The Same, Only Different: Contrasting Mobile Operator Behavior from CrowdSourced Dataset

Konstantinos Kousias*, Cise Midoglu*, Ozgu Alay*, Andra Lutu*, Antonios Argyriou† and Michael Riegler*
*Simula Research Laboratory, Norway, †University of Thessaly, Greece

***Abstract*—Crowdsourcing mobile network performance evaluation is rapidly gaining popularity, with new applications aiming to deliver more accurate and reliable results every day. From the perspective of end-users, these utilities help them estimate the performance of their service provider in terms of throughput, latency and other key performance indicators of the network. In this paper, we build ORCA: Operator Classifier, a Machine Learning (ML) based framework to define and determine the behavior of Mobile Network Operators (MNOs) from crowdsourced datasets. We investigate whether one can differentiate MNOs by using crowdsourced end-to-end network measurements. We consider different performance metrics (e.g. Download (DL)/Upload (UL) data rate, latency, signal strength) and study the impact of them individually but also collectively on differentiating MNOs. We use *RTR Open Data*, an open dataset of broadband measurements provided by the Austrian Regulatory Authority for Broadcasting and Telecommunications (RTR), to characterize the three major mobile native operators and two virtual operators in Austria. Our results show that ORCA can be used to identify patterns between various mobile systems and disclose their differences from the end-user perspective.**

I. Introduction and Motivation

The use of Mobile Broadband (MBB) networks has exploded over the last few years due to the immense popularity of mobile devices such as smart phones and tablets, combined with the availability of high-capacity 3G/4G mobile networks. Therefore, understanding the underlying mechanisms that dictate user performance and reliability of MBB networks is of great importance towards smooth operation and future improvements.

One challenge with understanding MBB networks is that they consist of several heterogeneous and complex components that are intertwined into a system. A simplified 4G Long-Term Evolution (LTE) network architecture has two main building blocks: a Radio Access Network (RAN) and a Core Network (CN). RAN is composed of radio base stations whose primary responsibility is to mediate access to the provisioned radio channel and transport the data packets to and from the user's device. CN connects the radio network to the public Internet and is responsible for the data routing, accounting, and policy management. Although these same components exist in every operator's network, end-users typically experience considerably disparate performance due to different deployment of base stations, number of users, network configurations, and traffic policies across operators. In addition, the environment is highly dynamic in terms of physical channel conditions, applications, and users. The first aim of this paper is to quantify how visible these differences are to the end-users. Can we differentiate the operators and characterize their behavior by just looking into the end-user measurements?

978-1-5386-3531-5/17/$31.00

In mobile networks, to assess the quality experienced by end-users, certain network performance metrics are collected via end-to-end network measurements [13], [9], [11], [8]. One popular approach for performing such measurements in MBB networks is to rely on end-users to run performance tests by using a measurement application. Such a crowdsource approach is accepted as norm today since it can collect millions of measurements from different regions, networks and User Equipment (UE). For example, Okla's speedtest [3] evaluates operators performance and provides awards to the fastest in a country [5]. Such results are then commonly used by the operators in their marketing announcements. Similarly, regulators use the results of crowdsourced mobile applications such as RTR Nettest [4] to evaluate whether the operators fulfill their obligations. In all above applications, DL data rate is the most popular parameter that every operator uses to differentiate from other operators. The second aim of this paper is to determine the fundamental network parameters that differentiate operators. Accordingly, we will investigate parameters, including DL/UL data rate, latency, and signal strength, to establish if differentiation is attributed to a single parameter or a combination of parameters.

The goal of this paper is to understand whether different parameters can be used to discriminate the performance of operators using crowdsourced datasets. To tackle this problem, we propose a methodology that captures and discloses the unique behavior of operators by identifying patterns, namely *ORCA: Operator Classifier*. ORCA is designed to leverage large crowdsourced datasets composed of features as latency, DL/UL data rate, wireless signal strength and other network-level characteristics, collected from a large number of measurement vantage points, networks and UE. However, crowdsourced datasets are known to be noisy and naturally unbalanced due to experiments running at users' own will [14]. To eliminate this noise and imbalance, ORCA introduces a set of pre-processing and filtering

steps. Then, it follows a learning process and builds a classification model by adopting features that are representative of selected operators. The derived model is then used to investigate whether one can differentiate operators using a crowdsourced dataset.

