

Using Blackbox ML Techniques to Diagnose QoE Problems for an IPTV Service

Jawwad Ahmed
Ericsson Research
Ericsson AB
Kista, Sweden

jawwad.ahmed@ericsson.com

Amadeu do Nascimento Júnior
Ericsson Research
Ericsson Telecomunicações S.A.
Indaiatuba, Brazil

amadeu.nascimento.junior@ericsson.com

Caner Kilinc
Ericsson Research
Ericsson AB
Kista, Sweden
caner.kilinc@ericsson.com

Dan Pan
Performance Management Networks
Telenor Sverige AB
Stockholm, Sweden
dan.a.pan@telenor.se

Jaume Rius i Riu
Performance Management Networks
Telenor Sverige AB
Stockholm, Sweden
jaume.riusiriu@telenor.se

Jorgen Gustafsson
Ericsson Research
Ericsson AB
Kista, Sweden
jorgen.gustafsson@ericsson.com

Abstract— IPTV services continue to disrupt the traditional cable-based TV delivery and grow in popularity. However, they also face challenges related to the delivery of stable and problem free video streams to the end users while minimizing downtime due to the inherent complexity and uncertainty associated with the IP based networks. In this work, we share our experience on addressing this service assurance problem for a commercial IPTV service provider. The key focus here is on the root cause inference aspects of the service assurance aimed at providing QoE problem hints with localization in the delivery network to the technicians of network service provider. To solve this problem, we investigate various ML-based model interpretability approaches to extract insights from predictive models built for the alarm generation. Key focus is on Blackbox model interpretability approaches namely LIME and SHAP with their associated pros and cons while at the same time comparing these techniques to the popular model agnostic (i.e., Whitebox) way of explainability based on XGB. One of the findings is that SHAP Blackbox technique is most suitable for diagnosing IPTV QoE problems. However, having the downside of long computational time, for cases where the predictive model is not tree-based such as XGB or Random Forest.

Keywords— Machine Learning (ML), STB (Set-top-box), Internet Protocol Television (IPTV), Service Assurance, Prediction, Quality of Experience (QoE), LIME (Local Interpretable Model-agnostic Explanations), SHAP (SHapley Additive exPlanations), Bayesian Network (BN)

I. INTRODUCTION

Globally, Internet traffic volume is growing exponentially since 4G (4th Generation) mobile networks support higher speed IP connectivity services and the number of internet users with broadband access are rapidly increasing in general. One of the major reasons for this explosive increase of Internet traffic is online video streaming services. In the mean-time there is also a trend of users moving away from traditional cable-based TV distribution to modern IPTV (Internet Protocol Television) [4] based services delivered through the Internet. This trend also called as ‘cord-cutting’ is indeed one of the major contributors to the rapid increase in the Internet video traffic in recent times. The main benefit for the users is the choice it provides in selecting various IPTV packages available from different service providers at more competitive prices. Also, it provides higher flexibility in using the service on various devices such as STB (Set-top-box) connected to TV, mobile devices, and personal computers etc. However, despite all the flexibility that IPTV provides, there are certain challenges associated particularly related to service assurance

i.e., making sure that service disruption is minimized for the end-users during the live operation. Of course, service assurance is important for non-IP based TV networks as well. However, due to the complexity and shared nature of IP-based networks (i.e., best-effort delivery) this task becomes significantly more challenging.

Service assurance in IPTV networks demand proactive detection of any potential QoE (Quality of Experience) related issues that may degrade the viewing experience for its users and then executing a diagnostic check on finding the main underlying cause of problem so that a preventive measure can be taken to resolve the problem. Ideally, these QoE problems should be detected promptly and problem diagnosis should be performed in a fast manner to minimize the downtime of the service. Traditional techniques of QoE problem detection and diagnosis are mostly based on manual work from humans. For example, to observe the service dashboard in the control room and then in case a service problem is detected to manually analyze various service (and delivery) network parameters to identify the problem. It is a tedious manual and error-prone process requiring significant human effort resulting in increased costs and debug time for the IPTV service provider in the event of an SLA (service level agreement) breach.

To resolve the above-mentioned problems and minimize the service downtime time, the crucial phases of service QoE problem detection, and problem diagnostic must be automated [1]. This is where modern data-driven approaches such as machine learning (ML) can help to solve this problem in an effective manner. On this front, a lot of work has been done in the literature on the IPTV service QoE problem detection and diagnostic [5]-[8]. Some of the existing work involves developing ML models for QoE prediction based on network parameters and KPIs while others are using system measurements crafted by the domain experts.

However, the automated diagnostic of QoE related issues is still a challenging task and is of key importance if the IPTV service downtime is to be minimized. In this paper, we focus on a specific service assurance task related to problem diagnostic right after a QoE problem is predicated to occur with a certain lead time. It refers to the diagnostic of the problem with some hints such as the location in the network, device or component most likely involved in the problem so that a suitable solution to resolve the problem can be approached in a swift manner. We investigate some promising ML-based model explainability techniques

recently proposed in the literature with a special focus on model agnostic techniques such as SHAP (SHapley Additive exPlanations) values [15], anchor generation [12], LIME (Local Interpretable Model-agnostic Explanations) [11], what-if analysis tools [16]. We assess how they can help us with an IPTV QoE problem diagnostic. We share our experience with the applicability of these techniques on a commercial IPTV service provided by Telenor as a service operator. Although, we mainly focus on performance, but we also briefly touch on reliability and deployment aspects. Our main objective is to design an effective and low overhead automated ML-based root cause inference sub-system which can be implemented as an add-on module of a real-time service assurance system for an IPTV service provider. Other key components of the service assurance system such as QoE problem detection, actuation and dashboarding will be discussed in a separate work.

