MDPI

Article

Online Anomaly Detection System for Mobile Networks

Jesús Burgueño [1]*, Isabel de-la-Bandera [1], Jessica Mendoza [1], David Palacios [2], Cesar Morillas [2] and Raquel Barco [1]

[1] Department of Communications Engineering, University of Malaga, Málaga, 29071, Spain (e-mail: jesusbr, ibanderac, jmr, rbm@ic.uma.es)
[2] Tupl Spain S.L., Tupl Inc., Málaga, 29010, Spain (e-mail: david.palacios, cesar.morillas@tupl.com)
* Correspondence: jesusbr@ic.uma.es

Version July 30, 2020 submitted to Journal Not Specified

Abstract: The arrival of the Fifth-Generation (5G) standard has further accelerated the need for operators to improve the network capacity. With this purpose, mobile network topologies with smaller cells are being currently deployed to increase the frequency reuse. In this way, the number of nodes that collect performance data is being further risen, so the amount of metrics to be managed and analyzed is being highly increased. Therefore, it is fundamental to have tools that automate these tasks and inform the network operator of the relevant information within the vast amount of metrics collected. In this manner, it is particularly important the continuous monitoring of the performance indicators and the automatic detection of anomalies for network operators to prevent the network degradation and users' complaints. Therefore, in this paper a methodology to detect and track anomalies in the mobile networks performance indicators in real time is proposed. The feasibility of this system is evaluated with several performance metrics and a real LTE-Advanced dataset. In addition, it is also compared with the performance of other state-of-the-art anomaly detection systems.

Keywords: Anomaly detection, network operation, LTE, self-healing

1. Introduction

In the recent years, the number of traditional users which are connected to the mobile networks has been constantly increasing. In addition, a massive increase in automatic devices from many different areas is being experienced in mobile networks [1]. This trend is referred to as Internet of Things (IoT). In this way, IoT and the arrival of 5G have driven demand for a wider range of services. To address these challenges, communications with higher data rates, more available bandwidth and lower interference have to be established. This has further accelerated the need for operators to improve the network capacity. With this purpose, the deployment of new mobile network is focused on the use of topologies with smaller cells in order to increase the frequency reuse [2]. This new trend is cited as network densification and it will enable to fulfill the more demanding user requirements over the next few years. The network densification will involve a deployment of many more base stations with a lower inter-site distance. Thus, this will allow a better resource reuse as well as an improvement in the network capacity. Likewise, each base station will provide resources to a lower number of users, improving the available bandwidth per user.

Since the number of nodes that collect data of the network performance will be highly increased, the amount of metrics to be managed and analyzed will be further risen. In this sense, each node collects hundreds of the most important network performance indicators, known as Key Performance Indicators (KPIs). Hence, it is essential to have tools that automate these tasks and bring out the

relevant information that is hidden among the vast amount of metrics collected [3]. With this objective, different approaches can be addressed. In a centralized approach, data collected in each network site are delivered to the upper network equipment, where data are analyzed by the developed tool. This approach implies a data overloading in this central equipment but also enables to use more complex tools that correlate data from different sites. On the other hand, a distributed approach allows to minimize the delay of results since the developed tool is deployed in each network site. In this case, if the developed tool has to be integrated into existing nodes so as not to shoot up the operating expenses (OPEX) of the network, it should require low computational cost.

In addition, data analysis is more important than ever as emerging functionalities are being implemented in mobile networks, such as Software Defined Networking (SDN) and Network Functions Virtualization (NFV) [4], which allow that the available network resources and services may be modified in real time. Thus, this causes an instantaneous effect in KPIs collected. Likewise, a network configuration change made by the operator as well as outside influences, such as social events, may lead to major changes in network performance. Hence, a tool that automatically detects behavior changes in KPIs before users complain about the degraded quality of experience is essential [5]. In this way, the tool should trigger an alert when the current data show an unusual change of behaviour in any KPI. Additionally, the subsequent monitoring of the KPI could notify the network operator of the current state of the KPI, i.e., if the anomaly has ended after a period of time or after a change in the network configuration. Also, the tracking of different KPIs anomalies may be highly useful to determine the root cause of the trouble.

In order to identify when the behaviour of a KPI is anomalous, it is very useful to have historical data which have been labeled. This implies that network experts have previously classified samples of a KPI into normal and anomalous samples. Nonetheless, the historical data are generally not labeled, excluding those interest cases that have been deeply studied and analyzed by the operator. Although it is difficult to get whole labeled datasets, these are essential to apply supervised machine learning algorithms since these techniques need tagged data to be trained [6]. Supervised techniques aim at learning the relationship between the input and the output of a dataset in order to forecast the output for a new input. Hence, these techniques can accurately identify the anomalous samples of a KPI when trained with datasets labeled by experts. In this manner, authors of [7] propose an anomaly detection system based on regression analysis. This system is trained with normal samples and synthetic anomalies. In this sense, a wide proportion of published studies add synthetic anomalies manually in order to get labeled datasets. After the training, regression techniques are used to forecast the next KPIs samples and the decision is taken based on the forecasting made on various KPIs. Hence, several correlated KPIs have to be used as input in order to detect network anomalies with this system. In [8], deep neural networks are used to detect outages and congested cells. This system is proposed for a centralized approach since it is computationally expensive because the large number of operations to be processed and stored for each unit in each layer of the neural network for each cell. Therefore, the OPEX would be highly increased in case this solution was deployed on the equipment of each network site.

