LinkForecast: Cellular Link Bandwidth Prediction in LTE Networks

Chaoqun Yue, *Student Member, IEEE*, Ruofan Jin, *Member, IEEE*, Kyoungwon Suh, *Member, IEEE*, Yanyuan Qin, *Student Member, IEEE*, Bing Wang, *Member, IEEE*, and Wei Wei, *Member, IEEE*

Abstract—Accurate cellular link bandwidth prediction can benefit upper-layer protocols significantly. In this paper, we investigate how to predict cellular link bandwidth in LTE networks. We first conduct an extensive measurement study in two major commercial LTE networks in the US, and identify five types of lower-layer information that are correlated with cellular link bandwidth. We then develop a machine learning based prediction framework, *LinkForecast*, that identifies the most important features (from both upper and lower layers) and uses these features to predict link bandwidth in real time. Our evaluation shows that LinkForecast is lightweight and the prediction is highly accurate: At the time granularity of one second, the average prediction error is in the range of 3.9 to 17.0 percent for all the scenarios we explore. We further investigate the prediction performance when using lower-layer features obtained through standard APIs provided by the operating system, instead of specialized tools. Our results show that, while the features thus obtained have lower fidelity compared to those from specialized tools, they lead to similar prediction accuracy, indicating that our approach can be easily used over commercial off-the-shelf mobile devices.

Index Terms—Cellular networks, cellular link bandwidth prediction, network measurement, machine learning

1 Introduction

Cellular infrastructures have been evolving at a fast speed. The current 4G Long Term Evolution (LTE) cellular networks provide significantly higher data rate and lower network latency than earlier generation systems. The advances provided by LTE, however, may not be fully utilized by upper-layer protocols due to the rapidly varying cellular link conditions. As an example, a recent large-scale study of LTE networks [8] shows that, for 71.3 percent of the large TCP flows, the bandwidth utilization rate is below 50 percent. The low link bandwidth utilization is partly because many existing transport protocols are designed for traditional networks and cannot readily adapt to the rapidly changing link conditions in cellular networks.

The performance of upper-layer protocols can be improved significantly when accurate link bandwidth prediction is available (see details in Section 2.1). Existing studies either use upper-layer information (e.g., historical throughput, delay, loss rate, inter-packet arrival times) or lower-layer (PHY/MAC) information to predict cellular link bandwidth. The approaches using upper-layer information [26], [30] may lead to inaccurate prediction under highly dynamic conditions. Complementary to upper-layer information, a rich set of lower-layer information is monitored by the user device and the base station, and is exchanged between these two entities in real time (used routinely for scheduling purposes). Lower-layer information provides important insights on the cellular link bandwidth. Existing studies that use lower-layer information [4], [15], [17], however, are conducted in limited settings and only consider one or two types of information (e.g., signal strength, signal noise ratio).

In this paper, we start with the observation that base stations in LTE networks typically use (variants of) proportional fair scheduling, which allocates bandwidth to user devices based on past throughput and link conditions [1], [7], [18]. This resource allocation mechanism indicates that upper-layer information (past throughput) and lower-layer information are both useful for cellular link bandwidth prediction. We therefore ask the following questions: what set of lower-layer information is most useful for cellular link bandwidth prediction? How to combine upper- and lower-layer information for prediction purposes? How much more accurate the prediction can be compared to using upper-layer information alone?

To answer the above questions, we first conduct an extensive measurement study to identify a set of lower-layer information that is correlated with cellular link bandwidth. We then develop a lightweight machine learning framework that determines the most important features, and uses these features to predict cellular link bandwidth in real time. The above measurement study uses Qualcomm eXtensible Diagnostic Monitor (QxDM) [19], which allows us to collect a wide variety of lower-layer information at fine granularity. For practical cellular link bandwidth prediction, we however cannot rely on QxDM since it is a

- *C. Yue, R. Jin, Y. Qin, B. Wang, and W. Wei are with the Computer Science & Engineering Department, University of Connecticut, Storrs, CT 06269. E-mail: {chaoqun.yue, yanyuan.qin, wei.wei}@uconn.edu, ruofan.jin@gmail.com, bing@engr.uconn.edu.*
- *K. Suh is with the School of Information Technology, Illinois State University, Normal, IL 61761. E-mail: kwsuh@ilstu.edu.*

Manuscript received 14 Feb. 2017; revised 14 Aug. 2017; accepted 18 Sept. 2017. Date of publication 26 Sept. 2017; date of current version 1 June 2018. (Corresponding author: Chaoqun Yue.)
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TMC.2017.2756937

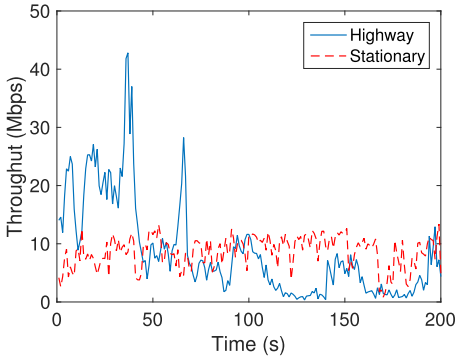


Fig. 1. Illustration of rapidly changing link bandwidth in LTE networks.

specialized tool, not readily available to end systems. Therefore, we further investigate the prediction accuracy when lower-layer information is collected directly at mobile phones, through APIs provided by the operating system.

Our main contributions are as follows.

- We have conducted an extensive measurement study over two commercial LTE networks and investigated a large set of lower-layer information obtained using QxDM. Through the measurement, we identify five types of lower-layer information that are correlated with cellular link bandwidth. The degree of correlation varies in different scenarios; none of them has direct functional relationship with link bandwidth. Our study significantly expands the scope of existing studies [4], [15], [17] that only focus on one or two parameters.
- We develop a machine learning based prediction framework, *LinkForecast*, to predict link bandwidth in real time. Using this framework, we quantify the importance of various upper- and lower-layer features and make the following findings. First, while historical throughput is the most important feature, using it alone can lead to much inferior performance compared to that using all the features, indicating the importance of combining upper- and lower-layer information for link bandwidth prediction. Second, using the four most important features achieves almost the same performance as when using all the features, and the prediction is highly accurate: at the time granularity of one second, the average prediction error is in the range of 3.9 to 17.0 percent for all the scenarios we explore. In addition, the prediction technique is lightweight and is insensitive to training data, making it easy to apply in practice.
- We further investigate link bandwidth prediction when the lower-layer features are obtained through the standard APIs provided by Android operating system (instead of QxDM). While the features thus obtained have lower fidelity than those from QxDM, they lead to similar prediction accuracy, indicating that our approach can be easily used over commercial off-the-shelf mobile devices.

The rest of the paper is organized as follows. Section 2 covers the background and presents our high-level approach. Section 3 investigates the correlation between various lower-layer information and cellular link bandwidth. Section 4 presents the design and performance of LinkForecast. Section 5 explores the performance of LinkForecast when the lower-layer features are obtained through the standard APIs provided by the operating system. Section 6 summarizes related work. Finally, Section 7 concludes the paper and presents future directions.

2 Background and High-Level Approach

In this section, we first describe the challenges and benefits of link bandwidth prediction in cellular networks. After that, we present our high-level approach.

2.1 Motivation

Link bandwidth in cellular networks is highly dynamic. Fig. 1 shows two examples using measurements collected from commercial LTE networks (see details in Section 3.2). Specifically, it plots per-second link bandwidth in two scenarios: one is a stationary scenario, and the other is a highway driving scenario. We observe that even in the stationary scenario, the link bandwidth changes rapidly over time. In the highway driving scenario, the dynamics are even more dramatic. The rapidly changing link bandwidth has been widely observed. It is due to the nature of wireless communication as well as the radio resource scheduling at the base stations in cellular networks (as illustrated in [32]).

The performance of upper-layer protocols and applications can be severely affected by the highly dynamic link bandwidth in cellular networks. Accurate cellular link bandwidth prediction can help these protocols and applications to proactively react to changing network conditions, leading to better performance. For instance, it can be used in realtime applications (e.g., VoIP, video conferencing and online gaming) to avoid large latencies [30]. It can also be used to design transport protocols that determine the sending rate in real time to achieve both high throughput and low latencies simultaneously [15], [26]. In addition, it can be used to schedule background data transfers [4] and optimize the performance of adaptive video streaming [33].

Accurate link bandwidth prediction in cellular networks is a challenging task [31], [32]. Several studies have developed techniques for link bandwidth prediction, using either upper- or lower-layer information (see details in Section 6). Our study uses a different approach: we leverage both past throughput and lower-layer information based on the radio resource scheduling algorithms used in cellular networks. This approach leads to a lightweight technique that provides accurate prediction and is readily usable on commercial off-the-shelf mobile devices.

