On The Accuracy of TCP Throughput Prediction for Opportunistic Wireless Networks

Mariyam Mirza*, Kevin Springborn†, Suman Banerjee*, Paul Barford*†, Michael Blodgett*, Xiaojin Zhu*
*University of Wisconsin-Madison, (mirza, suman, pb, mblodget, jerryzhu)@cs.wisc.edu
†Nemean Networks, springbo@nemeannetworks.com

Abstract—The increasing density of WiFi Access Points (APs) in metropolitan areas is enabling an opportunistic model of wireless networking, whereby a "guest" user within range of one or more wireless APs can gain temporary Internet access through these APs. In this paper, we address the problem of TCP throughput prediction for opportunistic networks. Applications of opportunistic networking can benefit from such predictions by adapting to prevailing network conditions. Our approach is different from prior efforts to model wireless network throughput in that only the two communicating endpoints participate in the prediction, and no information about network topology or traffic loads generated by interfering sources is required. Our goal is to understand how accurate throughput predictions can be under the above assumptions. The physical environment considered in our study includes varying degrees of interference, indoor and outdoor networks, and nodes that are stationary or moving at walking or driving speeds. We use throughput predictors based on time series analysis and machine learning techniques, as they are well-suited to predicting phenomena with unknown variables. The prediction accuracy that our methods yield is cause for cautious optimism. We find that 80% to 100% of predictions are within a factor of two of actual throughput. This bound on accuracy means that predictions are useful for certain applications, because this bound *(a)* can be achieved by measurements lasting for as little as 0.3 seconds, and *(b)* holds even when nodes are driving at speeds of 15-25 mph.

I. Introduction

Over the past decade, there has been a great increase in the deployment of WiFi Access Points (APs) in homes, neighborhoods, and public areas. The increasing density of WiFi APs is beginning to enable a new model of wireless network connectivity, where a guest user passing through the range of one or more APs can gain temporary Internet access. Users of these *opportunistic networks* can be stationary or mobile, possibly vehicular. A common example of a stationary opportunistic network is *Fon*, (www.fon.com). *Fon* hosts allow *Fon* guests Internet access when the guests are within range of the hosts' APs, in exchange for Internet access via the guests' APs at the guests' home location. The goal of this network access exchange is to eventually enable *Fon* users to have network connectivity worldwide. An example of a vehicular opportunistic network is the *CarTel* project [1], which has studied the feasibility of vehicular opportunistic networking by measuring how much connectivity and bandwidth is available to vehicles driving through a metropolitan area. The *CarTel* project is currently exploring a variety of vehicular networking applications such as using vehicular nodes to collect traffic information at a central server for traffic-aware route planning, and using cars as *data mules* to allow communication between nodes that would otherwise not be connected, such as sensor networks deployed in the field and servers that analyze the sensor data.

A forecast of the throughput an application can expect in a given opportunistic wireless environment can be quite valuable. It can help adaptive applications adjust their parameters according to prevailing network conditions. It can tell the guest user whether bandwidth-intensive and time-sensitive applications such as voice over IP and live video will be able to operate successfully in a given network environment. In dense wireless environments, knowledge of expected throughput can allow clients to select the best AP, *i.e.*, the highest throughput AP, if more than one AP is within range. TCP throughput prediction can provide the guest user with the TCP-friendly rate, which is useful for limiting non-TCP traffic that the guest can introduce to levels that are not disruptive for the network. This is an important application, because bounding guest traffic to benign levels is essential to the success of opportunistic networks. Guest users have an incentive to respect this bound, because if they do not, it is unlikely that APs will allow the open access required for opportunistic networking, and the opportunistic networking paradigm will fail.

Accurate TCP throughput prediction in wireless environments is challenging for several reasons. First, physical and MAC layer behavior has a significant impact on observed TCP performance. Next, mobile clients that move through an area can introduce complex interference dynamics. Finally, the short time span over which predictions in opportunistic networks must be made, and sensitivity to the impact of measurement, further complicate the problem.

A number of recent studies, such as [2]–[6], have considered the problems of modeling capacity and achievable throughput on both single-hop and multi-hop wireless paths. To predict wireless throughput in a given network, these approaches assume knowledge of the network topology, such as the number of interfering sources, their traffic demands, and the quality of the channel between the inter-

fering sources. Our work is different from these studies in that we assume no direct knowledge regarding the sources of interference in the network. In any real network, this information cannot be obtained without extensive passive monitoring (e.g., [7]) and/or active measurements between the AP and sources of interference (e.g., [6]) to determine link quality. Neither capability is available in opportunistic networks, so our prediction method relies only on communication between the AP and the wireless client that requests the prediction. Other studies often make simplifying assumptions about the environment, such as disabling auto rate fallback, using constant bit rate traffic instead of TCP and limiting the number of sources of interference: we make no such assumptions.

In general, high prediction accuracy, rapid generation of predictions in new conditions, and low measurement overhead are the goals of TCP throughput prediction. However, these goals are often in conflict: high accuracy can be achieved by using heavy-weight measurements or taking measurements for a longer period of time. Specifically, rapid generation of accurate predictions is critically important for both stationary and mobile opportunistic wireless environments. The time required to generate predictions will be part of an application's latency, so for any interactive application, a latency of at most 2-3 seconds is acceptable: if the selection takes much longer, the user could get frustrated by the delay. The time constraints for generating throughput predictions for mobile applications are even more stringent because the mobile client may spend very little time within the range of an AP. For example, if the mobile client is traveling at 30 mph and is within the range of an AP for 400 ft, the total time within the range of the AP is about 10 seconds. To be practical, *i.e.*, to be useful to an application, the prediction would have to be generated in considerably less than 10 seconds.

