2019 16th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)

NB-IoT Network Monitoring and Diagnosing

Zhenxian Hu, Guangtao Xue, Yi-Chao Chen, and Minglu Li

Shanghai Jiao Tong University, China
gt_xue@sjtu.edu.cn

***Abstract*—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 signalling messages transmitted by a Qualcomm NB modem. This system gives researchers and developers an unprecedented understanding of network behaviour as well as the ability to adjust them to their particular requirements. We employed the-state-of-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.**

I. 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, 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.

NB-IoT is an access technology defined in the 3rd Generation Partnership Project (3GPP) for massive Machine Type Communications (mMTC). According to [4], there were 83 Nb-IoT launches as of December 2018. NB-IoT includes a number of mMTC-oriented enhancements that are not found in mobile technologies, including (i) narrow-band transmission and the use of repeaters to reach devices in inaccessible locations, such as underground basements; (ii) the ability to differentiate the performance of user equipment (UE) according to coverage area, and the ability to tune parameters of the physical channels and network procedures; (iii) enhanced power saving mechanisms to improve the battery life; (iv) simplification of network procedures to reduce UE complexity. NB-IoT can be scaled down and/or 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 re-farming the GSM spectrum.

Unfortunately, cellular networks remain a blackbox to most users and developers. A lack of open access to fine-grained runtime network operations makes it exceedingly difficult for researchers and developers to understand and/or refine network behaviors. 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[10], XCAL[15], MTK Catcher[6], are then used to analyze cellular network messages and provide fine-grained information. Unfortunately, many such devices are out-dated and difficult to bring up to spec.

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 developed by Nokia[14][7], Huawei[3], and Actix[1], before undertaking the tedious task of manual checking each event.

In this study, we adopted the approach used in MobileInsight[32] in our development of a full-featured test bed referred to as NBPilot. 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-

978-1-7281-1207-7/19/$31.00 ©2019 IEEE

IoT 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.

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:

- 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 non-negligible influence on performance.
- The proposed system was applied to a metropolitan NB-IoT network for data trace collection, testing, and verifying thousands of individual site. To the best of our knowledge, this is the largest NB-IoT database ever explored.
- 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.

The rest of the paper is organized as follows. Section III then presents the design of the NBPilot system and Section IV presents our evaluation of its typical performance. Section V supplements performance evaluation with examples of machine learning models. Section VI outlines related work. Section VII concludes.

II. 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.

A. 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.

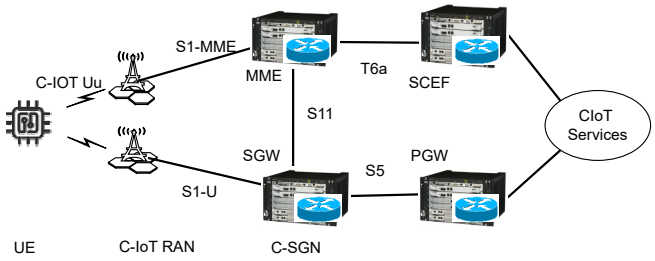


Fig. 1: NB-IoT Network Structure

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[17].

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.

B. 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

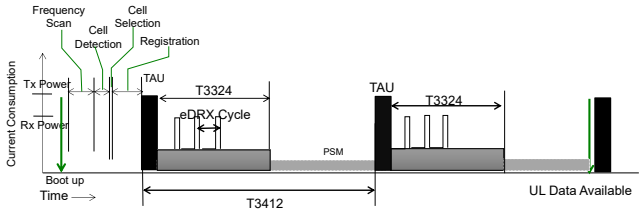


Fig. 2: Life cycle and related power levels of a NB-IoT activities

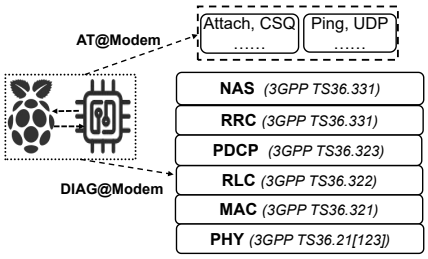


Fig. 3: NBPilot System Structure: Raspberry Pi and Qualcomm NB modem. The specific NB cellular stack(3GPP R13 version) is referring to [23][21][24][20][25][22][19]

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.

