Experimental Comparison of Machine Learning-Based Available Bandwidth Estimation Methods over Operational LTE Networks

Natsuhiko Sato[†], Takashi Oshiba[‡], Kousuke Nogami[‡], Anan Sawabe[‡], and Kozo Satoda[‡]
[†]Department of Physics, University of Tokyo, Japan
[‡]System Platform Research Laboratories, NEC Corporation, Japan

***Abstract*—We propose PathML, an available bandwidth (i.e., unused capacity of an end-to-end path) estimation method based on a data-driven paradigm that uses machine learning with a large amount of data. An experiment over an operational LTE network was performed to compare our method with prior work.**

Now we are living in the cloud era: an age in which large amounts of data are collected from numerous devices and accumulated on clouds. To take advantage of the characteristics of this cloud era, we propose a new method of available bandwidth estimation that uses the accumulated mass data. In prior work, network specialists constructed simplified models of complex network behavior based on a relatively small amount of measured data and designed estimation algorithms based on the simplified model. However, when network behavior not assumed in the model construction occurs, these algorithms are not equipped to deal with the newly obtained data and cannot extract sufficient information from it. This adverse simplification effect decreases the estimation accuracy. In contrast, our method based on machine learning can extract information that is ignored or overlooked by humans, thus enabling the estimation of available bandwidth with high accuracy even in the uncertain situations with which prior work struggles.

We selected four machine learning techniques suitable for available bandwidth estimation: namely, support vector regression, kernel ridge regression, random forests, and convolutional neural network. We performed an experiment over an operational LTE network of Japan's primary mobile operator to compare our method with prior work. Results showed that our method clearly outperformed the prior work in terms of available bandwidth estimation accuracy. The most accurate technique was the convolutional neural network: its estimation accuracy was 83.7% compared to the 74.2% of the prior work. Specifically, under the 40–50 Mbps broadband condition, the mean absolute error of the values estimated by our method was just 2.2 Mbps while that of the prior work was 16.4 Mbps—in other words, only about 1/8 that of the prior work.

Keywords—available bandwidth; machine learning; LTE

I. INTRODUCTION

Due to the rapid growth of mobile networks such as LTE and the spread of cloud computing, a large amount of data is being collected and accumulated from a vast number of smartphones. To extract valuable information from such data, machine learning is commonly used to analyze data on behalf of humans. For example, Evernote[1] collects images of documents (such as business cards and receipts) that are saved on smartphones by users on the cloud and then extracts text data from collected images of documents by means of machine learning-based optical character recognition (OCR). This enables users to search text in documents. Another example is Google photos[2], which collects photos saved on smartphones by users on the cloud and automatically generates albums and tags by machine learning-based classification. These services were infeasible not so long ago due to the communication cost for data collection, the computational cost for machine learning, and the social acceptance of data collection and usage but have recently come into favor with the advent of the cloud era. In addition, the diffusion of crowdsourcing now makes it easier to collect data on mobile network performance [1]. Strides are also being made with applying machine learning to the data collected by crowdsourcing [2].

In light of the above, we propose PathML, a new machine learning-based available bandwidth (i.e., unused capacity of an end-to-end path) estimation method. This method is based on a data-driven paradigm. In previous studies, network specialists constructed simplified models of complex network behavior with a relatively small amount of measured data and designed estimation algorithms on the basis of the model. The amount of data was necessarily small, as humans cannot handle or interpret large amounts of data. However, such prior works had a problem in that when network behavior that is not assumed in the simplified model occurs, the estimation algorithm cannot handle newly observed data and can extract only a little information from it, which decreases the estimation accuracy. Moreover, many operational LTE networks utilize a packet scheduler at the evolved Node B (eNB), and this complicates the network behavior. This also tends to decrease the estimation accuracy of prior work (see Section III).

We applied the machine learning approach for available bandwidth estimation over an operational LTE network and found that our method clearly outperformed the prior work. We focus on the downlink direction of LTE networks in this paper because the traffic volume of the downlink can be more than ten times as large as that of the uplink [3] and is thus dominant in LTE networks.

The main contributions of this paper are as follows:

1. We proposed a machine learning-based available bandwidth estimation method based on the data-driven paradigm with a large amount of data. We also selected four machine learning techniques suitable for available bandwidth estimation.
2. We conducted an experiment over an operational LTE network and applied our machine learning-based method for available bandwidth estimation to an operational LTE network for the first time.
3. In general, machine learning approaches require a certain amount of data. We revealed quantitatively how much data

[1] https://evernote.com/

[2] https://photos.google.com/

is needed to estimate available bandwidth with sufficient accuracy.

II. Related Work

Much prior work has been done on end-to-end available bandwidth estimation that actively sends probing packet trains (i.e., a set of multiple probing packets) [4]. These methods using measured data of the packet train are classified into (A) methods that assume simplified models of network behavior and perform estimation on the basis of the models and (B) methods that are based on machine learning.

A. Prior Work Based on a Simplified Model

The behavior of a network differs depending on the type of network (wired, Wi-Fi, etc.), so many of the prior methods were designed for specific network types. Representative conventional methods for available bandwidth estimation include pathChirp [5], Pathload [6], and PTR [7]. These methods were originally designed for wired networks, and after these received positive attention in the research community, many methods for Wi-Fi networks were proposed [8]–[12].

Recently, methods for mobile networks (e.g., PathQuick3 [13], NEXT-FIT [14], and PathQuick-based TPG [15]) have been proposed. Since PathQuick3 is a successor of our previous methods PathQuick [16] and PathQuick2 [17], and we have already verified that PathQuick3 is more accurate over LTE networks than a representative method (pathChirp) [13], we compare our newly proposed method with PathQuick3 as a prior work in Section V to verify whether utilization of machine learning improves the estimation accuracy. In this paper, because the same packet train structure as PathQuick and PathQuick3 is used for our method, we describe the estimation mechanism of PathQuick and PathQuick3 in detail in Section III.