2.2 High-Level Approach

In LTE networks, a user equipment (UE) is served by a base station, called evolved NodeB (eNodeB). Each eNodeB is responsible for managing scheduling for both downlink and uplink channels for a UE. In this paper, we focus on downlink channels as most data traffic in cellular networks is in the downlink direction [11]. Investigation for the uplink channels is also important as user-generated content uploading is in this direction and the amount of such data is increasing over time; we leave further investigation as future work. Henceforth, link bandwidth in this paper refers to downlink bandwidth.

TABLE 1
Data Collected from AT&T Network

Phone	Scenario	Time	Location	Duration (hour)	Traffic (GB)	Avg. rate (Mbps)	Unique cell IDs	Handovers
Samsung Note 3	stationary	day	campus	0.9	1.5	3.7	2	2
	stationary	night	campus	0.8	2.1	5.8	2	9
	stationary	day	residence	1.2	5.3	9.8	2	9
	stationary	day	office (NJ)	1.5	14.1	20.9	1	0
	walking	day	campus	0.6	1.8	6.8	3	8
	walking	night	campus	0.7	2.5	7.9	1	0
	local driving	day	campus to residence	1.2	4.7	8.7	14	58
	highway driving	day	CT	0.9	6.5	16.0	35	87
	highway driving	day	CT to NY	0.4	2.9	16.1	24	28
HTC M8	stationary	day	campus	9.2	31.6	7.6	2	8
	stationary	night	campus	2.7	15.5	12.8	1	0
	stationary	day	campus arena (crowded)	2.3	9.4	9.1	4	8
	stationary	day	Hartford, CT	0.7	2.8	8.9	1	0
	walking	day	campus	1.7	3.7	4.8	4	7
	local driving	day	campus to residence	0.6	2.1	7.8	15	46
	highway driving	day	CT	0.9	6.4	15.8	43	98

An eNodeB allocates the resources to a UE based on its resource scheduling algorithm, the channel condition, and the UE's capability. A UE estimates channel quality based on various information collected from the radio channels, and reports both the channel quality and its radio capabilities (e.g., power headroom) to the associated eNodeB. Correspondingly, the eNodeB allocates proper resource (e.g., the resource blocks, modulation schemes, transmission power) to the UE, based on its available resource and channel conditions. The actual algorithm used by an eNodeB for radio resource allocation is implementation dependent and not available to the public. On the other hand, typically base stations enforce some forms of proportional fairness that take into account past throughput of a UE and channel quality to maximize the aggregate throughput of the UEs, while ensuring fairness among UEs and observing the priorities of the UEs [1], [7], [18].

Since the scheduling of eNodeB considers both past throughput and wireless link conditions, it is natural to conjecture that both types of information are useful in predicting link bandwidth. The study in [30] has investigated using historical throughput to predict link bandwidth. We investigate the correlations between various lower-layer information and the link bandwidth, and then utilize a machine learning based prediction framework to identify important features and predict link bandwidth in cellular networks.

3 Analysis of Lower-Layer Information

We analyze the correlation of link bandwidth and a large set of LTE related lower-layer information. In the following, we first describe the data that we collected for this study, and then present the results of correlation analysis.

3.1 Data Collection Methodology

We collect two types of data, cellular link bandwidth and lower-layer information. Link bandwidth of a cellular link represents the bandwidth that is available to a UE over the link. It is not directly perceivable, and has to be obtained through active measurement. We use the methodology in [26] to obtain link bandwidth as the average throughput when a well-provisioned server sends data to a phone using UDP; the server adjusts the size of the sending window so that the observed RTT is in a range (set to between 750 and 3,000 ms [26]), indicating that the link is saturated but not too overloaded. While the link is saturated, we use a packet sniffer to record the network traffic. Lower-layer information is available at a UE, and is recorded using QxDM. The reason for using QxDM is two-fold. First, it allows us to collect a large variety of lower-layer information. Second, QxDM records the various types of information at fine-grained sampling intervals (tens to hundreds of milliseconds), which allows us to study the correlation at a wide range of time lags.

To record QxDM traces, we connect a phone using a USB cable to a laptop that runs QxDM. QxDM communicates with the phone through a diagnostic port and records the various lower-layer information periodically. We have measured the CPU usage on a phone when it collects radio information and communicates with QxDM, and confirmed that the increase in CPU usage is not noticeable. In other words, collecting radio information in real time does not affect the normal operation of a UE. In Section 5, we investigate using lower-layer information that is collected directly at the phones through standard APIs provided by the operating system.

3.2 Collected Data

We have performed extensive measurements during a period of 15 months (from July 2014 to October 2015) on two major commercial LTE networks in the US, AT&T and Verizon networks. Tables 1 and 2 list the data from these two networks, respectively. We used seven different phones, including Samsung S3, S4, S5, Note 3 and HTC M8 on AT&T network, and Samsung S4 and S5 on Verizon network to collect data. The four phones listed in Tables 1 and 2 were used to collect majority of the data; the other three phones were used to confirm the results. We only report the results from the four phones in Tables 1 and 2; the results from the other three phones are consistent.

TABLE 2
Data Collected from Verizon Network

Phone	Scenario	Time	Location	Duration (hour)	Traffic (GB)	Avg. rate (Mbps)	Unique cell IDs	Handovers
Samsung S4	stationary	day	campus	5.7	25.7	10.0	2	27
	stationary	night	campus	2.9	14.7	11.3	2	10
	local driving	day	campus to residence	1.5	2.1	3.1	20	110
	highway driving	day	CT	0.8	2.2	6.1	45	66
Samsung S5	stationary	day	campus	3.0	14.0	10.4	2	8
	stationary	night	campus	3.0	18.6	13.8	2	6
	local driving	day	campus to residence	1.2	3.1	5.7	20	98
	highway driving	day	CT	0.9	4.5	11.1	49	88

The data were collected in three states, CT, NY and NJ, in the US, covering four movement scenarios: stationary, walking, local driving, and highway driving. For the stationary scenario, the data were collected in five locations: (1) a office on a university campus in CT (specifically, the University of Connecticut), (2) an arena on campus (when there was a major event inside the arena), (3) a residence that is 7.5 miles away from the campus, (4) a major city (Hartford, CT) that is 30 miles away from the campus, and (5) an office building in NJ. The time of data collection was during day (morning, noon, afternoon) when there were more cellular users and night when there were less users. For the walking scenario, the data ware collected on the university campus, again during day and night. The local driving was between the residence and the campus, riding in a car with a speed of 35 to 40 mph. In the highway driving scenarios, data were collected while driving at 65 to 70 mph, inside CT (between the campus and Hartford, CT) and from CT to NY (between the campus and New York City).

For the cases listed in Tables 1 and 2, the average data downloading rate is from 3.1 to 20.9 Mbps;[1] the number of unique cell IDs varies from 1 to 49, and the number of handover events varies from 0 to 110. Not surprisingly, we observe more handover events in local and highway driving scenarios than stationary and walking scenarios. The spectrum band of a cell is 5, 10 or 20 MHz (observed from QxDM).

In summary, the collected datasets cover a wide range of scenarios, including different phones, carrier networks, times of the day, locations and movement speed. The total duration of the data collection is over 45 hours. The total amount of data that was downloaded is over 197 GB. The total number of unique cell IDs is 161. The data will be used for correlation analysis in Section 3.3, and for training and testing the prediction models in Section 4.

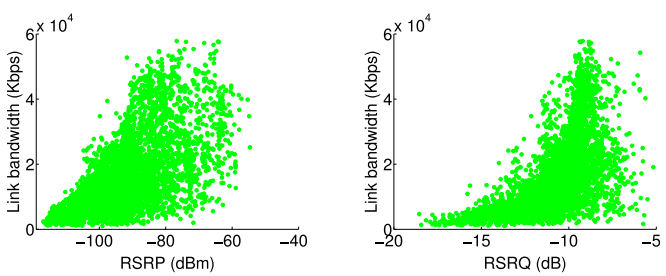
3.3 Correlation Analysis

We analyze the correlation of link bandwidth and a set of lower-layer information. The link bandwidth is the average throughput in a time unit of Δ, set to 0.5, 1, 2, 4, or 10 seconds. The lower-layer information is obtained by QxDM. For the lower-layer information that is captured at a much finer granularity, we aggregate multiple measurements over the interval of Δ, and use the average as the measurement value. In the following, we only present the results when $\Delta = 1$ second; the results for other time granularity are similar. For ease of exposition, we first present the results using the data collected from Samsung Note 3 on AT&T network (see Table 1), and then briefly describe the results in other settings.

