Estimating Network Performance Using Smart Experts

Y. Edalat, J. S. Ahn, and K. Obraczka

***Abstract*—Several network protocols, services, and applications adjust their operation dynamically based on current network states. Consequently, keeping accurate estimates of network conditions and performance as they fluctuate over time is critical. Notable examples include TCP and IEEE 802.11, both of which periodically adapt some of their key operating parameters, namely the retransmission timeout and the contention window size depending on the average round trip time and the number of collisions, respectively.**

In this paper, we present a novel mechanism to estimate "near-future" network performance based on past network conditions. We call our approach to network performance estimation SENSE for Smart Experts for Network State Estimation. SENSE uses a simple, yet effective algorithm combining a machine learning method known as Fixed-Share and Exponentially Weighted Moving Average (EWMA). SENSE also introduces novel techniques that improve the predictability of the Fixed-Share framework without increasing computational complexity. SENSE is thus able to respond to network dynamics at different time scales, i.e., long- and medium-term fluctuations as well as short-lived variations. We evaluate SENSE's performance using synthetic and real datasets. Our experimental results show that, when compared to Fixed-Share and EWMA, SENSE yields higher estimation accuracy for all datasets due to its ability to more closely track data fluctuations.

***Index Terms*—Machine learning, computation intelligence, network performance, network state estimation**

I. Introduction

COMPUTER networks have become one of our society's essential commodities and, like power- and water distribution systems, are now considered part of our critical infrastructure. Consequently, it is crucial to keep them operating continuously and delivering adequate performance. This is especially true as networks become increasingly more complex and the services they provide increasingly more sophisticated and demanding.

Like any complex dynamical system, computer networks' performance fluctuates over time influenced by a variety of factors such as traffic load, end system load, communication link conditions (e.g., propagation channel impairments especially in the case of wireless links), etc. In order to adapt to network dynamics, most computer network protocols and algorithms employ a number of operational parameters that constantly estimate current conditions in the network. Notable examples include the Transmission Control Protocol (TCP) and IEEE 802.11 (Wi-Fi) which adjust the retransmission timeout and contention window size, respectively, according to network congestion and wireless channel state. More specifically, to recover lost packets in a timely manner yet minimizing the number of unnecessary retransmissions, TCP periodically evaluates the degree of network congestion under the assumption that network conditions will stay almost the same until the next evaluation period. It uses the round-trip time (RTT), i.e., the time between sending a segment and receiving confirmation from the other end that the segment was received, as a way to gage network load. TCP adjusts its retransmission timeout, i.e., the interval of time the TCP sender will wait for a segment's acknowledgment from the TCP receiver before retransmitting the segment, based on TCP's current estimate of the RTT. To compute its estimate of the RTT, TCP runs a simple mechanism known as Exponentially Weighted Moving Average (EWMA) with one tunable parameter, which determines the relative weight between the current RTT measurement and the previous RTT estimate.

The IEEE 802.11 responds to congestion buildup in the network by exponentially inflating its back-off window, which stipulates the average amount of time that a node should wait to transmit after a collision has occurred. The rationale for this exponential back off is that collisions are used as congestion indicators; and, after a failed attempt to transmit due to a collision, the transmitter needs to wait longer before trying again. To estimate the "near-future" channel state, IEEE 802.11 counts the number of consecutive collisions that took place during the current estimation time window and exponentially expands the size of the back-off window according to this collision count.

Clearly, the performance of these widely used network protocols heavily relies on how correctly their prediction mechanisms forecast "near-future" network state. Their implicit assumption is that network conditions change smoothly, i.e., that "near-future" state is closely correlated to previous history. As a result, their performance can be negatively affected when their operational parameters are set without accurately accounting for network dynamics. TCP, for instance, statically presets the weight factor in its RTT EWMA equation irrespective of the target network environment and conditions. The fixed weight factor in TCP's RTT EWMA calculation is a relative ratio deciding how much the current RTT measurement and the current RTT estimate should influence the new RTT estimate. The more dynamic the network conditions, the more weight should be placed on the current RTT measurement. Therefore, to achieve better performance, the fixed weight factor should change dynamically depending on network conditions.