This paper is organized as follows: in Section II, we briefly describe the IPTV service use case and the network architecture used for service delivery. Furthermore, we give an overview of the network segment used for the studies and details about the different types of data gathered from network elements during the live operation of the IPTV service. In Section III, we briefly describe the concept of Whitebox and Blackbox model interpretability and describe the techniques investigated in this paper. In Section IV, we first formulate the problem of IPTV QoE prediction, and then describe the approach to do the root cause inference (RCI) once a QoE problem is predicted. In Section V, the results obtained for different learning approaches are presented along with some discussions, and Section VI concludes the paper.

II. IPTV USECASE

A traditional IPTV service consists of four domains: content provider, service provider, network provider, and end-user. Telenor has the role of service and network provider, thus being responsible for delivering the content with an adequate QoE for its' subscribers. Telenor is a leading IPTV provider in the Nordic countries, and it is currently delivering service for more than 2.7 million customers. This paper uses data from a small portion of Telenor's IPTV customer base in in Sweden but we plan to extend our analysis to a significantly larger sample size in the near future. The households used for this study belongs to a network segment composed of one backbone router, two aggregation switches, and a building switch as shown in Fig. 1. The backbone router connects the IPTV headend to the delivery network (composed of the Aggregation Switch 1 and Aggregation Switch 2) which in turn delivers the IPTV streams to households via the building switch (i.e., Building Switch). All these devices, including the household WiFi and STB devices are the network observation points (except the building switch). Data is collected for a total of three months spanning a time period from end of 2018 to start of 2019. It is a challenging scenario in the sense that different IPTV service users have widely different daily activity levels and service problems, resulting in a heavily imbalanced dataset. Next, we describe each data source namely called STB, WiFi and Links Traffic on the router and aggregation switches and its main characteristics.

A. QoE: STB Dataset

Each STB has a proprietary probe that reports objective measurements of QoE every 5 minutes. The overall QoE level for any client for a given period (5 minutes in the current

study) is summarized by 4 different QoE classes i.e., 'Unavailable', 'Major', 'Minor' and 'OK'. Where each performance metric varies between zero and one. Each QoE class depicts for how much time the service was in a state associated with that class for the current time-window. At a given time within an observation window the service state can only belong to one QoE class so the values of all QoE classes must sum up to one for any valid observation period for any active STB. As for the semantics of different QoE classes the 'OK' class represents the fraction of time the service performance was satisfactory. While QoE classes 'Unavailable', 'Major' and 'Minor' are self-explanatory representing different types of QoE problems related to the service, ranging from complete service unavailability to only minor problems. Actual categorization of each service QoE problem to these QoE classes is automatically handled by a proprietary IPTV QoE estimation software. QoE is measured only when users are active (e.g., watching TV). In the inactivity periods, STBs report zero values for all QoE classes, which allows to track the activity level for different users for

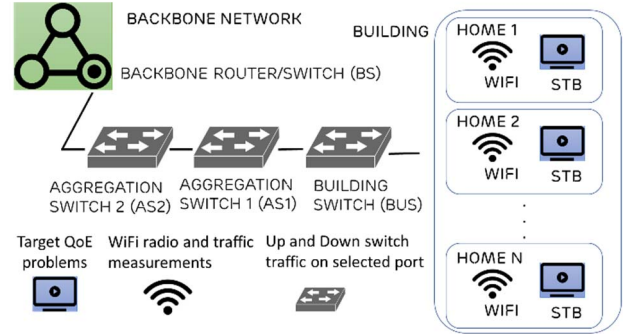


Fig. 1. Topology for IPTV delivery network.

analysis. Activity level for different users may vary significantly across households. Similarly, the frequency of QoE problem occurrences varies significantly per user which means some users are much more effected by the QoE problems than the other users. Above observations imply that we have significantly more STB data for some users which are very active to very small amount of data for some users which are mostly inactive during the observation period. Also, data is very unbalanced where many users do not experience any QoE problems most of the time so data samples where they do experience QoE problems is significantly less, bringing challenges for learning high quality ML models.

B. WiFi Radio Dataset

In addition to QoE measurements, WiFi radio data is also collected from each household every 15-seconds periodically. The performance metrics reported from each WiFi device are: 1) Medium available (the amount of available bandwidth in percentage during a time interval); 2) Physical rate; 3) Channel number and width; 4) Broadcasts packets received and transmitted; 5) Discarded packets received and transmitted; 6) Error packets received and transmitted; 7) Received and transmitted kilobytes; 8) Multicast packets received and transmitted; 9) Unicast packets received and transmitted; 10) Overall number of packets received and transmitted; and 11) WiFi frequency.

Also, STBs and WiFi devices report measurements on different time scales. To merge both data sources, we aggregate WiFi data with fine-grained resolution to match the coarse resolution of the STB data. Key statistics for some WiFi metrics computed from test data are shown in Table I.

TABLE I. Key statistics of important WiFi metrics from test data (Normalized with a scaler computed from the training data)

	Channel	ChannelWidth	Medium_available	RX.Errors	RX.KBytes	RX.Packets	TX.Errors	TX.KBytes	TX.Packets	RSSI
MEAN	0.510774	0.326519	0.777408	31.544760	0.401971	0.375839	0.699377	0.998381	0.553724	0.475602
STD	0.615181	0.468962	0.152567	86.052423	0.375029	0.375229	1.903619	0.028007	0.551310	0.230725
MIN	0.000000	0.000000	0.150499	0.000000	0.000062	0.000063	0.000000	0.201710	0.000300	0.086379
25%	0.000000	0.000000	0.696148	0.000000	0.038283	0.037452	0.000000	1.000000	0.094199	0.199336
50%	0.187500	0.000000	0.818919	0.000000	0.256397	0.164740	0.000000	1.000000	0.247729	0.559801
75%	1.000000	1.000000	0.882231	0.000000	0.767841	0.753788	0.000000	1.000000	0.968469	0.654485
MAX	1.562500	1.000000	1.000000	291.000000	0.982799	0.966607	6.000000	1.000000	1.555124	0.901993

C. Links Traffic Throughput Dataset

Additionally, data is also collected from packet switches and router as shown in Fig. 1. From the packet switches and the router (AS1, AS2 and BS), we have aggregated throughput information of the incoming and outgoing traffic through the device ports on a 5 minute interval.