3.3.1 RSRP and RSRQ

Reference Signal Received Power (RSRP) is the linear average (in watts) of the downlink reference signals across the channel bandwidth. It measures the absolute power of the reference signal. Reference Signal Received Quality (RSRQ) is defined as $(N \cdot \mathrm{RSRP})/\mathrm{RSSI}$, where N is the number of resource blocks across the downlink spectrum band, and Received Signal Strength Indicator (RSSI) is the *total* power that a UE observes across the entire download frequency band, including the main signal, co-channel non-serving cell signal, adjacent channel interference and the noise within the specified band.

We observe that both RSRP and RSRQ are correlated with link bandwidth. Fig. 2a is a scatter plot of RSRP and link bandwidth under the highway driving scenario (obtained using the two highway driving datasets in Table 1), where each point represents the average bandwidth over one second and the corresponding average RSRP over that second. We observe that when RSRP is very low, link bandwidth tends to be low as well (see lower left corner); similarly, when RSRP is very high, link bandwidth tends to be high as well (see upper right corner). Similar correlation is observed between RSRQ and link bandwidth (see Fig. 2b).



(a) RSRP (b) RSRQ

Fig. 2. Scatter plot of RSRP, RSRQ, and link bandwidth for the highway driving scenario.

1. Since the traces were collected at different times and locations, the data rate cannot be used to compare the services of the two commercial providers.

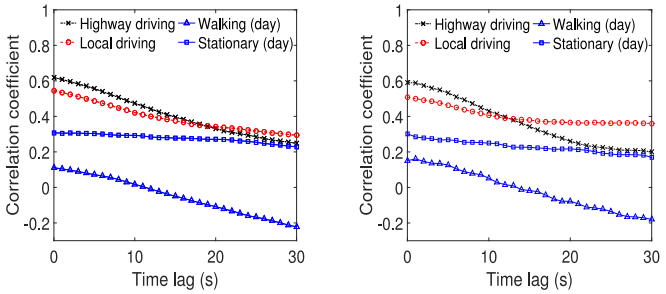


Fig. 3. Correlation coefficient between RSRP, RSRQ, and link bandwidth for various scenarios.

Fig. 3a plots the cross correlation between RSRP and link bandwidth under various scenarios. Specifically, for each scenario, we obtain two time series, $\{b_i\}$ and $\{r_i\}$, where b_i and r_i are the link bandwidth and RSRP during the ith second, respectively. Fig. 3a plots the cross correlation with lag ℓ, i.e., between $b_{i+\ell}$ and r_i, where $\ell = 0, 1, \ldots, 30$ seconds. Similarly, Fig. 3b plots the cross correlation between RSRQ and link bandwidth. In general, the correlation between RSRP and link bandwidth is comparable to that between RSRQ and link bandwidth. The correlation is significant even at a lag of tens of seconds except for the walking scenario (where the correlation is only significant at lags within a few seconds). We also observe that the correlation is larger for higher mobility scenarios (local and highway driving) compared to lower mobility scenarios (stationary and walking).

We also observe that, for both RSRP and RSRQ, its correlation with link bandwidth varies with day and night. Fig. 4 plots the correlation coefficients of RSRP and RSRQ with link bandwidth in a stationary setting at the university campus. Fig. 4a uses data collected during daytime, while Fig. 4b uses data collected during night time. Interestingly, we observe that the correlation between RSRQ and link bandwidth during night time is close to zero. This is the only scenario where RSRQ is not correlated with link bandwidth. Inspecting the trace, we think this might be because during night time, RSRQ is fairly stable while link bandwidth fluctuates over time.

3.3.2 Channel Quality Indicator (CQI)

CQI is an indicator carrying the information on the quality of the communication channel. It is a 4-bit integer (i.e., the value is between 0 and 15), and is determined by a UE [2]. The LTE specification does not state how a UE calculates CQI. In general, a UE takes into account multiple factors

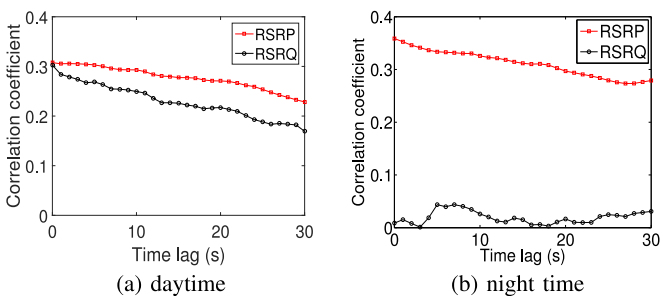


Fig. 4. Correlation coefficient between RSRP, RSRQ, and link bandwidth for the stationary scenario (data collected on campus).

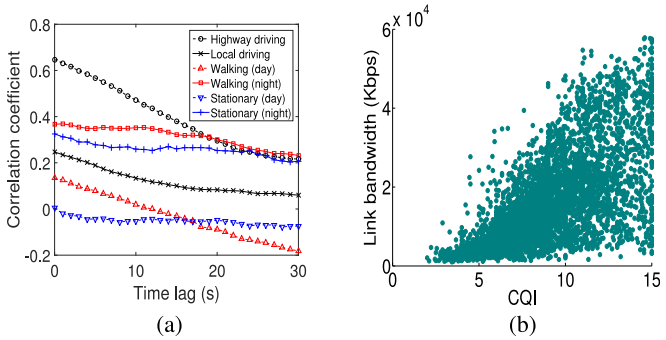


Fig. 5. (a) Correlation coefficient between CQI and link bandwidth for various scenarios. (b) Scatter plot of link bandwidth and CQI in the highway driving scenario.

(e.g., the number of antennas, signal-to-interference-noise-ratio (SINR), the capability of its radio) when calculating CQI. The calculated CQI is reported to the eNodeB, which performs downlink scheduling accordingly. As such, it is not surprising that CQI is correlated with link bandwidth. Fig. 5a plots the correlation coefficient of CQI and link bandwidth under various scenarios. We observe significant correlation even when the lag is up to a few or tens of seconds in most cases. However, in one scenario (stationary on campus during daytime), the correlation is very low for all the time lags.

Fig. 5b presents an example scatter plot using data collected from a highway driving scenario. We observe that when CQI is low, the observed link bandwidth is low as well (lower left corner). However, the scatter plot does not indicate a direct functional relationship between CQI and link bandwidth as that observed in [15].

3.3.3 Block Error Rate (BLER)

Hybrid-ARQ is used in LTE networks to achieve higher efficiency in transmission and error correction. It uses 8 stop-and-wait processes to transmit data. Once a packet is sent from a particular process, the process will be in active state and will not process other packets until it receives an ACK or NACK. If a NACK is received, then a retransmission will be initiated. In the face of too many retransmissions, the eNodeB has to adjust the modulation and coding scheme. Specifically, Hybrid-ARQ Block Error Rate is defined as the number of erroneous blocks divided by the total number of blocks that are sent. The target BLER is typically 10 percent [2]. If the actual BLER is larger than 10 percent (e.g., due to weak signal strength or interference), the link must be switched to a lower speed (leading to a lower bandwidth), and vice versa.

Fig. 6a is a scatter plot between link bandwidth and BLER under the highway driving scenario. We observe that most of the BLER values are between 5 and 10 percent. When BLER is above 10 percent, the link bandwidth tends to be low. On the other hand, when BLER is below 10 percent, the link bandwidth is in a wide range. As a result, the cross correlation between BLER and link bandwidth is not significant (figure omitted).

3.3.4 Handover Events

A handover is performed when a UE moves from the coverage of one cell to the coverage of another cell in the connected state. Unlike the case in UMTS, there are only hard

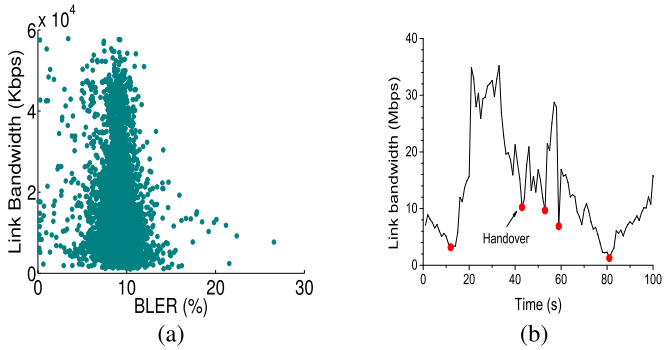


Fig. 6. (a) Scatter plot between link bandwidth and BLER (highway driving scenario). (b) Link bandwidth and handover events.

handovers in LTE networks, i.e., a UE cannot communicate with multiple eNodeBs simultaneously. Therefore there will be a short interruption in service when a handover happens. To capture handover events, we keep monitoring the serving cell ID. A change in serving cell ID indicates a handover event.