On the other hand, unsupervised machine learning algorithms are used when there is no available labeled datasets. Thus, network expert opinion is not considered to distinguish anomalies in these cases. Concretely, these techniques aim at splitting the samples into two clusters. Given a mobile network works correctly most of the time, it is assumed that most samples are normal. Therefore, the anomalous samples will be grouped into the smallest size cluster. In this manner, an online anomaly detection methodology based on dynamic k-Nearest Neighbors (k-NN) algorithm is detailed in [9]. This technique is one of the simplest unsupervised algorithms and only occasional outliers are detected by the proposed methodology. Authors of [10] introduce an anomaly detection framework that uses self-organizing maps (SOM) and k-medoids technique. It works at the network level, using a set of KPIs from each cell in the network as input. Hence, the framework detects the cells of a scenario that have an anomalous behavior but does not detect anomalies at the KPI level. In [11], two unsupervised

online anomaly detection techniques are proposed. Both techniques estimate the temporal properties of an input data stream based on adaptive learning, but each one uses different sliding window to involve the most recent behavior in the detection. Later, they statistically evaluate the deviations and decide if the new sample is anomalous. Both methods are tested with Internet traffic datasets and the results show that, although the performance of both methods is good for non-periodic streams, it is specially poor for periodic streams because of the high false alarm rates. In this last case, the results are even worse when there are level shift anomalies, which are characterized by a temporary increase or decrease of the KPI mean over the duration of the anomaly. In this sense, level shift anomalies are much more harmful than occasional outliers in the opinion of network operators because the behavior of the network changes over a period of time and is not a one-time anomaly. Therefore, level shift anomalies have more impact on quality of experience perceived by users since the new behavior may last until the network operator realizes the reason for this change and makes a decision about it. On the other hand, [12] presents an online anomaly detection system for mobile networks. This system uses an autoregressive integrated moving average (ARIMA) algorithm to forecast the expected value of the next coming sample. Then, the system makes a decision based on the similarity between the forecast sample and the next real sample. Finally, the system is evaluated using real network datasets where synthetic anomalies have been manually added. It achieves a high precision value for chance outliers but a low precision value for level shift anomalies. In a completely different way, many authors have proposed their own unsupervised anomaly detection system and have uploaded their corresponding code to online repositories for any user can test them with different datasets. In this manner, a Python package for anomaly detection is shared in [13]. This technique focuses on identifying new different behavior patterns by using Singular Spectrum Transformation (SST). It mainly detects anomalies when the frequency of the analyzed indicator changes considerably. Other Python package is shared in [14]. In this case, the authors propose a methodology that combines different detection algorithms to create a multi-purpose system that detects any type of anomaly. Therefore, the decision will be made based on the criteria of different algorithms, which implies that many anomalies are detected by one algorithm but not confirmed at subsequent stages. Thus, the number of true anomalies detected decreases in exchange for a decrease in the number of confirmed false anomalies. In addition to Python packages, the authors of [15] present a R package to detect anomalies by using Seasonal Hybrid Extreme Studentized Deviate (S-H-ESD). This technique calculates the statistical values that are used to decide whether a sample is anomalous based on the percentage of anomalous samples that the analyzed indicator has usually. Therefore, although this is an unsupervised method, it has to be configured with the expected percentage of anomalies in the KPI. This implies that the number of anomalies indicated by the proposed method is close to the set value, resulting in poor performance when the number of actual anomalies increases noticeably or approaches zero.

Additionally, there is a last type of machine learning algorithms: semi-supervised techniques. These techniques are a hybrid of both supervised and unsupervised algorithms. In this case, a small amount of labeled data is enough for the algorithm to start working correctly. In this context, authors of [16] present a semi-supervised online anomaly detection system focused on identifying unusually low or high user traffic areas in mobile networks. Nevertheless, the false positive rate is too high (more than 14%), so the system often warns the network operator of non-existent troubles. On the other hand, [17] introduces a generic semi-supervised system to detect anomalies in real time. However, this system has to be constantly tuned using the feedback provided by experts in order to reach a good performance.

In the introduced context, this paper proposes a novel methodology to detect and track anomalies in KPIs of mobile networks in real time. It consists of two stages: offline learning of KPIs' seasonal patterns and online block to detect and track anomalies in them. Additionally, the online block includes a mechanism to automatically adapt the functioning of the whole system to new network behaviors in run time. Also, it should be pointed out that the proposed implementation is a semi-supervised technique because once the setup parameters have been configured based on a reduced set of labeled

samples, it can feasibly work without manual intervention. Given the limitations of the state-of-the-art, this study makes the following contributions:

- The methodology designed has been specially optimized to detect and track the most harmful anomalies in accordance with the requirements of mobile network operators. These most damaging anomalies are level shift anomalies and anomalies maintained over time where the KPI totally changes its trend and not just its mean. Thus, these types of anomalies have been prioritized before chance outliers. Furthermore, the proposed methodology aims at minimizing the false positive rate so that network operators only spend resources on dealing with true anomalies. Finally, once the system is run, it works in real time, i.e. it produces immediate outputs as new KPI samples arrive.
- The proposed system has been designed based on a distributed approach in order to be easily implemented with a low cost of computing and storage in the current mobile network equipment already deployed. In addition, this allows that the system can also be adapted to other use cases that require low computing and storage capability, such as IoT or Device-to-Device decentralized networks. Likewise, its use can be extended to next generation networks. Furthermore, the design system works at the KPI level, i.e. the system detects anomalies for each KPI without taking into account the rest of the KPIs. Therefore, the system is scalable and can be used with a different number of KPIs. In addition, it automatically adapts to new network behavior profiles without any manual intervention or feedback.
- To evaluate the performance of the proposed system, the measures most often cited in the literature on machine learning have been used. It is evaluated with an actual LTE-Advanced dataset where real anomalies have been labeled by network experts and no synthetic anomalies have been manually injected. In addition, the performance of the proposed system is compared with the performance of the open-source packages previously introduced in the state-of-the-art. The results demonstrate the feasibility of the system, which can lead to a reduction of OPEX.

The rest of the paper is organized as follows. In Section 2 it can be seen an introduction about anomaly detection in real time. Section 3 details the proposed system. Its subsections describe each of the parts that form the whole system. In Section 4 it is defined the methodology used to evaluate the system. This section presents the dataset and the metrics used to analyze the performance in addition to the rest of systems that will be tested as well, whereas Section 5 analyzes the results. Section 6 concludes with a summary and directions for future work.

2. Online anomaly detection

An anomaly is defined as an atypical and significantly different behavior from the previous normal behavior of a KPI during a span of time. Hence, it is mandatory to study the previous KPI behavior in order to decide if the current one is anomalous or not. An anomaly may imply a negative change in the network, such as a decrease in the cell availability. Conversely, it may also be positive, like an unusual low interference level. However, this positive fact may have been caused by a negative event in other cell, e.g., the outage of a neighbor cell. Hence, all anomalies should be indicated to the network operator.

