Research Article

City-Wide NB-IoT Network Monitoring and Diagnosing

Zhenxian Hu, Guangtao Xue, Yichao Chen, Minglu Li, Ming Wang, and Feng Lv

1Department of Computer Science, Shanghai Jiao Tong University, China
2Department of Computer Science, University of Illinois at Urbana-Champaign, USA
3Department of Computer Science, Central South University, China

Correspondence should be addressed to Guangtao Xue; gt_xue@sjtu.edu.cn

Received 16 June 2021; Revised 21 February 2022; Accepted 1 March 2022; Published 11 April 2022

Academic Editor: Lei Zhang

Copyright © 2022 Zhenxian Hu et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

NarrowBand-IoT (NB-IoT) is a radio-access technology standardized by 3GPP to support a large set of use cases associated with the rapid deployment of massive machine-type communications. NB-IoT facilitates the connection of devices in inaccessible areas, extends battery life, and reduces device complexity. Unfortunately, the opacity of the underlying schema (i.e., the way that these benefits are achieved) makes it very difficult for most users and developers to manage deployment scenarios. In this study, we built an embedded system comprising a Raspberry Pi with an NB module, referred to as NBPilot, which interacts with NB networks to identify essential signaling messages transmitted by a Qualcomm NB modem. This system gives researchers and developers an unprecedented understanding of network behavior as well as the ability to adjust them to their particular requirements. We employed the state-of-the-art machine learning techniques for modeling and the analysis of NB performance. The efficacy of the proposed NBPilot system was established by applying it to a metropolitan NB-IoT network with over 2,000 NB sites for the collection and testing of data trace as well as the validation of a cellular station prior to going online. The developed machine learning techniques predict the actual performance (rate, delay, and power consumption) of the user equipment’s connection against the QoS advertised by the operator and pinpoint the affecting factors.

1. Introduction

The Internet of Things (IoT) is constantly being expanded to include an ever growing range of devices, such as sensors, actuators, meters (water, gas, and electric), cars, and appliances. Essentially, the IoT comprises an enormous number of networks that vary in their design objectives. For example, some networks were designed for local-area coverage (e.g., a single home), whereas others were designed for wide-area coverage.

NarrowBand-Internet of Things (NB-IoT) is a standards-based low-power wide area (LPWA) technology developed to enable a wide range of new IoT devices and services. NB-IoT significantly improves the power consumption of user devices, system capacity, and spectrum efficiency, especially in deep coverage. Battery life of more than 10 years can be supported for a wide range of use cases. NB-IoT can be scaled down and implemented in a greatly simplified form for low-throughput, delay-tolerant applications, thereby allowing data rates in the tens of kbps within a bandwidth of 200 kHz. NB-IoT can also be deployed within existing LTE bands, in the guard band between two regular LTE carriers, or in standalone mode, thereby providing easy migration paths for refarming the GSM spectrum. According to GSMA [1], there were 106 NB-IoT launches as of June 2021.

Unfortunately, cellular networks remain a blackbox to most users and developers. Cellular network is a closed network deployed by operators on a licensed spectrum which cannot be accessed by public monitoring. A lack of open access to fine-grained runtime network operations makes it exceedingly difficult for researchers and developers to understand and refine network behaviors [2]. For example, a particular device may differ in its energy characteristics under different operators, and many developers are unaware of why this is the case. Another example would be the use of NB-IoT to monitor and control vehicles, wherein forward control commands are difficult to reach and tend to be delayed. In this situation,
developers would find it difficult to determine whether the observed behavior is affected by the quality of the radio signal or is a side effect of an energy-saving feature.

In industry, many operators employ a large number of testing engineers and experts in analysis to keep their networks operating within designated specifications. In a typical scenario, engineers employ customized cellphones that are connected to a notebook PC to launch testing scripts. Debugging tools for baseband suppliers, such as QXDM [3], XCAL [4], and MTK Catcher [5], are then used to analyze cellular network messages and provide fine-grained information. Unfortunately, those tools are external (by attaching cellphones to PC) and proprietary software (cannot access). After field engineers have collected data traces for testing, analysts are tasked with identifying problems, such as low throughput and dropped calls reported by monitoring software such as Guardian [6], Optimizer [7], Nastar [8], and Actix [9], before undertaking the tedious task of manual checking each event.

In this study, we adopted the approach used in MobileInsight [2] in our development of a full-featured test bed referred to as NB Pilot. The proposed system can carry out active performance testing, while simultaneously acquiring user and signaling data by taking advantage of the transparent Qualcomm baseband decoding capabilities. We then applied this tool to explore the daily operations of a city-scale NB-IoT network from the perspective of site verification and network maintenance. To the best of our knowledge, the dataset used in this study is the fastest growing NB network in the world (more than 2,000 NB sites in just six months) and of high dimensionality.

There usually exist multiple network operators in the same area, offering different configurations and IoT backend platforms with constrains of APN selections, supported data rates, etc. Each network operator has its own network configurations, which may affect the measurement results in terms of power consumption, network coverage, etc. The sheer volume and dimensionality of this data would make it almost impossible to have human experts perform troubleshooting using conventional rule-based systems. In many cases, the functioning of an NB network relies on hundreds or even thousands of items and parameters. Thus, we employed the state-of-art machine learning techniques for modeling and the analysis of NB performance. This work makes the following main contributions:

(i) We reverse-engineered the Qualcomm NB chipset to decode essential NB frame details and illustrate the procedures used to deal with various energy footprints. We also describe several typical configurations (under different operators), which have a nonnegligible influence on performance

(ii) The proposed system was applied to a metropolitan NB-IoT network for data trace collection, testing, and verifying thousands of individual sites. To the best of our knowledge, this is the largest NB-IoT database ever explored. The open access to the dataset can go through https://github.com/Zhenxian-Hu/NBPilot

(iii) We applied machine learning to facilitate the interpretation of a dataset collected over a six-month period with the aim of identifying the most important features underlying low-throughput situations and long delays

(iv) We showed that nodes yield significantly imbalanced energy consumption across different locations, operators, and module vendors. Such performance variance can be attributed to several key factors including poor network coverage level, long tail power profile due to conservative inactivity timer settings, and excessive control message repetitions during random access control

The rest of the paper is organized as follows. Section 3 then presents the design of the NB Pilot system, and Section 4 presents our evaluation of its typical performance. Section 5 supplements performance evaluation with examples of machine learning models. Section 6 specifies energy performance and models with discussions of the optimization. Section 7 outlines related work. Section 8 concludes.