A Pearson correlation of a key IPTV QoE variable from user STB devices ('ok') with the other features from the data is shown in Fig. 2. As it can be observed that none of the input features show a strong correlation with this QoE variable. Some of the of most correlated variables show a correlation 0.30 at most both in the positive direction (e.g., 'channel' and 'medium_available') and negative direction (e.g., 'TX.Discards' and 'TX.Errors').

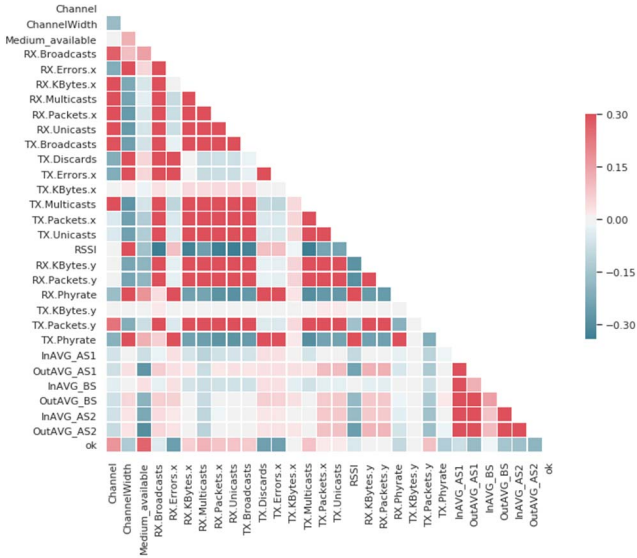


Fig. 2. Pearson Correlation of target variable ('ok') with input features.

III. MODEL INTERPETATION TECHNIQUES

In general, model interpretability or explainability techniques can be classified into model-based (Whitebox) or model agnostic techniques (Blackbox). Both approaches have been active area of research and has been explored quite extensively in the literature. In this section, we briefly look into the underlying concept behind these two main

approaches and describe some specific techniques that we will later use for problem diagnostic of the IPTV service.

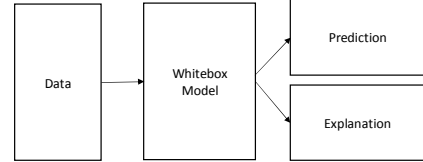


Fig. 3. Whitebox Model for cause inference.

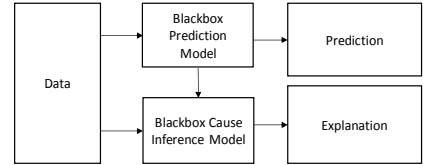


Fig. 4. Blackbox Model for cause inference.

A. Whitebox Methods

Some ML algorithms used for predictive analytics have inherent capabilities of interpretability, and hence give clues into the model explainability (Fig. 3). Main examples include ML models based on linear regression, decision trees, tree ensembles such as Random Forest (RF) and Self-Organizing Maps (SOMs) [17][23]. As an example, tree-based models can be translated into a set of rules that attempt to explain the model behavior. Here, we also need to distinguish between the models which only provide global explainability such as most important features extracted from a trained model to a more fine-grained local explainability for the prediction of a specific input sample to the model. It is the local explainability that is more important for the ML-based diagnostic than just the global understanding of the model from the perspective of a real-time service such as IPTV.

B. Blackbox Methods

In the Blackbox methods a trained model is not directly used for the model interpretability but instead the trained model is fed to a Blackbox method (i.e., *explainer* model) to learn the behavior of the model on different data inputs (Fig. 4). The Blackbox methods can provide not only global explainability but also local interpretability which is suitable for root cause inference purposes.

Methods belonging to this category include LIME (Local Interpretable Model-agnostic Explanations) [11], Anchor generation [12], Contrastive Explanation Method (CEM)

[13], What-if analysis tools [16], SHAP (SHapley Additive exPlanations) [15] and more recently Explainable Boosting Machine (EBM) [18]. Both LIME and Anchor generation create interpretable rules to provide the insights into the model prediction behavior based on different inputs. While CEM can generate *Pertinent Positives*, which behave like Anchor generation, and *Pertinent Negatives* which are conceptually similar to Counterfactuals [19].

SHAP has been recently proposed in [15] for model explainability and has a solid theoretical foundation. If the objective of trained prediction model is to have associated explainability for each input data sample, then SHAP approach is useful to interpret the impact (on prediction) of having a certain value for a given feature in comparison to a case where that feature took some other baseline value. Also, summing up all SHAP feature contributions and adding to the baseline prediction value equates to the output prediction value for the current input data sample (Eq. 1). The unit of SHAP values is the same as the target variable to be predicted.

$$\sum_{i=1}^N SHAP_value(Feature_i) = Pred(current) - Pred(baseline) \quad (1)$$

IV. PROBLEM FORMULATION

In this section, first we describe the methodology to build the QoE prediction model, including the target variable, data preparation and hyper parameters for the different learning algorithms. Secondly, we describe how these prediction models are used as a basis for root cause inference to diagnose the QoE problems in the IPTV service.