Fig. 6b plots a sample trace of link bandwidth where handover events are marked as red dots. It shows an instant decrease in bandwidth when handover happens. We represent handover events as a binary time series $\{h_i\}$, where $h_i = 1$ if there exists a handover event in the ith second and $h_i = 0$ otherwise. Let $\{b_i\}$ represent the time series of link bandwidth, where b_i is the average link bandwidth in the ith second. Calculating the cross correlation between $\{h_i\}$ and $\{b_i\}$ does not reveal significant correlation in any of the scenarios (the most significant correlation is under highway driving where the correlation under lag 1 is around -0.1). The low correlation might be because of the infrequency of handover events.

3.4 Other Phones and Networks

The analysis above is based on the traces collected using a Samsung Note 3 on AT&T network. To verify whether the correlation results are consistent across different phones and/or different carriers, we also evaluate the correlation results for other phones on both AT&T and Verizon networks. Overall, the results exhibit similar trends. For instance, for the datasets collected using Samsung S5 on Verizon network, at time lag of one second, the correlation coefficients of RSRP and link bandwidth range from 0.42 to 0.60 for various scenarios; for RSRQ and CQI, the correlation coefficients range from 0.31 to 0.52, and 0.32 to 0.61, respectively, all similar to what we have reported earlier for the Samsung Note 3 on AT&T network.

4 Link Bandwidth Prediction

In Section 3, we have identified five types of lower-layer information, RSRP, RSRQ, CQI, BLER and handover events, that are correlated with cellular link bandwidth. None of them has direct functional relationship with link bandwidth (unlike what is observed in [15], which shows that link bandwidth is a direct function of CQI). In this section, we develop a machine learning based framework called *LinkForecast* that predicts cellular link bandwidth in an online manner, using a prediction model that is learned offline. In the following, we first describe the prediction framework, and then describe how to select features. After that, we evaluate the performance of LinkForecast.

4.1 Prediction Framework

LinkForecast uses random forest [3] as the underlying machine learning technique for link bandwidth prediction. Random forest is an ensemble learning approach that uses multiple decision trees. The reason for using random forest is three-fold. First, since it uses the average of multiple trees (we use 20 trees), it is less sensitive to outliers in the training set and does not suffer from overfitting compared to using a single decision tree. Second, it requires low memory and computation overhead, and is suitable for prediction in an online manner on mobile phones. Third, it can determine the importance of the features (or predictors) that are used for the prediction, and hence can be used to identify the most important features.

Algorithm 1 summarizes the LinkForecast prediction framework. For convenience, we adopt a discrete time system $t = 1, 2, \ldots$, where each time unit is of length Δ. The prediction is therefore at the granularity of Δ. To predict the link bandwidth at time t, LinkForecast uses both the upper- and lower-layer information of the past k time units, where k is referred to as *prediction window size*, $k \geq 1$. Specifically, let $\hat{b}_t$ denote the predicted link bandwidth for time t. Suppose n types of information are used for prediction ($n = 6$ when throughput and the five types of lower-layer information are being used). Let x_t^i denote the measurement of information i in time t. Then the measurements of information i in the past k time units of t are $x_{t-k}^i, \ldots, x_{t-1}^i$. LinkForecast uses the past measurements $\mathbf{x} = (x_{t-k}^1, x_{t-k}^2, \ldots, x_{t-k}^n, \ldots, x_{t-1}^1, x_{t-1}^2, \ldots, x_{t-1}^n)$ as inputs and a prediction model learned offline from training data to predict $\hat{b}_t$. To obtain x_t^i, we may need to aggregate multiple samples of this information based on the sampling interval. Specifically, if the sampling interval is δ, then approximately Δ/δ samples will be used to obtain x_t^i.

Algorithm 1. LinkForecast: Link Bandwidth Prediction

Input: Historical upper- and lower-layer information (throughput, RSRP, RSRQ, CQI, BLER and handover events), k: prediction window size, $\mathcal{M}$: prediction model learned offline
Output: $\hat{b}_t$: predicted link bandwidth for time t
1: **for** each type of information **do**
2: obtain its values for each of the past k time units
3: **end for**
4: $\mathbf{x} = (x_{t-k}^1, \ldots, x_{t-k}^n, \ldots, x_{t-1}^1, \ldots, x_{t-1}^n)$; x_t^i is the value of information i in time t
5: $\hat{b}_t = \mathcal{M}(\mathbf{x})$
6: **return** $\hat{b}_t$

4.2 Feature Selection

LinkForecast uses both historical throughput and lower-layer information as features for link bandwidth prediction. In the following, we first identify the most important features and determine how to set prediction window size k. The time unit Δ is set to 1 second. That is, we predict the link bandwidth for the next second.

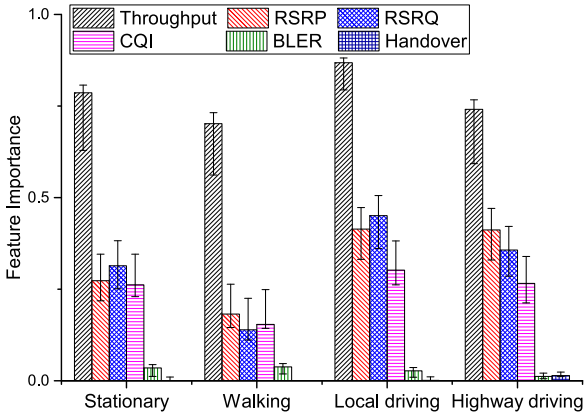
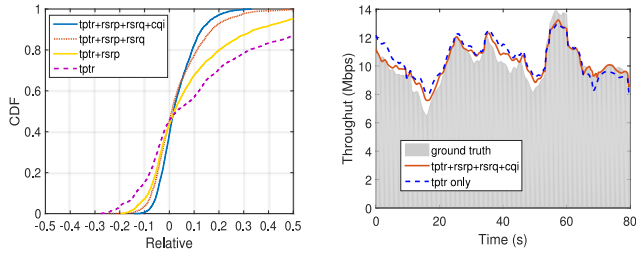


Fig. 7. Feature importance in various mobility scenarios.

4.2.1 Feature Importance

Since LinkForecast uses a random forest based prediction model, we use an approach [6] that is developed for random forest to determine the importance of the various features. In random forest, for each decision tree (recall that random forest uses multiple decision trees), a certain fraction of the samples are used to construct the tree; the remaining samples, called out-of-bag samples, are used to evaluate the prediction error and obtain Variable Importance (VI), an index that is used to quantify the importance of a feature. Specifically, for tree ℓ, consider the associated out-of-bag samples. Let err_ℓ denote the prediction error on the out-of-bag samples. Then the VI of feature j is $\sum_\ell (err_\ell^j - err_\ell)/N_{\text{tree}}$, where err_ℓ^j is the prediction error when randomly permuting the values of feature j in the out-of-bag samples, and N_{tree} is the number of trees used in the random forest. The intuition is that a feature is more important if perturbing it leads to a larger error. As an extreme case, if the VI of a feature is close to 0, then this feature will not affect the prediction result, and hence its importance is low.

We apply the above approach to obtain feature importance using the datasets that we collected. The prediction window size k is set to 1; we will come back to the choice of k in Section 4.2.2. Fig. 7 plots the VI of various features in four mobility scenarios: stationary, walking, local driving and highway driving. The average values as well as the maximum and minimum values from 20 runs (using different random seeds) are plotted in the figure. For each scenario, the results are obtained from a dataset that aggregates all the traces for that scenario (including those collected at different times, and using different phones and carriers). We observe that throughput has the highest importance, followed by RSRP, RSRQ and CQI (the importance of these three features varies in different scenarios; overall they have comparable importance), and then BLER and handover events. The relative low importance of BLER is consistent with the correlation results in Section 3, where we observe that it is correlated with link bandwidth only in certain cases. The importance of handover events is higher in the highway driving scenario than that in other three scenarios. This is not surprising since handover happens more frequently in the highway driving scenario than that in other scenarios. In scenarios with even higher moving speed (e.g., on high-speed rails) and hence even more frequent handovers [12], the importance of handover events might be even higher.



(a) Prediction using different features. (b) Example prediction results.

Fig. 8. Prediction accuracy when using different combination of features.

We make the following two conjectures based on the above observations: (i) while throughput has the highest importance, the importance of RSRP, RSRQ, and CQI is also significant, and hence they may also play important roles in predicting link bandwidth; and (ii) using the four most important features, i.e., throughput, RSRP, RSRQ, and CQI, might be sufficient to provide accurate predictions.