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 2.56s (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: $T3324$ is the duration of the idle phase (up to 3 hours); $T3412$ represents the TAU periodicity and thus determines the duration of the PSM cycle (up to 413 days).

III. The Design of NBPilot

We were inspired by the MobileInsight project[32] to build a portable hardware and software system for decoding NB-IoT network messages and conducting experiments. The resulting NBPilot system is used to dissect NB-IoT network procedures and parameters.

A. System structure

Figure 3 presents a schematic illustration showing the overall NBPilot 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 include 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 non-radio 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[9] 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.

B. 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)[16][18]. 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 and/or serial interfaces when connected via USB to receive those AT commands.

Table I lists a selection of the AT commands used in this work and comparison of AT command series provided by Qualcomm(Simcom7000c[12]), MTK(ME3616[5]), Huawei(BC95[8]). We observe that different chipset vendors share few common items but differentiate even for the same functions.

TABLE I: Selection of the AT commands used in this work of different chipset vendors.

	Simcom7000c	ME3616	BC95
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+CPSI	AT+CEREG	AT+CEREG
PDP activation	AT+CGATT	AT*MCGDEFCONF	AT+CGATT
set APN	AT+CGNAPN	AT+EGACT	AT+CGATT
ping remote address	AT+CIPPING	AT+PING	AT+NPING
open UDP socket	AT+CIPSTART	AT+ESOCON	AT+NSOCR
send UDP message	AT+CIPSEND	AT+ESOSEND	AT+NSOST

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).

C. Reverse Engineering an NB-IoT Stack

The first issue in reverse engineering an NB-IoT Stack is the fact 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. Figure 4 illustrates the decoding procedure for NB Master Information Block (MIB) message.

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[32]. Some formats can be extracted automatically including RRC and NAS.

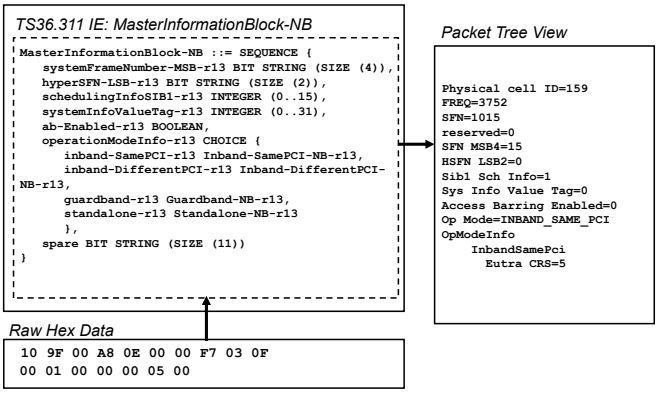


Fig. 4: NB Stack Decoding Examples

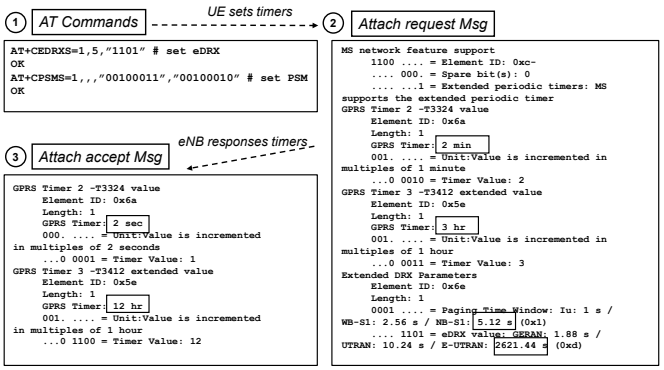


Fig. 5: Setting and negotiating PSM and eDRX between UE and network

For instance, the NB RRC standards[19] provide abstract message notations under ASN.1, which can be readily compiled into message decoders[2]. Other messages must be manually converted into machine-readable formats by comparing with Qualcomm QXDM.

D. Interactivity between AT and diag

Figure 5 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: $T3324$(Active time 2 min), $T3412$(TAU 3 h), eDRX:$(2621.44s(0xd))$, PTW(5.12s).
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: $T3412$(12h) and $T3314$(2s).