A. Prediction Problem Formulation

The goal here is to predict a QoE problem occurrence in advance before it manifests in the service. Here, we formulate it as a supervised learning problem. In an earlier section, we presented the QoE classes reported by the STB, and to make the problem formulation simpler, here we opt to focus on one specific QoE class i.e., ‘OK’ which is considered most important based on domain knowledge. It allows us to have a ground truth about QoE issues. To formulate the QoE prediction problem as a supervised ML classification problem we construct a new binary variable, QoE_flag. This variable is constructed as following (Eq. 2):

$$QoE_flag = \begin{cases} 0, & \text{if } QoE(OK) = 1 \\ 1, & \text{otherwise} \end{cases} \quad (2)$$

The intuition behind QoE_flag is straightforward. That is a QoE_flag is only raised if an STB is active, and QoE(OK) is < 1.0 meaning that a problem will occur.

As mentioned earlier, in this study we consider three different data sources namely, STBs, WiFi, and Links Traffic. Data from STBs represent the ground truth or the labels for the supervised learning problem. QoE_flag variable is generated only for active STBs and reported after every 5-minute interval. As the ground truth data is reported every 5 minutes, WiFi data is aggregated to match the same time resolution. This aggregation results in 29 features in total including 6 features composed of Links Traffic measurements (i.e., throughput). Note that non-causal features such as time stamps and device IDs are not considered for input feature construction. Since the main objective for us here is to construct an input dataset to train predictive models that are

going to be used as a *base* for investigating different root cause inference techniques described in the next section.

Since we are formulating a forecasting problem, we construct the input data frame in such a way that each input vector of data representing a sample at timestamp t is matched against the label from timestamp $t+1$, resulting in a prediction horizon of 5 minutes. The merged dataset is split into two parts where first part comprises of about 70% samples and second part comprising of 30% samples. First part of the data is used to train the model (including tuning model hyper parameters) while the second part is used to evaluate each model for prediction accuracy and the root cause inference. To evaluate predictive performance, ML models are built using three popular supervised learning algorithms namely: Random Forest [9], eXtreme Gradient Boosting (XGB) [10] and Multilayer Perceptron (MLP) [2]. These algorithms have a set of hyper parameters that were optimized by running repeated experiments to tune the performance. Key hyper parameters used for these algorithms are shown in Table II.

Table II: HYPER PARAMETERS FOR ML MODELS

MLP	RF	XGB
Neurons per Layer: 64 (L1), 32 (L2), 16 (L3), 8 (L4), 4 (L5), 1 (L6)	No. of Estimators: 40	No. of Estimators: 100
Learning Rate: 0.001	Min. samples split: 2	Learning Rate: 0.1
Batch size: 32	Max depth: None	Max depth: 3
Epochs: 20	Split criteria: Gini	Objective: Binary-logistic
Activation function: relu, sigmoid (output layer)		Evaluation Metric: Logloss
Optimizer: adam		
loss='binary_crossentropy'		

Note that in this study we only consider a centralized learning scenario, where only one generalized machine learning model is trained and deployed centrally to predict the QoE problems and do the root cause inference for all the STBs. During the learning phase, the measurements from all the households and network switches are collected and combined in a central node. The supervised model is then trained using input features extracted from WiFi and switch devices, along with QoE flag (i.e., labels) generated from STB devices. The Centralized model approach allows much simpler central training relaxing the need of distributed computational resources to run models at every household. For the sake of simplicity, we also assume that root cause inference model is co-located at the central location with the predictive model. While simple, this approach implies that every measurement needs to be pushed to the centralized place, consequently increasing link bandwidth consumption. As STB data need to be transferred, this approach may also suffer from delays. Performance related deployment aspects of centralized versus decentralized model training, prediction and root cause inference in an IPTV network context will be discussed in a separate work.

B. Root Cause Inference based on Predictive Models

We use the feature importance computation of XGB as a reference Whitebox model interpretability technique to compare with the Blackbox methods. There are couple of ways how the XGB can be used to compute the feature importance and can lead to a discrepancy in the computed feature importance results. Three common XGB feature importance computation methods are: *weight* based which

decides on the importance of a feature based on how many times that feature is used to split the input data across all the trees (during training), *cover* based which decides on feature importance on the basis of how many times that feature is used to split the input data across all trees but weighted by the samples that passthrough each tree branch, and finally the *gain* method (i.e., Gini Importance), which decides on feature importance based on how much error is reduced (on average) in the model when a feature is used to split the input data on tree branches in the model. We used the *gain* method to compute the feature importance which we experienced to be more consistent and stable.

On the other hand, SHAP is based on game theory and computes the marginal gain in terms of prediction accuracy when a feature is introduced into the model as compared to when its absent. Main design objective for SHAP is to have the property of *consistency* and *accuracy* which is missing from most other model explainability methods. *Consistency* here refers to the fact that when a model (structure) is changed in such a way so that it relies more on a specific feature (than in the earlier model) then the feature importance of that feature must not decrease otherwise it makes it impossible to compare the feature importance across different models. *Accuracy* refers to the property that sum of all feature's importance should equate to the total importance of the model. More details of SHAP computation algorithm can be found in [15]. Here, we consider Tree SHAP (an exact solution for tree-based models) for model explainability. Note that we didn't consider the Deep-SHAP (an approximate solution for neural network explainability based on Deep-LIFT [20]) because of the prohibitively long computation times. SHAP value computation approaches can be used to explain local behavior of a model (for a specific prediction) in addition to the global explainability of the model.