B. Prior Work Based on Machine Learning

There have been few studies on available bandwidth estimation using machine learning. To the best of our knowledge, only one method [18] was proposed using SVM [19]. This method forces users to have knowledge of the bottleneck physical capacity of a network a priori. However, on operational networks, because the bottleneck capacity is rarely known, this method is infeasible over operational networks. The performance evaluation was also limited to simulations; evaluation over operational networks was not performed. In contrast, we conducted a performance evaluation of our method over an operational LTE network. Moreover, our method can estimate available bandwidth without bottleneck physical capacity.

III. Overview of PathQuick3

Before going into the description of our machine learning-based method, we describe the estimation mechanism of PathQuick3, which uses the same packet train structure as our method. We also briefly describe the limitation of PathQuick3.

A. Basic Principle: Probe Rate Model

We describe one of the basic principles of available bandwidth estimation, called the Probe Rate model (PRM) [20]. PRM has been broadly utilized by prior works, including pathChirp, Pathload, and PTR, and also our own PathQuick3. In PRM, a sender transmits a UDP packet train to a receiver. The receiver then estimates the available bandwidth. PRM is based on the observation that (a) if the probing rate of a packet train at the sender is less than the available bandwidth, the probing packets will face no queuing delay inside the network, so the time interval for each probing packet observed at a receiver will be the same as at the sender, but (b) if the probing rate exceeds the available bandwidth, the packets will be queued inside the network, increasing the time intervals observed at the receiver. The available bandwidth can be estimated by observing the probing rate at which there is a transition from (a) to (b).

B. Probe Rate Model in PathQuick

1) Design of Packet Train Structure of PathQuick

As a concrete example of the PRM principle, we explain the estimation mechanism of PathQuick [16]. We designed the packet train structure of PathQuick for short estimation duration and probing over a wide range of rates as follows. In order to keep the entire transmission duration of a packet train short, the time interval for each packet within the packet train must be short. To this end, we designed the packet train so that each packet is placed at an equal time interval (Fig. 1). Also, in order to probe over a wide range of rates with a single packet train, the per-packet probing rate must be changed within the single packet train. To this end, we designed the structure so that each packet size linearly increases from the previous one as the packet sequence proceeds (Fig. 1).

Let us consider a packet train consisting of N probing packets. Each packet within the packet train is placed at equal time interval T_{quick} at the sender (Fig. 1). The entire transmission duration of a packet train (i.e., the packet train length) is $T_{train}^{(quick)} = T_{quick} \cdot (N-1)$. Thus, packet train length $T_{train}^{(quick)}$ is a linear function of the number of probing packets N.

The packet size of each probing packet is

$$P_i = P_1 + (i-1)\cdot \Delta P = \Delta P \cdot i + (P_1 - \Delta P), \tag{2}$$

where $i = 1, 2, \ldots, N$ and the constant value ΔP is the increase in the packet size (Fig. 1). Thus, each packet size P_i is a linear function of i, since P_1 and ΔP are constant values.

The per-packet probing rate at the i-th packet—i.e., the momentary probing rate of the packet train—is

$$R_i = \frac{P_i}{T_{quick}} = \frac{\Delta P}{T_{quick}} i + \frac{P_1 - \Delta P}{T_{quick}}. \tag{3}$$

Thus, each per-packet probing rate R_i is also a linear function of i.

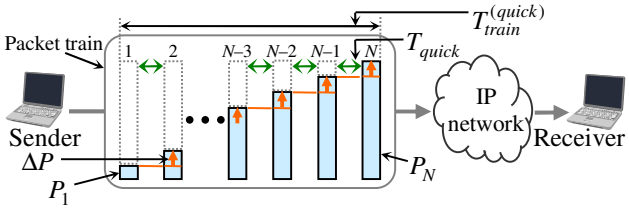

Fig. 1. Design of packet train structure.

2) PRM-based Available Bandwidth Estimation with Queuing Delays Observed at a Receiver

As discussed in Section III-A, finding the right transition point is essential in PRM. Here, we explain in detail how to find it in PathQuick.

In PathQuick, the transition point is identified by using the *queuing delay of each packet observed at a receiver* q_i $(i = 1, 2, \ldots, N)$. q_i is defined as

$$q_i = (r_i - s_i) - (r_1 - s_1), \tag{5}$$

where r_i is the receiving time of the i-th packet in the receiver clock and s_i is the transmission time of the i-th packet in the sender clock. Namely, q_i is the difference between the interval from the transmission of i-th packet s_i to reception r_i and the interval from the transmission of first packet s_1 to reception r_1 (Fig. 2). As described in Section III-A, (a) when the probing rate R_i is lower than the true available bandwidth A, probing packets are not queued inside the network, so the intervals from the transmission of a packet to the reception are almost constant. Thus, the difference q_i between them becomes almost zero. In contrast, (b) when probing rate R_i is higher than the true available bandwidth A, probing packets are queued inside the network, so the intervals from the transmission of a packet to the reception increase. Thus, q_i begins to increase. This can be rewritten as

$$\begin{aligned} &\text{(a) } q_i = 0, \quad \text{if } R_i \le A \\ &\text{(b) } q_i > 0, \quad \text{otherwise.} \end{aligned} \tag{6}$$

Therefore, as we increase probing rate R_i, $q_2 = q_3 = ,\cdots, = q_{k-1} = q_k = 0$ holds in the situation of (a) up to a certain k, and for $i > k$, $q_i > 0$ holds in the situation of (b). This k is the transition point of PRM and the per packet probing rate of the k-th packet is the estimated value of available bandwidth. For example, in the case shown in Fig. 2, the transition point k is 3, because $q_2 = q_3 = 0$ and $0 < q_4 < q_5 < q_6$. Thus, the estimated value of available bandwidth is $R_3 = P_3 / T_{quick}$.

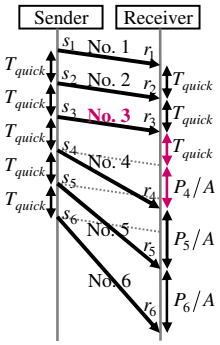

Fig. 2. Transmission time and receiving time of each probing packet.