We consider the problem of TCP throughput prediction empirically by taking measurements of wireless client and AP interactions over a broad range of operating conditions including varying number of APs in the network, conducting experiments indoors and outdoors, with client stationary or moving at walking or driving speeds. In each case, third party interference is created by systems running load generators that emulate web-like, bursty cross traffic. We use two standard methods for generating predictions based on operating conditions. The first is Exponentially Weighted Moving Averages (EWMA) which is a standard time series forecast. The second is Support Vector Regression (SVR), which is a machine learning-based approach.

The prediction accuracy that our methods yield is cause for cautious optimism. We find that for all wireless environment configurations and all prediction techniques used, wireless throughput prediction accuracy is lower than wireline prediction accuracy reported in the literature. We find in most of our experiments that only 10% to 30% of predictions are within 10% of actual throughput. In comparison, it has been reported in the literature that for wireline networks, 50% or more of predictions are within 10% of actual: this is a factor of 2 to 5 better than wireless prediction accuracy. However, we also find that 80% to 100% of predictions are within a factor of two of actual throughput. The factor of two bound on accuracy means that predictions are useful for certain (but not all) applications, specially because this bound can be achieved by measurements lasting for as little as 0.3 seconds, and because this bound holds even when nodes are driving at speeds of 15-25 mph.

In summary, we make the following contributions:

- We consider the problem of TCP throughput prediction empirically in a diverse set of wireless environments without making simplifying assumptions.
- We develop two predictors for TCP throughput forecasting, one based on time series analysis and the other based on machine learning.
- We provide empirical upper and lower bounds on the TCP throughput prediction accuracy that can be achieved using our methods.
- We provide evidence that the inherent dynamism of the wireless environment limits the level of prediction accuracy that can be achieved in opportunistic networks.

The remainder of this paper is organized as follows. In Section II, we discuss studies related to our own. We describe our predictions mechanisms, Exponentially Weighted Moving Averages and Support Vector Regression, in Section III. We present details of our indoor and outdoor experimental environments in Section IV. In Section V, we describe what measurements we take and how we use them to construct predictors. In Section VI we present the results of our experiments and their implications. We summarize and conclude in Section VII.

II. RELATED WORK

Wireless network performance is a widely-studied area. Seminal work by Gupta and Kumar [8] presented an asymptotic analysis of the capacity of general wireless networks. Bianchi [9] focused on 802.11-based wireless networks, analytically predicting system throughput based on 802.11 MAC parameters such as the minimum and maximum backoff windows. Work by Burmeister *et al.* in [10], [11] models wireless network performance analysis as a queuing theory problem and considers both TCP and non-TCP data transfers. Gopalakrishnan *et al.* [12] predict wireless throughput using a combination of analytical and measurement approaches. Another approach to wireless throughput prediction is through an understanding of interference in the wireless network: [2]–[5] use either analytical or measurement-based interference models to predict wireless throughput. All these studies differ from ours because they compute achievable throughput when the number and location of interfering sources, and their intended traffic

demands, are known. Kashyap *et al.* [6] study more general network configurations, however their traffic model is based on PHY and MAC layer characteristics and they do not consider TCP traffic. All studies mentioned above consider only stationary nodes while our study addresses TCP throughput prediction for both stationary and mobile nodes.

The problems of TCP throughput prediction and available bandwidth estimation have all been studied thoroughly in the wireline domain. Analytic models for TCP throughput estimation have been developed based on TCP's congestion control algorithm and active measurements of network path characteristics such as packet loss, round trip time, and delay [13]–[16]. The problem of available bandwidth estimation on end-to-end paths and an active probe-based estimation method are described by Jain and Dovrolis in [17]. Additional methods for available bandwidth estimation with active probe-based measurement include [18]–[20].

He *et al.* [21] and Mirza *et al.* [22] present two approaches to wireline TCP throughput prediction. He *et al.* present a history-based throughput prediction approach, using moving averages of past throughputs to predict future throughputs [21]. Two shortcomings of this method are that it considers only bulk transfers lasting tens of seconds, and that it is relatively slow to adapt to changing network conditions. Mirza *et al.* improve on He *et. al.*'s methodology by using Support Vector Regression (SVR), a state-of-the-art machine learning technique, to predict wireline TCP throughput over a wide range of conditions, including small and large transfers and quick adaptation to level shifts.

In this work, we borrow from He *et al.* and Mirza *et al.* in that we use both moving averages and SVR to predict wireless TCP throughput. However, the problem of TCP throughput prediction in the wireless environment is very different from that in the wireline environment, so we have to adapt the approaches used by He *et al.* and Mirza *et al.* considerably. Both approaches take tens of seconds to generate a prediction. In the wireless environment, a throughput prediction needs to be generated in 1-2 seconds to be useful for applications in both the stationary and drive-by environment, so in this work we target these short timescales. Due to our focus on short timescales, we cannot use the active measurement tools [23] and [19] that Mirza *et al.* use to obtain path property measurements for SVR. Another reason we cannot use these tools is because they are designed around assumptions that only hold true for the wireline environment, *e.g.*, loss is an indicator of congestion and queue build-up on the path instead of interference, so they are unlikely to perform well in the wireless environment. The state of the art for active path measurement tools in the wireless domain is currently not as developed as that in the wireline domain, so we do not have adequate substitutes for tools such as [23] and [19] for drive-by TCP throughput prediction.

The development of drive-by communication protocols is also related to our work. The IEEE 802.11p Task Group is standardizing a short range communication protocol to be used in intelligent transportation systems for data exchange between high-speed vehicles and roadside infrastructure. A number of research efforts have also focused on understanding the feasibility and performance of drive-by applications where communication is set up between a mobile vehicle and a roadside AP. Examples include work by Ott and Kutscher [24], Gass *et al.* [25], and in more general settings in the CarTel project [1]. Unlike our work, the first two studies look at background traffic-free environments. All three look at what TCP throughputs are achievable in the vehicular scenario: our focus is different in that we try to predict future throughput accurately in a given environment rather than measuring current throughput.