The SHAP value concept is best explained by an example. Here, we trained a Tree SHAP *explainer* model (built on top of an XGB prediction model) and asked it to provide the *local explainability* of a prediction. This explainability computed by the SHAP for current prediction is intuitively explained using a specific plot called *Force plot* as shown in Fig. 5.

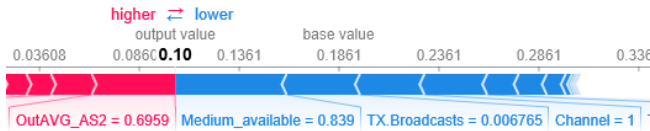


Fig. 5. Force plot for a single prediction of QoE_flag.

From Fig. 5, we can observe that for this specific prediction how different features are pushing above (or below) the predicted value of class 1 probability (0.10) from default prediction of base value (i.e., probability of 0.1861) by a certain 'force'. Recall that class 1 probability here refers to the probability of predicating a QoE_flag=1. As an example, in this case 'Medium_available' is pushing above (with a positive force) the class 1 probability from the *base value* with the highest value of 0.839. Second most important feature in this case is 'OutAVG_AS2' pushing down the class 1 probability with negative force of 0.69. Note that the consistency property of SHAP (Eq. 1) is satisfied here.

LIME is another popular method used for model interpretability which can generate human readable rules that drive the decisions taken by the model under consideration. LIME method works by *taking in* the original Blackbox

model, current sample, current sample prediction (with associated class probability) as input for which the explainability is desired. Then a set of synthetic examples are generated around the input sample using perturbation. In the next step, they are input to the Blackbox model to generate predictions (with class probabilities). Then this relatively small artificially generated dataset is used to train a new Whitebox linear model using an interpretable supervised learning technique (e.g., linear regression). Note that generated examples are weighted based on the distance from the original sample when used for training the newer Whitebox model. Although, the model explainability computed using LIME may not be globally accurate, but it should be locally accurate enough to explain the model prediction of the current example. A main advantage of the LIME *explainer* is the low computation time hence it can be used for large datasets.

In Fig. 6, we see an example set of rules generated for a test data sample when fed to the LIME algorithm. Prediction and class probabilities for input to LIME are generated from an XGB model pretending to be a Blackbox model in this case. Left part of Fig. 6 shows the features associated with different rules either supporting class 0 (blue color) or class 1 (orange color) for the prediction of current sample. Notice the values on top of the bars show the actual importance of those features/rules in making that prediction. Table in the right part of the figure shows the actual values of some of the top features in the current input data sample.

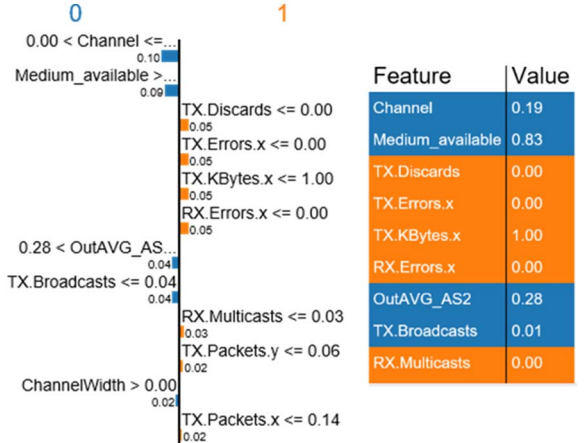


Fig. 6. Rule set generation using LIME.

V. RESULTS

In this section, first we present the performance of machine learning models trained on data collected from the operational IPTV service platform in terms of their predictive capabilities. Secondly, we show the results for the root cause inference based on Blackbox techniques namely SHAP and LIME then discuss various performance aspects including comparison to a Whitebox approach based on XGB.

We run the experiments on a lab server machine, which includes an Intel Xeon W-2123 CPU (16 physical cores) interconnected by a 1Gb Ethernet switch with 16 GB of RAM, a 500 GB SSD drive and Nvidia GeForce GTX 1080ti GPU. Server machine is running Linux CentOS 7.6 with Python 3.6 environment setup with a Nvidia container with relevant Python libraries such as NumPy, Scikit-learn, Matplotlib, SHAP and LIME installed.

To evaluate the predictive performance of the trained supervised ML models we used the performance metrics commonly used to assess the classification performance namely: Precision (accuracy on detected positive cases), Recall (percentage of positive cases detected out of total positive cases in the data), and F1 (a harmonic mean of Precision and Recall). Note that to maximize the F1 performance of classifier, Precision and Recall both need to increase at the same time.

$$\text{Precision} := \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \quad (3)$$

$$\text{Recall} := \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \quad (4)$$

$$\text{F1} := 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \quad (5)$$

A. Results for the Predictive Models

First, we present average accuracy results of the predictive models trained for forecasting QoE_flag in the next 5 minutes (summarized in Table III).

TABLE III. ACCURACY PERFORMANCE FOR CLASSIFICATION MODELS

Method	Precision	Recall	F1
Random Forest	0.207	1.0	0.343
XGB	0.373	0.555	0.446
MLP	0.336	0.345	0.340

In terms of the predictive performance we observe that XGB shows much better precision, recall and F1 performance than RF. MLP shows prediction performance similar to RF. Although, the training time is much longer for the MLP as compared to the RF and XGB models. One thing to note is that due to high class imbalance (of QoE_flag) and also the fact that non-causal features are removed from the input data used to train the model, overall prediction performance is on the low side. However, even a moderately accurate XGB model can still be used as the basis for root cause inference to understand the issues causing IPTV service QoE problems.