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.

IV. 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 HiSillicon(Huawei) and MTK decoding tools. We then sought to provide a comprehensive illustration of the NB-IoT typical procedures, covering cell information, random access procedures, uploading and downloading characteristics under different operators.

A. Decoding capabilities coverage

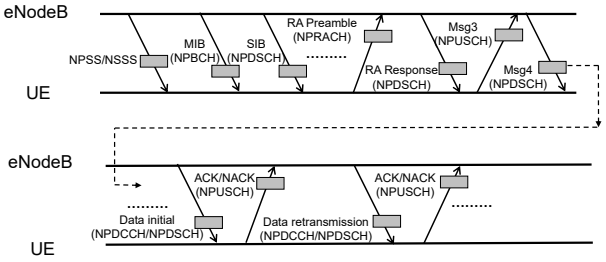
NBPilot decodes all signaling messages on RRC-r13(for NB), NAS(MM and SM) and partially supports PHY, MAC, RLC and PDCP messages. Huawei provides a tool called LogViewer[8] to examine the RRC messages. It first finds the corresponding decoder .xml file, and then uses the message ID to build a dictionary, the key of which is the message decoder node in the XML tree. MTK provides the similar tool called Genie Logging Tool[5] to observe RRC messages.

We manually check with QXDM,LogViewer and Genie tool. We find RRC and NAS formats are 100% the same among three chipsets, they have been endured the conformance testing before shipping into the market. NBPilot currently supports nearly 40 types(38) of messages. In a typical user study of data uploading activity, the top 5 signaling messages 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%), LTE NB1 ML1 GM TX Report(11.2%).

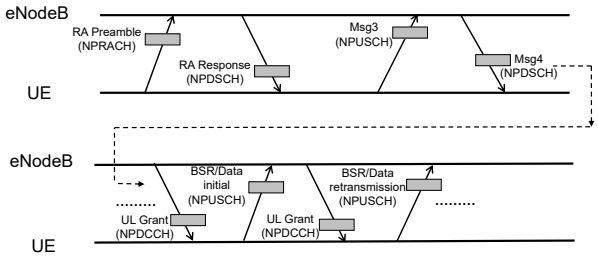
B. uploading and downloading activities

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 6(a) and Figure 6(b) present schematic diagrams illustrating the downloading of data from eNodeB to UE and an opposite data uploading procedure.

System information is used to broadcast information that is valid for all UEs within a cell. However, broadcasting imposes restriction on each EU in terms of computational resources and battery consumption, and should therefore be kept to a minimum in terms of broadcast size and frequency. This can be achieved by defining a set of System Information Blocks (SIBs).



(a) DL



(b) UL

Fig. 6: Data transfer procedures in downlink and uplink

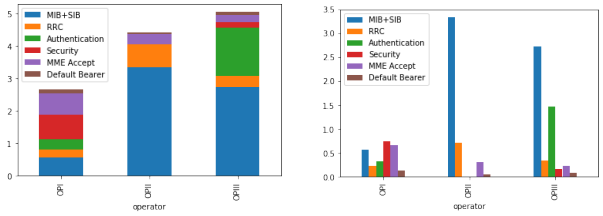


Fig. 7: Time consumption in each procedure when attaching NB network, each stack bar stands for one operator

Valid versions of MIB-NB, SIB1-NB, and SIB2-NB are always required for an UE, and any other SIBs required for additional operations must also be valid. For instance, if access barring is indicated in MIB-NB, then the UE requires a valid version of SIB14-NB.

We tested NBPilot under the three operators deployed in the city. Hereafter, the three operators are referred to as OPI, OpII, OpIII. Figure 7and Table II illustrate the time consumption in each procedure when attaching a particular NB network.

The time elapse for each specific procedure is calculated

We elaborate only a few differences of the SIB2 configurations. OPIII is using pp16/32/64 three contention resolution timer while OPI and OPII stick to pp16; as nprach periodicity, OPIII sets this parameter two times larger than OPI and OPII.