Regarding the study of the KPIs' previous behavior, seasonal KPIs are very common in mobile networks. The periodic behavior of these KPIs reflects the typical users' behavior and the network operator management policies. Also, these KPIs are typically business-related, e.g., number of connected devices or throughput per user. On the other hand, the seasonal patterns may change or shift over time. For example, the number of connected devices may rely on the month of the year, e.g., the number of users for the same cell may be different in August than in March. Thus, it is critical that the KPIs patterns are characterized and updated over time. In this sense, authors of [18] propose a system which detects the different seasonal patterns of a KPI in real time. This system uses a density-based spatial clustering of applications with noise (DBSCAN) algorithm. Moreover, if the

traffic profile changes, the system will be automatically adapted to the new seasonal patterns. Besides, [19] proposes a system that updates the data of the normal behavior when a system is reconfigured. An algorithm based on suffix trees is developed in this case.

Finally, the requirements of an ideal online anomaly detection system are shown below:

- Detection of all the anomalies as soon as possible.
- False alerts must be minimized.
- Automatic adaptation to the new behaviors.
- No parameters tuning must be manually made at runtime.

3. Proposed methodology

The proposed system aims at deciding the state of each mobile network KPI in real time. Thus, the system will ideally be deployed in every node in order to monitor the KPIs of each cell. Figure 1 shows the diagram of the proposed, which is applicable to a single KPI. However, this can be replicated to monitor several KPIs. The system consists of an offline block and an online block. The offline block enables to initialize the system. Once it is initialized, the online block can work without manual intervention over time. The online block consists of different subsystems. Firstly, the new KPI sample is scaled. Then, if this sample is possibly anomalous, an alert is triggered by the *Alert generator* stage. Later, the *Anomaly decision subsystem* determines if the new KPI sample is definitely anomalous or not on the basis of the previous generated alerts and the KPI state. Next, the KPI state is updated in the *State machine* stage. Finally, aggregate metrics of the normal data are updated by the *Normal data update* subsystem. In the following subsections all the parts are detailed as well as the inputs of the system.

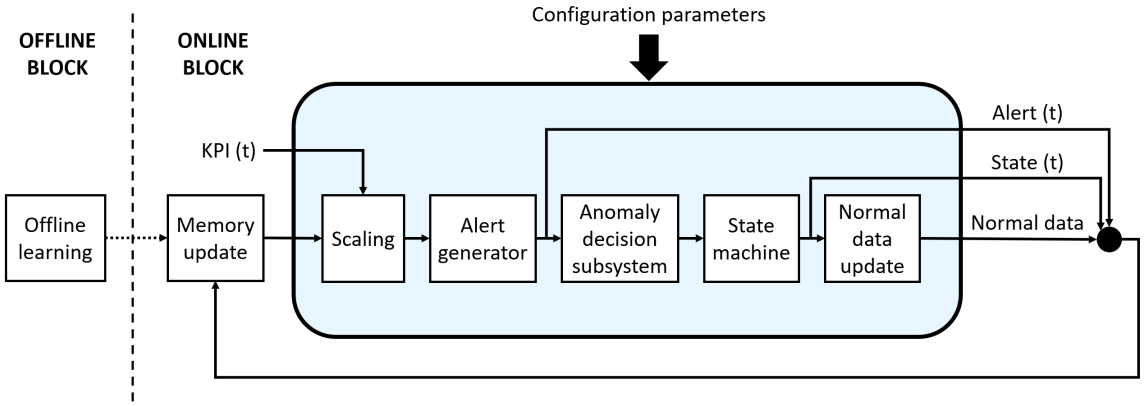

Figure 1. System diagram.

3.1. Offline block

This subsection explains the process that must be made to initialize the system. In order to carry out this process, the following two values are required:

- **'T'**: Period of the main seasonal pattern of the KPI. It can be calculated with the system proposed in [18] that was introduced in the previous section.
- **'S'**: Number of previous KPI samples that the system will use to detect new anomalous behaviors. They must have been previously storaged by the network operator. In order to ensure an ideal performance of the system, most of those data should have a normal behavior. Likewise, the more previous KPI samples the network operator can provide, the more reliable the anomaly detection will be.

The offline block aims at calculating and storing in memory each initial version of 'avg_buffer' that will be used by the online block. Each 'avg_buffer' is a buffer of 'T' values based on each KPI period. Each of the values is calculated as the mean of the previous KPI samples corresponding to

the same instant of time, e.g., if 'T' is 24 because the KPI period is 24 hours and the network operator collects one sample per hour, the first value of the 'avg_buffer' will be the mean of the previous KPI samples corresponding to midnight of the whole set of 'S' samples. Hence, 'S' samples will be used for calculating 'T' average values. Consequently, as 'S' increases, the average values will be calculated using more samples and the anomaly detection will be more reliable. However, the initial computational cost will grow.

3.2. Inputs of the online block

The online block takes three input sets. The first two sets include the new value of the KPI and the content of the memory whereas the third set consists of the configuration parameters. The three sets are detailed as following:

- **KPI (t)**: The new sample of the KPI.
- **The content of the memory**:
  - **'avg_buffer'**: Buffer that consists of the 'T' average values of the KPI for every instant of time over an entire period. Also, the system must storage two buffers in the memory for different traffic patterns. One buffer will be used on weekdays and the other one will be used on weekends and holidays.
  - **'d_buffer'**: Buffer that stores the last 'T' values of 'd'. This value 'd' represents the difference between the new KPI sample and its corresponding sample in the 'avg_buffer'.
  - **'alert_buffer'**: Buffer that stores the alerts which will be triggered over time by the *Alert generator* stage.
  - **'state_buffer'**: Buffer that stores the KPI state which will be decided over time by the *State machine*.
- **The configuration parameters of the system**:
  - **'k'**: Factor that is used in the *Scaling* stage to increase or decrease the difference between the minimum and maximum values obtained after scaling. It is an integer value.
  - **'th_low'**: Threshold that is used in the *Alert generator* stage to consider the triggering of a low alert.
  - **'th_med'**: Threshold that is used in the *Alert generator* stage to consider the triggering of a medium alert.
  - **'th_high'**: Threshold that is used in the *Alert generator* stage to consider the triggering of a high alert.
  - **'max_dif'**: Maximum difference that can exist between the new KPI sample and its corresponding average value in order to be considered as a normal sample.
  - **'max_lag'**: Number of normal samples that must be received to leave an anomaly.

3.3. Online block

Once the system is initialized, the online block is run every time a new KPI sample is received. Each of the parts that form the online block is detailed below.