Our results reveal that DL data rate alone is not enough to differentiate operators. Rather, latency is often a better differentiating parameter. In all cases, the combination of various performance parameters allows to better distinguish operators. More importantly, ORCA reveals the operating regime in which such differentiation can occur. Therefore, we can answer more detailed questions of the form: what is the range of the average latency that is representative of a certain operator? The end result is that ORCA can differentiate operators who own the network infrastructure with high accuracy. Further analysis illustrates how virtual operators are treated compared to the infrastructure owners: while native operators prioritize their own users, they do not discriminate between virtual operators. The basic framework of ORCA is relevant for many potential applications that could be explored as part of future work. Improving the robustness of ORCA, would allow a more accurate match between pricing and claimed offered Quality of Service (QoS), hints regarding the different CN and RAN structures, detection of performance bottlenecks and network problems.

II. Dataset

There is an increasing amount of attention from both academia and industry towards crowdsourced approaches for measuring MBB performance via end-user devices. Popular approaches include *Speedtest* [3], *OpenSignal* [1], *MobiPerf* [2], and *RTR-Nettest* [4]. Among available platforms, *RTR-Nettest* is the only one that provides its source code together with the complete open dataset, called *RTR Open Data*. Thus, we have used this dataset for the analysis and exploration of operators' characteristics as captured from the end-user perspective.

RTR-Nettest is a measurement platform launched by the RTR in 2013. It measures QoS parameters such as data rate and latency, as well as signal strength, geolocation, network and device type, with a timestamp for each measurement. A measurement in the *RTR-Nettest* platform consists of six stages: 1) *Initialization*, 2) *Pre-Test Download*, 3) *Ping Test*, 4) *Download Test*, 5) *Pre-Test Upload* and 6) *Upload Test*. Initialization consists of the client connecting to the Control Server and undertaking necessary authentication procedures before making a measurement request, which, when granted, starts the communication between the client and the Measurement Server. This exchange is very brief and consists of an almost-constant number of packets. Once the client establishes a connection with the server, the Pre-Test Download phase follows. Both of the Pre-Test phases are undertaken with the same purpose: to ensure that the Internet connection is in an "active" state, i.e. that dedicated radio resources are available. During this phase, the client requests and the server sends a data packet in each active thread. While the duration of the phase has not exceeded its nominal value, the client requests a data block of double size compared to the last iteration step. The transfer of the last data block will be completed even if the duration has already exceeded the nominal value. The Pre-Test Upload phase works analogously to the Pre-Test Download phase, but with the client as the sender and the server as the receiver.

The Ping Test consists of the client sending a certain number of Transmission Control Protocol (TCP) pings in short intervals to the server to test the latency of the connection. This exchange is also very brief and consists of an almost-constant number of packets. The Download Test and Upload Test are the main components of the measurement where multiple TCP threads are opened and within each of these, the receiver side simultaneously requests and the sender side continuously sends data streams consisting of fixed-size packets. After the nominal duration, the sender stops sending further packets on all connections, the last packet per each thread is allowed to transmit completely, and the DL/UL data rate of the connection is estimated. We refer the reader to the *RTR Open Data Interface Specification* for a complete list of available parameters and their descriptions [4].

