An Intelligent Customer Care Assistant System for Large-Scale Cellular Network Diagnosis

Lujia Pan[1], Jianfeng Zhang[1], Patrick P. C. Lee[2], Hong Cheng[3], Cheng He[1],
Caifeng He[1], Keli Zhang[1]
[1]Noah Ark's Lab, Huawei Technologies
[2]Department of Computer Science and Engineering, The Chinese University of Hong Kong
[3]Department of Systems Engineering and Engineering Management, The Chinese University of Hong Kong

ABSTRACT

With the advent of cellular network technologies, mobile Internet access becomes the norm in everyday life. In the meantime, the complaints made by subscribers about unsatisfactory cellular network access also become increasingly frequent. From a network operator's perspective, achieving accurate and timely cellular network diagnosis about the causes of the complaints is critical for both improving subscriber-perceived experience and maintaining network robustness. We present the Intelligent Customer Care Assistant (ICCA), a distributed fault classification system that exploits a data-driven approach to perform large-scale cellular network diagnosis. ICCA takes massive network data as input, and realizes both offline model training and online feature computation to distinguish between user and network faults in real time. ICCA is currently deployed in a metropolitan LTE network in China that is serving around 50 million subscribers. We show via evaluation that ICCA achieves high classification accuracy (85.3%) and fast query response time (less than 2.3 seconds). We also report our experiences learned from the deployment.

CCS CONCEPTS

•Information systems → Data stream mining; •Networks → Network monitoring;

KEYWORDS

Fault classification; Sequential pattern mining; Cellular network diagnosis

1 INTRODUCTION

With the advances of 3G and 4G cellular network technologies, we are witnessing the continuous growth of mobile data traffic worldwide [8]. By subscribing to a cellular carrier, users can use mobile devices (e.g., smartphones, tablets, or datacards) to access data services on the Internet anywhere and anytime through the cellular network. However, if they encounter any problem of accessing the cellular network, such as network disconnection or slow network performance, they often call the customer service center to file complaints. A customer care assistant then needs to diagnose the root cause of each received complaint, which can be classified as either a *user fault* or a *network fault*: a user fault is related to any hardware or software problem of the subscriber's mobile device (e.g., damages of phones or SIM card, incorrect phone configurations, malicious mobile apps, etc.), while a network fault is related to any failure in network elements (e.g., network hardware outages or mis-configurations, weak coverage, signal interference, etc.). If it is classified as a user fault, the assistant will guide the complaining subscriber step-by-step to resolve the problem, such as rebooting or replacing the mobile device; if it is a network fault, the assistant will submit a trouble ticket to ask engineers to check network hardware and localize and repair any failure or mis-configuration.

The scale of the complaints being processed by the customer service center is overwhelming. As a case study, we consider a cellular network in China that is currently serving around 50 million subscribers. Each day the customer service center receives over a thousand of complaints regarding the unsatisfactory cellular network access. Around 85% and 15% of the complaints are later diagnosed as user and network faults, respectively. According to the statistics from network operators, a customer care assistant spends on average about two minutes to resolve a customer call; if the complaint is classified as a network fault, it will take another two days to resolve.

From a network operator's perspective, it is critical to diagnose complaints in an accurate and timely manner. Specifically, this paper aims to address the following binary classification problem: *Should a subscriber's complaint be classified as a user fault or a network fault?* Although the classification problem looks simple, providing an accurate and timely answer is critical for several reasons: (1) it improves subscribers' Quality-of-Experience (QoE) by correctly addressing their complaints; (2) it maintains network dependability by quickly responding to any failure in network elements; and (3) it saves unnecessary personnel hours of network engineers to check falsely identified network faults.

Unfortunately, with the ever-growing volume and complexity of mobile data traffic we face today, achieving accurate and timely cellular network diagnosis becomes increasingly challenging. Traditional diagnosis approaches heavily depend on human factors, including the experience and domain knowledge of customer care assistants as well as the information provided by subscribers when they file complaints. To improve both accuracy and efficiency of diagnosis, network operators have deployed an *expert rule-based* approach, which pre-configures a set of rules defined by experts based on their past experience to guide the diagnosis process. However, the rules still require manual configurations, need to be exhaustive,

KDD'17, August 13--17, 2017, Halifax, NS, Canada.
© 2017 ACM. 978-1-4503-4887-4/17/08…$15.00
DOI: http://dx.doi.org/10.1145/3097983.3098120

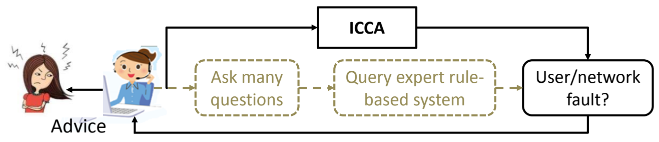

Figure 1: Diagnosis workflow with ICCA.

and are difficult to adapt to the current network conditions. Clearly, minimizing the human factors in the diagnosis workflow will be beneficial for network operators to manage cellular networks at scale.

This motivates us to take a *data-driven* approach, in which we analyze the characteristics of massive network data to help our diagnosis. To this end, we aim for the following design goals for our diagnosis solution:

- **Compatibility with heterogeneous cellular network technologies:** Today's cellular networks serve a mix of cellular network technologies (e.g., 2G, 3G, 4G), each of which has different signaling protocols and signaling message formats. Our solution should support heterogeneous types of input data for our analysis.
- **Automatic feature engineering:** Network data often has no pre-defined features, and its feature patterns vary over time. Our solution should automatically extract meaningful feature patterns based on current input data.
- **Accurate, real-time, and scalable analytics:** Our solution should achieve high accuracy in our binary classification (e.g., over 80%), achieve fast query response performance (e.g., in few seconds), and scale to a large number of users (e.g., 50 million subscribers in the network we consider).

In this paper, we design and implement *Intelligent Customer Care Assistant (ICCA)*, a distributed fault classification system that performs large-scale cellular network diagnosis. ICCA exploits data mining and machine learning techniques to analyze massive network data spanning the entire cellular network and provide accurate and timely answers towards subscribers' complaints. It eliminates the needs of customer care assistants to ask many questions and query the expert rule-based system for answers (see Figure 1).