To verify the above two conjectures, we investigate the performance of five prediction models with increasing number of features. The first one uses only throughput, the second one uses both throughput and RSRP, the third one uses throughput, RSRP, and RSRQ, the fourth one uses all the four most important features (throughput, RSRP, RSRQ, and CQI), and the last one adds BLER and handover events as features (i.e., it uses all the features). Fig. 8a plots the Cumulative Distribution Function (CDF) of relative prediction errors of the first four models; the result of the last model overlaps with that of the fourth one, and is not plotted for clarity. Here the relative prediction error of a prediction is the predicted value minus the actual value, and then divided by the actual value. Hence positive relative errors indicate overestimation and negative relative errors indicate underestimation. Fig. 8a is obtained using the dataset collected using Samsung S5 on Verizon network (stationary, day time, duration of 3 hours). Specifically, we divide the trace into 1,440-second long segments; for each segment, we use the first 1,200 seconds of data as training set and the remaining 240 seconds of data as testing set. From Fig. 8a, we observe that in general using more features indeed leads to a better prediction accuracy; using the four most important features achieves almost the same performance as that when using all the features. On the other hand, using the most important feature, throughput, alone can lead to much inferior performance: in Fig. 8a, when using the four most important features, around 81.8 percent relative prediction errors are within ± 10 percent (i.e., between $[-10, 10]$ percent); when using throughput alone, only 41.7 percent of the relative prediction errors are within ± 10 percent. The results in other scenarios are similar. As a further illustration, Fig. 8b plots a snippet of the prediction results when only using throughput and when using the four most important features. The shaded region represents the ground-truth link bandwidth. We see that, compared to using the four most important features, using throughput alone can lead to less accurate prediction. In the rest of the paper, unless otherwise specified, the prediction is based on using the four most important features.

The four most important features that we have identified are perhaps not surprising: they represent historical throughput and link quality (RSRP, RSRQ, and CQI are

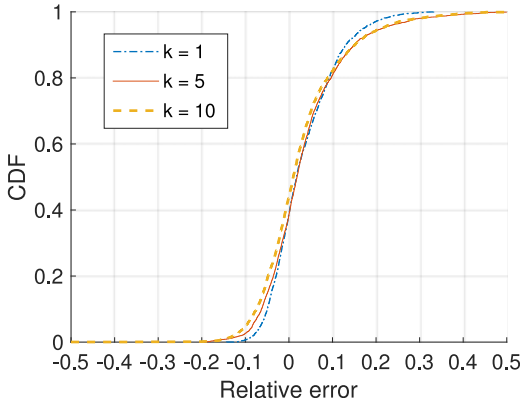


Fig. 9. Impact of prediction window size.

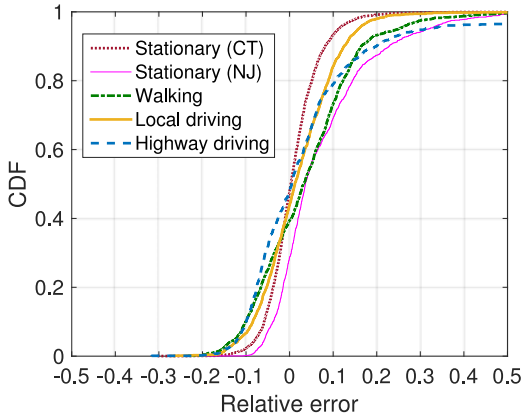


Fig. 10. Relative prediction error in various mobility scenarios (using Samsung Note 3 on AT&T network).

measures of link quality from different perspectives), the two important types of information that base stations use to enforce proportional fairness scheduling policy. On the other hand, the exact form of proportional fairness enforced by the base stations and their implementations vary across the vendors and networks. Therefore, it might be surprising that these four features can be used to provide accurate prediction for many base stations (we observe a total of 161 unique cell IDs in our datasets) in two large commercial cellular networks. In any case, our study is the first that quantifies the importance of these features in predicting link bandwidth and develops a real-time prediction framework using these features.

4.2.2 Prediction Windows Size

We now investigate the impact of prediction window size, k. Specifically, we set k to 1, 5, or 10, i.e., using historical information in the past one, five or ten seconds. We find that adding more historical data does not necessarily improve the performance of our prediction model. In fact, the performance may be degraded. The reason might be that in rapidly changing cellular environment, historical data can become obsolete quickly, and hence using data further away in the past may degrade performance. Fig. 9 shows an example. It is obtained using the same methodology and dataset as what are used for Fig. 8a. We observe similar trend when using other datasets. Considering that using a larger k may not improve prediction accuracy while leads to a higher computation overhead, we use $k = 1$ in the rest of the paper.

4.3 Prediction Accuracy

We evaluate the prediction accuracy of LinkForecast for each scenario listed in Tables 1 and 2. Specifically, the data for a scenario concatenates all the traces collected using the same phone, in the same carrier network, and with the same type of mobility. For the data in each scenario, we divide it into 1,440-second long segments; in each segment, the first 1,200 seconds and the remaining 240 seconds of the data are used as training and testing sets, respectively. For each instance of prediction, we obtain the relative prediction error (predicted bandwidth minus the actual bandwidth, and then divided by the actual bandwidth). Since relative prediction error can be positive or negative, we use the absolute value and then obtain the average prediction error for all the predictions in one scenario. The evaluation shows that LinkForecast provides accurate prediction in all the scenarios: when $\Delta = 1$ second, the average prediction errors are 3.9-9.0, 5.0-12.1, 6.0-8.4, 5.9-17.0 percent under stationary, walking, local and highway driving scenarios, respectively (the range of the prediction error in each mobility scenario is obtained from the multiple settings differentiated by the phone and carrier network). The prediction under lower mobility scenarios (stationary, walking and local driving) are more accurate than that under highway driving scenario. This is perhaps not surprising since the link bandwidth changes more rapidly in high mobility scenarios and hence is more challenging to predict accurately.

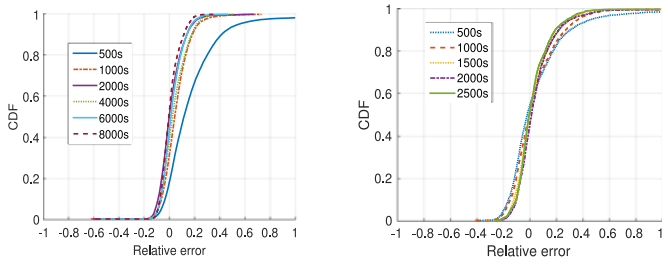
In the interest of space, we only describe the prediction results for the datasets collected using Samsung Note 3 on AT&T network in detail. Fig. 10 plots the CDF of the relative prediction error under stationary (CT), stationary (NJ), walking, local and highway driving scenarios, respectively (we differentiate the two stationary scenarios since the data are collected in CT and NJ, respectively). For these five scenarios, 91, 69, 63, 78 and 70 percent of the relative prediction errors are within ± 10 percent, respectively.

4.4 Sensitivity to Training Data

The prediction accuracy of LinkForecast depends on the training set: the training data needs to contain sufficient amount of variation and range. A natural question is how sensitive the performance is to training data. In the following, we first consider a fixed scenario (for a fixed phone, mobility and carrier), and investigate the length of training data that is needed for accurate prediction. We then present cross-scenario evaluation, which applies data collected in one scenario as the training data for another scenario.

4.4.1 Length of Training Data Needed

For a dataset corresponding to a certain scenario, we gradually increase the length of training data, starting from a length of 500 seconds. For each length, we find a continuous segment of data that has the largest variation (in terms of the difference between the maximum and minimum throughput), and use it as the training set; the remaining data is used as the testing set. We find that in general a training set of 1,000 seconds already yields accurate prediction. Fig. 11 shows two examples, one from a stationary scenario



(a) Stationary scenario. (b) Highway driving scenario.

Fig. 11. Impact of the length of the training data.

(corresponding to Samsung S5 on Verizon network, day time) and the other from a highway driving scenario (corresponding to Samsung Note 3 on AT&T network). In both cases, we observe significant improvement in prediction accuracy when increasing the length of the training data from 500 to 1,000 seconds; the improvement afterwards is less significant. Results for other datasets show similar trend.

In practice, when there is no knowledge about when throughput will exhibit the largest variation, we need to collect more data as training set. On the other hand, random forest incurs very low memory and computation overhead: even when using thousands of seconds long training data, the computation time on a modern processor is negligible. In addition, in our approach, since the training is done offline, the length of the training data does not affect the computation time of the online prediction.

4.4.2 Cross-Scenario Evaluation

We evaluate the sensitivity of the prediction accuracy to training data in four settings, on the sensitivity to mobility, phone, time, and cellular carrier, respectively.

Sensitivity to Mobility. In this setting, to predict the link bandwidth in one mobility scenario, we use the data collected in other mobility scenarios as training data. Specifically, the data collected using Samsung Note 3 from AT&T network (Table 1) contains four mobility scenarios: stationary, walking, local driving, and highway driving. For each testing scenario, we use the data collected from all the other scenarios as the training data, thus creating four pairs of training and testing sets given the four mobility scenarios. Fig. 12 (marked as "cross-mobility") presents the CDF of the relative errors of all the cross-mobility evaluation scenarios. It shows that the prediction is very accurate: 90 percent of the relative errors are within ±8.8 percent.