2. Background

NB-IoT is radio-access technology designed by 3GPP to meet the connectivity requirements of massive MTC applications. The aim of this scheme was to provide cost-effective connectivity to billions of IoT devices, with support for low power consumption and the use of low-cost devices, while ensuring excellent coverage.

2.1. NB-IoT Primer. The design of NB-IoT mimics that of LTE, due to the fact that they are both intended to facilitate radio network evolution and roll out in the form of software solutions implemented atop an existing LTE infrastructure. The overall structure of NB-IoT is illustrated in Figure 1.

UEs connect with eNB through the Uu air interface. The eNBs are then connected to the MME and the Serving Gateway (SGW) via the S1 interface for the transmission of NB-IoT messages and data packets. Two optimizations for the cellular Internet of Things (CIoT) have been defined for the evolved packet system (EPS) to enable the transmission of data to applications: User Plane CIoT EPS optimization and Control Plane CIoT EPS optimization [10].

Control Plane CIoT EPS optimization involves the transfer of UL data from the eNB to the MME, from which it may be transferred via SGW to the Packet Data Network Gateway (PGW) or to the Service Capability Exposure Function (SCEF). SCEF is used for the delivery of non-IP data over a control plane and provides an abstract interface for network services (authentication and authorization, discovery, and access network capabilities).

From these nodes, the UL data are forwarded to the application server. Downlink data is transmitted along the same paths in the reverse direction. Under this scheme, there is no need to establish a data radio bearer; i.e., data packets are sent to the signaling radio bearer instead. Thus, this scheme is particularly well-suited to the infrequent transmission of small data packets.
Under the User Plane CIoT EPS optimization scheme, user data is transferred in the same way as conventional data traffic; i.e., over radio bearers to the application server via SGW and PGW. Thus, there is a certain amount of overhead associated with establishing a connection; however, this scheme makes it far easier to transmit a sequence of data packets.

2.2. Power-Saving Techniques. The battery life of an MTC device depends to some extent on the technology used in the physical layer for transmitting and receiving data. However, longevity depends to a greater extent on the efficiency with which a device utilizes the idle and sleep modes that allow the powering down of many device components for extended periods.

Like LTE, NB-IoT uses two main RRC protocol states: RRC idle and RRC connected. In RRC-idle state, devices save power by freeing up resources that would otherwise be used to send measurement reports and uplink reference signals. In RRC-connected state, devices send and receive data directly. NB-IoT introduces two additional power-saving techniques: extended Discontinuous Reception (eDRX) and power-saving mode (PSM). Figure 2 illustrates the NB-IoT energy profiles. Upon boot up, the NB-IoT baseband will go through frequency scan, cell detection, and cell selection, then register to the network. In idle state, the UE periodically monitors the paging channel to check for incoming data. This periodicity (i.e., the DRX cycle) has been extended in NB-IoT from 2.56 s (maximum value in LTE) to a maximum eDRX of 175 minutes. A UE may also be allowed by the network to switch to PSM, in which the UE is registered to the network but is not reachable (i.e., paging is not monitored in terms of energy savings w.r.t. the idle state). At the expiration of the PSM cycle, the UE performs a Tracking Area Update (TAU). Two timers are defined for idle and PSM phases: $T_{3324}$ is the duration of the idle phase (up to 3 hours); $T_{3412}$ represents the TAU periodicity and thus determines the duration of the PSM cycle (up to 413 days).

3. The Design of NB Pilot

We were inspired by the MobileInsight project [2] to build a portable hardware and software system for decoding NB-IoT network messages and conducting experiments. The resulting NB Pilot system is used to dissect NB-IoT network procedures and parameters.

3.1. System Structure. Figure 3 presents a schematic illustration showing the overall NB Pilot system.

The hardware components are based on the Qualcomm modem chipset, due to the fact that our decoding scheme is currently operating in a Qualcomm series. Unlike LTE, which deals with smartphones as user equipment, we emulated IoT devices using a Raspberry Pi as the main control unit. NB-IoT cellular protocol stack at the modem includes PHY, MAC, RLC, and PDCP functionalities. Above the PHY and MAC layers is situated the control-plane protocol, RRC, which is used mainly for radio resource allocation and radio connection management. Note that this protocol is also involved in the transfer of signaling messages over the air. NAS is responsible for conveying nonradio signaling messages between the device and the core network.

We found several commercial NB modules that provide AT and DIAG ports, as long as the drivers are installed correctly. Quectel provides documentation [11] to guide configuration of the QMI WAN driver for the Qualcomm NB modem used in our prototype (Simcom7000c). The AT port can be read, and the log can be debugged at the same time. Commands are issued directly to the virtual device via AT commands in accordance with the chipset. These commands include the activation/deactivation of cellular message types, and callback registrations to receive hex logs. We set up the recorder to run on the Raspberry Pi, and the decoder was set up to run offline and obtain modem messages on a laptop.

The Raspberry Pi was used to run two sets of processes. The first set was for conducting user experiments, such as selecting and attaching NB networks, pinging websites, and establishing socket connections, all of which are wrapped AT commands and functions listed in chipset documents provided by vendors. The second set of processes operates like a daemon service to record raw modem logs, which are identified by the Qualcomm chips via the DIAG interface and then decoded in accordance with standard protocol written in the headers. To monitor the energy consumption, we attach a Monsoon power monitor to the Raspberry Pi. Figure 4 shows the system in a static environment, we use this scenario to establish the life cycle energy profile and in
the outdoor, we log the signal/signaling events in the MCU and model the energy consumption.