IEEE 802.11 rigidly cold-starts and counts collisions at every new frame's transmission without considering previous

channel state. This means that considerable resources may be wasted in the process of reaching an adequate congestion window since 802.11's network estimation technique does not keep track of the network state after successful transmissions.

Motivated by the need to accurately estimate near-term future network conditions whose rate of change may vary over time, this paper introduces Smart Experts for Network State Estimation (SENSE)[1]. SENSE is a simple, yet efficient machine learning predictor based on the Fixed-Share approach [1] [2] [6] [7]. Unlike conventional network state estimators, SENSE provides a general framework that can incorporate any traditional estimator as an "expert". SENSE can then dynamically select the best experts among the set of all experts being used depending on their performance. It swiftly chooses experts that more faithfully capture network dynamics by penalizing poorly performing experts.

The original Fixed-Share algorithm [1] has four main drawbacks. First, for every dataset, a fix value within the range we are trying to predict is assigned to each expert. Thus, the range of the estimation is required for proper assignment of these values. Second, its accuracy is sensitive to the number of experts and typically, the more experts, the more accurate the prediction since the algorithm basically singles out a few well-behaved experts among the set of competing experts. There is clearly, a "diminishing return" effect after the number of experts gets too high. Third, the "loss function" penalizing experts, relies exclusively on the magnitude of the current error, instead of whether errors have recently increased or decreased. Additionally, all poorly performing experts are equally penalized. Depending on the recent error variation history, the loss function should intensify or alleviate the penalty for each individual expert to accelerate convergence. Finally, to promptly adapt to abrupt changes even when recent measurements become distinctly different from previous ones, the original Fixed-Share algorithm constantly tries to boost the weight given to poorly performing experts while offsetting the weight of well performing ones. This feature leads to precision degradation as too much emphasis is placed on poorly performing experts, especially when sudden changes rarely happen.

To address these problems, SENSE introduces three techniques, namely: (1) smart experts, (2) META-learning, and (3) Level-shift. SENSE's smart experts reduce sensitivity to the number of experts and eliminate the need for a-priori knowledge of the data that we are trying to predict. SENSE employs EWMA equations with different weights as its experts and normalizes errors by the maximum observable output. SENSE's META-learning algorithm expedites convergence by tracing recent past history and adjusting each expert's penalty accordingly. Finally, the Level-shift mechanism [12] employed by SENSE improves its response to sudden data changes by bounding SENSE's learning time window, and upon detecting dissimilar data patterns, SENSE reinitializes its tunable parameters and starts to relearn.

We evaluate SENSE using a variety of datasets including synthetic and real data. In all cases, SENSE outperforms predictors based on pure EWMA as well as Fixed-Share. Furthermore, a key advantage of SENSE is that it automatically adjusts to the data it is trying to predict. As a result, SENSE yields superior performance for all datasets used in our experiments when compared to "pure" Fixed-Share and EWMA. Our results also indicate that the performance of EWMA is quite sensitive to its "smoothing" factor, which determines how much weight will be placed on the "past" versus the "present" when predicting the "future". Another key advantage of SENSE's ability to automatically adjust to the data is that, unlike Fixed-Share, it needs no a-priori information about the dataset and is minimally sensitive to the number of experts. In our experiments, SENSE yields higher prediction accuracy when compared to the Fixed-Share algorithm and EWMA.

The rest of the paper is organized as follows. Section II presents some background on history-based prediction algorithms, namely EWMA, Fixed-Share, and its predecessor, Static Experts. Section III describes SENSE in detail and Section IV compares the performance of SENSE against EWMA and Fixed-Share algorithm. Section V evaluates SENSE's new techniques and parameters, while Section VI provides an overview of related works. Finally, Section VII concludes the paper with directions for future work.