To summarize, we make the following contributions.

- We collect network data in a unified form called *subscriber records*, which describe the details of both control-plane and data-plane network data. Our analysis operates on subscriber records and is applicable for heterogeneous cellular network technologies.
- We formalize a fault classification model for ICCA, which builds on two types of features: (1) expert features, which are statically defined by experts, and (2) sequential pattern features, which represent discriminative features with temporal dependency based on current network conditions and can be extracted by the model-based search tree (M^{b}T) [10].
- We design and implement ICCA as a distributed architecture that realizes both offline model training and online feature computation, so as to enable real-time fault classification between user faults and network faults.
- ICCA is currently deployed in a metropolitan LTE network in China that is serving around 50 million subscribers. Our evaluation shows that ICCA achieves a classification accuracy of 85.3% and a query response time of less than 2.3 seconds, and it outperforms the traditional expert rule-based approach.
- We report our experiences learned from our deployment and provide insights into large-scale cellular network diagnosis.

The remainder of this paper proceeds as follows. Section 2 describes our data collection methodology. Section 3 formalizes the fault classification problem. Section 4 describes our feature engineering approach. Section 5 presents the design of ICCA. Section 6 presents evaluation results. Section 7 summarizes our lessons learned from our deployment. Section 8 reviews related work, and finally, Section 9 concludes the paper.

2 DATA COLLECTION

Our study targets a production LTE network in a metropolitan area in China that is serving around 50 million subscribers. We use this network to motivate the design of ICCA, yet our methodology is applicable for general cellular networks. In this section, we first provide a general overview of an LTE network architecture, and then describe the format of the data for our analysis.

2.1 LTE Network Architecture

Figure 2 shows a simplified LTE network architecture considered in this paper. An LTE network provides Internet access for each subscriber's mobile device, called a User Equipment (UE), through two subsystems: the Radio Access Network (RAN) and the Evolved Packet Core (EPC). The RAN comprises multiple base stations, called Evolved NodeBs (eNodeBs), such that each UE connects over wireless to an eNodeB to access the EPC and then the Internet. The EPC comprises the Mobile Management Entities (MMEs), which perform control-plane functions (e.g., subscriber authentication, location tracking, etc.) as well as the Serving Gateways (SGWs) and the Packet Data Network Gateway (PGWs), both of which perform data-plane functions (e.g., routing data traffic between UEs and the Internet). A typical metropolitan LTE network is composed of thousands of eNodeBs and tens of MMEs, SGWs, and PGWs. For example, the LTE network we consider has over 80,000 eNodeBs and around 50 MMEs, SGWs, and PGWs.

To access the Internet, a UE first sends an *attach* request to set up a radio connection with an eNodeB, which works with an MME to authenticate and manage the UE's data transfer. Both the eNodeB and the MME exchange signaling messages to synchronize the UE's states. The UE next sets up a data session with a pair of SGW and PGW on top of the radio connection via the eNodeB. It finally can send/receive data packets over the Internet via the SGW and PGW pair. If the UE is inactive for some timeout, it closes the radio connection with the eNodeB, which again exchanges signaling messages and synchronizes the UE's states with the MME.

We deploy *probes* at each MME, SGW, and PGW to collect data (see Figure 2). Since each UE is required to access the Internet through these network entities, we have a complete view of the control-plane and data-plane usage of the LTE network. In practice, the number of probes deployed is limited, since the MMEs,

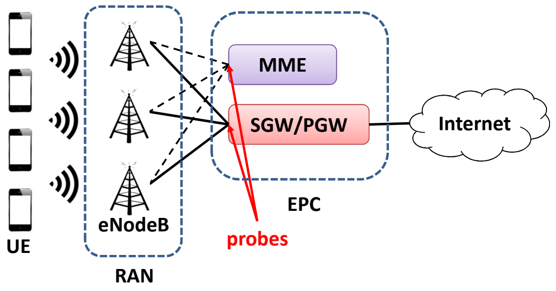

Figure 2: Simplified LTE network architecture.

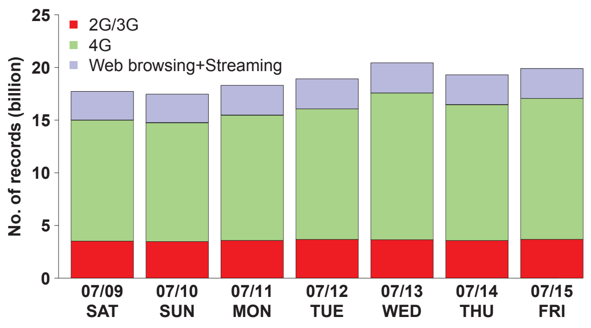

Figure 3: Subscriber records from July 9 (Saturday) to July 15 (Friday), 2016. The overall percentage distributions of different types of subscriber records are: 2G/3G (19.0%), 4G (66.1%), and web browsing and streaming (14.9%).

SGWs, and PGWs are often connected by a small number of network switches. We can deploy a probe per switch and let each probe monitor multiple network entities traversing the switch. Note that the probes do not reveal sensitive information, such as payloads and subscriber identity information, and they return a unique anonymized subscriber ID to identify each subscriber.

2.2 Subscriber Records

LTE mainly targets the latest 4G cellular technology. Note that while an LTE network supports 4G connectivity, it can also fall back to legacy 2G/3G connectivity if 4G connectivity performance is poor. Thus, we must have a unified format for our collected data so that our analysis works for heterogeneous cellular technologies.

We collect data from the probes in the form of *subscriber records*, which capture the statistics of signaling and data sessions between UEs and the Internet at the granularity of individual subscribers. Each subscriber record is keyed by the subscriber ID, and is attached with the timestamp when the record is collected by the probes. We classify subscriber records into two categories: control-plane and data-plane.