B. Root Cause Inference Results

For root cause inference, we first present the summary results (Fig. 7) for the SHAP value computation on the whole test data set (only for the positive test instances where (QoE_flag=1)) which gives us the overall picture in terms of important features responsible for causing the QoE problems in the IPTV service operation but also more granular insights that we will discuss here. Note that to facilitate the comparison among different techniques the trained XGB model from previous sub-section is used as input to build the explainer models described here.

In the Fig. 7, we show the computed tree SHAP values for the test data (only positive cases) to reveal the distribution of impacts each feature has on the model output. These distributions are shown for all notable features sorted by the average magnitude of SHAP value over the test samples. We observe the ‘Medium_available’ (WiFi radio parameter) is the most important one that triggers QoE problems. Another insight is that, increase in the ‘Medium_available’ value causes the SHAP value to become negative for this feature effectively reducing the probability of predicting a QoE problem. In other words, higher ‘Medium_avaiable’ values

are encouraging the smooth operation of the service avoiding any required bandwidth bottleneck for video streaming. Along the same lines, we observe that ‘Channel’ is another important WiFi related feature where lower values of ‘Channel’ can increase the chances of WiFi link problems and hence also the chances of QoE problem occurrence during the smooth operation of IPTV service.

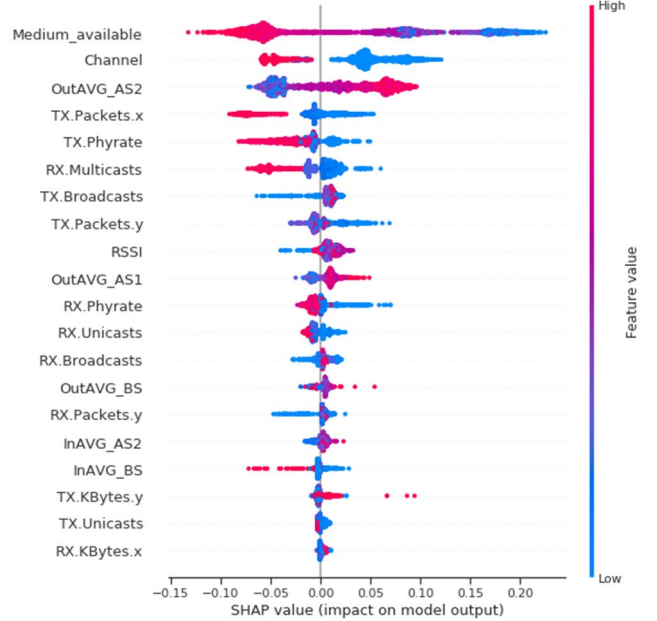


Fig. 7. SHAP summary plot for distributions of input features.

In general, we observe that top 6 input features have the most *weight* to influence that if an IPTV QoE problem is going to happen or not with top 2 features being part of the WiFi network. We also averaged the SHAP values for all positive case test samples for each of the input features in the data. Then we present it in a simpler SHAP summary plot sorted on the average SHAP value for each feature (Fig. 8). It provides a clear view on the average relative strength of SHAP values on a per feature basis.

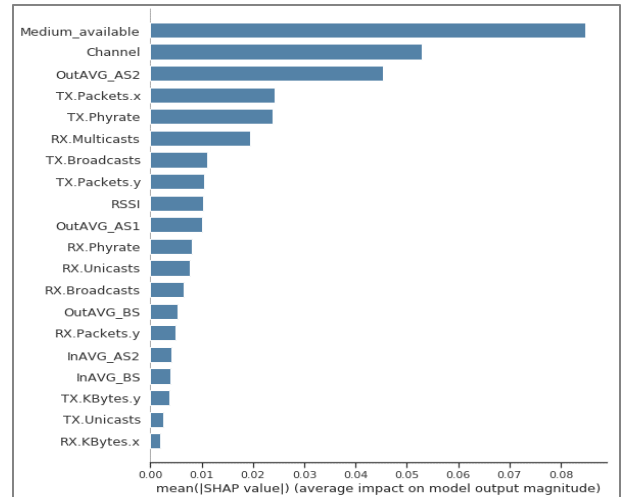


Fig. 8. Aaverage SHAP summary plot.

We also checked with the domain expert associated with the IPTV service provider and it was confirmed that most of the QoE issues faced during the time considered in this study were indeed related to the spotty WiFi. Part of the reason was the signal interference between the different IPTV household users using the same frequency band on their WiFi networks

in the small apartment building ('channel', 'RSSI') resulting in bandwidth throttling ('Medium_available').

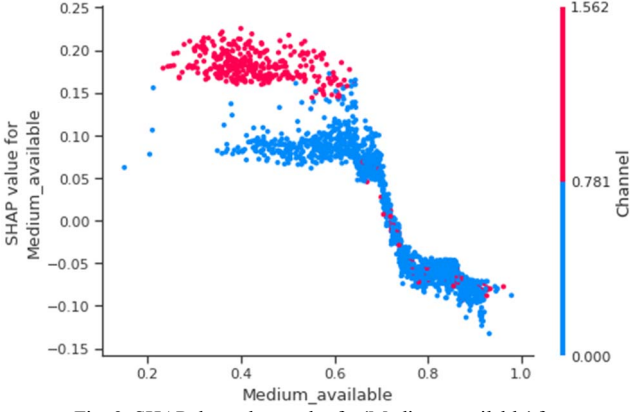


Fig. 9. SHAP dependence plot for 'Medium_available' feature.