II. Background

SENSE is based on a combination of history-based predictors, more specifically EWMA and Fixed-Share. In this section we review EWMA as well as Fixed-Share and its predecessor, Static Experts, both of which are examples of the Multiplicative Weight algorithmic family.

A. EWMA

EWMA based predictors, calculate an exponentially weighted mean of the previous data. Equation (1) shows the basic equation of exponential smoothing given by Hunter [14] where y_t and x_t represent, respectively, a sequence of data point that has been observed and a sequence of forecasts given by the predictor. Furthermore, α in (1) is the "smoothing factor", a value between 0 and 1 specifying how much relative weight is given to previous estimates (i.e., the "past") versus new samples (the "present").

$$x_t = \alpha \times y_{t-1} + (1-\alpha) \times x_{t-1} \quad (1)$$

The problem of using EWMA based predictors is choosing appropriate α, which should be based on the dataset. Even though there has been no generally accepted statistical technique for choosing α, our observations indicate that α needs to be determined based on the small-lag autocorrelation. Recall that the autocorrelation is a correlation coefficient that instead of measuring the correlation between two different variables, it measures the correlation between two values of the same variable at times t_i and t_{i+k}. The autocorrelation can be used to detect the degree of randomness in data i.e., whether data similar to the present data would appear in the

[1] An earlier version of this work appeared in [3]

future. The autocorrelation parameter named "time lag" measures how soon the same data pattern will repeat.

Our experiments with a variety of datasets indicate that α should be chosen based on the small-lag autocorrelation. If data is random, the small-lag autocorrelation should be near zero. In this case, low values of α are desirable: low α has EWMA act as a low-pass filter smoothing out sudden fluctuations occurred in the input data series. In other words, low values of α, favor the "past" over the "present" when computing the current estimate. On the other hand, if data is non-random, then small-lag autocorrelations will be significantly non-zero. In this case, high α acts as a high-pass filter hardly filtering out measurement noise. It means that, with high α, the "present" plays a more important role.

The problem of current EWMA based predictors is that they have to statically set α, for example, by trying to guess what the data will look like in the future. SENSE, however, runs a small number of EWMA experts with different α's and, using the Fixed-Share technique, dynamically picks the best performing EWMA depending on network dynamics.

B. Multiplicative Weights Method

The Multiplicative Weight algorithmic family has shown to yield performance improvements in a variety of on-line problems [8]. Aiming at minimizing the prediction error, this family of algorithms combines predictions of a set of experts $\{x_1, x_2, \dots, x_N\}$ to compute the overall prediction denoted by $\hat{y}_t$. To denote the impact of each expert on the overall predictor, it associates each expert with a weight from $\{w_1, w_2, \dots, w_N\}$. After each trial, the weight of each expert is updated depending on the difference between its prediction and the real data represented by y_t. Weights of "well-performing" experts are not changed, while the weights of experts that are not performing well are reduced.

As an illustration, Fig. 1 shows the implementation of the Multiplicative Weight algorithm with *N* experts using a hardware block diagram. The shaded boxes on the left- and middle columns correspond, respectively, to the experts denoted as x_i and the penalty function. The process of updating weights and generating the final predictions is represented as a circuit employing the addition, division, and multiplication operators.

Equation (2) represents the circuit of Fig. 1 as a mathematical expression. As shown in (2), $\hat{y}_{t+1}$ can be represented by a sum of products of $\alpha_{i,t}$ and $x_{i,t}$ where $\alpha_{i,t}$ is the experts' weights ($0 < \alpha_{i,t} < 1$) which are dynamically and systematically adjusted and $x_{i,t}$ is each expert's prediction. Equation (2) confirms that the Fixed-Share algorithm is a selection process, which favors experts whose predictions are closer to the real data by incrementally growing their weights, while reducing other experts' weights.