III. PREDICTION MECHANISMS

What distinguishes our work from existing approaches to TCP throughput prediction in the wireless environment (*e.g*, [5] and [6]) is that existing approaches assume that a large amount of information, such as network topology and the sending rates of sources of interference, is available, while our approach assumes no such knowledge. Having access to this information allows the existing approaches to use analytical models to predict wireless TCP throughput. Since we assume incomplete information about the network environment, we are unable to use analytical models for prediction. Hence, we turn to time series analysis and machine learning to predict throughput. In this section, we present our time series and machine learning predictors, and also describe how we evaluate prediction accuracy.

A. Evaluating Prediction Accuracy

We use the metric *relative prediction error E* introduced in [21] to evaluate the accuracy of an individual throughput prediction. We denote the actual throughput by R and the predicted throughput by $\hat{R}$. *Relative prediction error* is defined as

$$E = \frac{\hat{R} - R}{min(\hat{R}, R)}$$

Throughout this paper, we use the distribution of the absolute value of E to report our results.

B. Time Series-Based Predictor

For time series-based prediction, we use the exponentially weighted moving average (EWMA),

$$\hat{R}_{i+1} = \alpha R_i + (1 - \alpha)\hat{R}_i,$$

with an α value of 0.3, because this value is used in both [21] and [22].

C. Support Vector Regression-Based Predictor

This section introduces Support Vector Regression: for more details see [26] and [27].

We are interested in predicting TCP throughput based on observed characteristics of the wireless environment. In statistical machine learning, this can be considered in a regression framework.

We call each file transfer an instance $\mathbf{x}$. Usually the instance is represented by a d-dimensional feature vector $\mathbf{x} = (x^{(1)}, \ldots, x^{(d)})^\top \in \mathbb{R}^d$. Each dimension is called a feature. For example, the feature vector can consist of path properties such as packet loss rate and round trip time before a transfer. The target $y \in \mathbb{R}$ of an instance $\mathbf{x}$ is the throughput. Our goal is to train a regression function $f : \mathbb{R}^d \to \mathbb{R}$ that maps any file transfer instance $\mathbf{x}$ to its throughput y.

To accomplish this, a training set that consists of n previous known instance-target pairs $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$ is required. The training set acts as a teacher to the regression function. A family of candidate regression functions $\mathcal{F}$ must also be specified, *e.g.,* the set of all linear functions, or a Reproducing Kernel Hilbert Space induced by a kernel, which we use in this study (see [26]). In addition, one needs a loss function $L(f(\mathbf{x}), y)$ to measure how close a prediction $f(\mathbf{x})$ is to the actual throughput y. The squared loss $L(f(\mathbf{x}), y) = (f(\mathbf{x}) - y)^2$ is often used. We use the ε-insensitive loss $L(f(\mathbf{x}), y) = \max(|f(\mathbf{x}) - y| - \varepsilon, 0)$. This loss function is standard in support vector regression, and produces comparable results as the squared loss (see [26]).

Given these input, training a regression function involves finding a function $f \in \mathcal{F}$ that minimizes the loss on the training set, plus a regularization term:

$$\hat{f} = \arg\min_{f \in \mathcal{F}} C \sum_{i=1}^{n} L((f(\mathbf{x}_i), y_i) + \|f\|^2. \qquad (1)$$

The first term is known as the empirical loss. The function that minimizes the empirical loss "does best" on the training set. Such a function, however, may not be the one that produces the most accurate predictions on future test instances. This is because minimizing empirical loss has the danger of overfitting the training data, which is but a small sample with its own idiosyncrasies. One way to prevent overfitting is to regularize f by its norm $\|f\|^2$, with the intuition that we prefer a smoother regression function. The scalar C balances empirical loss and smoothness of f. The solution to the optimization problem (1) can be found efficiently using a quadratic program.

Our SVR uses a so-called Radial Basis Function kernel, which induces a rich candidate regression function family. The kernel enables learning of highly non-linear, yet well regularized, regression functions. There are freely available applications that implement SVR. We use the software SVM^{light} ([28]).

IV. EXPERIMENTAL ENVIRONMENT

In this section we present our indoor (laboratory) and outdoor (drive-by) experimental environments.

We use 802.11a for our experiments: this allows us to avoid interference with our department's and other departments' 802.11b/g networks. We use 802.11a channel 36 for all our experiments.

Figure 1 illustrates the experimental setup for our laboratory experiments. There are four primary components to the setup: the wireline nodes, the wireless nodes, two commodity access points (APs), and a monitor node.

The wireline and wireless nodes are connected in a dumbbell topology via the APs and a switch. The APs are Cisco AP1200, running IOS version 12.3(8), with single Rubber Duck antenna, and integrated 802.11a module/antenna. The switch is a commodity LinkSys 10/100 16-port Workgroup Hub. The wireline nodes and switch are connected to the AP via 100Mbps Ethernet connections. The maximum data rate for 802.11a is 54Mbps; having 100 Mbps wireline links insures that the wireless, rather than the wireline, part of the network is the throughput bottleneck.
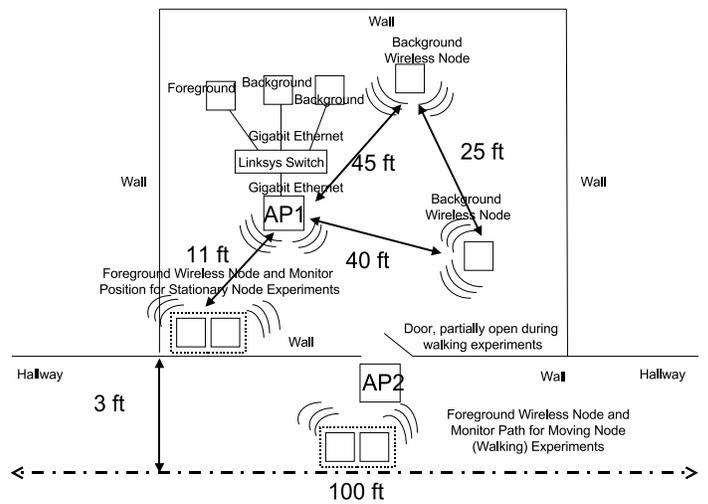

Fig. 1: Laboratory (indoor) setup for stationary-node and walking-node experiments.