Some more interesting insights are gained when we observe the SHAP dependence plot for 'Medium_available' feature against the 'Channel' feature (Fig. 9) revealing the *interaction effect* between these two top features. We observe that the 'Medium_available' value of less than 0.6 causes a rapid increase in the SHAP value in positive direction (meaning more chances for predicting the occurrence of an IPTV QoE problem). Secondly, we observe that high values for the 'Channel' feature in that region causes the SHAP value for 'Medium_available' to shoot even further as compared to the when the 'Channel' values are on the lower side (<0.78). These insights can be very helpful for the IPTV service technician sitting in the Network Operation Centre (NOC) to quickly find the root cause of a problem and quickly roll out a solution to minimize the service disruption. Although, due to space limitations we do not show here but using the SHAP it is also possible to calculate the higher order interaction effects between more than two features. However, we believe for the current use case it is enough to investigate the interaction effect between the feature pairs that have high SHAP values to gain insights.

Next, we show the most important features calculated directly using a model-based approach from a trained XGB model (as described in the Section V.A).

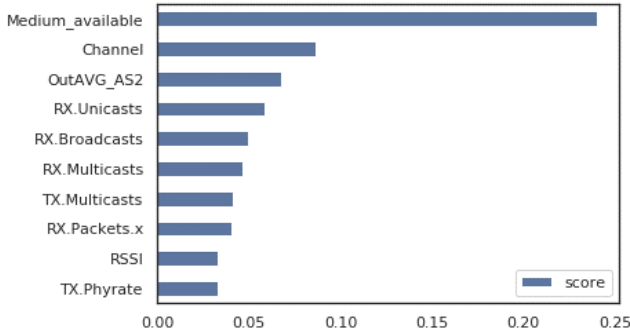


Fig. 10. Most important feature list using XGB model.

It is interesting to compare the most important features list extracted from the XGB model (Fig. 10) to the summary list computed from the SHAP values (Fig. 8). We immediately notice that top 3 feature are the same as computed by the SHAP algorithm. Although the relative strength of feature importance is not the same (XGB feature importance putting more weight on 'Medium_available' than SHAP), and features beyond top 3 do not match significantly.

Furthermore, as mentioned earlier, using SHAP provides per-sample based results which is important when running the algorithm in the live system to diagnose the currently predicted QoE problem. SHAP also has stronger theoretical foundations (i.e., consistency and accuracy properties). XGB only provides one global feature importance list which is useful for overall model explainability but of no use to gain a sample-by-sample runtime diagnostic of the IPTV service operation.

We also, trained an *explainer* using the kernel-based SHAP algorithm to compute the SHAP values on the test data (using the XGB predictive model described earlier as input pretending it to be a Blackbox) but we got very similar results to the tree SHAP as shown in the Fig. 7. Kernel SHAP only computes approximate SHAP values as compared to the tree SHAP and takes a lot more time to train the *explainer* model. However, it has general applicability and can be used for model explainability of almost any predictive algorithm beyond tree-based methods.

Now, we investigate another Blackbox model explainability method LIME for the diagnosis of the IPTV QoE problems. Results for the most important features computed based on the whole test data set (for the positive test instances) are summarized in the Fig. 11.

We observe that top 10 most important features are very similar to the ones that are computed using tree SHAP. Although now the positions are switched for top 2 most important features (i.e., 'Channel' becomes the top most important feature and 'Medium_available' the second most important feature). So, the agreement in this case with the SHAP results is not as strong as it was using model-based XGB approach.

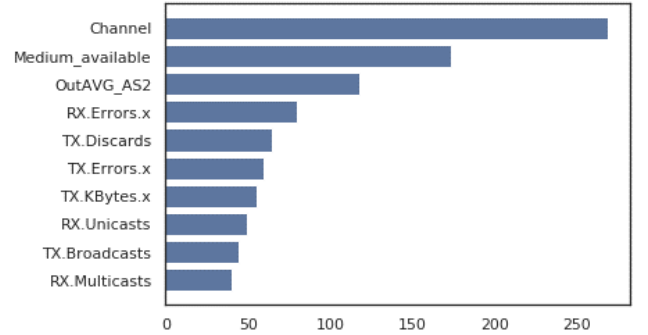


Fig. 11. Most important feature list using LIME.

VI. CONCLUSION

This paper presents our learnings when designing a service assurance system for an IPTV service operator. The first objective is to provide the insights in the running live service so that any potential QoE problems hindering the end-user video streaming experience are traced proactively. Second, important objective is to provide useful diagnostic to technicians in the NOC so that a problem can be localized to a specific network segment in an end-to-end network (e.g., core router, aggregation routers, WiFi, STB boxes). This assistance in terms of problem pinpointing can greatly reduce the service downtime for the affected end-users.

For this purpose, we evaluate two promising Blackbox techniques for model explainability namely SHAP values and LIME by comparing them to each other and to the traditional Whitebox feature importance attribution method based on the

XGB model. According to our experimental results SHAP is the most suitable approach for us to diagnose the IPTV QoE problems. SHAP capability to pinpoint the culprit feature(s) and localizing the problem to a specific segment of the network is also verified with the domain experts from the IPTV service provider. The downside is that computing SHAP values require some additional computation time. Although, it is known that tree-based learning methods such as XGB work very well for the tabular data. So, if the predictive model is XGB or RF then the tree SHAP can be used for much faster computation of SHAP values without resorting to the slower kernel SHAP. However, kernel SHAP must be used in case non tree-based learning algorithms are used (e.g., MLP). On the plus side, SHAP has a strong theoretical foundation providing consistent and accurate root cause inference results for each input test sample (i.e., local explainability).

On the other hand, LIME is a popular option for model explainability having the capability of providing detailed rule-based explanation for the prediction of each test sample. But it is not that accurate and reliable as SHAP [15]. For this study, we did consider the Anchor generation and CEM algorithm however we found the computation time to be significantly more even compared to the kernel SHAP making them less suitable for the time sensitive diagnostic applications such as the one we are considering here. One attractive aspect of Blackbox cause inference methods for service assurance applications is that the same model built for QoE prediction can be ‘fed’ to the cause inference algorithm for training an *explainer* model. This streamlines the functionality and interaction of two crucial components of the service assurance system namely *fault detection*, and *fault localization*.