The probe at each MME collects control-plane subscriber records, which store the key fields of signaling messages, such as radio connection start/stop times, signaling message type, radio access types (e.g., 2G/3G/4G), and error codes regarding radio connections. Note that there are hundreds of fields across different types of signaling messages. The probe only extracts around 30 of them for our analysis. Each signaling message will trigger a control-plane subscriber record.

The probe at each SGW/PGW collects data-plane subscriber records, which collect metrics of each subscriber's data transfer, such as uplink/downlink throughput, number of retransmissions, TCP connection times, and error codes regarding data sessions. The data-plane subscriber records are periodically collected (currently at 5-minute intervals), and their metrics are the aggregate statistics since the last collection time.

To understand the scale of subscriber records that need to be processed, Figure 3 shows the distributions of the subscriber records collected by the probes in our LTE network over a one-week period on July 9--15, 2016. We focus on three types of subscriber records: (1) 2G/3G, (2) 4G, and (3) web browsing and streaming. The first two refer to the control-plane subscriber records for the legacy 2G/3G and the latest 4G cellular technologies; the last one refers to the data-plane subscriber records for the web browsing and streaming applications, both of which account for the majority of data traffic. During this period, there were 132 billion subscriber records in total (130 TB of size), or equivalently, 18.9 billion records per day. We observe that more subscriber records are collected in weekdays than in weekends, although the difference is fairly small (10%). Among the records, 85.1% and 14.9% of them are of types control-plane (i.e., 2G/3G and 4G) and data-plane (i.e., web browsing and streaming), respectively. Control-plane subscriber records are the majority as they are triggered by individual signaling messages.

Our subscriber records can be viewed as extensions of call detail records (CDRs) [9], which are widely used in telecommunication. CDRs provide details of voice-based phone calls, while our subscriber records provide details of Internet-based data sessions in both control plane and data plane.

3 PROBLEM FORMULATION

ICCA is designed to diagnose subscribers' complaints based on the collected subscriber records. We provide a problem formulation to guide our ICCA design. Specifically, we formulate the complaint diagnosis problem as a classification problem where the outcome variable is whether a subscriber's complaint is due to a *user fault* or *network fault*. We model this outcome as a function of the observed subscriber records. Formally, suppose that we are given a set of training subscriber records with known fault type denoted as $D = \{(R_i, y_i)\}|_{i=1}^{n}$, where R_i is the set of subscriber records for subscriber i, y_i is the true fault type, and n is the number of observed subscribers. We train a classification model denoted as a function f. Given a set of subscriber records R for a subscriber's complaint, we can predict the fault type $y = f(R)$.

4 FEATURE ENGINEERING

To build a fault classification model for fault classification, we need to extract discriminative features from subscriber records. We define two types of features that are complementary to each other: *expert features* and *sequential pattern features*. In this section, we describe how to construct both types of features and acquire the ground-truth labels for the training instances.

4.1 Expert Features

Expert features are defined by experts based on their domain knowledge, expert rules, and 3GPP specifications [1]. For each complaint filed by a subscriber, we define 164 expert features from subscriber records (both control-plane and data-plane) keyed by the subscriber ID. Each feature value can be either numerical (discrete or continuous) or categorical. We classify the expert features into four groups, whose examples are shown in Table 1:

- **General**: It describes a subscriber's profile and usage behaviors (e.g., whether the subscriber has restarted a phone or not). It reveals key information about a user fault, such as phone misuse or service-level agreement issues.
- **Signaling:** It describes the features of signaling messages associated with the subscriber. For example, a high number of attach failures (i.e., a UE fails to set up a radio connection with an eNodeB) may indicate poor network coverage around the region in which the subscriber is located; the error code contained in the signaling operation is useful to classify a network fault as it may indicate a failure of connecting to the cellular network.
- **Web browsing:** It describes the performance of web browsing applications perceived by a subscriber. For example, the page response delay provides an indicator of web browsing performance. If the page response delay is significantly high (e.g., on the order of seconds), it is more likely to be caused by a network fault rather than a user fault.
- **Streaming:** It describes the performance of streaming applications perceived by a subscriber.

When a subscriber files a complaint, we extract the expert features from the subscriber records keyed by the subscriber ID that are within a time window before the complaint time. A long time window retrieves more features and hence improves classification accuracy, but needs more memory space for storing and processing the features. Currently, we set the default time window as three hours.

Expert features aim to capture the most common cellular network usage patterns of subscribers. For example, mobile traffic is dominated by web browsing and streaming applications, so expert features only focus on them and exclude other applications. On the other hand, since expert features are defined by human experts, they are *static* by nature and cannot readily adapt to current network conditions.

Table 1: Examples of expert features.

Types	Examples
General	Did the subscriber restart his phone before?
	Does the subscriber use 4G?
	Does the subscriber use streaming services with 2G?
Signaling	Number of attach messages
	Number of attach failures
	Number of handovers
	Error code of a signaling message
Web browsing	Number of out-of-sequence packets
	Connection time
	Page response delay
Streaming	Number of retransmission packets
	Average number of pauses
	Average number of stalls

4.2 Sequential Pattern Features

Sequential pattern features aim to capture the dynamic patterns of network usage by subscribers. For example, expert features cannot capture the detailed interaction process between the network and a subscriber in the form of a sequence of signaling messages. If we consider a subscriber's signaling messages in sequential order, we may discover some anomalous signaling information with temporal dependency, which is indicative of a user fault or network fault. This kind of temporal information can be modeled as sequential patterns, which can be extracted to form classification features.

Thus, we propose to extract sequential patterns from the signaling message sequences. Specifically, we select three key fields from control-plane subscriber records: signaling message type, succeed flag, and radio access type. We denote them as a set $I = \{i_1, i_2, i_3\}$. As an example, I={*UE-triggered attach request, succeed, 4G*} means that a user launched a successful attach request with 4G connectivity. Similar to expert features, we collect the control-plane subscriber records that are within three hours before the subscriber's complaint time, and form a sequence $S = \langle (I_1, t_1), (I_2, t_2), \ldots, (I_k, t_k) \rangle$, where I_k is the set of the three selected key fields at time t_k, and $t_1 < t_2 < \ldots < t_k$.