3.2. Issuing Tests via AT Commands. Since their inception in the 1980s, AT commands have been the preferred means of controlling modems. Standardized AT commands are issued by authorities, such as the International Telephone Union (ITU-T) and the European Telecommunications Standards Institute (ETSI) [12, 13]. These commands make it possible to perform a number of functions, including the selection of communication protocol, setting up the line speed, dialing numbers, and ending calls.

Manufacturers of baseband processors that provide cellular devices with modem functionality provide additional proprietary and vendor-specific AT commands with their chipsets. As a result, modem modules also support their own AT command sets and expose modem through serial interfaces when connected via USB to receive those AT commands.

Table 1 (AT commands of different NB modem vendors) lists a selection of the AT commands used in this work and comparison of AT command series provided by Qualcomm (Simcom7000c [14]), MTK (ME3616 [15]), and Huawei (BC95 [16]). We observe that different chipset vendors share few common items but differentiate even for the same functions.

The first step in checking whether a network connection is up is to ping a server on the Internet. The Simcom7000c has a special AT command specifically for this task. However, before the ping can go through, it is necessary to specify the Access Point Name (AT+CSTT), establish the IP bearer (AT+CIICR), and ensure that everything has been set up correctly by querying the local IP address that was assigned (AT+CIFS). As long as a proper IP address appears, then the technician can be sure that connectivity has been established and that the ping can be sent to the server (AT+CIPPING).

3.3. Reverse Engineering an NB-IoT Stack. The first issue in reverse engineering an NB-IoT stack is that ordinary NB development boards are unable to expose message-level cellular information. Thus, we leveraged an alternative side channel using the baseband chipset. The chipset supports an external diagnostic mode, which exposes the cellular interface to the USB port.

The cellular interface maps itself to a virtual device (e.g., /dev/diag) in the OS. This virtual device exposes all raw cellular messages as binary streams; the OS uses USB tethering to bind the virtual device to a USB port (e.g., /dev/ttyUSB). This enables the external collector to fetch cellular messages from the hardware interface. Next, we parse each message using the raw cellular logs.

Developing a message parser for each signaling message involves extracting the message format from the standards outlined for each protocol. We follow along the principles of another cellular decoding project MobileInsight [2]. Some formats can be extracted automatically including RRC and NAS. For instance, the NB RRC standards [17] provide abstract message notations under ASN.1, which can be readily compiled into message decoders [18]. Other messages must be manually converted into machine-readable formats by comparing with Qualcomm QXDM. Figure 5 illustrates the decoding procedure for Master Information Block message.

We extract events from modem message log. First, we identify modem procedures and timers from logged modem messages. The MAC layer is the lowest layer containing data packets related to logging; therefore, we treat the MAC-Transport-Block as a data procedure. Detailed timer values are obtained from SIB2 and RRC connection reconfiguration messages. When an event is parsed, we record (1) the state entering time as the event start time, (2) the state exiting time (not including reenter transition) as the event end time, and
settings and their energy feature effects in real networks:

(3) extra fields from procedures, such as signal strength and the number of sent/received bytes.

The range of solutions aimed at extending battery life must be balanced against requirements pertaining to reachability, the transmission frequency of different applications, and mobility.

We have developed Raspberry Pi plus different NB modem shields, including Quectel BC95/BG36/BG96, ME3616, and Simcom7000c/7070G/7080G. NBPilot offers APIs that allow developers to access and control heterogeneous UE modules. It hosts libraries to decode the radio-access logs of different UE modules and reports the decoded events in a unified format.

3.4. Interactivity between AT and Diag. Figure 6 illustrates the interactivity of the proposed system via energy feature settings and their effects in real networks:

(1) The AT command is used to set the PSM and eDRX parameters.

(2) The information is wrapped within an “Attach request Msg” from the UE to eNB, which includes the following parameters: $T_{3124}$ (active time 2 min), $T_{3412}$ (TAU 3 h), eDRX (2621.44 s), and PTW (5.12 s).

(3) In cases where the network returns an “Attach accept Msg” that does not include PSM or eDRX parameters, this is an indication that the network does not support eDRX functionality, such that we obtain the following parameters: $T_{3412}$ (12 h) and $T_{3324}$ (2 s).

NBPilot allows one to interact with and analyze the energy performance of the NB-IoT modules. For example, the developer can emit the AT command and observe the UE behavior immediately. In this case, the UART ports, including the main AT and debug log export functionalities, are routed to the USB-serial adaptors. We use a Monsoon power monitor, a state-of-the-art power measurement equipment, to measure the current consumed by the NB-IoT module. In this manner, the user can easily know when specific events occur and how they introduce the variation in the current.

In this section, we outlined the design and implementation of NBPilot. In the following section, we show how a number of NB network activities could be performed using the decoding capabilities of NBPilot.

4. Evaluation

We first evaluated the decoding capabilities of the selected modem by conducting a comparison using Qualcomm proprietary software as well as a crosswise comparison using LogViewer [16] from HiSilicon (Huawei) and Genie Logging Tool [15] from MTK for NB-IoT baseband debugging purpose. We then sought to provide a comprehensive illustration of the NB-IoT typical procedures, covering cell information, random access procedures, and uploading and downloading characteristics under different operators.

4.1. Decoding Capability Coverage. NBPilot decodes all signaling messages on RRC-r13 (for NB) and NAS (MM and SM) and partially supports PHY, MAC, RLC, and PDCP messages. Huawei provides a tool called LogViewer [16] to examine the RRC messages. It first finds the corresponding decoder .xml file and then uses the message ID to build a dictionary, the key of which is the message decoder node in the XML tree. MTK provides the similar tool called Genie Logging Tool [15] to observe RRC messages.