The wireline nodes are identically configured Sun 4200 AMD Opteron 275 (dual Core) nodes, with 4 GB RAM, Intel 82546EB (e1000) chips, running CentOS 5.0. The wireless nodes are MacBooks and MacBook Pros with default vendor configurations. For the wireless network interfaces, for both indoor and outdoor experiments, the defaults were *(a)* no RTS/CTS, *(b)* auto rate fallback enabled, and *(c)* no mac layer fragmentation. We used the default network interface configurations because in production opportunistic networks the client nodes will to be heterogeneous and independently configured by users and/or vendors.

The monitor node is located next to the foreground wireless node for all experiments. This node is a Dell 700M, 1.6Ghz P4, with a Cisco CB21AG 802.11a/b/g cardbus adapter. The monitor node does not transmit any packets, but captures all the packets it sees on a specified channel. The traces thus collected are used for throughput prediction.

Figure 2 illustrates the setup for the drive-by experiments. The outdoor setup is different from the indoor setup in three ways. First, we replaced the Sun 4200 AMD Opteron nodes (the wired nodes) with Powerbook G4 and MacBook Pros because the Sun nodes were rack-mounted machines and hence not portable. Second, instead of using a MacBook Pro as the foreground wireless node, we used a 1 Ghz VIA Nehemiah node, with 512MB RAM, Netgear WAG311 with integrated 5 DBi antenna, running Gentoo 1.12.9, which is a 2.6.20.7 Linux kernel. The antenna extended the range of the wireless signal sufficiently to allow us to communicate with the AP while driving to a distance of approximately 400 ft in each direction. Third, we mounted the AP 20 feet above the ground to stay within line of sight of the car for a greater distance.

We used a portable generator to power our stationary nodes, which we placed halfway along our driving path, which was between two parking lots. The wireless foreground node and the monitoring node were placed in the car; the wireless foreground node's 5 DBi antenna was placed on the roof of the car to prevent the body of the car from lowering the signal strength.
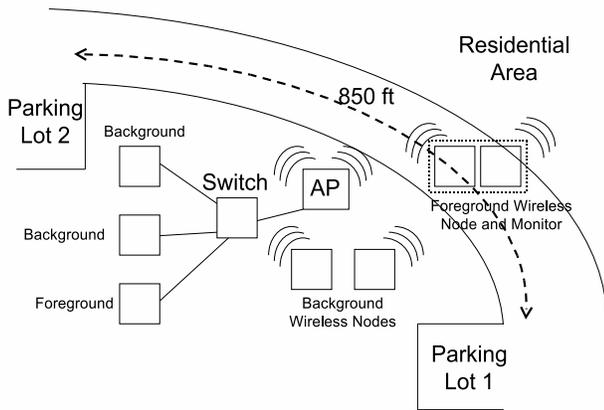

Fig. 2: Setup for the drive-by experiments

We ran two sets of indoor experiments. In the first set, the foreground wireless node was stationary; in the second set, the foreground node was moved around in the hallway next to our lab at walking speed over a distance of 100 ft. The position of the foreground node in both these scenarios is illustrated in Figure 1. How far we could move the foreground node was restricted by the layout of the hallway.

For the outdoor experiments, we drove along a stretch of road between two parking lots that were approx. 850 ft apart at speeds between 15 and 25 mph. An athletic field bordered the road on one side; the other side was bordered by a residential area. We chose this particular location because its low vehicular traffic volume allowed us to maintain driving speeds of 15-25 mph, and having an open field on one side allowed us to maintain line of sight, and hence association with the AP, for approximately 400 ft on either side, compared to approximately 50 ft on either size in the indoor experiments. We limit the driving distance from the AP such that we always stay associated with the AP. When we experimented with driving far enough to get out of range and then coming back into range, the re-association time was high: we spent over 80% of our driving time trying to associate, because the client was scanning channels on all APs it was receiving beacons from. For vehicular networking, not just vehicular throughput prediction, to be feasible, significantly faster association techniques will be required: coming up with such techniques is beyond the scope of this paper.

V. EXPERIMENTAL DESIGN

In this section, we describe the details of the traffic we generate, the measurements we take and how we use measurements to produce TCP throughput predictions.

In our experiments, communication between wireless and wireline nodes is pairwise, *i.e.*, during an experiment, each wireless node sends data to a single, pre-assigned, wireline node and vice versa. One wireline-wireless pair is designated the foreground pair: we predict throughput from the perspective of this node. The other two node pairs generate background traffic.
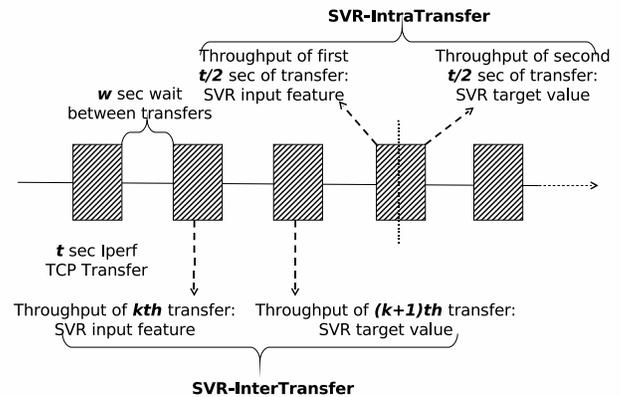
Measurement Protocol