RTR Fields: *RTR Open Data* currently provides up to 67 features which are grouped in six categories: test, location, device, network, coverage and performance. In this paper, we use 11 of these fields. Date and time (in UTC) is indicated by `time_utc`. Additional test-specific parameters, as identifiers of the test and relative start timers, do not contribute to our model and therefore are discarded. Moreover, we do not consider geo-location characteristics (e.g LAT and LONG of user's position, distance covered, etc), hence, no location related parameters are used. The device platform (Android/iOS) is indicated by `platform`, and `model` indicates the device name. For identifying networks, we use `network_type` which indicates the technology (e.g UMTS, GSM, 3G, LTE, etc.), and a combination of `sim_mcc_mnc` and `network_mcc_mnc` which indicate the Mobile Country Code (MCC) and Mobile Network Code (MNC) as read from the Subscriber Identification Module (SIM) card (i.e. home network), and the network that is currently used (i.e. access network), respectively. With this information we can identify cases of roaming. We use LTE signal strength information in the form of Reference Signal Received Power (RSRP) (`lte_rsrp`), and Reference Signal Received Quality (RSRQ) (`lte_rsrq`). Similar to several other datasets, *RTR Open Data* includes a series of QoS-related parameters, namely `download_kbit`, `upload_kbit` and `ping_ms`. Interface related parameters are out of the

TABLE I: ORCA model features.

Id	Feature	Description	IG
1	ping_ms	Latency (ms)	0.21
2	upload_kbit	UL data rate ($Kbps$)	0.18
3	download_kbit	DL data rate ($Kbps$)	0.12
4	lte_rsrp	Signal strength (dBm)	0.07
5	lte_rsrq	Signal quality (dB)	0.06
6	hour	Hour of the day	0.02
7	weekend	Weekend indicator	0.01

scope of in this paper.

Table 1 lists the selected model features along with their description. Features one to five are used directly from the dataset, where six and seven are derived. We use time_utc to obtain the hour of the day (hour) and add a weekend indicator (weekend, 1 if the measurement conducted during weekend, 0 otherwise) to investigate temporal effects. Overall, we focus on network related features that are available in every crowdsourced dataset.

Dataset Statistics: The total number of samples in the *RTR Open Data* between 2013 and 2016 is 3.67 millions. In this paper, we use a part of the dataset corresponding to six months of measurements (March 2016 - August 2016). During this period, we observe an average of 22568 samples per month in LTE, among which 16186 samples are collected from Android devices. There are 20 distinct SIM networks, including native operators, i.e. MNOs who manage their own infrastructure, virtual operators, i.e. Mobile Virtual Network Operators (MVNOs) who rely on others' infrastructure to operate via national roaming agreements, and operators who are roaming internationally in Austria. Measurements are collected from 378 device models.

Exploration and Filtering: As mentioned before, crowdsourced datasets are noisy and unbalanced due to the voluntary participatory initiation of measurements by users. In the *RTR Open Data*, for instance, we observed large variations in the number of samples per operator. Furthermore, there is a significant imbalance in the distribution of devices per operator (potentially due to joint smartphone and subscription deals), which implies that higher category devices might pull the average data rate up for a given operator. To overcome this bias, we first picked a representative distribution of device categories. Namely, we selected LTE Cat 4, 6, 9, and 12-13. We then randomly selected an equal number of samples from each operator per device category (see Table II). We accounted only for Android devices with LTE support.

TABLE II: Balanced dataset for Austria's top three MNOs (3 AT, A1, T-Mobile (TMA)) in LTE on Android platform (only native) collected during March 2016 - August 2016.

	3 AT	A1	TMA	Total
Original Dataset	17137	19338	22788	59263
Balanced Dataset	6,332	6,332	6,332	18,996

In Figure 1, we illustrate the characteristics of the balanced dataset in terms of latency, DL/UL data rate, and LTE signal strength (RSRP and RSRQ). Violin plots show the range of each parameter per operator. We observe that the range and the density of the parameters do not vary greatly from one operator to another, making it hard to find a single parameter that clearly differentiates operators. In addition, defining thresholds on statistical descriptors of the parameters' distributions does not suffice to capture the interplay of all these metrics. Hence, they fall short in contrasting operator behavior across multiple dimensions. This motivates our decision to consider all available parameters and leverage ML to build an operator classifier.