C. 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.

TABLE II: Time consumptions(in ms) in a RRC attach procedure,including system information reading, RRC connection setup, UE authentication, NAS security setup, MME accept, and EPS bearer establishment.

Operator	Sys	RRC	Auth	Sec	MME	Bearer
OPI	0.568	0.24	0.324	0.748	0.66	0.13
OPII	3.331	0.708	0	0	0.317	0.05
OPIII	2.729	0.348	1.472	0.175	0.233	0.08

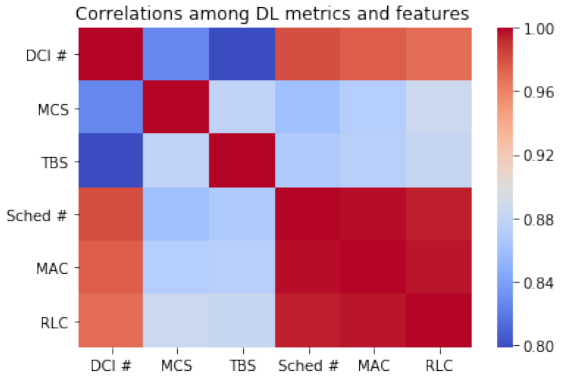


Fig. 8: feature correlations in downlink transport situation

In scenarios involving the transfer of uplink data, once resources have been granted after recieving 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.

In our dataset, the average RLC/MAC throughput of uploading data is $11.14kbps$ and $11.72kbps$, while in receiving data, the average RLC/MAC throughput is $16.96kbps$ and $17.31kbps$.

We also calculate the correlations between the resulting MAC/RLC throughput with DCI counts,MCS,and TBS in Figure 8, 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.

V. Performance Analysis and Modelling

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.

A. Metrics concerned

The proposed system was applied to a metropolitan NB-IoT system for data trace collection, testing, and verifying thousands of individual site 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,long signaling and data delays.

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 meta-data 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.

B. Feature Engineering

In NB-IoT, the Random Access procedure(RACH) 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 the contention resolution process. Finally, the contention resolution message is transmitted to the UE indicating the successful completion of the RACH procedure.

The RACH procedure illustrated in Figure 9 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.

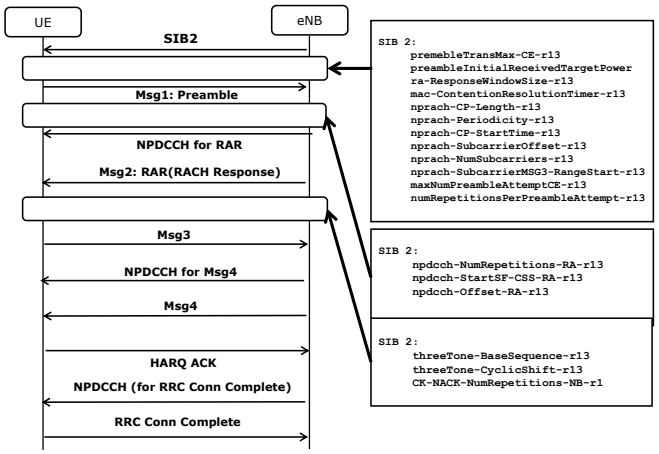


Fig. 9: Procedures of RACH

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 high 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 tend to influence the RACH delay:

- RSRP: Reference Signal Receiving Power
- SINR: Signal to Interference plus Noise Ratio
- eDRX, DRX cycle periodicity: The DRX cycle periodicity affects the time for DL reachability
- Power Saving Mode Idle timer: define the number of occasions for DL reachability
- Deployment Mode: inband or standalone
- System Information: MIB periodicity, SIB1 periodicity, SIB2 periodicity
- NPUSCH Transmission: Payload Size Subcarrier Spacing (3.75kHz, 15kHz)
- NPRACH occurrence: NPRACH Periodicity
- Preamble Transmission: Preamble Format 0 or 1, Number of Repetitions
- RA Backoff configuration
- NPDCCH Occasion periodicity, Number of repetitions
- RAR Reception,Packet Scheduling NPDCCH Occasion periodicity RAR Window Size
- NPDCCH Occasion periodicity Contention Resolution Window Size
- NPDSCH/NPDCCH Transmission, Payload Size Number of Repetitions

C. 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 pre-classified 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 meta-estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve predictive accuracy and control over-fitting. The sub-sample 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 non- probabilistic 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.