Cross-Phone Prediction. In this setting, we use the data collected from one phone as training data and the data collected from another phone as testing data. Specifically, we consider two phones, HTC M8 and Samsung Note 3. When using data collected from Samsung Note 3 as training set and the data from HTC M8 as testing set, 80 percent of the relative errors are within ±9.4 percent, and the average prediction error is 6.1 percent. The results for the other case are similar. Fig. 12 ("cross-phone" curve) presents the CDF of the relative errors of these two cross-phone testing scenarios.

Sensitivity to Time. We choose two datasets that were collected far apart in time: one in July 2014 and the other in October 2015. When using the first dataset as training data, 80 percent of the relative errors are within ±9.8 percent, and the average prediction error is 7.3 percent. When using the second dataset as the training data, 80 percent of the relative errors are within ±10.4 percent, and the average prediction error is 8.1 percent. Fig. 12 ("cross-time" curve) presents the CDF of the relative errors of these two cross-time testing scenarios.

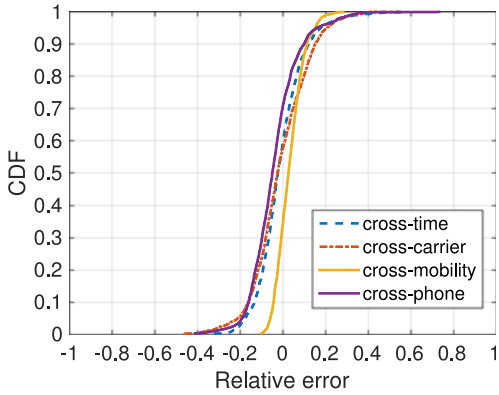


Fig. 12. Sensitivity to training data: Cross-scenario evaluation.

Sensitivity to Carrier. To investigate whether a prediction model obtained from one carrier network can be used for another carrier network, we perform a "cross-carrier" test. Specifically, we compile two datasets: one containing all the data collected from the Verizon network, and the other containing all the data collected from the AT&T network. When we use the Verizon data as the training set and the AT&T data as testing set, 80 percent of the relative errors are within ±10.8 percent (the average prediction error is 8.5 percent). When we switch the training and testing sets, 80 percent of the relative errors are within ±11.2 percent (the average prediction error is 9.1 percent). In both cases, our approach achieves good prediction accuracy. Fig. 12 ("cross-carrier" curve) presents the CDF of the relative errors of these two cross-carrier testing scenarios.

4.5 Summary

In this section, we have designed and implemented a machine learning framework, LinkForecast, that predicts link bandwidth in real time using past throughput and lower-layer information. Our evaluation shows that it achieves good prediction accuracy in all the scenarios we investigated. In addition, we observe good prediction accuracies in all the cross-scenario evaluation settings, indicating that the prediction results are not sensitive to the training data. The insensitivity to training data indicates that LinkForecast can be easily applied in practice: it is easy to collect training data and build prediction models; the training data do not need to be collected from very specific situations and be only useful to those settings.

5 Lower-Layer Information in Real Systems

So far, we have been using lower-layer information obtained using QxDM, a specialized diagnostic tool. Since QxDM is not widely available to end users, in this section, we explore cellular link bandwidth prediction when using lower-layer information obtained through standard APIs provided by the operating system. In the following, we first describe data collection methodology and the collected datasets, and then compare the lower-layer information collected from the

TABLE 3
Data Collected Using an HTC M8 Phone on the AT&T Network

Phone	Scenario	Time	Location	Duration (hour)	Traffic (GB)	Avg. rate (Mbps)	Unique cell IDs	Handovers
HTC M8	stationary	day	campus	4.2	10.8	5.7	3	4
	walking	day	campus	1.7	3.7	4.8	4	7
	local driving	day	campus to residence	1.8	5.3	6.5	16	72
	highway driving	day	CT	0.9	6.4	15.8	43	98

Lower-layer information are collected using both QxDM and Android APIs for comparison purposes.

standard APIs with that collected from QxDM. In the end, we compare the link bandwidth predicted using these two methodologies.

5.1 Data Collection

Currently, not all lower-layer information that is accessible through QxDM is directly accessible at end devices. On the other hand, the top three lower-layer information, RSRP, RSRQ, and CQI, have been made available by Android OS. We use an HTC M8 phone (running Android 4.4, API level 19 on AT&T network) for data collection in this section. On this phone, both RSRP and RSRQ are available, while CQI is still a developing function that is not yet available (it is likely to be available in the future). Therefore, in the following, link bandwidth prediction at the end system uses past throughput as well as RSRP and RSRQ information directly from the operating system.

We collect two types of data, one directly through standard Android APIs and the other using QxDM. These two types of data are collected simultaneously so that they are comparable. Specifically, we use the same methodology as described earlier (see Section 3.1) to collect QxDM data. To collect data through Android APIs, we write an app that registers an *event receiver* to the operating system. When a signal change event happens (specifically, a change in RSRP or RSRP), the Android core module will send out an event to the event receiver in the app, which will catch the event and write the current values of the signal to the SD card of the phone. The timestamps of all the data (from QxDM and the standard APIs) are based on the clock of the phone.

Table 3 lists the data collected. It contains 8.6 hours and 26.2 GB downloaded data, covering four mobility scenarios: stationary, walking, local driving, and highway driving.

5.2 Lower-Layer Information from Android APIs

We observe lower-layer information collected using Android APIs has lower fidelity compared to that collected using QxDM. This is expected since QxDM collects data periodically at fine-grain intervals (in tens to hundreds of milliseconds) from the chip-set, while when using Android APIs, a signal value is collected only after a signal change event is triggered, i.e., when the extent of change exceeds a threshold (which is set in the core module). Fig. 13 shows a snippet of RSRP values collected using QxDM and Android APIs. We see that the values collected using QxDM change more frequently than those collected using Android APIs. In addition, the values from QxDM are floating-point numbers, while those collected using Android APIs are integers (and thus have a lower granularity).

We next quantify the differences using all the data we have collected (see Table 3). Specifically, we calculate the relative difference between a value obtained using the Android APIs and the corresponding value obtained using QxDM as the difference of these two values divided by the latter. Fig. 14 plots the CDF of the relative differences. Both the results of RSRP and RSRQ are plotted in the figure. For RSRQ, the relative differences are close to zero; for RSRP, around 80 percent of the relative differences are within ± 10 percent.

5.3 Prediction Accuracy

We now compare the prediction accuracy when using the data collected through Android APIs with those collected through QxDM. Fig. 15 plots CDFs of the relative prediction errors under four mobility scenarios using the data in Table 3. For the data in each mobility scenario, we again divide the data into segments of 1,440 seconds, using the first 1,200 seconds of data as training set and the remaining 240 seconds of data as testing set. Both the results when

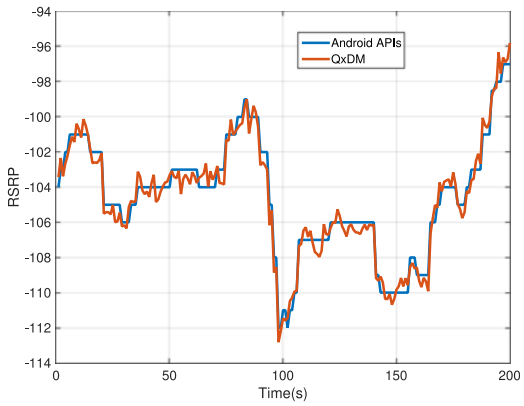


Fig. 13. Comparison of RSRP values collected using QxDM and Android APIs.

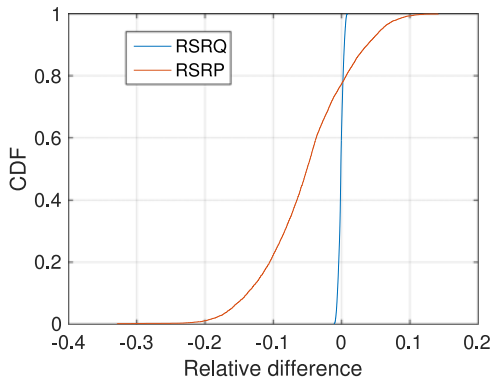


Fig. 14. Relative differences of the values collected using Android APIs and those collected using QxDM.