Fig. 3: The foreground node measurement protocol used for all experiments.
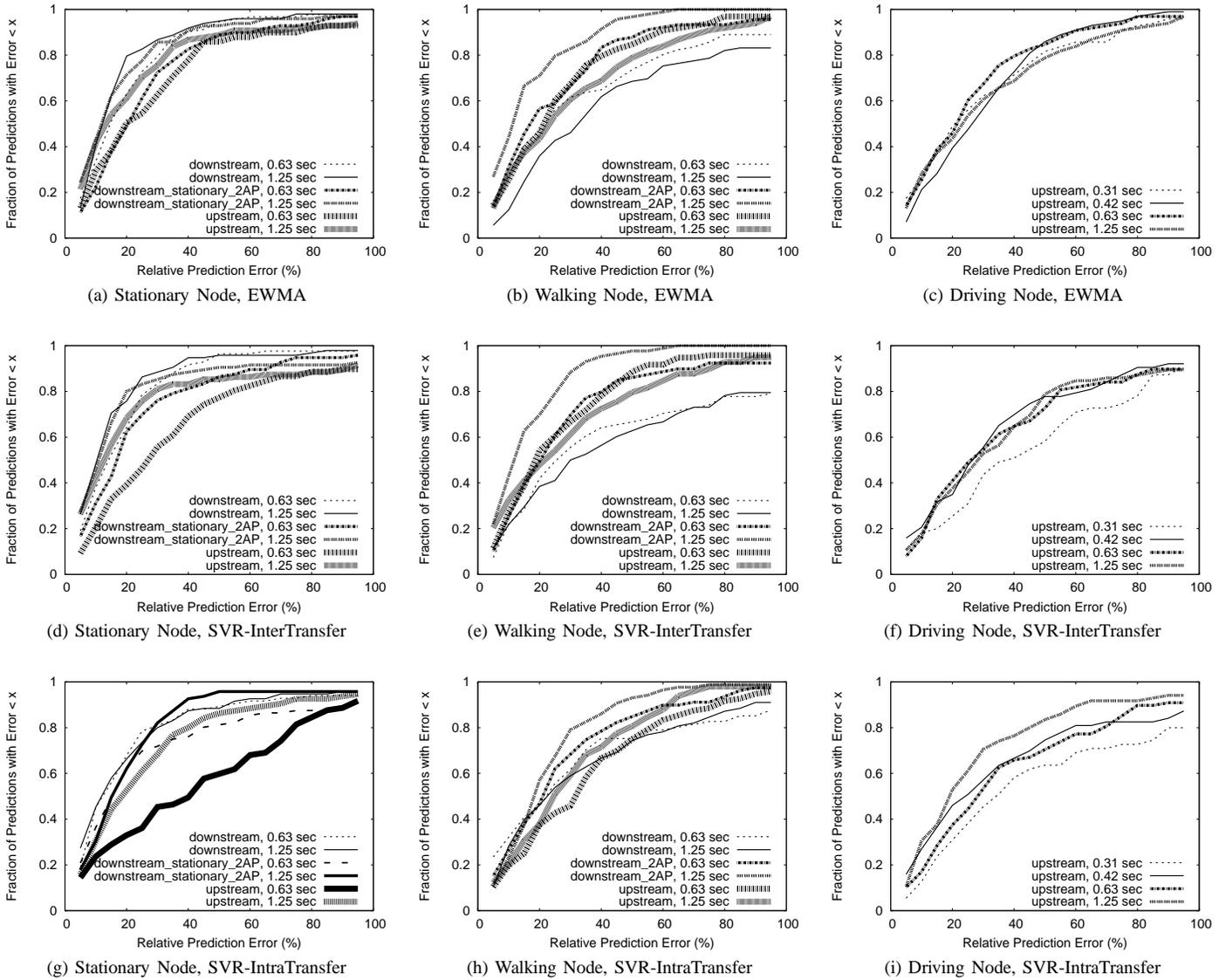
(a) Stationary Node, EWMA

(b) Walking Node, EWMA

(c) Driving Node, EWMA

(d) Stationary Node, SVR-InterTransfer

(e) Walking Node, SVR-InterTransfer

(f) Driving Node, SVR-InterTransfer

(g) Stationary Node, SVR-IntraTransfer

(h) Walking Node, SVR-IntraTransfer

(i) Driving Node, SVR-IntraTransfer

Fig. 4: Prediction accuracy using EWMA and SVR predictors for stationary, walking, and driving wireless nodes.

(a) EWMA

(b) SVR-InterTransfer
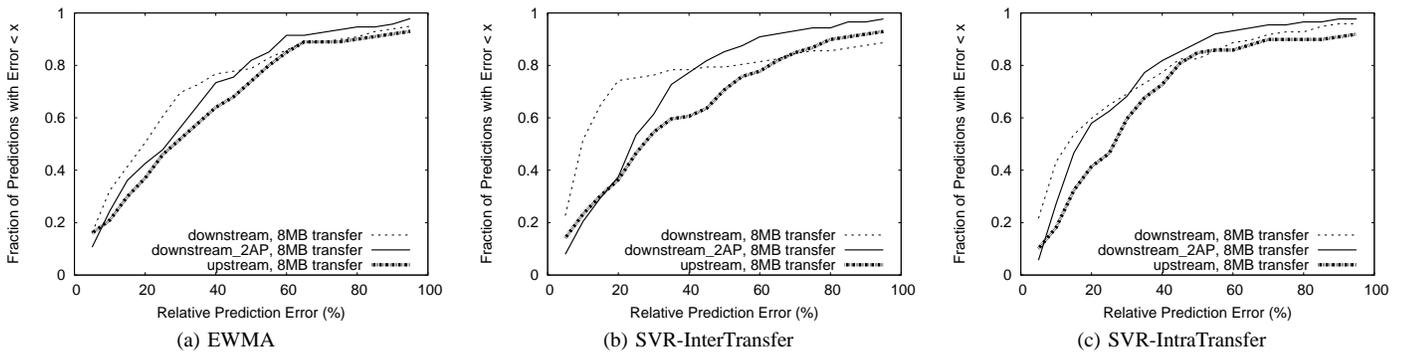
(c) SVR-IntraTransfer