(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.

D. 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 *mutual_info_classif* method from Scikit-learn[11] to derive the information gain (IG) introduced by each of the attributes to the overall binary classification of a target metric as important or non-important. Table IV 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.

TABLE III: Top 5 Features in RACH delay classification according to average IG.

Feature	ID	Average IG
SINR	1	0.346
RSRP	2	0.275
MCL	3	0.201
NPRACH repetitions	4	0.179
NPDCCH repetitions	5	0.165

TABLE IV: RACH delay, Ping delay, MAC throughput prediction using machine learning techniques.

Problem	ML Technique	Accuracy	Recall	Precision	F-Score
RACH delay	C4.5 Decision Tree	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 classification	0.84	0.85	0.85	0.84
Ping delay	C4.5 Decision Tree	0.79	0.79	0.79	0.79
	Random forest	0.92	0.92	0.92	0.92
	SVM	0.90	0.91	0.90	0.89
	Logistical classification	0.83	0.84	0.84	0.84
MAC UL rate	C4.5 Decision Tree	0.80	0.80	0.80	0.80
	Random forest	0.93	0.93	0.93	0.93
	SVM	0.90	0.91	0.91	0.90
	Logistical classification	0.85	0.84	0.84	0.85

We found that PRACH repetition counts are also relatively important features in terms of information gain.

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 10kbps is regarded as poor performance and in attach delay where the threshold is to be 5s.

We evaluate the data-driven prediction models by testing with the 10-fold cross validation approach, using four machine learning techniques from Scikit-learn library. The four techniques perform similarly in terms of different metrics(accuracy, precision, recall and F-Score). Overall average accuracy of prediction is more than 87%. 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 showed in Table IV.

E. 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 data delays. Thus, accurate performance modeling depends on capturing these particular features.

VI. Related work

Industry players and researchers develop lots of tools to dig into cellular networks. Many commercial solutions for baseband chip signal porting and parsing, such as Qualcomm QXDM[10], MediaTek Catcher[6], XCAL[15].

MobileInsight[32] provides the principle under capturing baseband cellular message in an ordinary smartphone and opens its codebase. Snoopsnitch[13] 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[31] provide a similar decoding capability on the air using USRP, which was further used by piStream[34] to decode the LTE resources information for optimizing the video performance.

Operators put lots of energy to build large software to monitor and diagnosing their networks. Many solutions are existed and serving as the basic toolchain for their regular tasks, such as Nokia[14][7], Huawei[3], and Actix[1].

Our work differs from the above in that we implement a full-fledge 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.

The authors of [33] [30] use both user and control plane data to diagnose real-world Femtocell and VoLTE(Voice over LTE) problems, Understanding and diagnosing real-world Femtocell performance problems seperately. Authors in [28] 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.[29] 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). [27] modeled web quality-of-experience on cellular networks using machine learning framework, illustrating radio network characteristics (such as signal strength, handovers, load, etc). [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.

VII. Conclusions

In this study, we adopted the approach used in the MobileInsight project[32] in developing a portable hardware and software system for decoding NB-IOT network messages and conducting experiments 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[26]. 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.

Acknowledgements

This work was supported by the Joint Key Project of the National NSFC (No. U1736207), NSFC (No. 61572324), National Key R&D Program of China (2017YFC0803700), Shanghai Talent Development Fund.