3.3.1. Memory update

At first, the system memory must store the two 'avg_buffer' values which are calculated when the system is initialized. Once the online block is running, the *Normal data update* stage decides whether the corresponding 'avg_buffer' value has to be updated in each iteration. On the other hand, 'd_buffer', 'alert_buffer' and 'state_buffer' are updated in all iterations. In these cases, the new values are stored at the end of their corresponding buffer. The *Alert generator* stage stores the new 'd' value at 'd_buffer'. Likewise, this stage stores at 'alert_buffer' whether or not an alert has been triggered in this iteration. In case an alert has been generated, the type of this alert will be the stored value. Finally, the *State machine* stores the current KPI state at 'state_buffer'.

3.3.2. Scaling

This stage is responsible for scaling the new KPI sample and 'avg_buffer' in each iteration. Thus, the system works with scaled values although the memory stores the original ones.

At first, both 'min' and 'max' values are calculated as Eq. 1 and Eq. 2 show. In these equations, $\bar{X}$ and S_x indicate the mean and the standard deviation of the 'S' values that are used to calculate 'avg_buffer'. Once both values have been calculated, Eq. 3 is used to scale the new KPI sample and 'avg_buffer'. These scaled values will be used by the following stages.

$$min = \bar{X} - k \cdot S_x \tag{1}$$

$$max = \bar{X} + k \cdot S_x \tag{2}$$

$$value'(t) = \frac{value(t) - min}{max - min} \tag{3}$$

3.3.3. Alert generator

The *Alert generator* stage is mainly responsible for triggering an alert in case a possible anomaly has been detected. Concretely, this system generates an alert when the new KPI sample has a noticeable different behavior compared to its normal behavior and, simultaneously, there was not a relevant behavior change in the previous sample or in the sample from the previous period. Algorithm 1 presents the corresponding pseudocode.

Additionally, the alerts are classified into three types: low, medium and high. These levels depend on the severity of the behavior change. They can be configured with the parameters introduced above: 'th_low', 'th_med' and 'th_high'. On the other hand, both positive and negative anomalies are detected. Thus, an alert will be generated even if the KPI has been enhanced, e.g., the download throughput per user has been increased.

Algorithm 1: Alert generator pseudocode

$d = |KPI'(t) - avg\_buffer'[mod(t,T)]|$
Storing 'd' value at the end of d_buffer
if $d > th\_low$ **and** $(|d - d\_buffer[t-T]| > th\_low$ **or** $|d - d\_buffer[t-1]| > th\_low)$ **then**
    if $d > th\_high$ **then**
        Triggering High Alert
        Storing 'high' at the end of alert_buffer
    else if $d > th\_med$ **then**
        Triggering Medium Alert
        Storing 'medium' at the end of alert_buffer
    else
        Triggering Low Alert
        Storing 'low' at the end of alert_buffer
    end if
else
    Storing 'no' at the end of alert_buffer (no alert is triggered)
end if

3.3.4. State machine

Although this block, *State machine*, goes after the *Anomaly Decision Subsystem*, for understandability, it will be presented before. This stage decides the current state of the KPI. Therefore, it allows to track the state of a KPI and to know the start and the end of an anomaly. Three states have been defined:

- **Non-anomalous**: The current KPI behavior is similar to its normal average behavior.

- **Anomalous**: The current KPI behavior is anomalous.
- **Border**: The KPI is leaving an anomalous period of time. If the KPI fulfills the conditions that can be seen in detail in Figure 2 and Algorithm 2, the KPI will return to the non-anomalous state.

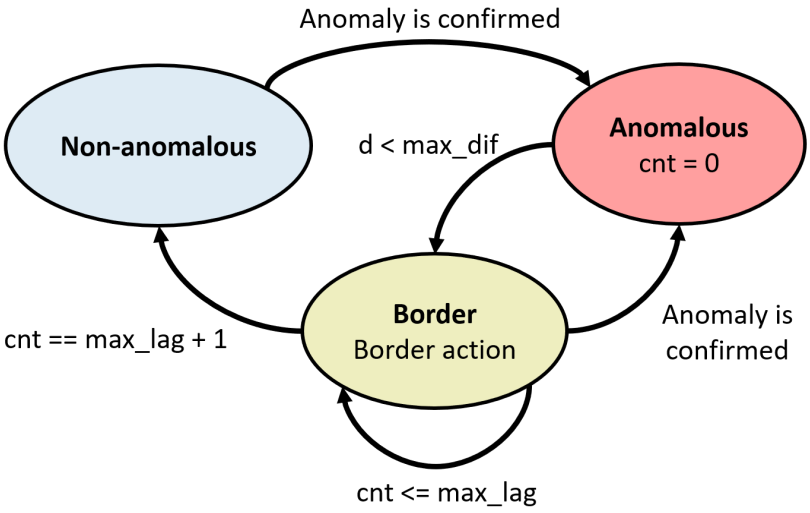

Figure 2. State machine.

Algorithm 2: Border action pseudocode

1: $zero\_samp = 0$
2: **for** $i = 1, 2, \ldots, T$ **do**
3: **if** $avg\_buffer'[i] == 0$ **then**
4: $zero\_samp = zero\_samp + 1$
5: **end if**
6: **end for**
7: **if** $d < max\_dif$ **and** $(KPI'(t) > 0$ **or** $zero\_samp > T/2)$ **then**
8: $cnt = cnt + 1$
9: **end if**

3.3.5. Anomaly decision subsystem

This subsystem confirms whether there is an anomaly based on the generated alerts and the state of the KPI in the previous instants of time. The decision is made on the basis of three conditions. These conditions are detailed in Algorithm 3.

Line 1 indicates the first condition. It confirms an anomaly if a medium or high alert has been triggered and, at the same time, another alert was generated or the KPI state was 'border' in the previous 'max_lag' samples. The second condition (line 3) confirms an anomaly if a low alert has been generated and, at the same time, a low alert was triggered in the previous sample or a higher severity alert was generated in previous 'max_lag' samples. Finally, line 5 indicates the last condition. It specifies that if no alert has been generated but the KPI state is 'border' and the KPI behavior is still different enough from the normal behavior, an anomalous state will return to be confirmed. Therefore, in order to confirm an anomaly in the beginning, it is necessary that two alerts have been triggered. This will allow to reduce the triggering of anomalies caused by sporadic outliers. In this way, the

proposed system mainly aims at detecting and tracking the anomalies that are maintained over time in accordance with the network experts' directives.