Fig. 5: Prediction accuracy using EWMA and SVR Predictors for Large (8MB) transfers from a Stationary Wireless Node.
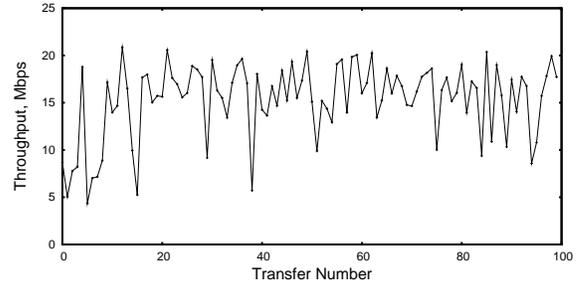
We experiment with three different data flow scenarios. First, we download data from the wireline nodes to the wireless nodes: in this case, the principle direction of TCP data flow is from the AP to the wireless nodes, and only TCP ACKs travel in the reverse direction. This is how wireless clients are typically used. Second, we upload data from the wireless nodes to the wireline nodes. This maximizes contention for the wireless medium in a single-AP wireless network. In the upload case, there are three nodes, the wireless nodes, transmitting, while in the download case only the access point is transmitting for the most part, so in the upload case there is greater contention for the medium. Third, we use the 2-AP setup. Here, two APs are within range of each other, so even in the download case, there is contention for the medium. For our indoor experiments, in the 2-AP setup, the background traffic nodes are associated with the AP inside the lab (AP1), and the foreground node is associated with the AP in the hallway (AP2), which is mounted on the ceiling of the hallway (Figure 1). In the single AP upload and download cases, AP2 is turned off and all nodes are associated with AP1.

We use Harpoon [29], a flow-level network traffic generator, to generate background traffic. The traffic is open loop on/off TCP traffic, with file sizes drawn from a Pareto distribution and the time between transfers drawn from an exponential distribution, a pattern that closely resembles actual Internet traffic [30]. Each background node pair runs 0-6 data transfer threads at any given time: this represents a user doing a few different things on a laptop simultaneously, such as browsing the web and downloading software. We set the maximum number of threads per background node to 6. This varies the background traffic enough to make prediction non-trivial without overloading the wireless network to the point of congestion collapse and near-zero throughput.
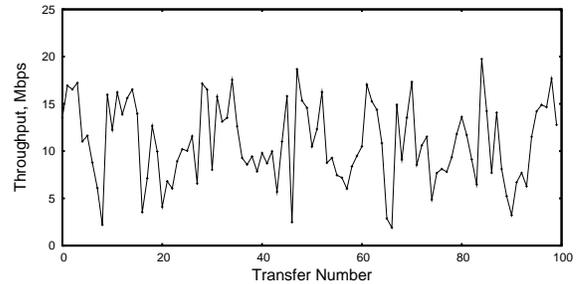
Figure 3 presents the measurement protocol we run on the foreground nodes. We run a series of Iperf transfers t seconds long separated by w seconds of wait time. Since our focus is on predicting throughput quickly, we use t and w values of 0.31, 0.42, 0.63, and 1.25 seconds. However, we also experiment with larger values for comparison. Background traffic runs continuously, during both the transfer phase and the wait phase of the foreground traffic.

For each value of t and w, we collect 200 measurement samples. For SVR-based prediction, the first 100 are used for training, and the second 100 for testing. For EWMA, the moving average of the throughputs of the first k transfers serves as the prediction for the $(k+1)th$ transfer. For SVR, the training set is every set of k consecutive transfers and the test set is the $(k+1)th$ transfer; given a sequence of samples, the first training set is made of samples 1 to k and the test set of sample $k+1$, the second training set of samples 2 to $k+1$ and the test set of sample $k+2$, and so on.

We use two schemes for SVR, called *SVR-InterTransfer* and *SVR-IntraTransfer*, as illustrated in Figure 3. In *SVR-*

(a) Stationary Node, Upstream Traffic, $t = 1.25$

(b) Driving Node, Upstream Traffic, $t = 1.25$

Fig. 6: Throughput timeline for stationary and driving transfers, for upstream traffic with $t = 1.25$ sec

InterTransfer, the input feature to the SVR is the throughput of the kth transfer, and the target, *i.e.*, the value to be learned, is the throughput of the $(k+1)th$ transfer. In *SVR-IntraTransfer*, the throughput of the first half of a transfer is the input feature, and the throughput of the second half is the target value. We divide the transfer into halves in terms of time elapsed rather than bytes transferred.

VI. RESULTS

Figure 4 presents the results of our experiments. We obtain the relative error of an individual prediction using the formula described in Section III-A. We present our results as cumulative distributions of relative error. The x-axis of each graph is the relative error of prediction, and the y-axis is the fraction of predictions with relative error of x or lower. A higher y value for a given x value means better prediction accuracy than a lower value of y.

We begin by looking at the general trends in Figure 4, and then look at error distribution values in detail.

We first consider the error distribution for each individual graph. Different lines in a given graph represent combinations of data flow setups (downstream, downstream_2AP, upstream), and the length of the TCP transfer (0.31, 0.42, 0.63, or 1.25 seconds).[1] The overall trend is that, in each

---

[1] We present results for only 0.63 and 1.25 second transfers for the stationary and walking case due to space limitations; results for 0.31 and 0.42 second transfers were similar. We present only upstream results for drive-by experiments because our downstream traces turned out to be corrupted.

graph, all lines are clustered relatively close together. This means that at a given speed and for a given predictor (*EWMA*, *SVR-InterTransfer*, or *SVR-IntraTransfer*), traffic direction, number of APs, and transfer duration have little impact on prediction accuracy.

Next, we compare the different graphs in two ways: the speed of the wireless client (stationary, walking, or driving), and the predictor used.