C. Available Bandwidth Estimation over LTE networks

1) Disturbance of Queuing Delays by a Packet Scheduler

The estimation method described in Section III-B works well over wired networks [16]. However, this naive method does not work well over LTE networks [13], for the following reasons. In the LTE downlink, multi-path fading, interference with neighboring cells, and path loss (i.e., attenuation of signal) due to propagation distance lead to rapid time-varying radio quality of the wireless channel [21]. Moreover, due to the mobility of user equipment (UE), the number of UEs in the cell of a base station (i.e., an Evolved Node B, or eNB) is also time-varying. Many operational LTE networks utilize a packet scheduler (e.g., a proportional fair scheduler [22][23]) that takes the time-varying radio channel quality and number of UEs into account [24]. The packet scheduler periodically assigns radio resources, called resource blocks, and transmits packets with the assigned resource block to UEs for every transmission time interval (TTI). In LTE networks, the TTI is 1 ms [24], which means that the packet scheduler repeats (1) buffering and (2) transmission every 1 ms. Namely, it (1) *buffers* incoming packets from a wired packet core network and (2) transmits the buffered multiple packets at once, i.e., in a *bursty* manner. This behavior enables eNBs to dynamically adapt to the time-varying nature of the LTE network, and is effective for high throughput, low latency, and fairness among UEs. However, this behavior is quite harmful from the point of view of available bandwidth estimation using a probing packet train [13]. In short, this repetitious *stop-and-go* or *ON-OFF* behavior injects strong *burstiness* into the probing packets, resulting in severe disturbance of queuing delays observed at a receiver (i.e., q_i).

Fig. 3-(ii), a conceptual example, illustrates what happens when a packet train arrives at an eNB. We assume all per-packet probing rates are less than the actual available bandwidth, i.e., $R_i < A$. We also assume the time interval for each packet is 0.25 ms, and thus $s_i = s_1 + 0.25 \times (i-1)$. This means, if the size of a packet is 1,500 bytes, the probing rate of the packet is $8 \times 1{,}500 / 0.25 = 48$ Mbps. Since the 48-Mbps bandwidth is comparable to today's LTE downlink, the 0.25-ms time interval is a realistic assumption. At the eNB, the single packet train is split into multiple chunks, and multiple probing packets (four in this case) are collected on each chunk. At the receiver, the four packets arrive at the same time, so $r_2 = r_3 = r_4 = r_5$ and $r_6 = r_7 = r_8 = r_9$. Let us calculate the queuing delays of several packets with Eq. (5):

$q_2 = (r_2 - r_1) - (s_2 - s_1) = (r_2 - r_1) - ((s_1 + 0.25 \times 1) - s_1) = 1 - 0.25 \times 1 = 0.75,$
$q_5 = (r_5 - r_1) - (s_5 - s_1) = (r_5 - r_1) - ((s_1 + 0.25 \times 4) - s_1) = 1 - 0.25 \times 4 = 0,$
$q_6 = (r_6 - r_1) - (s_6 - s_1) = (r_6 - r_1) - ((s_1 + 0.25 \times 5) - s_1) = 2 - 0.25 \times 5 = 0.75,$
and thus $(q_2, q_3, q_4, q_5) = (q_6, q_7, q_8, q_9) = (0.75, 0.50, 0.25, 0)$.
Note that this means the cyclic expansion and contraction of queuing delays, i.e., cyclic ((b), (b), (b), (a)) in Eq. (6). In contrast, in a wired network (Fig. 3-(i)), $q_i = 0$ for all probing packets. This means it is always (a) in Eq. (6).

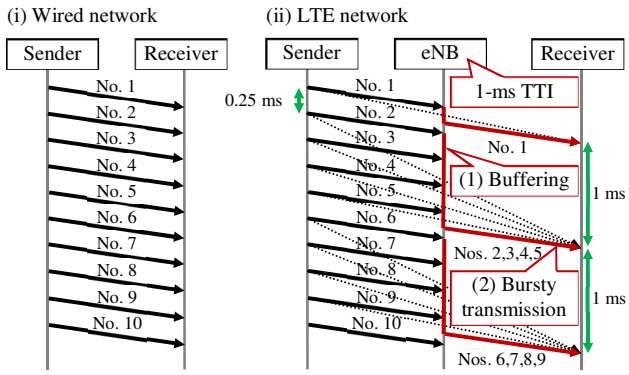

Fig. 3. Disturbance of queuing delays in LTE networks. The black and red solid arrows show the actual packet flow, i.e., white box view to network nodes, while the gray dotted arrows show the packet flow in an end-to-end black box view to network nodes.

Fig. 4-(i) shows the queuing delays of ten packet trains of PathQuick obtained from a private wired network without cross-traffic. We designed the packet trains so that the per-packet

probing rates near the tail of the packet train exceeded the actual available bandwidth. There is no cross-traffic, so each of the ten queuing delays are quite similar. Since these ten queuing delays strictly agree with Eq. (6), we can determine the exact transition point of PRM easily: it is the 46th or 47th packet, depending on the packet trains.

Fig. 4-(ii) shows the queuing delays of a single packet train of PathQuick obtained from an operational LTE network. We designed the packet trains so that the per-packet probing rates near the head of the packet train exceeded the actual available bandwidth. In contrast to Fig. 4-(i), due to the repetitious stop-and-go behavior of the packet scheduler, queuing delays repeat stretch and shrink behavior. Consequently, the shape of the queuing delays becomes like the teeth of a saw, and thus the position of the transition point of PRM seems quite indistinct.

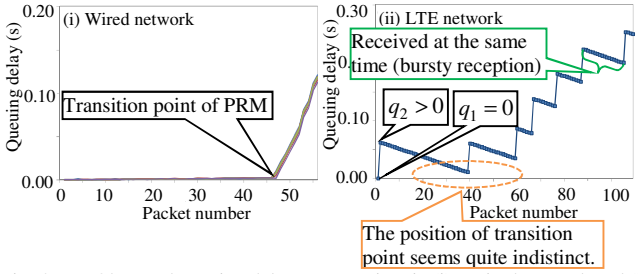

Fig. 4. Observed queuing delays at a receiver in (i) a wired network and (ii) an LTE network.