Algorithm 3: Anomaly decision subsystem pseudocode

if $(alert\_buffer[t] == high$ **or** $alert\_buffer[t] == medium)$ **and** (any $alert\_buffer[n]$ **or** $state\_buffer[n] == border$ for any $n \in [t - max\_lag, t - 1])$ **then**
    Anomaly is confirmed
else if $alert\_buffer[t] == low$ **and** $(alert\_buffer[t-1] == low$ **or** $alert\_buffer[n] == high$ **or** $alert\_buffer[n] == medium$ for any $n \in [t - max\_lag, t - 1])$ **then**
    Anomaly is confirmed
else if $state\_buffer[t] == border$ **and** $d > th\_med$ **then**
    Anomaly is confirmed
else
    No anomaly is confirmed
end if

3.3.6. Normal data update

As explained before, the seasonal patterns may change or shift over time. Hence, a mechanism that automatically adapts the system to seasons of the year with different traffic patterns is required. Therefore, 'avg_buffer' should be updated with the future incoming samples. However, since this buffer represents the non-anomalous values, it should be updated only with normal samples, disregarding those iterations in which the KPI state is 'border' or 'anomalous'. This update is carried out based on Eq. 4. In this way, the proposed system will be automatically adapted to long-term changes in the network without generating alerts.

$$avg\_buffer_{updated}[mod(t,T)] = avg\_buffer[mod(t,T)] \cdot (1 - \frac{T}{S}) + KPI(t) \cdot \frac{T}{S} \tag{4}$$

4. Evaluation methodology

This section details both the dataset and the performance metrics which are used to evaluate the feasibility of the proposed system. Given that the current use of commercial 5G networks is still quite low, tests have been carried out with data from a LTE-Advanced network, without loss of generality. Furthermore, a last subsection further details the previously introduced open-source packages whose performance is going to be compared with that of the proposed system.

4.1. Dataset

The present study has been carried out with a dataset of a real LTE-Advanced mobile network. It covers 24725 cells of a metropolitan area during 45 days. In this way, 650 performance measurements are collected for each cell with a fifteen minutes time interval. These lower level information indicators have been used to calculate 240 KPIs with one hour granularity. It should be pointed out that if the performance measurements of a time interval are not collected because of occasional technical errors, the KPI will be calculated as the average of the rest of samples collected in the same hour. Following a network trouble, the engineers identified 80 cells with an unusually high number of anomalies in 15 key business related KPIs which were chosen in accordance with the network experts' directives (Table 1). These cells had a mean of 12% anomalous samples whereas mobile networks have a mean of about 3-4% anomalous samples in real-world scenarios [7]. In addition, 300 more cells have been labeled until the mean of anomalous samples from all labeled cells is consistent with this data. Hence, 15 KPIs of 380 labeled cells with a mean of 3.8% anomalous samples have been used to carry out the tests. As regards labeling, the engineers have added a label to each hourly sample indicating whether or not it is anomalous. In this sense, the engineers have tagged as anomalies the samples that

represent occasional uncommon values that usually are collected for an hour or two straight (outliers) in addition to changes in the KPI behaviour or changes in the magnitude of the KPI values that are maintained over time and that often are correlated with anomalies in other KPIs. Finally, it should be noted that the dataset does not include synthetic anomalies.

Table 1. Labeled key business related KPIs.

ID	Description
A	Average of Channel Quality Indicator
B	Average of connected users
C	Average of download throughput per user
D	Connection restablishment attempts
E	Control Channel Element blocking rate
F	Control Channel Element usage rate
G	Download data traffic volume in Megabytes
H	Physical Resource Block usage in the downlink
I	Ping-pong handover rate
J	Restablishment scheduling requests
K	Uplink data traffic volume in Megabytes
L	Uplink Received Signal Strength Indicator for the Physical Uplink Control Channel
M	Uplink Received Signal Strength Indicator for the Physical Uplink Shared Channel
N	VoLTE call setup success rate
O	VoLTE drop call rate

4.2. Performance metrics

The evaluation of the proposed system can be approached in two different ways. A system configuration can be proposed for each KPI or a global configuration for all KPIs. In order to demonstrate the feasibility of the proposed methodology, all KPIs will be used to achieve a global configuration. For this purpose, each of the measures will indicate the performance achieved by the set of all KPIs. All performance measures can be calculated from the confusion matrix entries [20]. These entries are detailed as follows:

- **True Positive (TP)**: Number of anomalous labeled samples which are correctly classified as anomalous by the system.
- **False Positive (FP)**: Number of non-anomalous labeled samples which are wrongly classified as anomalous by the system.
- **True Negative (TN)**: Number of non-anomalous labeled samples which are correctly classified by the system.
- **False Negative (FN)**: Number of anomalous labeled samples which are wrongly classified by the system.

Hence, once the confusion matrix is obtained, six of the mostly cited performance measures in the machine learning literature can be calculated [16]. They are introduced below.

- **Accuracy**: Proportion of correctly classified samples of the total samples (Eq. 5). This metric is useful if both false positives and false negatives have similar cost.
- **Error Rate**: Proportion of wrongly classified samples of the total samples (Eq. 6).
- **False Positive Rate (FPR)**: Proportion of non-anomalous labeled samples that have been wrongly classified (Eq. 7).
- **Precision**: Proportion of correctly classified anomalous samples of the total number of anomalous classified samples (Eq. 8).
- **Recall**: Proportion of correctly classified anomalous samples of all anomalous labeled samples (Eq. 9).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{5}$$

$$ErrorRate = \frac{FP + FN}{TP + TN + FP + FN} = 1 - Acc. \tag{6}$$

$$FPR = \frac{FP}{TN + FP} \tag{7}$$

$$Precision = \frac{TP}{TP + FP} \tag{8}$$

$$Recall = \frac{TP}{TP + FN} \tag{9}$$

4.3. Other systems analyzed

In order to measure how good is the performance of the proposed system, it is going to be compared with the performance of other systems previously introduced in the state-of-the-art. These systems have the same goal than the proposed system but they use different techniques. Given that they have been published as Python or R packages, the performance of all systems can be tested with the introduced dataset. Therefore, the pros and cons of each system will be exposed when tested in the following section, allowing to draw more precise conclusions about the proposed system. These systems are described below:

- **Banpei**: This technique focuses on identifying new different behavior patterns and level shift anomalies maintained over time by using SST [13]. Thus, the detection of occasional outliers will be minimum.
- **ADTK**: This system uses different detection techniques to identify the maximum number of anomalies and a subsequent module combines the multiple lists of anomalies into the definitive one [14]. Hence, many anomalies may be detected by one or more techniques but not confirmed at the subsequent stage. The different techniques that the system uses are the autoregression, detection based on percentiles, sliding window and Principal Component Analysis (PCA).
- **Hochenbaum**: It uses S-H-ESD to detect any type of anomaly [15]. This proposed technique calculates the statistical values that are used to identify anomalies based on the average percentage of anomalous samples that each KPI usually has. In this way, this system has been configured with the mean percentage of anomalous samples in the dataset, i.e., 3.8% of anomalous samples. However, the dataset used is unbalanced as indicated in *Dataset* subsection, so this system might not reach an optimal performance.

