(a) Precision (b) Recall (c) F1-score

Fig. 8. Comparisons between CELLPAD and TWITTER.



(a) Sudden drop

(b) Correlation change

Fig. 9. Effects of model retraining.

two cases: (i) the baseline case, which uses all instances (including normal instances and anomalies) to update the predictor and (ii) our CELLPAD design, which uses only the normal instances to update the predictor. In addition, we test different thresholds in anomaly detection by varying the number of standard deviations from the mean; here, we consider 2σ, 2.5σ, and 3σ. We use RF_all (with trend removal) and HR (without trend removal) as the predictors for sudden drop detection and correlation change detection, respectively.

Figure 9 shows the results for both sudden drop and correlation change detection; the former considers all six KPIs, while the latter considers six KPI pairs which show high PC (see Table II). We make the following observations. First, both RF_all and HR maintain high accuracy for different KPIs and KPI pairs in sudden drop detection and correlation change detection, respectively. Second, the baseline and CELLPAD do not show significant difference in sudden drop detection, while CELLPAD achieves higher accuracy than the baseline in correlation change detection. This justifies the need of retraining the predictor using normal instances only. Finally, we do not see significant difference for different thresholds in CELLPAD, meaning that CELLPAD remains robust in threshold selection.

F. Summary

We summarize our main findings as follows:

- In sudden drop detection, random forest regression with trend removal achieves high PRAUC using different features, although some simple statistical modeling algorithms such as EWMA and WMA can also achieve high PRAUC.
- In correlation change detection, Huber regression without trend removal achieves the highest PRAUC across all predictors.
- Trend removal improves detection accuracy in sudden drop detection across all predictors, while its accuracy varies across predictors in correlation change detection.
- CELLPAD achieves higher F1-Score than TWITTER in sudden drop detection (note that TWITTER currently does not support correlation change detection).
- Retraining the predictor with normal instances only improves PRAUC in correlation change detection.
- CELLPAD remains robust for different choices of thresholds in anomaly detection.

V. RELATED WORK

In this section, we review related work on performance characterization and anomaly detection specifically in the context of cellular networks.

Performance characterization: Several measurement studies analyze real-world traffic traces collected at the cellular network core. Most studies focus on production 3G UMTS cellular networks. For example, Qian *et al.* [30] characterize the cellular network state machine and analyze how control parameters affect radio resource usage and mobile devices' energy consumption. He *et al.* [18] and Qian *et al.* [31] study the interactions between cellular data traffic and signaling overhead. Chen *et al.* [10] uses the supervised regression approach RuleFit [16] to how the round-trip time and loss rates are influenced by different factors such as traffic load and application types. Shafiq *et al.* [32] study the performance degradations in two crowded events. Given the emergence of 4G LTE, Huang *et al.* [20] study the TCP performance based on 10-day traffic traces collected in an LTE network and identify the limitations of TCP over LTE. Our work also analyzes real-world traces based on the measurements at the network core, with specific emphasis on anomaly detection.

Anomaly detection: Some measurement studies pay special attention to anomaly detection in cellular networks. For example, Theera-Ampornpunt *et al.* [34] use classification models to predict network drops and drop duration. Chen *et al.* [11] use customer care calls to infer anomalies through regression. Ahmed *et al.* [3] infer end-to-end performance degradations in four aspects: user locations, content providers, device types, and application types, and their inference models build on robust regression and associative mining. Casas *et al.* [8] apply decision-tree-based classification for anomaly detection, and specifically focus on DNS query performance.