Each horizontal row of graphs represents increased client speed for a given predictor. The general trend across rows is that as the client speed increases, the gradient of the lines becomes less steep, *i.e.*, prediction accuracy decreases. This is consistent with intuition: as the client speed increases, the wireless environment becomes more dynamic, so predicting throughput becomes more difficult.

Each vertical column of graphs represents different predictors for a fixed client speed. The general trend across columns is that there is little difference in the gradient of lines, *i.e.*, prediction accuracy does not change significantly with a change in predictors. This observation is surprising: we had expected SVR-based predictors to do better than EWMA because they are quicker to respond to level shifts in background traffic. We also expected *SVR-IntraTransfer* to perform better than *SVR-InterTransfer*, because it has more current information about network conditions. One possible reason for *SVR-IntraTransfer* doing no better than *SVR-InterTransfer* could be TCP slow-start having a large impact on throughput because our TCP transfers are short, 1.25 seconds or less. To investigate this possibility, we experimented with larger timescales to minimize the effect of slow-start on throughput. Specifically, we used 8MB transfers separated by 15 second wait times. The results are presented in Figure 5. Once again, we observe that *SVR-IntraTransfer* does no better than *SVR-InterTransfer*, so we can rule out slow-start effects as being the reason why the relative prediction accuracy of *SVR-InterTransfer* and *SVR-IntraTransfer* is contrary to expectation.

Next, we consider the absolute prediction accuracy that is achieved. Here, there is cause for cautious optimism.

The prediction accuracy for wireless opportunistic networks using our methods is not comparable to the high prediction accuracy for wireline networks reported in [22]. For the opportunistic wireless case, typically, 10% to 30% of predictions are within 10% of actual throughput, and 30% to 70% of predictions are within 20% of actual throughput. In comparison, for the wireline case using SVR, 50% or more of predictions are within 10% of actual [22], which is a factor of 2 to 5 better than the results we obtain for the wireless case.

However, in all cases, 80% to 100% of predictions are within a factor of two of the actual throughput. This bound on prediction accuracy means that predictions can still be useful for some purposes, such as TCP-friendly rate calculation. Applications wishing to use the TCP-friendly rate can be conservative and assume a factor of two over-prediction, and still be able to get a reasonable throughput without overloading the network. The alternative would be knowing only that available bandwidth is somewhere between 1 and 54 Mbps in an 802.11a or g network, which is a factor of 54, so a factor of two bound is considerably more useful to an application. Also, the prediction can be made with a measurement lasting only 0.3 seconds[2], which is specially useful if the node is mobile, because *(a)* the factor of two bound holds even for nodes driving at speeds of 15-25 mph, and *(b)* prediction time is short enough to be negligible, so almost all the time within range of an AP can be devoted to transferring application data. On the other hand, predictions are not useful for applications such as highest-throughput access point selection, because the difference in achievable throughput between candidate APs is generally less than a factor of two (see, for example, [31]).

Next, we try to understand why it is difficult to achieve prediction accuracy comparable to the wireline case. We look at how throughput varies with time in Figure 6, for both stationary and driving nodes. Time is represented on the x-axis in terms of transfer numbers, and the y-axis shows the actual throughput for a particular transfer number. The main difference between the two graphs in the figure is that, on average, throughput is higher for the stationary case compared to the driving case. This is to be expected, because the average distance between the node and the AP is greater in the driving case. What is similar about the two graphs is the high variability of TCP throughput over time, represented by the sharp peaks and valleys in both graphs. While there is observable decrease in prediction accuracy as the speed of a node increases (compare rows of graphs in Figure 4) this decrease is not very large. For stationary nodes, 30% to 70% of predictions are typically within 20% of actual throughput, for walking nodes 30% to 60%, and for driving nodes 20% to 40%, so the decrease in accuracy due to movement is about a factor of 1.5. In comparison, the decrease in accuracy relative to wireline TCP throughput prediction is a factor of 2 to 5. Also, *SVR-IntraTransfer* does no better than *EWMA* or *SVR-InterTransfer*, even though it has up-to-date path information available to it. All these observations suggest that TCP throughput in the wireless environment is inherently variable, regardless of whether nodes are stationary or mobile.

VII. CONCLUSION

In this paper, we study the problem of TCP throughput prediction for opportunistic networking for stationary clients, clients moving at walking speed and clients driving at 15-25mph. We consider the problem in a variety of wireless settings including long and short timescales, one AP

---

[2]We assume that the predictors are maintained by the AP, and the client supplies the current throughput measurement and its location to get a prediction.

and two AP networks, upstream and downstream traffic. We use Exponentially Weighted Moving Averages (EWMA)-based and Support Vector Regression (SVR)-based predictors. We find that for all wireless environment configurations and prediction techniques that were considered, 80% to 100% of predictions are within a factor of two of the actual throughput. We believe that this accuracy is sufficient for making some decisions in opportunistic networks, but the inherent dynamism of wireless environments presents significant challenges in making more accurate predictions. In future work, we will investigate whether one particular property of the wireless environment, such as bursty cross-traffic, auto rate fallback or variation in channel quality, causes this dynamism, or whether it is the result of the interaction of a number of factors.

VIII. Acknowledgements

References

[1] V. Bychkovsky, B. Hull, A. Miu, H. Balakrishnan, and S. Madden, "A Measurement Study of Vehicular Internet Access Using In Situ Wi-Fi Networks," in *Proceedings of ACM MOBICOM*, Los Angeles, CA, USA, September 2006.

[2] K. Jain, J. Padhye, V. N. Padmanabhan, and L. Qiu, "Impact of interference on multi-hop wireless network performance," in *Proceedings of ACM MOBICOM*, San Diego, California, September 2003.

[3] J. Padhye, S. Agarwal, V. N. Padmanabhan, L. Qiu, A. Rao, and B. Zill, "Estimation of link interference in static multi-hop wireless networks," in *Internet Measurment Conference*, Berkeley, California, October 2005.