5. Results

In this section, the proposed system is evaluated with several tests. A configuration of the parameters of the online block is then proposed. Finally, a comparison between the proposed system with this settled configuration and the other introduced systems is addressed, and the final conclusions are remarked.

To that end, the 380 labeled cells have been divided into twenty sets of the same size in order to apply the k-fold cross-validation technique [21]. This technique aims at demonstrating both the reliability and the feasibility of the system with different training data in addition to maximizing the use of the available labeled data. Nevertheless, unlike most cases where this technique is applied, the least amount of data (one of twenty sets) will be used to setup the configuration parameters of the online block in this study in order to demonstrate that few labeled data are required for reliable system performance. In this way, the proposed methodology could be considered as a semi-supervised

technique. Hence, network operators would not have to spend a lot of resources on classifying data if they use the proposed method.

Before testing, 'S' value must be indicated based on the dataset size limitation. As the dataset contains 45 days in this study, 10 days will be used in order to have a considerable percentage of data to calculate 'avg_buffer' and a high amount of data for the tests. Therefore, 240 samples will be used as 'S' value to calculate the normal behavior and to initialize the system.

Since each of the twenty training data sets of the k-fold cross-validation technique is labeled, it is possible to calculate the optimal configuration of the parameters of the online block for each of these training data sets. Figure 3 shows all the optimal configurations that have been calculated for these twenty tests. It shows that these values are very similar even though the training data are not the same. Both 'k' and 'max_lag' values obtained have been the same for all the tests. On the other hand, it should be pointed out that 'th_high' is manually configured once the rest of the configuration parameters have been calculated because it only decides the severity of the anomalies already confirmed. Once the optimal configuration of the parameters of each training data set has been calculated, a single configuration of the parameters of the online block is proposed to be evaluated with the 380 labeled cells. This parameters configuration is proposed with the objective of achieving a balanced performance of all the analyzed metrics. Therefore, the median values of the previous tests results have been proposed as configuration parameters.

Next, Figure 4 represents the system performance that the proposed configuration and the other configurations of the previous twenty tests achieve with the 380 labeled cells. The results indicate that the proposed system reaches a high level of accuracy and, therefore, a low error rate. Likewise, the recall is high at the same time as the FPR is low, which implies that most of the anomalies in the dataset are identified without triggering the number of false alerts. And finally, even though the precision is less than the recall, most of the anomalies indicated by the proposed system are real. On the other hand, the variance is low for all the performance metrics except for the recall, which is slightly higher. This higher variance is given by the difference between a configuration that risks more or less at the time of identifying anomalies. In this sense, it is possible to modify the proposed configuration in order to decrease the FPR at the expense of decreasing the recall according to the network operator interests. Finally, the figure shows how the proposed configuration achieves a balanced performance compared to the performance obtained with the other configurations.

In addition, a graphical user interface has been proposed to monitor each KPI. It shows the triggered alerts as well as the KPI state. In this sense, the triangles represent the generated alerts that have not been considered as anomaly as can be seen in Figure 5. This figure shows an example of visualization where there are two level shift anomalies in the KPI H (Physical Resource Block usage in the downlink). The colors represent the severity of the alert or anomaly. Likewise, the KPI is colored with a lighter green when its state is 'border'. Therefore, this interface would provide an user-friendly work environment for the network engineers. Also, this allows to check that the anomalies have been correctly identified.

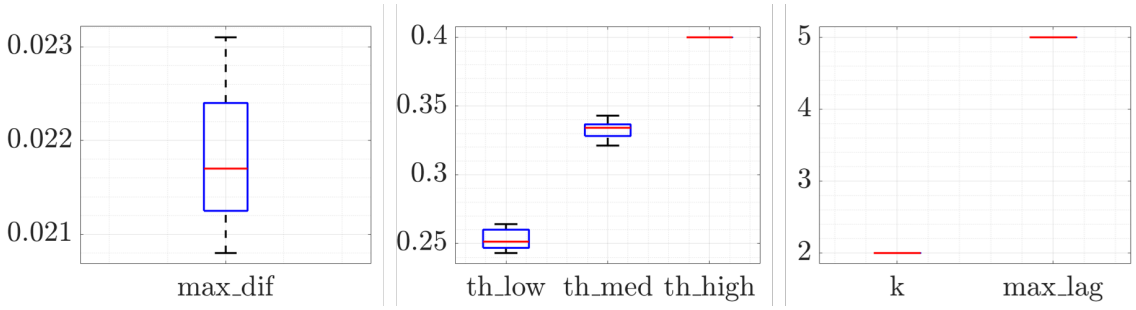

Figure 3. Optimal configuration of the parameters of the online block for each of the twenty training data sets of the k-fold cross-validation technique.

Figure 4. System performance that the proposed configuration and the other configurations of the previous twenty tests achieve.

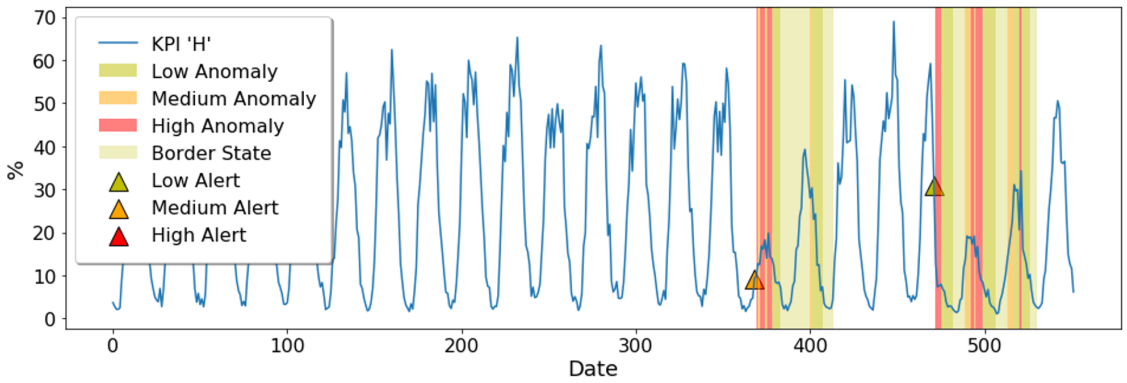

Figure 5. Example of visualization of the graphical user interface.