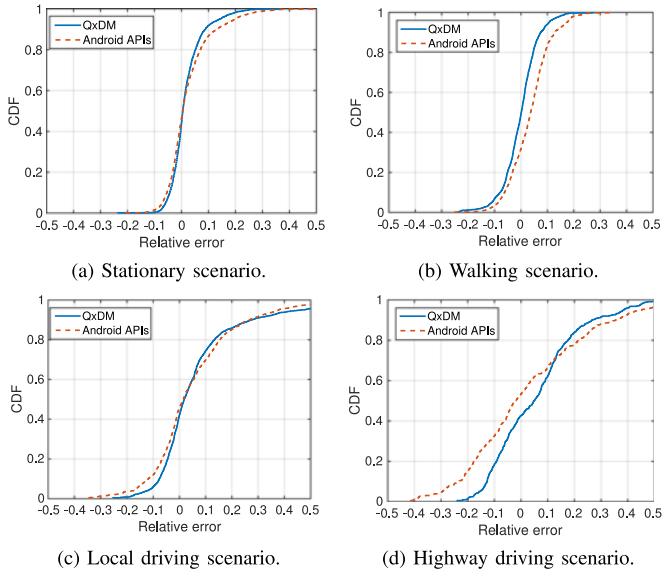


Fig. 15. Prediction errors when using data collected through Android APIs in comparison with the results when using data collected through QxDM.

using data from Android APIs (obtained using historical throughput, and lower-fidelity RSRP and RSRQ) and QxDM (obtained using historical throughput, and higher-fidelity RSRP, RSRQ and CQI) are plotted in the figure. We observe that, while RSRP and RSRQ collected using Android APIs are of lower fidelity and CQI is not yet available from the APIs, the prediction accuracy when using Android APIs is only slightly worse than that when using QxDM. Compared to the ground truth, the average prediction error when using Android APIs is 5.6, 6.8, 12.3, and 18.4 percent under stationary, walking, local and highway driving scenarios, respectively; the corresponding values are 4.1, 5.0, 11.1, and 13.5 percent when using QxDM.

In summary, we have demonstrated that lower-layer information collected directly from Android APIs can be used to provide accurate link bandwidth prediction. Such information is readily available at end systems, without any change to the operating system or the need of rooting the phone. Therefore, our link bandwidth prediction approach can be easily deployed in practice.

6 Related Work

Bandwidth estimation has been studied extensively in wired networks and wireless LANs. The study in [10] demonstrates that existing bandwidth estimation techniques for wired networks and wireless LANs are not effective in cellular networks. The studies in [14], [16], [30] indicate that data rate is predictable in cellular networks. Xu et al. [30] develop a system interface called PROTEUS that forecasts future network performance (throughput, loss, and one-way delay) based on current measurements, and integrate it into real-time communication applications. Winstein et al. [26] use packet interarrival time to infer link bandwidth and further determine the number of packets that can be transmitted. The prediction techniques in these two studies do not use lower-layer information.

Lower-layer information is used in several studies to predict link bandwidth. Chakraborty et al. develop a Support Vector Machine (SVM) based technique that categorizes bandwidth into two classes (high and low bandwidth, respectively), and propose a technique for UEs to coordinate cellular background transfer in an efficient manner [4]. Margolies [17] et al. generate throughput prediction for a mobile device based on the observation that the signal quality of a device is reproducible for multiple drives on the same path. The study in [15] uses CQI and DRX (discontinuous transmission) ratio to predict link bandwidth in HSPA+ networks. The prediction is by looking up a mapping table at a base station. Since the mapping table is vendor implementation dependent, the authors propose to use crowdsourcing to obtain the mapping table at each base station. Our work differs from the above studies in that we investigate an extensive set of lower-layer information, and construct a prediction model that uses both upper-layer (past throughput) and lower-layer information to predict link bandwidth in real time. We do not find a simple functional relationship between CQI and link bandwidth. Instead, we develop a machine learning technique that uses various features to predict link bandwidth, and demonstrate that our approach leads to accurate prediction and is not sensitive to training data.

The study in [20] investigates instantaneous throughput prediction in cellular networks. The focus is on predicting the available bandwidth before a phone establishes a connection to the content provider. The authors show that physical-layer information (RSRP and RSRQ) and radio access network (RAN) measurements collected at the operator's networks are most helpful in achieving accurate prediction. Our study differs from it in that we study link bandwidth prediction when a connection has been established, and show that past throughput and lower-layer information are both important for accurate link bandwidth prediction. Our approach does not require any information from the operator's network.

Several studies [11], [13], [27], [28], [29] develop tools to extract detailed cellular network information, which can be used to infer the network bandwidth that is potentially available to a phone. These tools, however, require software defined radio or access to certain diagnostic information that demands rooting a phone. Our prediction approach uses standard APIs provided by the operating system and can be used directly on commercial off-the-shelf phones without the need of rooting the phones.

Last, the relationship between lower-layer information and link bandwidth has also been investigated and leveraged in [21], [23]. Schulman et al. find that signal strength is correlated with link bitrate and power consumption, and propose an energy-aware scheduling algorithm for different workloads [21]. Sorous et al. find that signal strength and throughput are correlated to some extent [23], and demonstrate that measurement of throughput at a server can be used to reveal user location. Several studies investigate the relationship between radio status and performance at an end device. Liu et al. study the interplay between wireless channels and applications in CDMA networks [14]. To capture the radio information, Vallina-Rodriguez et al. implement a tool named RilAnalyzer [24], which can record the radio connection status and application behaviors. Our work differs in scope from the above studies.

7 Conclusion and Future Work

In this paper, we first conducted an extensive measurement study in two major commercial LTE networks in the US and identified a comprehensive set of lower-layer information that is correlated with cellular link bandwidth. We then developed LinkForecast, a machine learning framework that utilizes both past throughput and lower-layer information to predict link bandwidth in real time. Evaluation results in a wide range of scenarios demonstrated that our approach leads to accurate prediction, incurs low computation overhead, and is insensitive to the training set. Last, we investigated using lower-layer information obtained through standard Android APIs for link bandwidth prediction, and compared the prediction performance with that using QxDM. We showed that the prediction is accurate, and hence our approach can be easily deployed on commercial off-the-shelf mobile devices.

As future work, we plan to explore using LinkForecast in developing congestion control protocols and video streaming techniques. While recent studies [5], [9], [22], [25], [26], [32] have proposed congestion control protocols to tackle the challenges in cellular networks, leveraging realtime bandwidth prediction from LinkForecast can lead to alternative designs that achieve both high throughput and low latency. For video streaming, while the study [33] has demonstrated theoretically that accurate link bandwidth prediction can significantly benefit adaptive bitrate streaming, how to use realtime bandwidth prediction for video streaming for cellular network needs further investigation. We will design rate adaptation strategies using the bandwidth prediction from LinkForecast.

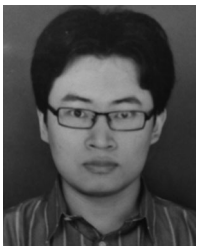
Acknowledgments

K. Suh was partially supported by an Illinois State University IT research grant. B. Wang was partially supported by US National Science Foundation CAREER award 0746841. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the funding agencies.