2) Estimation Method of PathQuick3

In PathQuick3, the simplified model is constructed to deal with the difficulty of identifying the transition point. This model is intended to reduce the harmful effects of stretch and shrink behavior on estimation. At the *microscopic view*, queuing delays repeat stretch and shrink behavior (as mentioned in Section III-C-1). However, this model also utilizes the *macroscopic view*, and assumes that the queuing delay is small until the transition point. It also assumes that the queuing delay increases after the transition point. More specifically, the shape of queuing delays is the *horizontal line* of $q_i = 0$ until the transition point, and the shape changes to *parabola* after the transition point because the delays *accumulate*. This means that queuing delay q_i is a function of i, where a horizontal line (if $i \le k$) and a parabola (if $i > k$) are connected at joint point k. PathQuick3 utilizes this change of the shape of queuing delays to identify the transition point. Fig. 5 shows the gray curves where joint point k moves on from left (i.e., $k = 1$) to right (i.e., $k = N$). The number of curves is N. These are called *ideal curves*. The detailed shape of ideal curves is described in [13]. With these ideal curves, the transition point is identified in the following way:

(1) Calculate squared error between an ideal curve and observed queuing delays for all ideal curves.

(2) Select an ideal curve that had minimum error in (1).

(3) Identify the transition point with the joint point of the selected curve.

Available bandwidth is estimated from the identified transition point. This method can identify the transition point and estimate available bandwidth well even if queuing delays are distorted, as in Fig. 5-(ii).

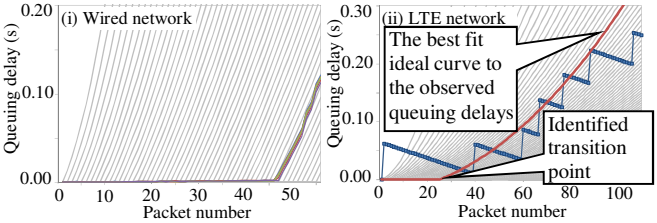

Fig. 5. Ideal curves of queuing delays (gray), overlapped with observed queuing delays in (i) wired network and (ii) LTE network.

3) Limitation of PathQuick3

In PathQuick3, it is assumed that the shape of the observed queuing delays is the horizontal line of $q_i = 0$ until the transition point and then become parabola at a macroscopic view. However, in reality, different behaviors are observed over operational networks and the estimation accuracy decreases in such cases. Fig. 6 shows the queuing delays of a packet train observed over an operational LTE network. The queuing delay of the second packet q_2 is already not equal to zero and becomes a horizontal line $q_i = c(\neq 0)$ at a macroscopic view. This behavior is outside the scope of PathQuick3. (Determining the cause of this behavior, which is currently unknown, will be the focus of our future work.) On the basis of results from a speed test (the relation between speed test and available bandwidth is mentioned in Section V-A-1)), we expect the transition point to be $k \sim 115$ th packet (estimated value: 39.0 Mbps). However, PathQuick3 estimates the transition point at $k = 48$ (estimated value: 17.4 Mbps), and so the estimation accuracy gets worse.

As seen above, over operational networks, complex behaviors that are outside the scope of the simplified model may be observed. The estimation algorithm does not work well in such cases, especially if there are unexpected queuing delays, and thus the estimation accuracy gets worse.

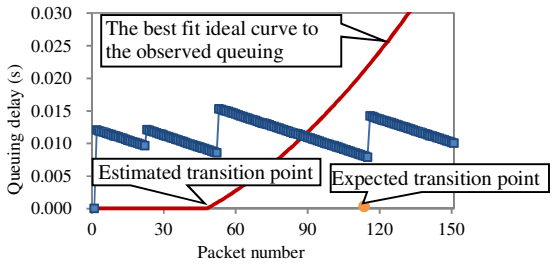

Fig. 6. Identified transition point from an unexpected queuing delay is far from the transition point expected from the speed test.

IV. Proposed Method

We propose PathML, an available bandwidth estimation method based on the data-driven paradigm using machine learning. Our method aims to deal with the complex behavior of queuing delays over operational networks and estimate available bandwidth accurately by machine learning with a large amount of data.

Note that proposed method can be used not only with PathQuick and PathQuick3, but also with estimation methods which have different packet train structures such as pathChirp and Pathload.

A. *Outline of Procedure of Proposed Method*

In our method, we generate a predictor using a machine learning algorithm (Fig. 7) before estimation (Fig. 8). The procedure for this is as follows. First, we prepare plenty of pairs of observed queuing delays at a receiver (Fig. 7-(1-1)) and corresponding true value of available bandwidth (Fig. 7-(1-2)) as a training dataset. Second, a predictor is generated by machine learning techniques with this training dataset to predict the true value of available bandwidth from observed queuing delays (Fig. 7-(2)). The predictor learns patterns in the relationship between queuing delays and available bandwidth, e.g., a transition point at a small packet number implies low available bandwidth and a transition point at a large packet number implies high available bandwidth. By inputting new queuing delays (Fig. 8-(3)) into the predictor, we can obtain an estimated value (Fig. 8-(4)). The specific machine learning techniques that we selected are described in the next subsection.

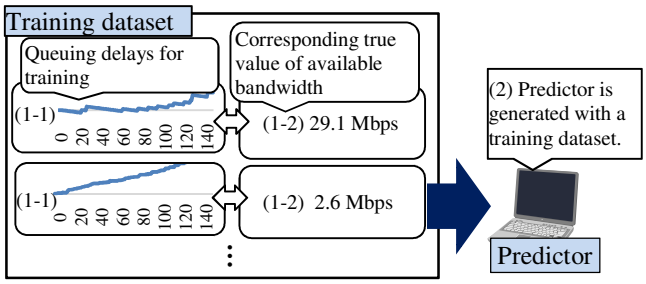

Fig. 7. (2) Generation of predictor using pairs of (1-1) observed queuing delays and (1-2) corresponding true value of available bandwidth.