We manually check with QXDM, LogViewer, and Genie Tool. We find that RRC and NAS formats are 100% the same among three chipsets; they have endured the conformance testing before shipping into the market. We further compared NBPilot with another research tool MobileInsight [2], which can also decode NB RRC and NAS signaling information, but lacking L2/L1 capabilities. NBPilot currently supports nearly 40 types (38) of messages. In a typical user study of data uploading activity, the top 5 signaling message statistics are LTE RLC DL AM All PDU (20.9%), LTE MAC DL Transport Block (17.8%), LTE NB1 ML1 Sum Sys Info (14.0%), LTE NB1 ML1 GM DCI Info (11.2%), and LTE NB1 ML1 GM TX Report (11.2%).

4.2. Uploading and Downloading Activities. UE first searches for a cell on an appropriate frequency, reads the associated SIB information, and begins the random access procedure to establish an RRC connection. Using this connection, the UE registers with the core network via the NAS layer (if this has not already been done). The UE then returns to RRC IDLE state, whereupon it may again implement the random access procedure (when it has mobile-originated data to send), or waits until it is paged. Figure 7 presents schematic diagrams illustrating the downloading of data from eNodeB to UE and an opposite data uploading procedure.

We tested NBPilot under the three operators deployed in the city. Hereafter, the three operators are referred to as OPI, OPii, and OPiii. Figure 8 illustrates the time consumption in each procedure when attaching a particular NB network.

The RRC attach procedure includes system information reading, RRC connection setup, UE authentication, NAS security setup, MME accept, and EPS bearer establishment. In the above example, OPI spent 0.568 s/0.24 s/0.324 s/0.748 s/0.66 s/
0.13 s in each step. The time differences of the same procedure were raising our interest. As each signaling message containing tens of configuration items, it can explain some differences in the performance result. We elaborate here a few differences of the SIB2 configurations:

- mac-ContentionResolutionTimer means contention resolution timer; if the UE does not receive msg4 (contention resolution message) of RACH procedure for initial access, within this timer, then it goes back to msg1 of RACH procedure; OPIII set this parameter accordingly same or two or four times as OPI and OPII’s option. Also, OPIII has two times larger nprach-Periodicity time than OPI and OPII which indicates the time resource allocated by the base station.

4.3. Cell Selection and Mobility. NB-IoT is designed for infrequent and short messages between the UE and the network. It is assumed that the UE can exchange these messages while being served from one cell; therefore, a handover procedure during RRC connected is not needed. If such a cell change would be required, the UE first goes to the RRC-idle state and selects another cell therein.

For the RRC-idle state, cell reselection is defined for both intrafrequency and interfrequency cells [19]. Interfrequency refers here to the 180 kHz carrier, which means that even if two carriers are used in the in-band operation embedded into the same LTE carrier, this is still referred to as an inter-frequency reselection.

In order to find a cell, the UE first measures the received power and quality of the NRS. These values are then compared to cell-specific thresholds provided by the SIB-NB. The S-criteria states that if both values are above these thresholds, the UE considers itself to be in coverage of that cell. If the UE is in coverage of one cell, it camps on it. Depending on the received NRS power, the UE may have to start a cell reselection. The UE compares this power to a reselection threshold, which may be different for the

### Table 1: AT commands of different NB modem vendors.

|                      | Simcom7000c | ME 3616 | BC 95  |
|----------------------|-------------|---------|--------|
| Turn on/off modem    | AT+CFUN     | AT+CFUN | AT+CFUN|
| Query signal strength| AT+CSQ      | AT+CSQ  | AT+CSQ |
| Network registration | AT+COPS     | AT+COPS | AT+COPS|
| Network status       | AT+CSI      | AT+CEREG| AT+CEREG|
| PDP activation       | AT+CGATT    | AT+MCGD| AT+CGATT|
| Setting APN          | AT+CGNAPN   | AT+EGACT| AT+EGACT|
| Ping remote address  | AT+CIPPING  | AT+PING | AT+NPING|
| Open UDP socket      | AT+CIPSTART | AT+ESOCON| AT+NSOCR|
| Send UDP message     | AT+CIPSEND  | AT+ESOEND| AT+NSOST|

![Figure 5: NB MIB decoding procedure.](image-url)
intrafrequency and the interfrequency case. All required parameters are received from the actual serving cell; there is no need to read SIB-NBs from other cells.

Among all cells fulfilling the S-criteria, the UE ranks the cells with respect to the power excess over another threshold. A hysteresis is added in this process to prevent too frequent cell reselection, and also, a cell-specific offset may be applied for the intrafrequency case. Contrary to LTE, there are no priorities for the different frequencies.

The UE finally selects the highest ranked cell which is suitable, i.e., from which it may receive normal service. When the UE leaves RRC-connected state, it does not necessarily select the same carrier to find a cell to camp on. The RRCConnectionRelease message may indicate the frequency on which the UE first tries to find a suitable cell. Only if the UE does not find a suitable cell on this frequency it may also try to find one on different frequencies.

4.4. MAC and RLC Throughput. After paging (or if the UE is already connected), data reception can begin. DCI format N1 indicates resource allocation, the number of subframes spanned by DL transmission, the number of repetitions, and whether an ACK is expected.

In scenarios involving the transfer of uplink data, once resources have been granted after receiving msg4, the UE begins transmitting its payload on NPUSCH using HARQ. ACK/NACK for HARQ is carried within the UL using the New Data Indicator (NDI) bit to distinguish between a request for a new transmission and a request for retransmission of the previous packet. In the case of failure, the eNB sends another UL grant in which the NDI bit is used as a NACK, and the UL grant informs the UE about the resources assigned for retransmission.

Figure 9 illustrates the RLC rate calculation of the typical data transport activity. In our dataset, the average RLC/MAC throughput of uploading data is 11.14 kbps and 11.72 kbps, while in receiving data, the average RLC/MAC throughput is 16.96 kbps and 17.31 kbps.

We also calculate the correlations between the resulting MAC/RLC throughput with DCI counts, MCS, and TBS in...
Figure 10 and find that those features are all highly correlated with each other.

In conclusion, we noticed that operators vary in their configuration parameters with the result that UEs differ in performance. An enormous number of parameters affect the RACH results, delay range, and data transfer rates; therefore, a new framework is required for the analysis of NB performance.