References

[1] Actix analyzer. http://actix.com/analyzer/.
[2] Asn.1 to c compiler. http://https://github.com/vlm/asn1c.
[3] Huawei nastar. http://www.huawei.com/in/products/oss/mbb-om-product/imanager-nastar/.
[4] Iot commercial launches. https://www.gsma.com/iot/mobile-iot-commercial-launches/.
[5] Mediatek mt2625. https://www.mediatek.com/products/nbIot/mt2625.
[6] Mtk catcher. http://www.finetopix.com/showthread.php/40844-MTK-Catcher.
[7] Network performance optimizer. https://networks.nokia.com/products/9959-network-performance-optimizer.
[8] Queltel bc95 at commands manual. https://www.quectel.com/UploadImage/Downlad/Quectel_BC95_AT_Commands_Manual_V1.9.pdf.
[9] Queltel lte linux usb driver. https://www.quectel.com/UploadImage/Downlad/Quectel_WCDMA&LTE_Linux_USB_Driver_User_Guide_V1.8.pdf.
[10] Qxdm professional - qualcomm extensible diagnostic monitor. http://www.qualcomm.com/media/documents/tags/qxdm.
[11] Scikit-learn. http://scikit-learn.org/stable/index.html.
[12] Sim7000 at commands manual. https://www.waveshare.com/w/upload/6/6a/SIM7000_Series_AT_Command_Manual_V1.03.pdf.
[13] Snoopsnitch. https://opensource.srlabs.de/projects/snoopsnitch.
[14] Wireless network guardian. https://networks.nokia.com/products/wireless-network-guardian.
[15] Xcal-mobile. http://www.accuver.com.
[16] 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.
[17] 3GPP. Architectecture enhancements to facilitate communications with packet data networks and applications. Technical Specification (TS) 23.682, 3rd Generation Partnership Project (3GPP), 12 2016. Version 13.8.0.
[18] 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), 12 2016. Version 13.6.0.
[19] 3GPP. Evolved Universal Terrestrial Radio Access (E-UTRA); Radio Resource Control (RRC); Protocol specification. Technical Specification (TS) 36.331, 3rd Generation Partnership Project (3GPP), 12 2016. Version 13.4.0.
[20] 3GPP. Medium Access Control(MAC) protocol specification. Technical Specification (TS) 36.321, 3rd Generation Partnership Project (3GPP), 12 2016. Version 13.4.0.
[21] 3GPP. Multiplexing and channel coding(Release 13). Technical Specification (TS) 36.212, 3rd Generation Partnership Project (3GPP), 12 2016. Version 13.4.0.
[22] 3GPP. Packet Data protocol(PDCP) specification. Technical Specification (TS) 36.322, 3rd Generation Partnership Project (3GPP), 12 2016. Version 13.4.0.
[23] 3GPP. Physical channels and modulation(Release 13). Technical Specification (TS) 36.211, 3rd Generation Partnership Project (3GPP), 12 2016. Version 13.4.0.
[24] 3GPP. Physical layer procedures(Release 13). Technical Specification (TS) 36.213, 3rd Generation Partnership Project (3GPP), 12 2016. Version 13.4.0.
[25] 3GPP. Radio Link Control(RLC) protocol specification. Technical Specification (TS) 36.322, 3rd Generation Partnership Project (3GPP), 6 2016. Version 13.2.0.
[26] 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.
[27] A. Balachandran, V. Aggarwal, E. Halepovic, J. Pang, S. Seshan, S. Venkataraman, and H. Yan. Modeling web quality-of-experience on cellular networks. In *Proceedings of the 20th annual international conference on Mobile computing and networking*, pages 213–224. ACM, 2014.
[28] 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*, volume 43, pages 339–350. ACM, 2013.
[29] 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*, page 24. ACM, 2015.
[30] Y. J. Jia, Q. A. Chen, Z. M. Mao, J. Hui, K. Sontinei, A. Yoon, S. Kwong, and K. Lau. Performance characterization and call reliability diagnosis support for voice over lte. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, pages 452–463. ACM, 2015.
[31] 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). ACM, New York, NY, USA*, pages 211–222. Association for Computing Machinery (ACM), 2014.
[32] 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 (Mobicom'16)*, New York, USA, Oct. 2016.
[33] C. Peng, Y. Li, Z. Li, J. Zhao, and J. Xu. Understanding and diagnosing real-world femtocell performance problems. In *Computer Communications, IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on*, pages 1–9. IEEE, 2016.
[34] 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*, pages 413–425. ACM, 2015.
[35] Z. Yuan, Y. Li, C. Peng, S. Lu, H. Deng, Z. Tan, and T. Raza. A machine learning based approach to mobile network analysis. In *2018 27th International Conference on Computer Communication and Networks (ICCCN)*, pages 1–9. IEEE, 2018.