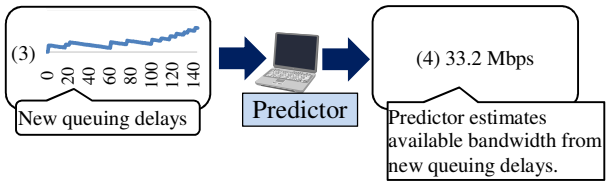

Fig. 8. By (3) inputting new queuing delays, predictor outputs (4) estimated value.

B. *Selected Machine Learning Techniques for Available Bandwidth Estimation*

In general, machine learning techniques are classified into two types: (1) unsupervised learning, which includes clustering techniques such as the k-means algorithm and (2) supervised learning, which aims to predict an output corresponding to an input with training data consisting of pairs of input and output values such as SVM and neural networks.

We used (2) supervised learning because our task is to estimate available bandwidth (output) from queuing delays (input). We selected four machine learning algorithms that are suitable for available bandwidth estimation: 1) support vector regression (SVR), 2) kernel ridge regression (KRR), 3) random forests (RF), and 4) convolutional neural network (CNN). We implemented our available bandwidth estimation system with these four techniques by using state-of-the-art machine learning libraries: Scikit-learn [25] and TensorFlow [26]. We provide an overview of these four techniques and the reasons for their selection below.

1) Support Vector Regression (SVR)

SVR [19] using the kernel method is a regression technique for nonlinear functions in which linear regression is performed in high dimensional kernel-induced feature space. We selected SVR because it is used in various fields and shows good performance [27][28]. The proposed method differs from prior work [18] in that our method does not require bottleneck capacity to estimate available bandwidth. We used the radial basis function (RBF) kernel (Gaussian kernel) and optimized hyper-parameters using grid search with 10-fold cross validation [29].

2) Kernel Ridge Regression (KRR)

KRR [30] is similar to SVR. The predictor of KRR makes it necessary to store more data than SVR, but the error can be smaller [31], and the number of hyper-parameters is lower, which makes it easier to control. This is why we selected KRR. We used the RBF kernel and optimized hyper-parameters using grid search with 10-fold cross validation in a manner similar to SVR.

3) Random Forests (RF)

RF [32] is an ensemble learning method. Many decision trees are constructed in training and the output is obtained by combining the outputs of all the trees. Each tree is trained by different data and thus is predicted differently. A prediction is obtained by calculating the mean of outputs of all trees. We selected RF because it can perform as well as SVM but at cheaper computational cost [33].

4) Convolutional Neural Network (CNN)

Due to their high recognition ability, humans can find the approximate position of the transition point in Fig. 5-(ii) by recognizing the macroscopic trend of the queuing delays. For this reason, we felt we could estimate the available bandwidth more accurately by treating queuing delays not as *time-series* like the PRM-based available bandwidth estimation mentioned in Section III-A but as *images* and then *recognizing the patterns*. CNN, a machine learning technique inspired by visual neuroscience [34], has recently achieved record-breaking results in image recognition [35]. In general, CNN is designed to extract *macroscopic* features in deep layers by combining *microscopic* features in shallow layers [34]. We selected CNN because we felt CNN would be able to estimate the available bandwidth accurately by utilizing both *microscopic* features (e.g., a shape like saw teeth, Fig. 5-(ii)) and the *macroscopic* features (trends) of queuing delays.

In this study, we designed a 6-layer CNN that has two fully connected layers and two pairs of convolutional and pooling layers for available bandwidth estimation.

V. Performance Evaluation

We evaluated the available bandwidth estimation of the four machine learning techniques and compared it with PathQuick3 as a conventional method in terms of mean absolute error (MAE) and estimation accuracy (defined later). We also investigated the amount of data needed to obtain sufficient accuracy for particularly promising techniques.

A. *Experimental Setup*

1) Ground Truth of Available Bandwidth

Since we cannot access the network nodes of the mobile operator directly, the *ground truth* of the available bandwidth is

unknown to us. Instead, although available bandwidth and bulk TCP throughput are not the same network metric [36], we follow [37] as our precedent and treat bulk TCP throughput as a *reference* to the ground truth (or *best effort* ground truth [37]). We obtained bulk TCP throughput with a well-known speed test application in Japan [38].

2) The Structure of Packet Train

For a fair comparison, we used the same packet train structure for PathQuick3 and the four machine learning techniques.

We chose the probable bandwidth range of PathQuick3 as follows. The current fastest average LTE downlink speed in the world is 37 Mbps [39], so for sufficient coverage, we chose 50 Mbps as the maximum probable bandwidth of the probing packet train of PathQuick3.

To realize this 50-Mbps target with PathQuick3, we set the packet size of the first packet $P_1 = 60$ bytes, the increase of the amount of packet size $\Delta P = 8$ bytes, the number of packets in a packet train $N = 151$, and the equal time interval $T_{quick} = 0.2$ ms. Therefore, the packet size of the last packet $P_N = 60 + 8 \times (151 - 1) = 1{,}260$ bytes and thus the maximum probable bandwidth is $P_N / T_{quick} = 8 \times 1{,}260 / (0.2 \times 10^{-3}) = 50.4$ Mbps. Then, the minimum probable bandwidth with PathQuick3 becomes $P_2 / T_{quick} = 8 \times (60 + 8) / (0.2 \times 10^{-3}) = 2.7$ Mbps.

3) System Setup

We used an Android smartphone (SO-03F [40]) for the packet train receiver. A Linux server (quad-core 1.7 GHz CPU, 8 GB RAM, Ubuntu 14.04) with a 1-Gbps FTTH connection was deployed for the packet train senders (Fig. 9).

The experiment was performed at diverse locations in Tokyo and Kanagawa. At each location, we (1) ran a downlink speed test once and (2) received probing packet trains of PathQuick3.

We obtained 6,532 pairs of queuing delays and speed test and randomly divided the pairs into $n_{training} = 5{,}000$ pairs of training data and $n_{test} = 1{,}532$ pairs of test data. We used the training data for the learning algorithms in our method (Fig. 7). We used the test data as new queuing delays (Fig. 8-(3)) and evaluated the estimation accuracy and error of the proposed machine learning method and PathQuick3. We did not use the training data for PathQuick3, since PathQuick3 is based on the model of network behavior and does not need to be trained.