5. Performance Analysis and Modeling

The sheer volume and dimensionality of data make it almost impossible to have human experts perform troubleshooting using conventional rule-based systems. In this section, we perform modeling and analysis on NB performance using the state-of-art machine learning techniques.

5.1. Metrics Concerned. The proposed system was applied to a metropolitan NB-IoT system for data trace collection, testing, and verifying thousands of individual sites over a period of six months. To the best of our knowledge, this is the largest NB-IoT database ever explored. Our primary objective was to identify the features responsible for low throughput and long delay.

A number of issues are a concern for every operator. The term delay refers to a number of signaling procedures (e.g., aging delay and RRC connection delay) as well as ping delays associated with user activity. Data throughput also requires optimization by operators. Many mechanisms and algorithms, such as adaptive coded modulation, have been developed to improve data transfer rates by enhancing spectral efficiency in wireless channels and access control in cases of congestion.

Any given network can be tasked with dozens of procedures, each of which can have many possible metadata fields. As a result, the measurement data often contains hundreds of fields. Our first objective is to elucidate the relationships between the various network factors and target metrics, with the end goal of building models that could be used to improve performance through the tuning of network parameters.

5.2. Feature Engineering. In NB-IoT, the random access (RACH) procedure is contention based and begins with the transmission of a preamble. After obtaining a response from the eNB, a scheduled message, msg3, is transmitted to begin...
NPDCCH for RAR

SIB2

UE

eNB

Msg1: Preamble

NPDCCH for RAR

Msg2: RAR (RACH Response)

Msg3

NPDCCH for Msg4

Msg4

HARQ ACK

NPDCCH (for RRC Conn Complete)

RRC Conn Complete

Figure 11: RACH procedures and key features.

Figure 10: Feature correlation map during downlink transferring.
Finally, the contention resolution message is transmitted to the UE indicating the successful completion of the RACH procedure.

The RACH procedure illustrated in Figure 11 includes the following:

1. UE sends a RACH preamble carrying RA-RNTI and eNB to decode the preamble and obtain RA-RNTI
2. eNB sends a RACH response using RA-RNTI, which is calculated from the preamble resource (time, frequency allocation). The UE decodes the RACH response to obtain an RB assignment and MCS configuration for use in configuring itself to receive the "RRC connection request"
3. The UE sends an RRC connection request using the C-RNTI obtained from the RACH response
4. The UE receives an RRC connection setup using the C-RNTI obtained from the RACH response. The RRC connection setup message carries C-RNTI.

From this point, the UE and network exchange messages with C-RNTI.

In NB-IoT, choices pertaining to coverage level depend on channel conditions. Extreme coverage levels correspond to low received power, whereas normal coverage corresponds to higher received power. The selection of coverage class determines the transmission parameters, including the number of repetitions. Deploying systems in this manner makes it possible to serve UEs under a range of coverage conditions, as characterized by path loss (MCS). Except the features showed in sib2 items and the coverage level we discussed, there are still many features that tend to influence the RACH delay:

(i) RSRP: reference signal receiving power
(ii) SINR: signal to interference plus noise ratio
(iii) eDRX, DRX cycle periodicity: the DRX cycle periodicity affects the time for DL reachability
(iv) Power-saving mode idle timer: define the number of occasions for DL reachability
(v) Deployment mode: in-band or standalone
(vi) System information: MIB periodicity, SIB1 periodicity
(vii) NPUSCH transmission: payload size subcarrier spacing (3.75 kHz, 15 kHz)
(viii) NPRACH occurrence: NPRACH periodicity
(ix) Preamble transmission: preamble format 0 or 1, number of repetitions
(x) RA backoff configuration
(xi) NPDCCH occasion periodicity, number of repetitions
(xii) RAR reception, packet scheduling NPDCCH occasion periodicity RAR window size
(xiii) NPDCCH occasion periodicity contention resolution window size
(xiv) NPDSCH/NPDCCH transmission, payload size number of repetitions

5.3. Machine Learning-Based Prediction. A number of machine learning algorithms have been used to predict network performance:

(i) C4.5-based decision tree model

C4.5 builds decision trees based on the concept of information entropy. The training data is a set of preclassified samples. At each node of the tree, C4.5 selects the data attribute that most effectively splits its set of samples into subsets enriched in one class or the other. The splitting criterion is the normalized information gain (difference in entropy). The attribute with the highest normalized information gain is selected to direct the decisions. The C4.5 algorithm then recurs in the smaller sublists.

(ii) Random forest model

A random forest is a metaestimator that fits a number of decision tree classifiers on various subsamples of the dataset and uses averaging to improve predictive accuracy and control overfitting. The subsample size is always the same as the original input sample size but the samples are drawn with a replacement when bootstrap = true (default).

(iii) Support vector machine (SVM)

SVMs are supervised learning models that operate with learning algorithms to analyze the data used for classification and regression analysis. Given a set of training examples (each of which is marked as belonging to one or the other of two categories), an SVM training algorithm builds a model that enables the assignment of new examples to one category or the other. In other words, SVM is a nonprobabilistic binary linear classifier. An SVM model represents the examples as points mapped within a space in a manner intended to ensure that the examples of the two categories are divided by a clear gap (i.e., as wide as possible). New examples are then mapped into the same space, thereby making it possible to predict the category to which they belong based on the side of the gap on which they fall.
We model low-throughput and long-delay problems into classification problems of machine learning. We set the threshold for good/bad performance according to the specification of operators, like in throughput scenario; below 10 kbps is regarded as poor performance and in attach delay where the threshold is to be 5 s.

We evaluate the data-driven prediction models by testing with the 10-fold cross-validation approach, using four machine learning techniques from the Scikit-learn library. The four techniques perform similarly in terms of different metrics (accuracy, precision, recall, and F-score). The overall average accuracy of prediction is more than 87%, which signifies that we will be able to classify and predict the performance. Random forest performs best in terms of all four metrics and also shows comparably less variation. On the other hand, the decision tree model performs best in terms of time without compromising too much on accuracy. In comparison, SVM is the slowest.