Finally, Figure 6 shows a comparison of the performance of the proposed system and the rest of introduced systems with the same dataset. Firstly, it should be noticed that the accuracy and, therefore, the error rate is similar for the proposed system and for Banpei because they are mainly focused on identifying level shift anomalies and anomalies with different behavior patterns maintained over time, which are the most relevant in mobile networks. On the other hand, Hochenbaum achieves the worst performance in terms of these metrics because its use is improper for an unbalanced dataset. This is also reflected in FPR, since the number of FP will be high for KPIs that have a low level of anomalous samples. On the contrary, ADTK achieves the lowest FPR since its requirement to decide whether a sample is anomalous or not is the hardest because of the combination of different anomaly detection techniques. This requirement is also noticed in the high precision of the method in the anomalies

identified by the system. However, ADTK achieves the lowest recall since many anomalies are not confirmed by the last stage of the system. Hence, most network anomalies are not identified by ADTK. Otherwise, the proposed system reaches the highest FPR because the tracking of a KPI implies that some samples are identified as anomalous after an anomaly finishes and the KPI returns to its normal state. In addition, the proposed system also generates alerts when occasional outliers are received. Therefore, although the precision of the proposed system is not the highest one due to the number of FP, it is the system that reaches the highest recall with difference over the rest of systems. Hence, the proposed system identifies the most network anomalies while maintaining a high level of precision, which is important to achieve the reduction of OPEX in mobile networks. In this sense, the main difference between the proposed system performance and Banpei's performance is that the latter only identifies the most severe anomalies maintained over time, so its FPR is lower and its precision is higher in exchange for a much lower recall. Finally, Hochenbaum does not stand out either in terms of precision and recall.

In a nutshell, although Banpei and ADTK achieve low FPR and high precision, they don't identify enough anomalies. Regarding Hochenbaum, its performance is poor for this use case where some indicators may have many more anomalous samples than others indicators. Thus, sometimes it will detect most of the anomalies while triggering many FP and other times it will identify few anomalies. Finally, even though the proposed system obtains a higher FPR and a slightly lower precision than other systems, it identifies almost 80% of all network anomalies. This enables that the overall performance of the proposed system is the best of the analyzed methods for a use in mobile networks. In this sense, the proposed system allows the network operator to be aware of the most network anomalies, which implies that a further optimization can be achieved in increasingly complex mobile networks.

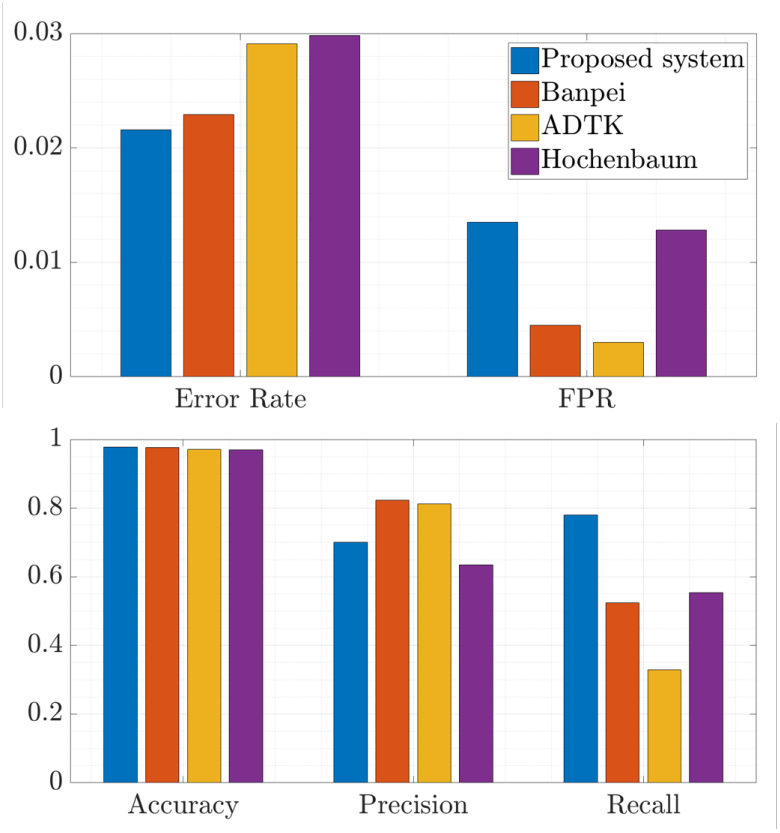

Figure 6. Comparison of each system performance.

6. Conclusions

In this paper, a system to detect anomalies in mobile networks in real time has been presented. The system consists of several stages that allow to distinguish and track harmful anomalies from occasional outliers. Specifically, the system is focused mainly on the detection of new different behavior patterns and level shift anomalies maintained over time according to network experts' directives. With this purpose, the system uses the seasonal patterns of the KPIs. The system has been tested with a large dataset from a live LTE-Advanced network where real anomalies have been labeled by network experts. Its performance has been then compared with the performance of other state-of-the-art anomaly detection systems.

Results have shown that the system can work feasibly without manual intervention once the configuration parameters have been setup based on a reduced set of labeled samples. The analyzed performance metrics show that the proposed methodology enables to identify most network anomalies, maintaining a high precision and a low level of false positives. In this sense, the proposed system takes a qualitative leap forward with respect to the rest of systems analyzed.

In addition, the proposed system might be integrated into the current deployed equipment throughout the mobile network because of its low computational complexity, which can lead to a reduction of OPEX. Also, the proposed methodology can be extended to different radio access technologies. Likewise, it is possible to add a following phase that automatically uses this information to correlate anomalies of different KPIs in order to identify the root cause of the anomaly. The design of this block is left for future work.