In the future, we plan to compare the Blackbox approach to a promising model-based approach named Bayesian Networks (BN). Idea with the BNs is to train probabilistic graphical models (PGM) [21] that are highly explainable and at the same time provide decent predictive performance. We are also interested to investigate the deployment aspects of Blackbox *explainer* models when *predictive* models which they explain are running in a distributed setting [22].

REFERENCES

- [1] V. Berggren, “Towards zero-touch network operations,” Kista, Sweden, September 2018. [Online]. Available: <https://www.ericsson.com/en/blog/2018/9/towards-zero-touch-network-operations>
- [2] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning: With Applications in R*. Springer Publishing Company, Incorporated, 2014.
- [3] R. Yanggratoke, J. Ahmed, J. Ardelius, C. Flinta, A. Johnsson, D. Gillblad, and R. Stadler, “A service-agnostic method for predicting service metrics in real time,” *International Journal of Network Management*, vol. 28, no. 2, p. e1991, 2018. [Online]. Available: <https://onlinelibrary.wiley.com/doi/abs/10.1002/nem.1991>
- [4] International Telecommunication Union (ITU), “IPTV service requirements,” in *Internet Protocol Television (IPTV) Focus Group*, 2008.
- [5] H. H. Song, Z. Ge, A. Mahimkar, J. Wang, J. Yates, Y. Zhang, A. Basso, and M. Chen, “Q-score: Proactive service quality assessment in a large iptv system,” in *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, ser. IMC ’11. New York, NY, USA: ACM, 2011, pp. 195–208. [Online]. Available: <http://doi.acm.org/10.1145/2068816.2068836>
- [6] A. Balachandran, V. Sekar, A. Akella, S. Seshan, I. Stoica, and H. Zhang, “Developing a predictive model of quality of experience for internet video,” *SIGCOMM Comput. Commun. Rev.*, vol. 43, no. 4, pp. 339–350, Aug. 2013. [Online]. Available: <http://doi.acm.org/10.1145/2534169.2486025>
- [7] V. Menkovski, A. Oredope, A. Liotta, and A. C. S’anchez, “Predicting quality of experience in multimedia streaming,” in *Proceedings of the 7th International Conference on Advances in Mobile Computing and Multimedia*, ser. MoMM ’09. New York, NY, USA: ACM, 2009, pp. 52–59. [Online]. Available: <http://doi.acm.org/10.1145/1821748.1821766>
- [8] V. Menkovski, G. Exarchakos, and A. Liotta, “Online qoe prediction,” in *2010 Second International Workshop on Quality of Multimedia Experience (QoMEX)*, June 2010, pp. 118–123.
- [9] L. Breiman, “Random forests,” *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, Oct. 2001. [Online]. Available: <https://doi.org/10.1023/A:1010933404324>
- [10] T. Chen and C. Guestrin, “Xgboost: A scalable tree boosting system,” in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD ’16. New York, NY, USA: ACM, 2016, pp. 785–794. [Online]. Available: <http://doi.acm.org/10.1145/2939672.2939785>
- [11] Ribeiro, Marco Tulio, Sameer Singh, and Carlos Guestrin. “Why should i trust you?: Explaining the predictions of any classifier.” *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. ACM, 2016.
- [12] Ribeiro, Marco Tulio, Sameer Singh, and Carlos Guestrin. “Anchors: High-precision model-agnostic explanations.” *Thirty-Second AAAI Conference on Artificial Intelligence*. 2018.
- [13] Dhurandhar, Amit, et al. “Explanations based on the missing: Towards contrastive explanations with pertinent negatives.” *Advances in Neural Information Processing Systems*. 2018.
- [14] Jiang, Heinrich, et al. “To trust or not to trust a classifier.” *Advances in Neural Information Processing Systems*. 2018.
- [15] Lundberg, S. M., & Lee, S. I. (2017). A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems* (pp. 4765–4774).
- [16] What-if tool for machine learning model inspection. Online URL: <https://pair-code.github.io/what-if-tool/>.
- [17] Dean, Daniel Joseph, Hiep Nguyen, and Xiaohui Gu. “Ubl: Unsupervised behavior learning for predicting performance anomalies in virtualized cloud systems.” *Proceedings of the 9th international conference on Autonomic computing*. ACM, 2012.
- [18] Caruana, Rich, et al. “Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission.” *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2015.
- [19] Morgan, Stephen L., and Christopher Winship. *Counterfactuals and causal inference*. Cambridge University Press, 2015.
- [20] Shrikumar, A., Greenside, P., & Kundaje, A. (2017, August). Learning important features through propagating activation differences. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70* (pp. 3145–3153). JMLR. org.
- [21] Velasco, F. J. M. (2012). A Bayesian Network approach to diagnosing the root cause of failure from Trouble Tickets. *Artif. Intell. Research*, 1(2), 75–85.
- [22] D. Bacon, “Federated learning: Strategies for improving communication efficiency,” in *NIPS Workshop on Private Multi-Party Machine Learning*, 2016.
- [23] Ahmed, Jawwad, Tim Josefsson, Andreas Johnsson, Christofer Flinta, Farnaz Moradi, Rafael Pasquini, and Rolf Stadler. “Automated diagnostic of virtualized service performance degradation.” In *NOMS 2018-2018 IEEE/IFIP Network Operations and Management Symposium*, pp. 1–9. IEEE, 2018.