From the sequence data collection for training, we mine discriminative sequential pattern features which can help classify the fault type. A sequential pattern is discriminative if it appears in many subscriber sequences from one class, but is very rare in subscriber sequences from the other class. We can measure the discriminative degree of a sequential pattern, for example, by information gain. Consider the following pattern as an example: P=⟨{*UE-triggered attach request, succeed, 4G*}, {*Cell update, succeed, 4G*}, {*Network-triggered routing modification, succeed, 2G*}⟩. It means that a user first successfully requested the service with 4G connectivity, but then the network side updated the cell connectivity and caused the next signaling message to show routing information changes and a fall-back to 2G connectivity. This sequential pattern shows that the subscriber's connectivity changed from 4G to 2G over time, which indicates a network congestion problem that caused the fall-back. We find that this pattern appears in 38% of all network fault instances while only 6% of all user fault instances. Thus, it can be viewed as a discriminative feature of the network fault.

PrefixSpan [21] is a widely used algorithm for mining *frequent sequential patterns* in the literature. However, for mining *discriminative sequential patterns*, we cannot simply apply PrefixSpan on the training subscriber sequences. This is because a discriminative sequential pattern may not have a high frequency in the whole training set, whereas many frequent sequential patterns discovered by PrefixSpan are not discriminative. If we set the minimum support threshold very low in PrefixSpan in order not to miss the discriminative patterns, the mining process may be very long and produce an explosive set of frequent patterns. To overcome this problem, we adopt the *model-based search tree* (M^bT) [10], which takes a top-down data partition approach and integrates frequent pattern mining and feature selection into a unified decision-tree-based framework. The M^bT procedure starts with the whole training sequence set and mines a set of frequent sequential patterns from the data using PrefixSpan. The best sequential pattern is selected according to information gain and used to divide the training set into two subsets, one containing this sequential pattern and the other not. The mining and pattern selection procedure is repeated on each of the subsets until the subset is small enough or the examples in the subset have the same class label. After the procedure completes, a small set of discriminative sequential pattern features, i.e., the best feature selected at each node of the model-based search tree, are discovered. Due to its ‘‘*divide-and-conquer*” nature, M^{b}T can efficiently discover discriminative features even with very low support, but not overwhelm the mining process or the result set.

4.3 Feature Vector Representation

Given the set of extracted expert features and sequential pattern features, we transform the training subscriber records into the feature vector representation, where each feature corresponds to a dimension in the feature vector. Specifically, for each expert feature, we compute the feature value from a set of subscriber records of a subscriber; for each sequential pattern feature, the feature value is 1 if the subscriber sequence contains the pattern, and 0 otherwise. The training set is thus transformed to the feature vector representation $\{(\mathbf{x}_i, y_i)\}|_{i=1}^{n}$, where $\mathbf{x}_i$ is the feature vector for the set of subscriber records R_i for subscriber i, y_i is the class label, and n is the number of observed subscribers.

4.4 Ground-Truth Label Acquisition

We have two ways to acquire the ground-truth labels for the training instances. First, network engineers, after receiving a trouble ticket, check the network entities as depicted in Figure 2 and identify the fault type, which in turn can be treated as the ground-truth label. Second, customer care assistants conduct feedback surveys by calling back complaining subscribers, whose responses can be used as the ground-truth labels as well.

5 ICCA DESIGN

In this section, we present the design details of the overall ICCA architecture and each of its components. The main idea of ICCA is to combine offline model training and online feature computation to achieve real-time fault classification. We demonstrate how ICCA processes subscriber records and preforms feature engineering in a scalable and real-time manner.

5.1 Architectural Overview

Figure 4 shows the system architecture of ICCA. It includes four key components: (1) a distributed key-value store for raw data and feature vector storage, (2) a distributed batch computing engine for offline model training, (3) a distributed stream processing engine for online feature computation, and (4) a fast query system for real-time fault classification. In addition, ICCA includes other components for data input, result output, label feedback, etc.

Currently, we implement the key-value store and the batch computing engine on the open-source systems HBase [2] and Spark [3], respectively. Both systems have been widely adopted for distributed analytics. On the other hand, we implement the stream processing engine on our in-house system called StreamSMART, which has

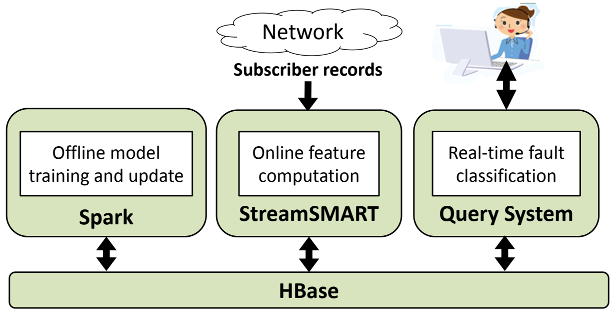
Network
Subscriber records
Offline model training and update
Spark
Online feature computation
StreamSMART
Real-time fault classification
Query System
HBase

Figure 4: ICCA architecture.

a similar architecture to the open-source distributed stream processing system Storm [4]. At a high level, StreamSMART partitions the processing of a continuous data stream across multiple worker nodes like Storm, but makes specific optimizations to processing performance and fault tolerance.

The workflow of ICCA can be summarized as follows. ICCA performs offline model training on Spark based on an initial training set (see Section 5.2). Also, ICCA performs online feature computation on StreamSMART, which collects a continuous stream of subscriber records, specifies the feature values for each subscriber, and stores the results in HBase (see Section 5.3). Upon the receipt of a complaint, the query system of ICCA retrieves the features from HBase, performs fault classification, and returns the classification result (see Section 5.4). In addition, ICCA periodically uses the classification results and computed features to re-train the model (see Section 5.5).

5.2 Offline Model Training

We perform model training in offline mode on Spark. When ICCA is first deployed, we bootstrap an initial training set of subscribers' complaints and their ground-truths (see Section 6.1.1), and build a classifier on the training set. While there are many possibilities to construct the classifier, we currently choose the Gradient Boosting Decision Tree (GBDT) [11] as it achieves high classification accuracy based on our deployment experience. We set the model parameters by grid search and 5-fold cross validation on the training set.