Author Contributions: The contribution of authors are: Conceptualization, J.B. and I.d.-l.-B.; methodology, J.B. and C.M.; software, J.B. and J.M.; validation, J.B. and C.M.; formal analysis, J.B.; investigation, J.B. and J.M.; resources, R.B.; data curation, D.P. and C.M.; writing–original draft preparation, J.B.; writing–review and editing, I.d.-l.-B., D.P. and R.B.; visualization, J.B.; supervision, I.d.-l.-B. and R.B.; project administration, R.B.; funding acquisition, R.B. All authors have read and agreed to the published version of the manuscript.

Funding: This work has been performed in the framework of the project NEREA (RTC-2017-6661-7), receiving funds from the Spanish Ministry of Economy and Competitiveness.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

ARIMA	Autoregressive Integrated Moving Average
CAPEX	Capital Expenditures
DBSCAN	Density-based Spatial Clustering of Applications with Noise
FN	False Negative
FP	False Positive
FPR	False Positive Rate
IoT	Internet of Things
KPI	Key Performance Indicator
NFV	Network Functions Virtualization
OPEX	Operating Expenses
PCA	Principal Component Analysis
SDN	Software Defined Networking
S-H-ESD	Seasonal Hybrid Extreme Studentized Deviate
SOM	Self-organizing Maps
SST	Singular Spectrum Transformation
TN	True Negative
TP	True Positive

References

1. L. Chettri and R. Bera, "A Comprehensive Survey on Internet of Things (IoT) Toward 5G Wireless Systems," *IEEE Internet of Things Journal*, vol. 7, no. 1, pp. 16-32, Jan. 2020.
2. J. G. Andrews, S. Buzzi,W. Choi, S. V. Hanly, A. Lozano, A. C. K. Soong, and J. C. Zhang, "What Will 5G Be?," *IEEE Journal on Selected Areas in Communications*, vol. 32, pp. 1065–1082, June 2014.
3. A. Imran, A. Zoha, and A. Abu-Dayya, "Challenges in 5G: how to empower SON with big data for enabling 5G," *IEEE Network*, vol. 28, no. 6, pp. 27–33, 2014.
4. G. C. Valastro, D. Panno, and S. Riolo, "A SDN/NFV based C-RAN architecture for 5G Mobile Networks," *2018 International Conference on Selected Topics in Mobile and Wireless Networking (MoWNeT)*, pp. 1-8, June 2018.
5. A. Asghar, H. Farooq, and A. Imran, "Self-Healing in Emerging Cellular Networks: Review, Challenges, and Research Directions" *IEEE Communications Surveys Tutorials*, vol. 20, no. 3, pp. 1682–1709, 2018.
6. R. Li, Z. Zhao, X. Zhou, G. Ding, Y. Chen, Z. Wang, and H. Zhang, "Intelligent 5G: When Cellular Networks Meet Artificial Intelligence," *IEEE Wireless Communications*, vol. 24, no. 5, pp. 175–183, 2017.
7. J. Wu, P. P. C. Lee, Q. Li, L. Pan, and J. Zhang, "CellPAD: Detecting Performance Anomalies in Cellular Networks via Regression Analysis," *2018 IFIP Networking Conference (IFIP Networking) and Workshops*, pp. 1-9, 2018.
8. B. Hussain, Q. Du, S. Zhang, A. Imran, and M. A. Imran, "Mobile Edge Computing-Based Data-Driven Deep Learning Framework for Anomaly Detection," *IEEE Access*, vol. 7, pp. 137656-137667, 2019.
9. R. Song and F. Liu, "Real-time anomaly traffic monitoring based on dynamic k-NN cumulative-distance abnormal detection algorithm," *2014 IEEE 3rd International Conference on Cloud Computing and Intelligence Systems*, pp. 187-192, Nov 2014.
10. X. Qin, S. Tang, X. Chen, D. Miao, and G. Wei, "SQoE KQIs anomaly detection in cellular networks: Fast online detection framework with Hourglass clustering," *China Communications*, vol. 15, pp. 25–37, 10 2018.
11. O. Ibidunmoye, A. Rezaie, and E. Elmroth, "Adaptive Anomaly Detection in Performance Metric Streams," *IEEE Transactions on Network and Service Management*, vol. 15, pp. 217-231, March 2018.
12. M. Wang and S. Handurukande, "A Streaming Data Anomaly Detection Analytic Engine for Mobile Network Management," *2016 Intl IEEE Conferences on Ubiquitous Intelligence Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress*, pp. 722-729, July 2016.
13. Banpei. Available online: https://github.com/tsurubee/banpei (accessed on 26 July 2020).
14. Anomaly Detection Toolkit (ADTK). Available online: https://adtk.readthedocs.io/en/stable (accessed on 26 July 2020).
15. J. Hochenbaum, O. S. Vallis, and A. Kejariwal, "Automatic anomaly detection in the cloud via statistical learning," *arXiv preprint arXiv:1704.07706*, 2017.
16. B. Hussain, Q. Du, and P. Ren, "Semi-supervised learning based big data-driven anomaly detection in mobile wireless networks," *China Communications*, vol. 15, pp. 41-57, April 2018.
17. M. R. Alam, I. Gerostathopoulos, C. Prehofer, A. Attanasi, and T. Bures, "A Framework for Tunable Anomaly Detection," *2019 IEEE International Conference on Software Architecture (ICSA)*, pp. 201-210, March 2019.
18. N. Zhao, J. Zhu, Y. Wang, M. Ma, W. Zhang, D. Liu, M. Zhang, and D. Pei, "Automatic and Generic Periodicity Adaptation for KPI Anomaly Detection," *IEEE Transactions on Network and Service Management*, vol. 16, pp. 1170-1183, Sep. 2019.
19. F. Rammig and K. Stahl, "Online Behavior Classification for Anomaly Detection in Self-X Real-Time Systems," *2014 IEEE 17th International Symposium on Service-Oriented Real-Time Distributed Computing*, pp. 334-341, June 2014.
20. A. Tharwat, "Classification assessment methods: a detailed tutorial," *Applied Computing and Informatics*, 9 2018.
21. S. Yadav and S. Shukla, "Analysis of k-Fold Cross-Validation over Hold- Out Validation on Colossal Datasets for Quality Classification," *2016 IEEE 6th International Conference on Advanced Computing (IACC)*, pp. 78-83, 2016.