III. ORCA: Operator Classifier

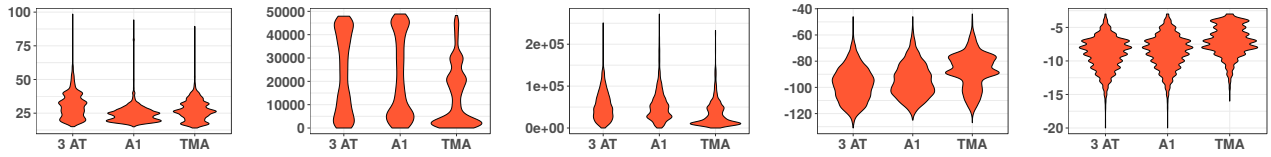
In this section, we expand on the design and methodology behind ORCA. The flowchart depicted in Figure 2 contains three main building blocks: Dataset, Study Design and Decision Tree Induction.

A. Study Design

The learning process is designed with a methodology that uses sequential training, validation, and testing. In particular, we first train and refine the classification model (training and validation) and then we measure how it behaves on an independent never-before-seen dataset (testing). This approach implies splitting the data into *known data*, which we use for training and validation, and *unknown data*, or *hold-out test data*. For a robust evaluation, we perform K-Fold cross-validation for the training and validation phase. Cross-validation splits the known data into disjoint training and validation subsets, in order to estimate the average accuracy of the model. In the following, we explain in detail the data structure and error metrics we used.

Data Structure: When splitting the dataset into training, validation and testing, we need to ensure that all datasets are perfectly disjointed. We first isolate five months (March 2016 - July 2016) for the known data, and one month (August 2016) for the hold-out test data. We then filter the data to ensure a balanced distribution across the three MNOs. With 10-fold cross-validation, we train our model and evaluate its performance under ten different splits of the known data. In this case, the training set has 90% of the known data while validation set contains the remaining 10%. At each repetition (i.e., fold), we do not reuse the 10% of the known data used for validation.

Error Measures: To determine the best model, we evaluate a set of error measures. The accuracy of a classifier is, by definition, the percentage of instances that are classified correctly when a set of unknown data is tested. Accuracy is commonly used as a performance metric in binary classification problems. However, in a multi-classification problem, accuracy is not enough to



(a) Latency (ms) (b) UL Data Rate (Kbps) (c) DL Data Rate (Kbps) (d) RSRP (dBm) (e) RSRQ (dB)

Fig. 1: Characteristics of the Training Dataset.

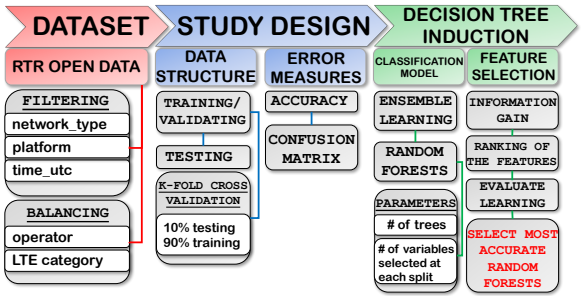


Fig. 2: ORCA flowchart.

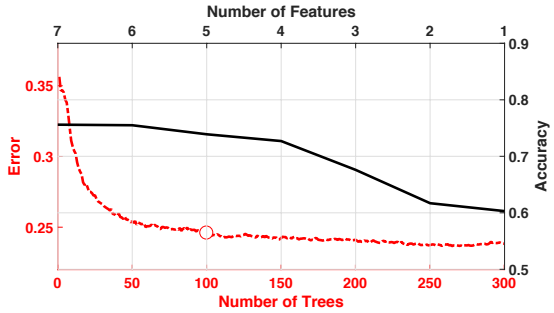


Fig. 3: (i) Left bottom axes (dotted line): Classifier error as a function of the number of trees in RF. Based on these results, we select to use 100 decision trees in the forest. (ii) Top right axes (solid line): The impact of total number of features used for model building on the accuracy of the classifier.

reflect the performance and efficiency of a classifier. Therefore, we produce a confusion matrix, that maps predicted operator labels to the rows and ground-truth operator labels to the columns.

B. Decision Tree Induction

In this section, we discuss the learning and the feature selection method we adopt for building the decision model.