We further model the ping delay and MAC throughput problem to the classification learning problem and still use those machine learning techniques; all the results are shown in Table 3.

### 5.5. Scope and Limitations

Our machine learning framework is highly generalizable; however, it is still limited in a number of aspects. Cell reselection required system information SIB4 (intrafrequency of neighbouring cell) and SIB5 (interfrequency of neighbouring cell); however, we found that the operator had not configured those SIBs. The NB-IoT network also failed to open a congestion control or access barring algorithm. We have to assume that the NB-IoT network was in the process of being deployed and tested prior to large-scale commercial implementation. As a result, we were unable to use machine learning to identify the mobility factors or determine what factors are important in cases of congestion.

In this study, we adopted an engineering approach to the identification of key features underlying the performance of NB networks. Experiment results revealed that wireless signal qualities had a profound influence on the perceived

---

### Table 3: Prediction results using different ML methods.

| Problem     | ML technique | Accuracy | Recall | Precision | F-score |
|-------------|--------------|----------|--------|-----------|---------|
| RACH delay  | C4.5         | 0.81     | 0.81   | 0.81      | 0.81    |
|             | Random forest| 0.93     | 0.93   | 0.93      | 0.93    |
|             | SVM          | 0.90     | 0.91   | 0.90      | 0.89    |
|             | Logistical   | 0.84     | 0.85   | 0.85      | 0.84    |
|             | C4.5         | 0.79     | 0.79   | 0.79      | 0.79    |
| Ping delay  | Random forest| 0.92     | 0.92   | 0.92      | 0.92    |
|             | SVM          | 0.90     | 0.91   | 0.90      | 0.89    |
|             | Logistical   | 0.83     | 0.84   | 0.84      | 0.84    |
|             | C4.5         | 0.80     | 0.80   | 0.80      | 0.80    |
| MAC UL Tput| Random forest| 0.93     | 0.93   | 0.93      | 0.93    |
|             | SVM          | 0.90     | 0.91   | 0.91      | 0.90    |
|             | Logistical   | 0.85     | 0.84   | 0.84      | 0.85    |

---

### Table 4: Energy footprints of the Simcom7000c NB modem.

| Procedure      | Latency | Current |
|----------------|---------|---------|
|                | 144 dB  | 164 dB  |
|                | 15 mA   | 15 mA   |
|                | 52 mA   | 52 mA   |
|                | 220 mA  | 320 mA  |
|                | 15 mA   | 15 mA   |
|                | 400 μA  | 400 μA  |
|                | 3.5 μA  | 3.5 μA  |

---

### Table 5: Energy prediction results using different ML methods.

| Problem   | ML technique | Accuracy | Recall | Precision | F-score |
|-----------|--------------|----------|--------|-----------|---------|
| Energy    | C4.5         | 0.87     | 0.88   | 0.86      | 0.88    |
|           | Random forest| 0.91     | 0.93   | 0.92      | 0.92    |
|           | SVM          | 0.91     | 0.90   | 0.90      | 0.88    |
|           | Logistical   | 0.86     | 0.85   | 0.87      | 0.86    |

---

(iv) Logistical classification

Linear regression is unsuitable for classification of this sort, due to the fact that it assigns too much weight to data located at a distance from the decision frontier. One alternative approach is to fit a sigmoid function or logistic function.

5.4. Feature Ranking and Prediction Accuracy. The value of these features can be determined by ranking them according to the information gained when they are used to predict performance. In this study, we adopted the method from Scikit-learn [20] to derive the information gain (IG) introduced by each of the attributes to the overall binary classification of a target metric as important or nonimportant. Table 2 lists the average information gains of the features. Our results show that the wireless signal quality features (e.g., SINR and RSRP, MCL) are the most important. We found that PRACH repetition counts are also relatively important features in terms of information gain.
data delays. Thus, accurate performance modeling depends on capturing these particular features.

6. Hidden Energy Consumption

One of the primary objectives of NB-IoT is to facilitate low-power operations. Thus, energy profiles are important for users and developers. In this study, we attached a Monsoon power meter [21] directly to the modem to measure the instantaneous power draw at a resolution of 5 kHz.

6.1. Energy Profiles. In this section, we present our observation in breaking down the NB-IoT UE power consumption under different signal conditions. A complete packet cycle consists of the same procedures among the three ECLs. MSG1, MSG3, ACK, packet data, and RRC connection release request are transmitted one by one, and the UE listens to the DL channels between these messages.

Table 4 presents the current and latency values associated with a device connecting and transmitting data. The data rates for worst-case coverage (+20 dB) are lower than those at the cell edge (0 dB), which results in higher latency of 7.6 seconds (under 144 dB) compared to 1.6 seconds (under 164 dB). The uplink data rate is the main cause of this degradation, wherein the per day energy consumption is 176.54 μAh (under 144 dB) and 712.26 μAh (under 164 dB).

The energy level for different procedures under different configurations is complicated. In reality, the configurations are vast in numbers, yet mostly are static; however, several key parameters vary in different operators.

One observed instance is a complaint about the same development board which differs in energy consumption which utilizes NB-IoT module to transfer data. The board plus modem consumed 47.5 mA in city A while 33 mA in city B, almost one half expected battery life gap.

By tracing the log, we find that the problem results from two parameters of the network in those two sites. In city A, the signals in NPDCCCH, named npdcch-NumRepetitions-r13 sets to r1, and the npdcch-StartSF- USS-r13 is V32. The npdcch-NumRepetitions field indicates a maximum repetitive transmission configuration value in a range of 1, 2, 4, 7, 16, 32, 64, 128, 256, 512, 1024, and 2048 for an NPDCCCH during a single period. Also, the npdcch-StartSF-US S-r13 field indicates the start subframe of a UE-specific search space and indicates the UE-specific search space which begins from a subframe indicated by a predetermined offset value (npdcch- Offset-US S) field based on the start subframe.