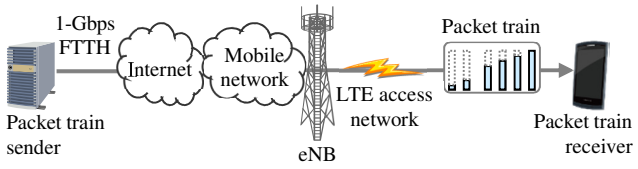

Fig. 9. Experimental environment over an operational LTE network.

B. Experimental Results

1) Estimation Accuracy

After training the four machine learning techniques with the training data consisting of $n_{training} = 5{,}000$ pairs of queuing delays and speed test, we evaluated the estimation accuracy and error of PathQuick3 (PQ3), SVR, KRR, RF, and CNN with test data consisting of $n_{test} = 1{,}532$ pairs of queuing delays and speed test. The estimation accuracy for evaluation is defined with the aid of mean absolute percentage error (MAPE), which is the most commonly used metric for percentage error [41]. In concrete terms, estimation accuracy is defined as $(100 - (\text{MAPE}))[\%]$. This is represented as

$$100 \times \left(1 - \frac{1}{n_{test}} \sum_j \left| \frac{S_j - E_j}{S_j} \right| \right),$$

where S_j is the value of speed test and E_j is the estimated available bandwidth $(j = 1, 2, \ldots, n_{test})$.

Fig. 10 shows the estimation accuracy of each method. Our method with machine learning techniques outperformed PathQuick3 in terms of estimation accuracy. As shown in Fig. 10, the estimation accuracy of PQ3 was 74.1% while those of SVR and KRR were comparable at a little less than 80% and those of CNN and RF were higher than 80%.

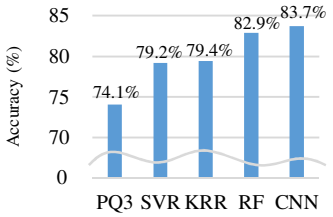

Fig. 10. Every machine learning technique outperformed PathQuick3 in terms of estimation accuracy. The accuracies of SVR and KRR were comparable at a little less than 80% and those of CNN and RF were higher than 80%.

2) Analysis of Speed Test vs. Estimated Available Bandwidth

For a more detailed analysis, we replotted graphs as shown in Fig. 11. These graphs show the values of the speed test (Mbps) on the horizontal axis and the estimated available bandwidth (Mbps) on the vertical axis. The closer the red points are distributed to the oblique line, the higher the estimation accuracy is. For PQ3, the red points of the right side of the graph distributed under the oblique line indicate that PathQuick3 underestimated the available bandwidth when the true available bandwidth was high. In contrast, the machine learning techniques were able to improve this underestimation.

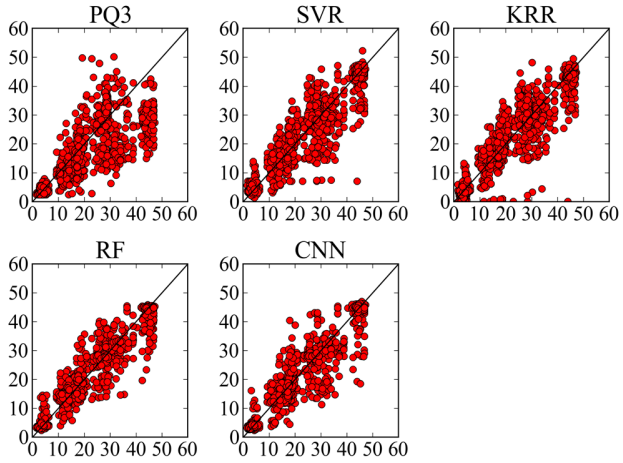

Fig. 11. Values of speed test (Mbps) on horizontal axis and estimated available bandwidth (Mbps) on vertical axis. The closer the red points are distributed to the oblique line, the higher the estimation accuracy is.

3) Estimation Errors vs. the Range of Available Bandwidth

Improvement of underestimation can be seen more clearly in Fig. 12, which shows the evaluated estimation error by the range of available bandwidth. The mean absolute errors (MAE) of estimation by each method when the corresponding speed tests were 0–10 Mbps, 10–20 Mbps, 20–30 Mbps, 30–40 Mbps, and 40–50 Mbps are plotted in Fig, 12. In the range of 0–10, 10–20, 20–30, 30–40, and 40–50 Mbps, there are $\{n_{test\_r}\} = \{454, 481, 250, 116, 231\}$ pairs, respectively. The mean absolute error is defined as

$$\frac{1}{n_{test\_r}} \sum_j |S_j - E_j|.$$

Although it may seem from Fig. 12 that RF is more accurate than CNN, since RF showed less error than CNN in the range of 30–40 Mbps, we know from Fig. 10, which shows the estimation accuracy of the entire bandwidth range, that CNN is actually more accurate. This is because, in this experiment, we obtained fewer pairs whose speed test was in the range of 30–40 Mbps than those in the range of others, and thus the other four ranges dominated the influence on the estimation accuracy of the entire bandwidth range. In Fig. 12, the MAE of PathQuick3 was high when the corresponding available bandwidth was high (30–50 Mbps), but our method with machine learning techniques did not show such tendency and tended to show less MAE than PathQuick3 in each range. Specifically, when the available bandwidth was high (40–50 Mbps), the estimation accuracy of PathQuick3 decreased and its mean absolute error (MAE) was 16.4 Mbps, whereas the MAE of our method using CNN was 2.2 Mbps, or just 13.2% (only about 1/8) that of PathQuick3.It seems that unexpected behavior for the model of PathQuick3 occurred frequently when the available bandwidth was high, so the ideal curves of PathQuick3 do not fit with the observed queuing delays like in Fig. 6. Our method with machine learning techniques was able to extract information related to true available bandwidth in such cases. As a specific example, for the queuing delay shown in Fig. 6, estimation accuracy of PathQuick3 was 44.6%, which is much lower than the accuracy of the entire bandwidth range of PathQuick3 (74.1% in Fig. 10), but the estimation accuracy of CNN was 82.7% (estimated value: 32.3 Mbps), which is similar to the accuracy of the entire bandwidth range of CNN (83.7% in Fig. 10).