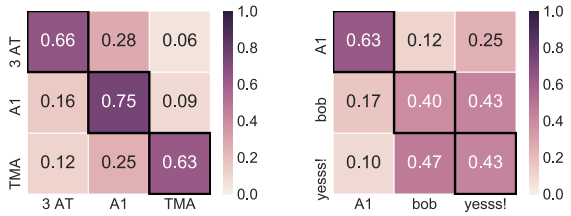
Random Forests (RF): We derive and tune a decision tree model based on the ML method of Classification and Regression Trees (CART) [6]. Tree-based learning methods rely on iteratively partitioning the data into smaller groups of similar elements [12]. The key idea is to choose the splits which maximize the group homogeneity ,or until the small groups are sufficiently *pure*. Choosing the right number of splits is a challenge, since the model can easily overfit by considering splits that are very specific to the training data, or, contrarily, underfit it by considering shallow general splits. Finding the correct balance is conditioned by finding the optimal set of features used to partition the data.

The next step is to adopt *ensemble learning*, that is, generate many classifiers and aggregate their results. For this purpose the RF algorithm is selected [7]. The used bagging approach builds independent decision trees using a bootstrap sample of the training dataset. In the end, a simple majority vote is taken for finding a prediction. RF adds randomness to the bagging approach. In RF, each split uses a subset of features randomly chosen at each repetition. This algorithm is known to outperform many other algorithms, including discriminant analysis, support vector machines and neural networks, and is robust against overfitting [7].

The number of trees in RF is an important parameter that dictates the performance and the computational complexity of a classifier. We need to select a number of trees that provides a good compromise between accuracy, computational complexity, and probability of overfitting to a given training dataset [15]. We vary the number of trees between 2 and 300 and evaluate our classifiers using 10-fold cross-validation. The bottom left axes of Figure 3 presents the Out-of-bag mean error of the classifier as a function of the number of trees used. Based on these results, we select a forest that consists of 100 decision trees. Another parameter that dictates the performance of RF is the number of features that are randomly sampled in each individual tree. We set this parameter to its default value that is defined as the square root of the total number of features.

Feature Selection: Subsequently, we apply feature selection using Information Gain (IG) as the primary metric. IG, also known as entropy, is a widely accepted method for evaluating the contribution of a feature in distinguishing between instances of different classes [10]. It varies between 0 and 1 with the latter one representing maximum information. We use a ranking approach to sort the features based on the scoring assigned by IG. The decreasing order of the features and the associated IG value is being reflected on Table I. We observe that latency dominates the pool of features along with UL and DL data rate following close behind. In the contrary, entropy of `weekend` is close to zero meaning that it hardly provides useful information to the classifier.

In order to select the subset of features that ensures the optimal performance of RF, we adopt a progressive approach. We first train RF with all available features using 10-fold cross-validation and estimate its performance by generating the confusion matrix. Subsequently, we



(a) Native MNO scenario. (b) MVNOs compared to their native operator.

Fig. 4: Confusion matrices illustrated by heatmaps. The gradient encodes the accuracy for each block of the confusion matrix. Rows and column direction indicate the ground-truth and the predicted operator respectively.

eliminate the feature with the lowest IG (i.e., weekend), re-train the model and calculate yet again the confusion matrix. With this approach, we further iterate through the remaining features in an increasing order of IG and, at each iteration, the feature with the lowest contribution is eliminated. This helps us estimate whether a subset of the features confuses RF instead of helping the algorithm to produce a higher-performance classifier. The top right axes of Figure 3 depict the accuracy of each classifier as a function of the number of features used. We observe that the generated classifier with all seven features is the best performing one.

IV. Performance Evaluation

Describing the performance of a multi-classification algorithm with a single number is often not enough to unveil it's overall behavior. Next, we evaluate the performance of ORCA by leveraging visualization tools such as heatmaps to better illustrate the confusion matrices.