[4] C. Reis, R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan, "Measurement-based Models of Delivery and Interference in Static Wireless Networks," in *Proceedings of ACM SIGCOMM*, Pisa, Italy, September 2006.

[5] L. Qiu, Y. Zhang, M. K. Han, and R. Mahajan, "A general model of wireless interference," in *Proceedings of ACM MOBICOM*, Montreal, Canada, September 2007.

[6] A. Kashyap, S. R. Das, and S. Ganguly, "A Measurement-Based Approach to Modeling Link Capacity in 802.11-based Wireless Networks," in *Proceedings of ACM MOBICOM*, Montreal, Canada, September 2007.

[7] R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan, "Analyzing the MAC-level Behavior of Wireless Networks in the Wild," in *Proceedings of ACM SIGCOMM*, Pisa, Italy, September 2006.

[8] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Transactions on Information Theory*, vol. 46, no. 2, pp. 388–404, 2000.

[9] G. Bianchi, "Performance analysis of the ieee 802.11 distribted coordination function." *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 3, 2000.

[10] C. Burmeister, U. Killat, and J. Bachmann, "An analytical model of rate-adaptive wireless lan and its simulative verification," in *WMASH '05: Proceedings of the 3rd ACM international workshop on Wireless mobile applications and services on WLAN hotspots*. New York, NY, USA: ACM Press, 2005, pp. 65–73.

[11] C. Burmeister, U. Killat, and B. Bachmann, "Tcp over rate-adaptive wlan - an analytical model and its simulative verification," in *WOW-MOM '06: Proceedings of the 2006 International Symposium on on World of Wireless, Mobile and Multimedia Networks*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 339–348.

[12] P. Gopalakrishnan, P. Spasojevic, L. Greenstein, and I. Seskar, "A method for predicting the throughput characteristics of rate-adaptive wireless lans," in *Vehicular Technology Conference (6)*, 2004, pp. 4528–4532.

[13] M. Mathis, J. Semke, J. Mahdavi, and T. Ott, "The macroscopic behavior of the TCP congestion avoidance algorithm," *Computer Communications Review*, vol. 27(3), July 1997.

[14] J. Padhye, V. Firoiu, D. F. Towsley, and J. F. Kurose, "Modeling TCP Throughput: A Simple Model and Its Empirical Validation," in *Proceedings of ACM SIGCOMM*, Vancouver, B.C., Canada, September 1998.

[15] N. Cardwell, S. Savage, and T. Anderson, "Modeling TCP latency," in *Proceedings of IEEE INFOCOM*, Tel-Aviv, Israel, March 2000.

[16] M. Goyal, R. Guérin, and R. Rajan, "Predicting TCP Throughput From Non-invasive Network Sampling," in *Proceedings of IEEE INFOCOM*, New York, NY, USA, June 2002.

[17] M. Jain and C. Dovrolis, "End-to-end Available Bandwidth: Measurement Methodology, Dynamics, and Relation to TCP Throughput," in *Proceedings of ACM SIGCOMM*, Pittsburgh, PA, August 2002.

[18] J. Strauss, D. Katabi, and F. Kaashoek, "A Measurement Study of Available Bandwidth Tools," in *Proceedings of ACM Internet Measurement Conference '03*, Miami, FL, November 2003.

[19] J. Sommers, P. Barford, and W. Willinger, "A Proposed Framework for Calibration of Available Bandwidth Estimation Tools," in *ISCC*, 2006, pp. 709–718.

[20] R. Kapoor, L.-J. Chen, L. Lao, M. Gerla, and M. Y. Sanadidi, "Capprobe: A simple and accurate capacity estimation technique." in *Proceedings of ACM SIGCOMM*, Portland, OR, August 2004.

[21] Q. He, C. Dovrolis, and M. Ammar, "On the Predictability of Large Transfer TCP Throughput," in *Proceedings of ACM SIGCOMM*, Philadelphia, PA, August 2005.

[22] M. Mirza, J. Sommers, P. Barford, and X. Zhu, "A Machine Learning Approach to TCP Throughput Prediction," in *Proceedings of the ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, San Diego, California, USA, June 2007.

[23] J. Sommers, P. Barford, N. Duffield, and A. Ron, "Improving Accuracy in End-to-end Packet Loss Measurements," in *Proceedings of ACM SIGCOMM*, Philadelphia, PA, August 2005.

[24] J. Ott and D. Kutscher, "Drive-thru internet: Ieee 802.11b for automobile users," in *Proceedings of IEEE INFOCOM*, Hong Kong, China, March 2004.

[25] R. Gass, J. Scott, and C. Diot, "Measurements of in-motion 802.11 networking," in *Proceedings of the Seventh IEEE WSMCA*, 2006.

[26] B. Schölkopf and A. J. Smola, *Learning With Kernels*. Cambridge, MA: MIT Press, 2002.

[27] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statistics and Computing*, vol. 14, pp. 199–222, 2004.

[28] T. Joachims, "Making large-scale svm learning practical," in *Advances in Kernel Methods - Support Vector Learning*, B. Schölkopf, C. Burges, and A. Smola, Eds. MIT Press, 1999.

[29] J. Sommers and P. Barford, "Self-Configuring Network Traffic Generation," in *Proceedings of the ACM Internet Measurement Conference*, Taormina, Italy, October 2004.

[30] M. E. Crovella and A. Bestavros, "Self-similarity in World Wide Web traffic: evidence and possible causes," *IEEE /ACM Transactions on Networking*, vol. 5, no. 6, pp. 835–846, 1997.

[31] A. J. Nicholson, Y. Chawathe, M. Y. Chen, B. D. Noble, and D. Wetherall, "Improved Access Point Selection," in *Proceedings of the 4th International Conference on Mobile Systems, Applications, and Services (MobiSys)*, Uppsala, Sweden, June 2006.