Prior studies perform anomaly detection based on cellular KPIs as in our work. Ciocarlie et al. [13] propose an adaptive

ensemble learning method to address concept drifts in cell anomaly detection. Some studies [4], [14], [23], [27], [33] present automated diagnosis to further identify the root causes of detected KPI anomalies. Chernogorov et al. [12] propose a data mining approach to detect unavailable cells that do not trigger alarms. Besides cellular network management, Twitter [2], [36] proposes an anomaly detection framework for long-term time-series data by addressing seasonality and trend components, yet our evaluation shows that it cannot achieve high detection accuracy as in CELLPAD based on our KPI dataset. Opprentice [25] focuses on KPI anomaly detection in a global search engine and applies machine learning techniques for anomaly detection. In contrast, CELLPAD focuses on providing a unified framework to detect both sudden drops and correlation changes, while correlation changes are not considered by any previous work.

VI. CONCLUSIONS

We study the problem of detecting performance anomalies in cellular networks, and motivate the problem based on a large-scale real-world KPI dataset collected from an operational LTE network. We present CELLPAD, a unified performance anomaly detection framework for cellular networks. CELLPAD targets two types of anomaly detection problems, namely sudden drop detection and correlation change detection. It has the following design elements: (i) support of various statistical and machine-learning-based regression algorithms, (ii) addressing the seasonality and trend patterns in anomaly detection, and (iii) providing a feedback loop for prediction model retraining. Our trace-driven evaluation demonstrates how CELLPAD achieves automated and accurate anomaly detection.

ACKNOWLEDGMENTS

This work was supported by Research Grants Council of Hong Kong (GRF 14204017) and National Natural Science Foundation of China (61572278). This work was done while Jun Wu was visiting the Chinese University of Hong Kong.