In city A, the NumRepetitions and the StartSF-US S are 8 and 2 separately. The period of NPDCCCH is about 32 ms, including 2 ms of repetition; when NPDCCCH is collided with PSS/SSS/SIB1/SIB2, it will prolong for a bit; thus, the remaining time is about 26 ms. According to the Qualcomm 9206 chipset, if the time interval is beyond 12 ms, it will shut down the RF. In opposite, city B needs to monitor 8 times of NPDCCCH within 16 ms; it will continue to keep the RF working, and the current will remain the same as normal state. We further validate the analysis through a Monsoon power monitor.

6.2. Energy Breakdown and Predicting Models. As discussed above, the nodes have significant energy imbalance, where the node under the same type of location profile drains energy at a distinct rate.

We see that ECL can be one of the major factors that cause such an imbalance. Since the NB-IoT network relies on the ECL to determine the Tx power, the UE drains different levels of energy under different ECLs. We observe that the UE selects ECL0 and ECL1 in a majority of the outdoor locations. However, for indoor parking, building corridors, and water meter deployment spots, ECL1 and ECL2 begin to dominate. Understanding the ECL selection mechanism is important; the UE decides its Tx power adaptively by itself based on the channel RSRP. Since UE consumes much more power in ECL2 than ECL0 and ECL1, we are interested in the factors that affect the selection of ECL. According to the 3GPP specification, the UE decides its ECL by comparing the latest RSRP with the two thresholds given by the eNodeB.

We model energy problems into classification problems of machine learning as in the previous section. We evaluate the data-driven prediction models by testing with the 10-fold cross-validation approach, using four machine learning techniques. The overall average accuracy of prediction is more than 88%, which is shown in Table 5.

7. Related Work

The importance of cellular network has attracted significant works ranging from traffic characterization to performance analysis to troubleshooting and optimization.

7.1. Cellular Measurement and Monitoring Tool. The authors of [22] deploy 3GTest to collect and analyze performance measurements from users of four major US cellular carriers. Reference [23] extends the tool then recognized as MobiPerf to analyze LTE users. In [24], the MobiPerf functions are distributed into a library for conducting network measurements.

MobileInsight [2] provides the principle under capturing baseband cellular message in an ordinary smartphone and opens its codebase. Snoopsniff [25] collects and analyzes mobile radio data to make you aware of your mobile network security and to warn you about threats like fake base stations (IMSI catchers), user tracking, and over-the-air updates. LTEye [26] provides a similar decoding capability on the air using USRP, which was further used by piStream [27] to decode the LTE resource information for optimizing the video performance. iCellular [28] exploits low-level cellular information at the device to improve multicarrier access. NBScope [29] illustrates the power profiles of typical NB-IoT networks and applications in the wild.

Operators put lots of energy to build large software to monitor and diagnose their networks. Many solutions are existing and serving as the basic toolchain for their regular tasks, such as Nokia [6, 7], Huawei [8], and Actix [9].

Our work differs from the above in that we implement a full-fledged NB prototype capable of conducting and collecting both user plane and control plane data. It gives the testing engineers, or any user, the freedom to know NB cellular characteristics and test its performance.
7.2. Cellular Performance Diagnosing and Machine Learning
The authors of [30] use both user and control plane data to diagnose real-world Femtocell and VoLTE (Voice over LTE) problems; the authors in [31] understand and diagnose real-world Femtocell performance problems. The authors in [32], developing a predictive model of quality of experience for internet video, presents a data-driven approach to model the metric interdependencies and their complex relationships to engagement and propose a systematic framework to identify and account for the confounding factors. The authors in [33] developed a machine learning framework for diagnosing the root cause of mobile video QoE issues for different video types (e.g., bitrate, duration) and contexts (e.g., wireless technology, encryption). The authors in [34] modeled web quality of experience on cellular networks using machine learning framework, illustrating radio network characteristics (such as signal strength, handovers, and load). The authors in [35] describe a device-centric machine learning approach and use the latency analysis on two popular mobile apps (web browsing and instant messaging).

Our work extends the analyzing task into NB-IoT network and device. By using classical machine learning methods, the delay/throughput problems can be located and well predicted out of hundreds of NB specific items.

8. Conclusions
In this study, we adopted the approach used in the MobileInsight project [2] in developing a portable hardware and software system for decoding NB-IoT network messages and experimenting by which to analyze network performance. We also built machine learning framework to diagnose delay/throughput efficiency under the conditions typically found in the IoT industry. In the future, NB-IoT modules and networks will be extended to include positioning methods and multicast services based on 3GPP specifications [36]. The efficacy of the proposed NBPilot system was established by applying it to a metropolitan NB-IoT network with over 2,000 NB sites for the collection and testing of data trace as well as the validation of a cellular station prior to going online. Our study showed that NB-IoT nodes yield significantly imbalanced energy consumption in the wild, due to poor network coverage level, long tail power profile, and excessive control message repetitions.

Data Availability
We release the data sample through https://github.com/Zhenxian-Hu/NBPilot. Requests for access to full data should be made to zhenxian.hu@qq.com.

Disclosure
An earlier version of this paper is presented in IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks (SECON) 2019.

Conflicts of Interest
The authors declare that they have no conflicts of interest.

Acknowledgments
This work is supported by the National Key R&D Program of China (2018YFB2101102), National Natural Science Foundation of China (U1736207, 62072306), and Program of Shanghai Academic Research Leader (20XD1402100).