A. Classification of Native Operators

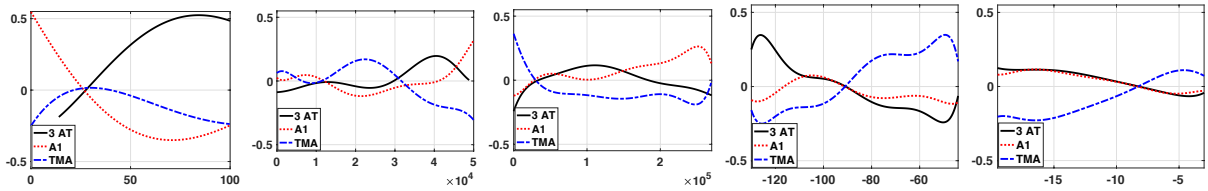
For the classification of native operators (i.e. no roaming case), we use the balanced training dataset described in Section III (March 2016 - July 2016) and evaluate the performance of ORCA on the hold-out test dataset (August 2016) which consists of 3462 samples. Figure 4a shows the heatmap of the confusion matrix where the correctly classified instances are located in the main diagonal of the matrix.

We observe that ORCA can identify MNOs with an accuracy of 66%, 75% and 63% for 3 AT, A1, and TMA, respectively. This is a rather good result considering that a random guess has a 33% probability to be successful. There exists a slight confusion between 3 AT - A1 and TMA - A1 while A1 is equally likely to be confused with either one. It is important to point out that it was difficult to differentiate the operators by using statistical representations (violin or box plots) of single features. However, with ORCA, the operators can be classified quite accurately indicating that each operator has a certain pattern that is uniquely identifiable. Note that, the accuracy of the classifier differs among operators. Given that we optimize RF to it's full potential, the classifier percentages are restricted by the similarities in the data. The more similarities exist between the MNOs, the less accurate the classifier is.

To understand the contribution of each feature to the classification performance of ORCA, we use the *forest floor* approach described in [16]. Recall that, by using a pluralistic vote mechanism, RF serves a probabilistic prediction for each class. The connections between samples are described by the change in the predicted probability for each operator and they sum to zero. Therefore, feature contributions can be defined as the sum of these changes over trees for each sample. Figure 5 depicts the change in the predicted probability compared to the value range of each feature, per operator. Subplots corresponding to the most important five features are sorted in a decreasing order of importance according to IG. We notice that observations with latency higher than $30ms$ are more likely to be classified as 3 AT. Moreover, RSRP higher than $-90dBm$ imply a higher possibility to be labeled as TMA. In addition, we observe that UL and DL data rates can identify the correct operator with an adequate likelihood within different intervals. In summary, operators have different likelihood for the value range of each feature and ORCA exploits this to differentiate the operators.

B. Contrasting MNOs Using a Single Feature

In this section, we assume there is only one feature in the dataset and investigate the dominant one for contrasting operators network. Figure 6 depicts the accuracy per MNO for five features ordered with respect to the IG ranking presented in Section III. The foreground bars illustrate the accuracy when only a single feature is used and the background bars represent the accuracy when all seven features are available. We observe that for the RTR dataset, we can identify MNOs with an average accuracy of 60% by using only latency. Note that this is only 8% lower than the native MNO scenario and clearly indicates that latency is the most important feature contributing to differentiating operators. Furthermore, we observe that UL and DL data rate are not good for differentiating operators, providing a 42% and 38% accuracy respectively, which is slightly better than choosing an MNO randomly. This shows that DL and UL data rates are very similar for all operators. The same results apply for the LTE power-related parameters. However, the distribution of accuracy between the MNOs is different. RSRP identifies 3 AT and TMA with an average accuracy of 43%, while for A1, this number goes down to 29%. Finally, while RSRQ is also a good identifier for TMA, it totally fails when it comes to A1, where accuracy reaches 6%. In summary, our results show that latency is the most important feature for operator classification, but using all available features clearly increases the overall accuracy.



(a) Latency (ms) (b) UL Data Rate (Kbps) (c) DL Data Rate (Kbps) (d) RSRP (dBm) (e) RSRQ (dB)

Fig. 5: Feature contributions for the training dataset, for each feature and each operator. Y' depicts the change in the predicted probability. X' is the value range for each feature. We fit seven degree polynomial curves to ease readability of the plots.