We elaborate the steps of our model training as follows. The training set is represented in the form $\{(\mathbf{x}_i, y_i)\}|_{i=1}^{n}$ (see Section 4.3). GBDT is a boosting model with decision trees as base learners. For the binary classification problem, the logit function $F(.)$ can be estimated as

$$\hat{F}(\mathbf{x}) = \sum_{i=0}^{N} \alpha_i F_i(\mathbf{x}),$$

where $F_i(\mathbf{x})$ is the i-th base learner, α_i is the corresponding weight, and N is the number of base learners. We initialize $\hat{F}(\mathbf{x})$ to 0 and estimate it by an iterative method. In the first iteration, $F_0(\mathbf{x})$ is initialized to 1, and α_0 is calculated by

$$\alpha_0 = \arg\min_{\alpha} \sum_{i=1}^{n} L(y_i, \hat{F}(\mathbf{x}) + \alpha F_0(\mathbf{x})),$$

where L is a pre-defined loss function. Then $\hat{F}(\mathbf{x})$ is updated as $\hat{F}(\mathbf{x}) \leftarrow \hat{F}(\mathbf{x}) + \alpha_0 F_0(\mathbf{x})$ and the pseudo-residual or gradient of each training instance is calculated by $r_i = -\frac{\partial L(y_i, F(\mathbf{x}_i))}{\partial F(\mathbf{x}_i)}|_{F(\mathbf{x}_i)=\hat{F}(\mathbf{x}_i)}$. In the second iteration, the next base learner $F_1(\mathbf{x})$ is trained in data $\{(\mathbf{x}_i, r_i)\}|_{i=1}^{n}$ using the decision tree model with the pre-defined model parameters. α_1 is calculated in the same way as α_0. Then we update $\hat{F}(\mathbf{x}) \leftarrow \hat{F}(\mathbf{x}) + \alpha_1 F_1(\mathbf{x})$, and calculate the gradient r_i of each training instance similarly as above. We repeat this process until the number of decision trees is greater than N. In our implementation, the classifier is defined as $f(\mathbf{x}) = \text{sign}(\hat{F}(\mathbf{x}) - c)$, where c is a threshold.

5.3 Online Feature Computation

The probes at the cellular network continuously generate subscriber records that are streamed into ICCA via StreamSMART. One challenge is that we do not know in advance which subscriber will file complaints. Thus, we choose to compute feature values for *all* observed subscribers based on the input subscriber records and store the computed feature values in HBase. This offloads the query system, which now simply leverages the computed feature values for a given complaining subscriber and returns the classification result (see Section 5.4). Currently, StreamSMART distributes the processing of subscriber records by hashing a subscriber ID to one of the worker nodes; how to more evenly distribute the workload across worker nodes is subject to future work.

StreamSMART computes the values of both expert features and sequential pattern features. For expert features, we compute their feature values from subscriber records in discrete time intervals. Here, we set a one-minute time window to collect subscriber records, and compute the expert feature values from the subscriber records. The calculated feature values are then stored in HBase keyed by the subscriber ID at the one-minute granularity. For the sequential pattern features, since they are mined from the training sequences within a three-hour time window of a complaint, we also use a three-hour sliding time window to collect the incoming subscriber records for pattern matching. For any sequence in the sliding window, if it contains a sequential pattern, the feature value is set to 1. The feature values for the sequential pattern features are also stored in HBase for efficient retrieval. Furthermore, the raw subscriber records are stored in HBase as well for periodic model update (see Section 5.5).

Since we compute the features for all observed subscribers, the storage space for the features in HBase can grow significantly large. Thus, we regularly remove the outdated data of a subscriber if it is not associated with any complaint to reclaim storage space. Currently, we perform the removal on a daily basis.

5.4 Real-Time Fault Classification

When a subscriber files a complaint, a customer care assistant can issue a request to the query system of ICCA, which immediately retrieves the slices of feature vectors keyed by the subscriber ID for the past three hours from HBase. The slices of feature vectors are combined into one feature vector by the logic ‘‘OR” operator. Then the query system applies the classification model on the feature vector and predicts the fault type. The predicted fault type is then returned to the customer care assistant to help handle the subscriber complaint. ICCA currently can handle 800 concurrent subscribers' complaints in less than 2.3 seconds (see Section 6.2).

5.5 Pattern and Model Update

ICCA updates the sequential pattern features and classification model periodically on Spark, so as to adapt to the dynamic changes of subscribers' network access behaviors and network conditions. Whenever some latest complaints have been properly handled and the true fault types have been identified, the corresponding subscriber records are retrieved from HBase and treated as additional training instances. On the enriched training set, ICCA performs sequential pattern mining to update the sequential pattern features. The stored feature vectors in HBase are updated with the new sequential pattern features. ICCA re-builds the classification model based on the new features and the enriched training set.

5.6 Deployment

We have deployed ICCA in a metropolitan LTE network in China since December 2016. ICCA currently runs on a cluster of 10 commodity rack servers, each of which has two Intel Xeon E5-2450 2.1 GHz CPUs and 256 GB RAM. Each server runs in hyper-threading mode, and has 32 logical CPU cores in total. All servers are interconnected by a 10Gb/s network. We deploy Spark, StreamSMART, and HBase in full distributed mode across all servers, and deploy the query system on one of the servers. Such a deployment setting enables ICCA to process billions of subscriber records (see Section 2.2).

6 EVALUATION

In this section, we present evaluation results of ICCA. We compare ICCA with the traditional expert rule-based approach (see Section 1) in terms of classification accuracy and query performance.

6.1 Classification Accuracy

6.1.1 Effects of Features. We first study the classification accuracy due to expert features, sequential pattern features, and the combination of both. To build the ground truths, we have collected 3,458 subscribers' complaints during December 2015 to July 2016. We obtain their ground-truth labels from network operators as the training set. We construct 164 expert features (see Section 4.1) and around 350 sequential pattern features from the training set, and train the ensemble classifier using GBDT (see Section 5.2). We use grid search and 5-fold cross validation to find the best parameters for GBDT such that the accuracy is maximized.