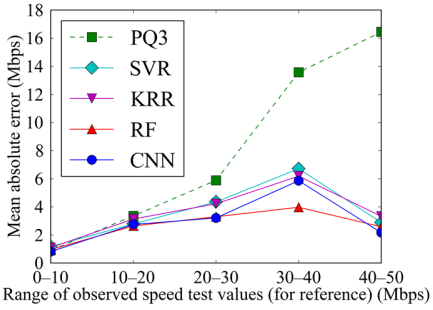

Fig. 12. The range of speed test values on horizontal axis and mean absolute errors on vertical axis.

C. The Required Amount of Data for Sufficient Estimation Accuracy

If the amount of training data (Fig. 7) is too small, the machine learning techniques cannot obtain sufficient estimation accuracy. Therefore, we investigated sufficient amount of data to obtain higher estimation accuracy than PathQuick3 for RF and CNN, which were the two most promising techniques (as discussed in Section V-B).

Fig. 13 shows the average estimation accuracy when increasing the amount of training data. The accuracies were calculated as follows. First, (a) we randomly chose the same amount of data shown on the horizontal axis (39, 78, 156, 312, 625, 1250, 2500, 5000) from the training data consisting of $n_{trainings} = 5{,}000$ pairs for machine learning. Then, (b) we generated predictors using the chosen data and evaluated estimation accuracy for test data consisting of $n_{test} = 1{,}532$ pairs (regardless of the amount of data chosen in (a), the estimation accuracy was evaluated for the same test data). If we evaluate the estimation accuracy by executing (a) and (b) only once, the estimation accuracy is highly affected by the randomness of chosen data in (a), so we executed (a) and (b) fifty times and plot the average accuracy to reduce the effect of randomness. The estimation accuracy of PathQuick3 (which needs no training data) is indicated by the green horizontal line (74.1%) for a baseline comparison.

Results showed that when we had training data consisting of 625 or more pairs, both RF and CNN improved the estimation accuracy by more than five points compared to PathQuick3, and the accuracies exceeded 80%. In addition, the estimation accuracy of RF and CNN continued to increase when the amount of training data was increased above 625. If we can obtain larger amounts of data hereafter, the estimation accuracy could improve even more.

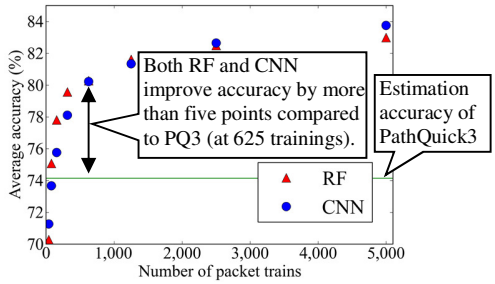

Fig. 13. The average estimation accuracy when increasing the amount of training data.

VI. Conclusion and Future Work

We proposed PathML, a machine learning-based available bandwidth estimation and conducted an experimental comparison with a conventional model-based method (PathQuick3 [13]) over an operational LTE downlink. Results showed that the proposed machine learning method clearly outperformed PathQuick3 in terms of estimation accuracy.

For future work, we plan to prepare larger amounts of training data and examine how much the estimation accuracy increases. We also plan to investigate the estimation accuracy if we use packet trains of PathQuick which have different packet train structures, while we used a single packet train structure in this paper. Additionally, we also plan to perform experiment of

PathML with estimation methods which have different packet train structures such as pathChirp and Pathload. Moreover, we will examine the robustness of PathML in situations that the type of packet scheduler, true available bandwidth, and/or the number of users vary. We also plan to conduct simulations (e.g., with ns-3 [42]) to confirm that treating bulk TCP throughput as the ground truth of available bandwidth is valid.

We believe our method can be widely deployed in the real world by mobile operators. These days it is common for mobile operators to collect data from their widely distributed smartphone applications. They can embed our method into their smartphone applications, collect data of queuing delays and global positioning system (GPS) data on cloud servers as big data, and thus deploy large-scale available bandwidth estimation system which covers their entire mobile networks. As a results, they can identify which locations are weak points of their mobile networks by using the system, and thus improve their mobile networks efficiently by installing eNBs into the weak points preferentially.

ACKNOWLEDGMENTS

Natsuhiko Sato was supported by the Japan Society for the Promotion of Science through a Program for Leading Graduate Schools (MERIT).

REFERENCES

[1] A. Frömmgen, J. Heuschkel, P. Jahnke, F. Cuozzo, I. Schweizer, P. Eugster, M. Mühlhäuser, and A. Buchmann, "Crowdsourcing measurements of mobile network performance and mobility during a large scale event," *PAM*, pp. 70–82, 2016.

[2] M. Lease, "On quality control and machine learning in crowdsourcing," *AAAI Workshops*, pp. 97–102, 2011.

[3] Nokia Solutions and Networks White Paper, "What is going on in mobile broadband networks?," 2014, [online] http://networks.nokia.com/system/files/document/nokia_smartphone_traffic_white_paper.pdf (accessed February 18, 2016).

[4] R. Prasad, C. Dovrolis, M. Murray, and K. Claffy, "Bandwidth estimation: Metrics, measurement techniques, and tools," *IEEE Network*, Vol. 17, Issue 6, pp. 27–35, 2003.

[5] V. J. Ribeiro, R. H. Riedi, R. G. Baraniuk, J. Navratil, and L. Cottrell, "pathChirp: Efficient available bandwidth estimation for network paths," *PAM Workshop*, 2003.

[6] M. Jain and C. Dovrolis, "End-to-end available bandwidth: Measurement methodology, dynamics, and relation with TCP throughput," *ACM SIGCOMM*, pp. 295–308, 2002.

[7] N. Hu and P. Steenkiste, "Evaluation and characterization of available bandwidth probing techniques," *IEEE JSAC*, Vol. 21, No. 6, pp. 879–894, 2003.