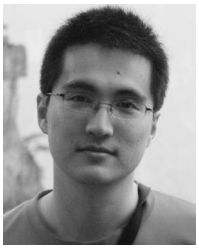
References

[1] Nokia Solutions and Networks Smart Scheduler, NSN (Nokia Solutions and Networks) white paper, Feb. 2014.
[2] 3GPP. 3GPP TS 36.213, "Evolved universal terrestrial radio access. (E-UTRA); physical layer procedures," 2014.
[3] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.
[4] A. Chakraborty, V. Navda, V. N. Padmanabhan, and R. Ramjee, "Coordinating cellular background transfers using LoadSense," in *Proc. Annu. Int. Conf. Mobile Comput. Netw.*, 2013, pp. 63–74.
[5] M. Dong, Q. Li, D. Zarchy, B. Godfrey, and M. Schapira, "PCC: Re-architecting congestion control for consistent high performance," in *Proc. Netw. Syst. Des. Implementation*, 2015, pp. 395–408.
[6] R. Genuer, J.-M. Poggi, and C. Tuleau-Malot, "Variable selection using random forests," *Pattern Recognit. Lett.*, vol. 31, no. 14, pp. 2225–2236, 2010.
[7] A. Ghosh, J. Zhang, J. G. Andrews, and R. Muhamed, *Fundamentals of LTE*, 1st ed. Englewood Cliffs, NJ, USA: Prentice Hall, Sep. 2010.
[8] J. Huang, et al., "An in-depth study of LTE: Effect of network protocol and application behavior on performance," in *Proc. Conf. Appl. Technol. Archit. Protocols Comput. Commun.*, 2013, pp. 363–374.
[9] H. Jiang, Y. Wang, K. Lee, and I. Rhee, "Tackling bufferbloat in 3G/4G networks," in *Proc. ACM SIGCOMM Internet Meas. Conf.*, 2012, pp. 329–342.
[10] D. Koutsonikolas and Y. C. Hu, "On the feasibility of bandwidth estimation in 1xEVDO networks," in *Proc. 1st ACM Workshop Mobile Internet Through Cellular Netw.*, Sep. 2009, pp. 31–36.
[11] S. Kumar, E. Hamed, D. Katabi, and L. E. Li, "LTE radio analytics made easy and accessible," in *Proc. ACM Conf. SIGCOMM*, 2014, pp. 211–222.
[12] L. Li, K. Xu, D. Wang, C. Peng, Q. Xiao, and R. Mijumbi, "A measurement study on TCP behaviors in HSPA+ networks on high-speed rails," in *Proc. IEEE Conf. Comput. Commun.*, 2015, pp. 2731–2739.
[13] Y. Li, C. Peng, Z. Yuan, J. Li, H. Deng, and T. Wang, "MobileInsight: Extracting and analyzing cellular network information on smartphones," in *Proc. Annu. ACM Int. Conf. Mobile Comput. Netw.*, 2016, pp. 202–215.
[14] X. Liu, A. Sridharan, S. Machiraju, M. Seshadri, and H. Zang, "Experiences in a 3G network: Interplay between the wireless channel and applications," in *Proc. Annu. ACM Int. Conf. Mobile Comput. Netw.*, 2008, pp. 211–222.
[15] F. Lu, H. Du, A. Jain, G. M. Voelker, A. C. Snoeren, and A. Terzis, "CQIC: Revisiting cross-layer congestion control for cellular networks," in *Proc. Int. Workshop Mobile Comput. Syst. Appl.*, 2015, pp. 45–50.
[16] J. Manweiler, S. Agarwal, M. Zhang, R. R. Choudhury, and P. Bahl, "Switchboard: A matchmaking system for multiplayer mobile games," in *Proc. ACM Int. Conf. Mobile Syst. Appl. Services*, 2011, pp. 71–84.
[17] R. Margolies, et al., "Exploiting mobility in proportional fair cellular scheduling: Measurements and algorithms," in *Proc. IEEE INFOCOM*, 2014, pp. 1339–1347.
[18] K. I. Pedersen, T. E. Kolding, F. Frederiksen, I. Z. Kovacs, D. Laselva, and P. E. Mogensen, "An overview of downlink radio resource management for UTRAN long-term evolution," *IEEE Commun. Mag.*, vol. 47, no. 7, pp. 86–93, Jul. 2009.
[19] Qualcomm, "Qualcomm eXtensible diagnostic monitor (QXDM professional)," 2008. [Online]. Available: http://www.qualcomm.com/media/documents/tags/qxdm
[20] A. Samba, Y. Busnel, A. Blanc, P. Dooze, and G. Simon, "Instantaneous throughput prediction in cellular networks: Which information is needed?" in *Proc. IFIP/IEEE Int. Symp. Integr. Netw. Manage.*, May 2017, pp. 624–627.
[21] A. Schulman, et al., "BarTendr: A practical approach to energy-aware cellular data scheduling," in *Proc. Annu. ACM Int. Conf. Mobile Comput. Netw.*, 2010, pp. 85–96.
[22] A. Sivaraman, K. Winstein, P. Thaker, and H. Balakrishnan, "An experimental study of the learnability of congestion control," in *Proc. ACM Conf. SIGCOMM*, Aug. 2014, pp. 479–490.
[23] H. Soroush, K. Sung, E. Learned-Miller, B. N. Levine, and M. Liberatore, "Turning off GPS is not enough: Cellular location leaks over the Internet," in *Proc. Privacy Enhancing Technol. Symp.*, 2013, pp. 103–122.
[24] N. Vallina-Rodriguez, A. Auçinas, M. Almeida, Y. Grunenberger, K. Papagiannaki, and J. Crowcroft, "RILAnalyzer: A comprehensive 3G monitor on your phone," in *Proc. ACM SIGCOMM Internet Meas. Conf.*, 2013, pp. 257–264.
[25] K. Winstein and H. Balakrishnan, "TCP ex machina: Computer-generated congestion control," in *Proc. ACM Conf. SIGCOMM*, Aug. 2013, pp. 123–134.
[26] K. Winstein, A. Sivaraman, and H. Balakrishnan, "Stochastic forecasts achieve high throughput and low delay over cellular networks," in *Proc. USENIX Conf. Netw. Syst. Des. Implementation*, 2013, pp. 459–472.
[27] B. Wojtowicz, "OpenLTE," (2011). [Online]. Available: http://openlte.sourceforge.net
[28] X. Xie, X. Zhang, S. Kumar, and L. E. Li, "piStream: Physical layer informed adaptive video streaming over LTE," in *Proc. Annu. ACM Int. Conf. Mobile Comput. Netw.*, 2015, pp. 413–425.
[29] X. Xie, X. Zhang, and S. Zhu, "Accelerating mobile web loading using cellular link information," in *Proc. ACM Annu. Int. Conf. Mobile Syst. Appl. Services*, 2017, pp. 427–439.
[30] Q. Xu, S. Mehrotra, Z. Mao, and J. Li, "PROTEUS: Network performance forecast for real-time, interactive mobile applications," in *Proc. ACM Annu. Int. Conf. Mobile Syst. Appl. Services*, 2013, pp. 347–360.

[31] J. Yao, S. S. Kanhere, and M. Hassan, "An empirical study of bandwidth predictability in mobile computing," in *Proc. ACM Int. Workshop Wireless Netw. Testbeds Exp. Eval. Characterization*, 2008, pp. 11–18.
[32] Y. Zaki, T. Pötsch, J. Chen, L. Subramanian, and C. Görg, "Adaptive congestion control for unpredictable cellular networks," in *Proc. ACM Conf. Special Interest Group Data Commun.*, 2015, pp. 509–522.
[33] X. K. Zou, et al., "Can accurate predictions improve video streaming in cellular networks?" in *Proc. Int. Workshop Mobile Comput. Syst. Appl.*, 2015, pp. 57–62.

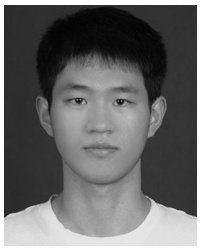


Chaoqun Yue received the BS degree in software engineering from Xi'an Jiaotong University, China, in 2011 and the MS degree in computer science from Shanghai Jiao Tong University, China, in 2014. He is currently working toward the PhD degree in the Computer Science & Engineering Department, University of Connecticut. His research interests include the wireless networks and wireless sensing applications. He is a student member of the IEEE.

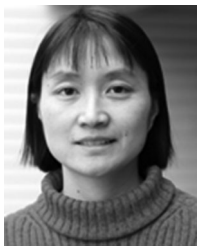


Ruofan Jin received the BS and MS degrees in computer science and engineering from Beihang University, China, in 2007 and 2010, respectively, and the PhD degree from the University of Connecticut, in 2015. His research interests include the wireless networks and performance optimization. He is a member of the IEEE.

Kyoungwon Suh received the BS and MS degrees in computer engineering from Seoul National University, Korea, in 1991 and 1993, respectively. He received the other MS degree from the Department of Computer Science, Rutgers University, New Jersey, in 2000. In 2007, he received the PhD degree in computer science from the University of Massachusetts at Amherst. He is currently an associate professor with Illinois State University, Normal, Illinois. His research interests include mobile hand-held devices, wireless networks, network measurement and inference, network security, and multimedia content distribution. He is a member of the ACM and the IEEE.



Yanyuan Qin received the BS degree in automation from the Nanjing University of Aeronautics and Astronautics, China, in 2011 and the MS degree in control science and engineering from Shanghai Jiao Tong University, China, in 2014. He is currently working toward the PhD degree in the Computer Science & Engineering Department, University of Connecticut. His research interests include wireless networking and software defined networking (SDN). He is a student member of the IEEE.



Bing Wang received the BS degree in computer science from the Nanjing University of Science & Technology, China, in 1994, and the MS degree in computer engineering from the Institute of Computing Technology, Chinese Academy of Sciences, in 1997. She then received the MS degrees in computer science and applied mathematics, and the PhD degree in computer science from the University of Massachusetts, Amherst, in 2000, 2004, and 2005, respectively. She is currently a professor of computer science and engineering with the University of Connecticut. Her research interests include computer networks and distributed systems. She received an NSF CAREER award in February 2008. She is a member of the IEEE.

Wei Wei received the BS degree in applied mathematics from Beijing University, China, in 1992 and the MS degree in statistics from Texas A & M University, in 2000. He then received the MS degrees in computer science and applied mathematics, and the PhD degree in computer science from the University of Massachusetts, Amherst, in 2004, 2004, and 2006 respectively. He is currently an assistant professor in residence in the Computer Science and Engineering Department, University of Connecticut. His research interests iclude the computer networks, statistical inference, and performance modeling. He is a member of the IEEE.