We evaluate the classification accuracy of different types of features using 5-fold cross validation and the area under curve (AUC) as the metric. Table 2 shows the results of five runs in 5-fold cross validation. We observe that both expert features and sequential pattern features have comparable classification accuracy (0.791 and 0.795, respectively). Recall that expert features cover over a hundred of fields, while sequential pattern features only cover three fields. Although expert features cover significantly more fields, they are statically defined by experts based on domain knowledge. On the other hand, sequential pattern features are more adaptive to the current network conditions as they are extracted by the M^{b}T algorithm on-the-fly. Nevertheless, the combined features

Table 2: Classification accuracy (in AUC) of different types of features under 5-fold cross validation.

	Expert features	Sequential pattern features	Combined features
Run 1	0.821	0.820	0.833
Run 2	0.773	0.766	0.787
Run 3	0.775	0.789	0.807
Run 4	0.790	0.789	0.803
Run 5	0.798	0.812	0.823
Average	0.791	0.795	0.811

further improve the average AUC to 0.811. Also, each run of the 5-fold cross validation shows that the combined features achieve higher accuracy than the individual ones.

We emphasize that although the combined features seem to only increase the AUC by a slight margin (less than 0.02 on average), the improvement is indeed significant. We conduct the paired t-test to compare the AUCs of different types of features, under the null hypothesis that any two types of features have the same classification accuracy. We measure the p-value, such that a p-value that is smaller than a threshold (currently set as 0.05) will reject the null hypothesis. We find that when we compare the expert features and the sequential pattern features, the p-value is 0.47, so we cannot reject the null hypothesis. On the other hand, when the combined features are compared to the expert features and the sequential pattern features, the p-values are 0.008 and 0.001, respectively. This implies that the combined features show statistically different classification accuracy from the individual features; in other words, they actually bring accuracy gains.

6.1.2 Classification in Production. We now evaluate the classification accuracy of ICCA in production. In parallel with ICCA, we also run the traditional expert rule-based approach that performs diagnosis by pre-configuring a set of rules defined by experts for our comparisons.

Our evaluation methodology is based on surveying the feedbacks from the subscribers who have filed complaints. Specifically, we collected 2,863 complaints during December 29 to 31, 2016 for our evaluation. Among them, we randomly selected 300 cases, and called the subscribers to survey whether the user or network faults were correctly classified. Most of the subscribers were unreachable, refused to participate in the survey, or provided ambiguous answers. Nevertheless, we successfully collected the feedbacks from 95 of them. We use the 95 responses as true labels.

Table 3 shows the confusion matrices of the classification results of the expert rule-based approach and ICCA. The expert rule-based approach can only make correct classifications for 65 out of 95 cases; for the 30 remaining cases, 13 of them cannot be classified at all as they cannot be covered by the rules. On the other hand, ICCA can make correct classifications for 81 out of 95 cases. Overall, the expert rule-based approach achieves an accuracy of 68.4%, while ICCA achieves an accuracy of 85.3%.

We further evaluate the classification of user faults based on the 95 responses, among which 82 of them are actual user faults. We measure both precision (i.e., the fraction of reported cases that are true) and recall (i.e., the fraction of true cases that are reported).

Table 3: Confusion matrices for the expert rule-based approach and ICCA.

(a) Expert rule-based approach

	Predicted: Network fault	Predicted: User fault	Unable to predict
Actual: Network fault	5	3	5
Actual: User fault	14	60	8

(b) ICCA

	Predicted: Network fault	Predicted: User fault
Actual: Network fault	4	9
Actual: User fault	5	77

Although the expert rule-based approach achieves a precision of 95.2%, many of the actual user faults cannot be predicted and its recall is only 73.2%. On the other hand, ICCA achieves a precision of 89.5% and a recall of 93.9%.

Unfortunately, due to insufficient cases for actual network faults (13 cases only), we cannot make fair evaluation on the classification of network faults based on the 95 responses. Instead, we look into a different evaluation approach, and focus on evaluating the precision of classifying network faults. Specifically, among the 2,863 received complaints, the expert rule-based approach classifies 400 of them as network faults. We called the subscribers of those complaints and collected 185 valid responses, among which 58 of them are actual network faults. Thus, the precision of the expert rule-based approach on classifying network faults is 31.4%. On the other hand, ICCA classifies 260 of the 2,863 complaints as network faults. We called the subscribers of those complaints and collected 133 valid responses, among which 74 of them are actual network faults. Thus, the precision of ICCA on classifying network faults is 55.6%.

We remark that the above approach cannot be used to evaluate the recall of classifying network faults, as the missing actual network faults cannot be identified; we pose this issue as future work. In addition, the same subscriber may be involved in different parts of the above analysis, yet we consolidated our questions and ensure that each involved subscriber is only called once.

6.2 Query Performance

We now examine the query performance of ICCA, and demonstrate via stress tests that it can achieve real-time fault classification even under high query load. Since it is difficult to directly stress-test the query performance of our production system, we set up a controlled experiment to conduct our measurements as follows. Specifically, we set up a local testbed of three servers that have the same hardware configurations as in our production system. We simulate three-hour subscriber records of up to 800 complaining subscribers, compute their expert features and sequential pattern features, and load the results into HBase. We issue queries concurrently for a

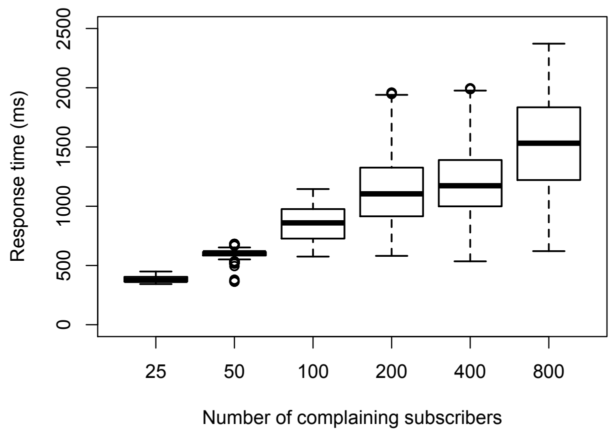

Figure 5: Query performance of ICCA versus the number of complaining subscribers.

number of complaining subscribers, varied from 25 to 800 in our evaluation. We measure the *query response time*, defined as the time from issuing a query for each complaining subscriber until the classification result is obtained. We collect query time results over 10 runs. We present *boxplots* for all obtained query time samples (i.e., 10 times the number of complaining subscribers); each boxplot shows the minimum, lower quartile, median, upper quartile, and maximum query times.