[8] M. Li, M. Claypool, and R. Kinicki, "WBest: A bandwidth estimation tool for IEEE 802.11 wireless networks," *IEEE LCN*, pp. 374–381, 2008.

[9] M. Portoles-Comeras et al., "Impact of transient CSMA/CA access delays on active bandwidth measurements," *ACM IMC*, pp. 397–409, 2009.

[10] A. Farshad et al., "On the impact of 802.11n frame aggregation on end-to-end available bandwidth estimation," *IEEE SECON*, pp. 108–116, 2014.

[11] A. Johnsson and M. Björkman, "On measuring available bandwidth in wireless networks," *IEEE LCN*, pp. 861–868, 2008.

[12] H. K. Lee et al., "Bandwidth estimation in wireless LANs for multimedia streaming services," *IEEE ICME*, pp. 1181–1184, 2006.

[13] T. Oshiba, K. Nogami, K. Nihei, and K. Satoda, "Robust available bandwidth estimation against dynamic behavior of packet scheduler in operational LTE networks," *IEEE ISCC*, pp. 1276–1283, 2016.

[14] A. K. Paul, A. Tachibana, and T. Hasegawa, "NEXT-FIT: Available bandwidth measurement over 4G/LTE networks—A curve-fitting approach," *IEEE AINA*, pp. 25–32, 2016.

[15] S. Shiobara and T. Okamawari, "A novel available bandwidth estimation method for mobile networks using a train of packet groups," *ACM IMCOM*, pp. 1–7, 2017.

[16] T. Oshiba and K. Nakajima, "Quick end-to-end available bandwidth estimation for QoS of real-time multimedia communication," *IEEE ISCC*, pp. 162–167, 2010.

[17] T. Oshiba and K. Nakajima, "Quick and simultaneous estimation of available bandwidth and effective UDP throughput for real-time communication," *IEEE ISCC*, pp. 1123–1130, 2011.

[18] L. Chen, C. Chou, and B. Wang, "A machine learning-based approach for estimating available bandwidth," *IEEE TENCON*, pp. 1–4, 2007.

[19] A. J. Smola and Bernhard Schölkopf, "A tutorial on support vector regression," *Statistics and Computing*, Vol. 14, Issue 3, pp. 199–222, 2004.

[20] L. Lao, C. Dovrolis, and M. Y. Sanadidi, "The probe gap model can underestimate the available bandwidth of multihop paths," *ACM SIGCOMM CCR*, Vol. 36, Issue 5, pp. 29–34, 2006.

[21] A. Larmo et al., "The LTE link layer design," *IEEE Communications Magazine*, Vol. 47, Issue 4, pp. 52–59, 2009.

[22] K. Winstein, A. Sivaraman, and H. Balakrishnan, "Stochastic forecasts achieve high throughput and low delay over cellular networks," *USENIX NSDI*, pp. 459–471, 2013.

[23] Q. Xu, S. Mehrotra, Z. M. Mao, and J. Li, "PROTEUS: Network performance forecast for real-time, interactive mobile applications," *ACM MobiSys*, pp. 347–360, 2013.

[24] F. Capozzi, G. Piro, L. A. Grieco, G. Boggia, and P. Camarda, "Downlink packet scheduling in LTE cellular networks: Key design issues and a survey," *IEEE Communications Surveys & Tutorials*, Vol. 15, Issue 2, pp. 678–700, 2013.

[25] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, Vol. 12, pp. 2825–2830, 2011.

[26] M. Abadi et al., "TensorFlow: A system for large-scale machine learning," *USENIX OSDI*, pp. 265–283, 2016.

[27] J. Lee, A. Seko, K. Shitara, K. Nakayama, and I. Tanaka, "Prediction model of band gap for inorganic compounds by combination of density functional theory calculations and machine learning techniques," *Phys. Rev. B*, Vol. 93, Issue 11, 115104, pp. 1–12, 2016.

[28] Y. Li, S. Gong, and H. Liddell, "Support vector regression and classification based multi-view face detection and recognition," *IEEE FG*, pp. 300–305, 2000.

[29] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," *IJCAI*, Vol. 14, No. 2, pp. 1137–1145, 1995.

[30] C. Saunders, A. Gammerman, and V. Vovk, "Ridge regression learning algorithm in dual variables," *ICML*, pp. 515–521, 1998.

[31] L. Wang, L. Bo, and L. Jiao, "Sparse kernel ridge regression using backward deletion," *PRICAI*, pp. 365–374, 2006.

[32] L. Breiman, "Random forests," *Machine Learning*, Vol. 45, Issue 1, pp. 5–32, 2001.

[33] A. Bosch, A. Zisserman, and X. Munoz, "Image classification using random forests and ferns," *IEEE ICCV*, pp. 1–8, 2007.

[34] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, Vol. 521, pp. 436–444, 2015.

[35] O. Russakovsky et al., "ImageNet large scale visual recognition challenge," *IJCV*, Vol. 115, Issue 3, pp. 211–252, 2015.

[36] M. Jain and C. Dovrolis, "Ten fallacies and pitfalls on end-to-end available bandwidth estimation," *ACM IMC*, pp. 272–277, 2004.

[37] D. Koutsonikolas and Y. C. Hu, "On the feasibility of bandwidth estimation in wireless access networks," *Wireless Networks*, Vol. 17, Issue 6, pp. 1561–1580, 2011.

[38] NTT DOCOMO, "DOCOMO SPEED TEST," *Google Play*. [online] https://play.google.com/store/apps/details?id=jp.co.nttdocomo.areainfo

[39] OpenSignal, "The state of LTE (February 2016)." [online] https://opensignal.com/reports/2016/02/state-of-lte-q4-2015/

[40] Sony Mobile Communications, "Xperia Z2 SO-03F." [online] http://www.sonymobile.co.jp/xperia/docomo/so-03f/

[41] R. J. Hyndman, "Another look at forecast-accuracy metrics for intermittent demand," *Foresight*, Vol. 4, Issue 4, pp. 43–46, 2006.

[42] G. Piro, N. Baldo, and M. Miozzo, "An LTE module for the ns-3 network simulator," *ICST SIMUTools*, pp. 415–422, 2011.