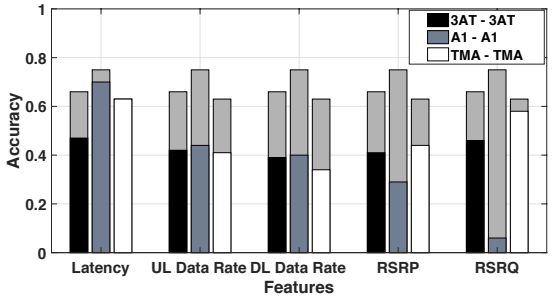


Fig. 6: Accuracy per MNO using a single feature. Background bars indicate the accuracy of the native MNO scenario.

C. MVNOs Identification using ORCA

We further investigate the roaming scenario, where a national agreement between MVNOs and a native operator exists. MVNOs are allowed to utilise the physical infrastructure of their service provider while applying different tariffs or QoS to their customers. Our aim is to understand if the customers of both sides are treated in the same manner. We use the same datasets, and consider A1 and two of its MVNOs: bob and yesss!. Figure 4b shows the heatmap of the confusion matrix for A1, bob and yesss!. We observe a clear differentiation between A1 and the MVNOs, most likely due to resource constraints enforced by A1 to the MVNOs. Moreover, *upload_kbit* is among the features with the largest entropy according to the IG analysis, which means that MVNOs customers are not allowed to exploit all the available bandwidth when uploading. On the other hand, ORCA suffers more when it comes to bob and yesss!, showing an accuracy of 40% for bob and 43% for yesss! respectively. This results indicates that MVNOs are treated similarly and ORCA cannot distinguish them.

V. Conclusions and Future Work

In this paper we introduced *ORCA: Operator Classifier* for identifying patterns and disclosing exclusive aspects of MNOs under noisy crowdsourced datasets. Contrary to what is extensively presented in the literature, in which the focus is centered upon operator performance with respect to a single parameter, ORCA leverages the power of the rich set of features commonly found in crowdsourced datasets and jointly considers multiple features. In addition, a learning process is used to build a classifier for identifying operators. RF is used for capturing this characterization. Results show that, latency is the most important feature to differentiate MNOs behavior. However, using all available features clearly increases the accuracy of the classifier. Moreover, MVNOs are treated differently compared to their native operator while they behave similarly among themselves and are not easily detectable.

The basic framework of ORCA is relevant for many potential applications. For example, improving the robustness of ORCA would allow a more accurate match between pricing and claimed offered QoS, hints regarding the different CN and RAN structures, detection of performance bottlenecks and network problems. Future work includes analysis of benchmarking classification algorithms and exploration of crowdsourced datasets.

Acknowledgments

This work is funded by the Norwegian Research Council project No. 250679 (MEMBRANE).

References

[1] https://opensignal.com.
[2] https://sites.google.com/site/mobiperfdev/.
[3] https://speedtest.net.
[4] https://www.netztest.at/en/.
[5] http://www.speedtest.net/awards.
[6] Breiman et al. *Classification and Regression Trees*. Wadsworth and Brooks, Monterey, CA, 1984.
[7] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.
[8] Cainey et al. Modelling download throughput of LTE networks. *IEEE LCN Workshops*, 2014.
[9] Chen et al. Understanding the Complexity of 3G UMTS Network Performance. In *IFIP Networking Conference*, 2013.
[10] R. L. De Mántaras. A distance-based attribute selection measure for decision tree induction. *Machine learning*, 6(1):81–92, 1991.
[11] Ferlin et al. Measuring the QoS characteristics of operational 3g mobile broadband networks. In *WAINA*, pages 753–758. IEEE, 2014.
[12] J. Han. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005.
[13] Huang et al. An In-depth Study of LTE: Effect of Network Protocol and Application Behavior on Performance. In *Proc. of SIGCOMM*, 2013.
[14] Midoglu et al. Opportunities and challenges of using crowdsourced measurements for mobile network benchmarking a case study on RTR Open Data. *SAI Computing*, pages 996 – 1005, 2016.
[15] Oshiro et al. How many trees in a random forest? In *International Workshop on Machine Learning and Data Mining in Pattern Recognition*, pages 154–168. Springer, 2012.
[16] Welling et al. Forest floor visualizations of random forests. *arXiv preprint:1605.09196*, 2016.