REFERENCES

[1] scikit-learn. http://scikit-learn.org/.
[2] Twitter: AnomalyDetection R package. https://github.com/twitter/AnomalyDetection.
[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 *Proc. of IEEE INFOCOM*, 2016.
[4] R. Barco, V. Wille, and L. Diez. System for Automated Diagnosis in Cellular Networks Based on Performance Indicators. *European Trans. on Telecommunications*, 16(5):399–409, 2005.
[5] S. Bolton and C. Bo. Linear regression and correlation. *Nurse Anesthesia*, 1990.
[6] L. Breiman. Random Forests. *Machine Learning*, 45(1):5, 2001.
[7] L. I. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. Classification and Regression Trees (CART). *Biometrics*, 2015.
[8] P. Casas, P. Fiadino, and A. D'Alconzo. Machine-learning Based Approaches for Anomaly Detection and Classification in Cellular Networks. In *Proc. of IFIP TMA*, 2016.
[9] V. Chandola, A. Banerjee, and V. Kumar. Anomaly Detection: A Survey. *ACM Computing Surveys*, 41(3):15, 2009.
[10] Y. Chen, N. Duffield, P. Haffner, W.-L. Hsu, G. Jacobson, Y. Jin, S. Sen, S. Venkataraman, and Z.-L. Zhang. Understanding the Complexity of 3G UMTS Network Performance. In *Proc. of IFIP Networking*, 2013.
[11] Y.-C. Chen, G. M. Lee, N. Duffield, L. Qiu, and J. Wang. Event Detection Using Customer Care Calls. In *Proc. of IEEE INFOCOM*, 2013.
[12] F. Chernogorov, S. Chernov, K. Brigatti, and T. Ristaniemi. Sequence-based Detection of Sleeping Cell Failures in Mobile Networks. *Wireless Networks*, 22(6):2029–2048, 2016.
[13] G. Ciocarlie, U. Lindqvist, K. Nitz, S. Nováczki, and H. Sanneck. On the Feasibility of Deploying Cell Anomaly Detection in Operational Cellular Networks. In *Proc. of NOMS*, 2014.
[14] G. F. Ciocarlie, C. Connolly, C.-C. Cheng, U. Lindqvist, S. Nováczki, H. Sanneck, and M. Naseer-ul Islam. Anomaly Detection and Diagnosis for Automatic Radio Network Verification. In *Proc. of MONAMI*, 2014.
[15] J. Davis and M. Goadrich. The Relationship Between Precision-Recall and ROC Curves. In *Proc. of ICML*, 2006.
[16] J. H. Friedman and B. E. Popescu. Predictive Learning via Rule Ensembles. *The Annals of Applied Statistics*, 2008.
[17] H. He and E. A. Garcia. Learning from Imbalanced Data. *IEEE Trans. on Knowledge and Data Engineering*, 21(9):1263–1284, Sep 2009.
[18] X. He, P. P. C. Lee, L. Pan, C. He, and J. C. S. Lui. A Panoramic View of 3G Data/Control-Plane Traffic: Mobile Device Perspective. In *Proc. of IFIP Networking*, 2012.
[19] C. C. Holt. Forecasting Seasonals and Trends by Exponentially Weighted Moving Averages. *International Journal of Forecasting*, 20(1):5–10, 2004.
[20] J. Huang, F. Qian, Y. Guo, Y. Zhou, Q. Xu, Z. M. Mao, S. Sen, and O. Spatscheck. An In-depth Study of LTE: Effect of Network Protocol and Application Behavior on Performance. In *Proc. of ACM SIGCOMM*, 2013.
[21] P. J. Huber. *Robust Statistics*. Wiley-Interscience, 2011.
[22] M. G. Kendall and A. Stuart. The Advanced Theory of Statistics. Vol.3: Design and Analysis, and Time-series. *Journal of the Royal Statistical Society*, 1983.
[23] 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 Trans. on Vehicular Technology*, 57(4):2451–2461, 2008.
[24] N. Laptev, S. Amizadeh, and I. Flint. Generic and Scalable Framework for Automated Time-series Anomaly Detection. In *Proc. of ACM SIGKDD*, 2015.
[25] 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 *Proc. of ACM IMC*, 2015.
[26] E. Lundquist. Implement Additive and Multiplicative Holt-Winters Time Series Forecasting Algorithm. https://github.com/etlundquist/holtwint.
[27] S. Nováczki. An Improved Anomaly Detection and Diagnosis Framework for Mobile Network Operators. In *Proc. of DRCN*, 2013.
[28] S. Papadimitriou, J. Sun, and S. Y. Philip. Local Correlation Tracking in Time Series. In *Proc. of IEEE ICDM*, 2006.
[29] B. Pfaff. *Weighted Moving Average*. Springer US, 2001.
[30] F. Qian, Z. Wang, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck. Characterizing Radio Resource Allocation for 3G Networks. In *Proc. of ACM IMC*, 2010.
[31] L. Qian, E. W. W. Chan, P. P. C. Lee, and C. He. Characterization of 3G Control-Plane Signaling Overhead from a Data-Plane Perspective. In *Proc. of ACM MSWiM*, 2012.
[32] M. Z. Shafiq, L. Ji, A. X. Liu, J. Pang, S. Venkataraman, and J. Wang. Characterizing and Optimizing Cellular Network Performance During Crowded Events. *IEEE/ACM Trans. on Networking*, 24(3):1308–1321, Jun 2016.
[33] P. Szilagyi and S. Nováczki. An Automatic Detection and Diagnosis Framework for Mobile Communication Systems. *IEEE Trans. on Network and Service Management*, 9(2):184–197, 2012.
[34] N. Theera-Ampornpunt, S. Bagchi, K. R. Joshi, and R. K. Panta. Using Big Data for More Dependability: A Cellular Network Tale. In *Proc. of HotDep*, 2013.
[35] P. Tune and M. Roughan. Internet Traffic Matrices: A Primer. In *Recent Advances in Networking*. ACM SIGCOMM, 2013.
[36] O. Vallis, J. Hochenbaum, and A. Kejariwal. A Novel Technique for Long-Term Anomaly Detection in the Cloud. In *Proc. of USENIX HotCloud*, 2014.
[37] P. R. Winters. Forecasting Sales by Exponentially Weighted Moving Averages. *Management Science*, 1960.