Figure 5 shows the query performance of ICCA versus the number of complaining subscribers. The query response time and its variance increase with the number of complaining subscribers, as more concurrent queries are issued to HBase. Nevertheless, each query can be completed quickly. Even for 800 complaining subscribers, the worst-case query response time is no more than 2.3 seconds.

We point out that ICCA performs much faster than our traditional expert rule-based approach. Based on our evaluation results in production, the expert rule-based approach takes around 20 seconds to issue queries for 100 complaining subscribers. The reason is that the expert rule-based approach configures a number of rules, each of which has one to many logical operations. It queries HBase multiple times in order to obtain a classification result. In contrast, ICCA only queries HBase twice for fetching both expert features and sequential pattern features by subscriber ID, and performs immediate matching to return the query result.

7 LESSONS LEARNED

This paper presents our experiences of applying data mining and machine learning techniques to process massive network data and achieve intelligent cellular network diagnosis at large scale, and realize our ideas in a production system called ICCA. We summarize our lessons learned from the deployment of ICCA.

Data collection: Subscribers often file complaints based on their experiences, which are subjective by nature. Even though we can standardize the procedures and questions for customer care services and feedback surveys, the answers provided by subscribers may still have a high degree of variance and hence are not fully trustworthy. Thus, we take a data-driven approach. By mining the useful features from massive network data, we capture the characteristics of network conditions that help diagnose subscribers' complaints. One important lesson is that our collected data should be complete and have full coverage of both the control plane and the data plane of the cellular network, in order to achieve accurate diagnosis. Another important lesson is that since the collected data comes from various cellular network technologies, its format should be well-defined and unified, so as to facilitate cellular network diagnosis. In future work, we also examine the inclusion of other data sources, such as subscriber profiles and network device logs, for more accurate diagnosis.

Automatic sequential pattern mining: Traditional diagnosis is based on the expert rule-based approach (see Section 1), which heavily relies on the domain knowledge of network engineers to configure the right set of rules for cellular network diagnosis. These expert rules can only address part of the fault classification problem, and ignore the dynamic features provided by the massive network data itself. In our case, the correlations of the control-plane signaling messages provide important hints for diagnosis, but they are ignored by expert rules. Nevertheless, our system design does not exclude expert features as they still provide valuable information, but instead we complement them by adding sequential pattern features that are dynamically extracted from the network data. An important lesson is that combining both expert features and sequential pattern features can provide more accurate diagnosis, as justified by our evaluation (see Section 6.1.1).

Large-scale system design: Processing data of up to 50 million subscribers online and providing accurate classification results for subscribers' complaints in real time are non-trivial tasks. We design ICCA based on both public, well-proven distributed computing and storage systems (Spark and HBase) and our in-house distributed stream processing system (StreamSMART). We also require that ICCA be deployable on commodity off-the-shelf rack servers and network switches. This simplifies the deployment of ICCA if we want to deploy it in a different cellular network. Also, we can readily add or upgrade servers for ICCA if we want to scale its resources.

Our experience is that the performance bottleneck of ICCA is mainly on CPU computations for feature extraction and pattern matching, compared to the disk I/Os or network communications among the servers. Thus, if more subscribers need to be supported, we can increase the number of servers for more CPU resources.

Generality and portability: Our ICCA system design is modularized to ensure that the components are general and portable. While some components are specific for the fault classification problem solved by ICCA (e.g., discriminative sequential pattern mining, ensemble learning algorithms, and online matching methods), many other components are general and can be smoothly ported to other applications that perform analytics on subscriber records. Examples include (1) the distributed stream processing engine, (2) the algorithmic framework, and (3) the related API and other interfaces. In addition, the trained model parameters in one cellular network can be applied to other cellular networks or similar applications with the same data schema.

8 RELATED WORK

In this section, we review related work, with emphasis on characterization and analytics of cellular networks.

Cellular network characterization: Our work takes a network operator's perspective by performing passive measurements (i.e., without injecting measurement traffic to the network) and analyzing collected data at the cellular core. Several studies also follow this direction and focus on different cellular network characteristics. Examples include: achievable download throughput [12], radio resource usage and power consumption [22], network gateway coverage and impact of content placement [27], usage patterns of mobile applications [26], control-plane signaling overheads due to IP-level packets [13, 20, 23], TCP flow characteristics in 3G UMTS [7] and 4G LTE [14], web browsing quality-of-experience [5], etc. Our work differs from above by focusing on fault classification about cellular network access. In addition, the above studies perform characterization offline, while we emphasize real-time characterization of network data.

Cellular network analytics: Machine learning and other analytics techniques have been widely used to identify sophisticated patterns from network measurement (see survey [18]). Specifically for cellular networks, machine learning has been used to predict all drops [24] and diagnose performance anomalies [6, 7].

Some studies focus on building analytics systems for cellular network analytics. For example, CellIQ [15] performs real-time graph analytics to identify spatial-temporal traffic hotspots and handoff sequences in cellular networks. CellScope [16] applies multi-task learning to study the trade-off between data collection time and analytics accuracy in cellular network diagnosis. Our proposed ICCA is also designed for real-time large-scale analytics in cellular networks, with a specific focus on fault classification. Note that some cellular carriers have also developed their own large-scale analytics systems to manage cellular networks (e.g., [19, 25]), but their design details are proprietary and cannot be directly compared with ICCA.