References
[1] Iot commercial launcheshttps://www.gsma.com/iot/mobile-iot-commercial-launches/.
[2] Y. Li, C. Peng, Z. Yuan, J. Li, H. Deng, and T. Wang, “MobileInsight: extracting and analyzing cellular network information on smartphones,” in The 22nd ACM Annual International Conference on Mobile Computing and Networking (Mobimon’16), New York, USA, 2016.
[3] QxDM professional - Qualcomm extensible diagnostic monitorhttp://www.qualcomm.com/media/documents/tags/qxdm.
[4] Xcal-mobilehttp://www.accuver.com.
[5] MTK Catcherhttp://www.finetopix.com/showthread.php?40844-MTK-Catcher.
[6] Wireless network guardianhttps://networks.nokia.com/products/wireless-network-guardian.
[7] Network performance optimizerhttps://networks.nokia.com/products/9959-network-performance-optimizer.
[8] Huawei Nastarhttp://www.huawei.com/in/products/oss/mbbom-product/imanager-nastar/.
[9] Actix analyzerhttp://actix.com/analyzer/.
[10] 3GPP, Architecture enhancements to facilitate communications with packet data networks and applications, Technical Specification (TS) 23.682, 3rd Generation Partnership Project (3GPP), 2016. Version 13.8.0, 2016.
[11] Quetel LTE Linux USB driverhttps://www.quectel.com/UploadImage/Download/Quetel_WCDMA&_LTE_Linux_USB_Driver_User_Guide_V1.8.pdf.
[12] 3GPP, ETSI. Digital cellular telecommunications system (Phase 2+); AT command set for GSM Mobile Equipment (ME), Technical Specification (TS) 07.07, 3rd Generation Partnership Project (3GPP), 1998. Version 7.8.0, 1998.
[13] 3GPP, ETSI. Digital cellular telecommunications system (Phase 2+)(GSM); Universal Mobile Telecommunications System (UMTS); LTE; AT command set for User Equipment (UE), Technical Specification (TS) 27.007, 3rd Generation Partnership Project (3GPP), 2016. Version 13.6.0, 2016.
[14] Sim7000 at commands manualhttps://www.waveshare.com/w/upload/6/6a/SIM7000_Series_AT_Commands_Manual_V1.03.pdf.
[15] Mediatek mt2625https://www.mediatek.com/products/nbIot/mt2625.
[16] Quetel bc95 at commands manualhttps://www.quectel.com/UploadImage/Download/Quetel_BC95_AT_Commands_Manual_V1.9.9.pdf.
[17] 3GPP, Evolved Universal Terrestrial Radio Access (E-UTRA); Radio Resource Control (RRC); Protocol specification, Technical Specification (TS) 36.331, 3rd Generation Partnership Project (3GPP), 2016. Version 13.4.0, 2016.
[18] Asn 1 to c compilerhttps://github.com/vlm/asn1c.
3GPP, *Multiplexing and channel coding (release 13)*, Technical Specification (TS) 36.212, 3rd Generation Partnership Project (3GPP), 12 2016. Version 13.4.0, 2016.

Scikit-learn [http://scikit-learn.org/stable/index.html](http://scikit-learn.org/stable/index.html).

Monsoon power monitor [https://www.msoon.com/](https://www.msoon.com/).

J. Huang, Q. Xu, B. Tiwana, Z. M. Mao, M. Zhang, and P. Bahl, "Anatomizing application performance differences on smartphones," in *Proceedings of the 8th international conference on Mobile systems, applications, and services*, pp. 165–178, San Francisco, California, USA, 2010.

J. Huang, F. Qian, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck, "A close examination of performance and power characteristics of 4G LTE networks," in *Proceedings of the 10th international conference on Mobile systems, applications, and services*, pp. 225–238, Low Wood Bay Lake District, UK, 2012.

A. Nikravesh, H. Yao, S. Xu, D. Choffnes, and Z. M. Mao, "Mobilyzer: an open platform for controllable mobile network measurements," in *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*, pp. 389–404, Florence, Italy, 2015.

Snoopsnitch [https://opensource.srlabs.de/projects/snoopsnitch](https://opensource.srlabs.de/projects/snoopsnitch).

S. Kumar, E. Hamed, D. Katabi, and L. E. Li, "LTE radio analytics made easy and accessible," in *Proceedings of the 2014 ACM conference on SIGCOMM (SIGCOMM’14)*, pp. 211–222, New York, NY, USA, 2014.

X. Xie, X. Zhang, S. Kumar, and L. E. Li, "piStream: physical layer informed adaptive video streaming over LTE," in *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, pp. 413–425, Santa Clara, CA, USA, 2015.

Y. Li, H. Deng, C. Peng et al., "iCellular: device-customized cellular network access on commodity smartphones," in *13th USENIX Symposium on Networked Systems Design and Implementation (NSDI 16)*, pp. 643–656, 2016.

D. Yang, X. Zhang, X. Huang et al., "Understanding power consumption of NB-IoT in the wild: tool and large-scale measurement," in *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*, pp. 1–13, London, United Kingdom, 2020.

C. Peng, Y. Li, Z. Li, J. Zhao, and X. Jiaqi, "Understanding and diagnosing real-world femtocell performance problems," in *Computer Communications, IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on*, pp. 1–9, San Francisco, CA, USA, 2016.

Y. J. Jia, Q. A. Chen, Z. M. Mao et al., "Performance characterization and call reliability diagnosis support for voice over LTE," in *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, pp. 452–463, Paris, France, 2015.

A. Balachandran, V. Sekar, A. Akella, S. Seshan, I. Stoica, and H. Zhang, "Developing a predictive model of quality of experience for internet video," in *ACM SIGCOMM Computer Communication Review*, vol. 43, pp. 339–350, 2013.

G. Dimopoulos, I. Leontiadis, P. Barlet-Ros, K. Papagiannaki, and P. Steenkiste, "Identifying the root cause of video streaming issues on mobile devices," in *Proceedings of the 11th ACM Conference on Emerging Networking Experiments and Technologies*, p. 24, Heidelberg, Germany, 2015.

[34] A. Balachandran, V. Aggarwal, E. Halepovic et al., "Modeling web quality-of-experience on cellular networks," in *Proceedings of the 20th annual international conference on Mobile computing and networking*, pp. 213–224, Maui Hawaii, USA, 2014.

[35] Z. Yuan, Y. Li, C. Peng et al., "A machine learning based approach to mobile network analysis," in *2018 27th International Conference on Computer Communication and Networks (ICCCN)*, pp. 1–9, Hangzhou, China, 2018.

[36] 3GPP, *Study on scenarios and requirements for next generation access technologies*, Technical report (TR) 38.913, 3rd Generation Partnership Project (3GPP), 6 2018. Version 15.0.0, 2018.