Analysis of customer care calls: Customer care calls provide useful information about network anomalies. Chen et al. [7] analyze customer care calls to detect anomaly events, and further leverage user tweets on Twitter to recover more attributes of the detected events. TREAT [17] mines sentences and phrases from text-based customer care calls and trouble tickets to classify reported issues from customers. However, the content of customer care calls can be highly subjective (see Section 7). Instead of only relying on the content of customer care calls, ICCA extracts patterns from massive network data to achieve timely and accurate fault classification.

9 CONCLUSION

We design and implement Intelligent Customer Care Assistant (ICCA), a distributed fault classification system for large-scale cellular network diagnosis. ICCA takes a data-driven approach: it combines traditional domain knowledge (i.e., expert features) and dynamic sequential patterns of control-plane and data-plane subscriber records (i.e., sequential pattern features) to achieve real-time fault classification between user faults and network faults. We demonstrate how ICCA realizes state-of-the-art data mining and machine learning techniques to achieve the goal. ICCA is deployed in a metropolitan LTE network in China with around 50 million subscribers. Evaluation shows that ICCA can achieve high classification accuracy and high query performance.

REFERENCES

[1] 3GPP. http://www.3gpp.org/.
[2] Apache HBase. http://hbase.apache.org/.
[3] Apache Spark. http://spark.apache.org/.
[4] Apache Storm. http://storm.apache.org/.
[5] A. Balachandran, V. Aggarwal, E. Halepovic, J. Pang, S. Seshan, S. Venkataraman, and H. Yan. Modeling Web Quality-of-Experience on Cellular Networks. In *Proc. of ACM MobiCom*, 2014.
[6] P. Casas, P. Fiadinoy, and A. D'Alconzo. Machine-Learning Based Approaches for Anomaly Detection and Classification in Cellular Networks. In *Proc. of IFIP TMA*, 2016.
[7] Y.-C. Chen, G. M. Lee, N. Duffield, L. Qiu, and J. Wang. Event Detection using Customer Care Calls. In *Proc. of IEEE INFOCOM*, 2013.
[8] Cisco. Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2015 - 2020. http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html, Feb 2016.
[9] Cisco. Understanding CDR. https://supportforums.cisco.com/document/53056/understanding-cdr-call-detail-records, 2016.
[10] W. Fan, K. Zhang, H. Cheng, J. Gao, X. Yan, J. Han, P. S. Yu, and O. Verscheure. Direct mining of discriminative and essential frequent patterns via model-based search tree. In *KDD*, pages 230--238, 2008.
[11] J. H. Friedman. Greedy Function Approximation: A Gradient Boosting Machine. *The Annals of Statistics*, 29(5):1189--1232, 2001.
[12] A. Gerber, J. Pang, O. Spatscheck, and S. Venkataraman. Speed Testing without Speed Tests: Estimating Achievable Download Speed from Passive Measurements. In *Proc. of ACM IMC*, 2010.
[13] X. He, P. P. C. Lee, L. Pan, C. He, and J. C. S. Lui. A Panoramic View of 3G Data/Control-Plane Traffic: Mobile Device Perspective. In *Proc. of IFIP Networking*, 2012.
[14] J. Huang, F. Qian, Y. Guo, Y. Zhou, Q. Xu, Z. M. Mao, S. Sen, and O. Spatscheck. An In-depth Study of LTE: Effect of Network Protocol and Application Behavior on Performance. In *Proc. of ACM SIGCOMM*, 2013.
[15] A. P. Iyer, L. E. Li, and I. Stoica. CellIQ : Real-Time Cellular Network Analytics at Scale. In *Proc. of USENIX NSDI*, 2015.
[16] A. P. Iyer, I. Stoica, M. Chowdhury, and L. E. Li. Fast and Accurate Performance Analysis of LTE Radio Access Networks. Technical Report UCB/EECS-2016-114, EECS Dept, UC Berkeley, 2016.
[17] R. Li, X. Huang, S. Song, J. Wang, and W. Wang. Towards Customer Trouble Tickets Resolution Automation in Large Cellular services: Demo. In *Proc. of ACM MobiCom*, 2016.
[18] T. T. Nguyen and G. Armitage. A Survey of Techniques for Internet Traffic Classification using Machine Learning. *IEEE Comm. Surveys and Tutorials*, 10(4), 2008.
[19] Nokia. Nokia Wireless Network Guardian. https://resources.alcatel-lucent.com/?cid=163631, 2016.
[20] U. Paul, A. P. Subramanian, M. M. Buddhikot, and S. R. Das. Understanding Traffic Dynamics in Cellular Data Networks. In *Proc. of IEEE INFOCOM*, 2011.
[21] J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M. Hsu. Prefixspan: Mining sequential patterns by prefix-projected growth. In *ICDE*, pages 215--224, 2001.
[22] F. Qian, Z. Wang, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck. Characterizing Radio Resource Allocation for 3G Networks. In *Proc. of ACM IMC*, 2010.
[23] L. Qian, E. W. W. Chan, P. P. C. Lee, and C. He. Characterization of 3G Control-Plane Signaling Overhead from a Data-Plane Perspective. In *Proc. of ACM MSWiM*, 2012.
[24] N. Theera-Ampornpunt, S. Bagchi, K. R. Joshi, and R. K. Panta. Using Big Data for More Dependability: A Cellular Network Tale. In *Proc. of HotDep*, 2013.
[25] Verizon. Verizon Adds Cloudera's Cloud-Based Big Data Analytics Solution to Verizon Cloud Ecosystem. http://www.verizon.com/about/news/verizon-adds-clouderas-cloudbased-big-data-analytics-solution-verizon-cloud-ecosystem/, 2013.
[26] Q. Xu, J. Erman, A. Gerber, Z. M. Mao, J. Pang, and S. Venkataraman. Identifying Diverse Usage Behaviors of Smartphone Apps. In *Proc. of ACM IMC*, Nov 2011.
[27] Q. Xu, J. Huang, Z. Wang, F. Qian, A. Gerber, and Z. M. Mao. Cellular Data Network Infrastructure Characterization and Implication on Mobile Content Placement. In *Proc. of ACM SIGMETRICS*, 2011.