As highlighted in [1], several schemes have been proposed for updating experts' weights in multiplicative weight algorithms. In the remaining of this section, we discuss two well-known Multiplicative Weight algorithms, namely Static Experts and Fixed-Share.

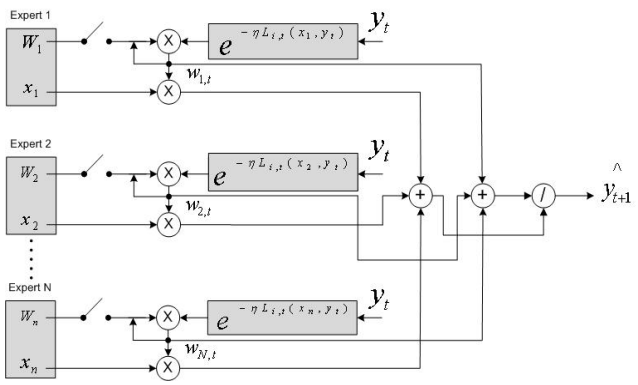


Fig. 1. Hardware block diagram of Multiplicative Weight algorithm

$$\hat{y}_{t+1} = \frac{\sum_{i=1}^{N} x_{i,t} \times w_{i,t} \times \exp\{-\eta \times \sum_{i=1}^{N} L_{i,t}(x_i, y_t)\}}{\sum_{i=1}^{N} w_{i,t} \times \exp\{-\eta \times \sum_{i=1}^{N} L_{i,t}(x_i, y_t)\}} = \sum_{i=1}^{N} \alpha_{i,t} \times x_{i,t}$$

$$\text{Where } \alpha_{i,t} = \frac{w_{i,t} \times \exp\{-\eta \times \sum_{i=1}^{N} L_{i,t}(x_i, y_t)\}}{\sum_{i=1}^{N} w_{i,t} \times \exp\{-\eta \times \sum_{i=1}^{N} L_{i,t}(x_i, y_t)\}} \qquad (2)$$

1) Static Experts Algorithm

Static Experts, whose pseudo code is presented in Algorithm 1, is the simplest version of the Multiplicative Weight algorithmic family. Its steps, as described in Algorithm 1, are common to all Multiplicative Weight algorithms with *N* experts. The Prediction step in Algorithm 1 computes the current prediction by (1) summing, over *N* experts, the products of the expert multiplied by its current weight and then (2) normalizing the result by the sum of the weights. Using a given "loss function", the Loss function step checks, at each prediction trial, how good of a prediction each expert yields. Then, in the Exponential updates step, the loss computed in the Loss function step is used to adjust the experts' weights, which will be used in the next trial.

Algorithm 1. Static Expert Algorithm

Parameters:

$$\eta > 0, \quad 0 \le \alpha \le 1$$

Initialization:

$$w_{1,1} = \cdots = w_{N,1} = \frac{1}{N}$$

Prediction:

$$\hat{y}_t = \frac{\sum_1^N w_{i,t} * x_i}{\sum_1^N w_{i,t}}$$

Loss Function:

$$L_{i,t}(x_i, y_t) = \begin{cases} (x_i - y_t)^2 & , x_i \ge y_t \\ 2 * y_t & , x_i < y_t \end{cases}$$

Exponential Update:

$$\acute{w}_{i,t} = w_{i,t} * e^{-\eta * L_{i,t}(x_i, y_t)}$$

The Static Experts algorithm has one main drawback: it is not able to adjust to abrupt changes in data fast enough. This is

because it takes a relatively long time for the weight of an expert to either shoot up or down when the expert's performance suddenly changes following an abrupt change is the data.

2) Fixed-Share Algorithm

To solve the problem of slow convergence of well-performing experts in the Static Experts algorithm, [1] introduced Fixed-Share. The main goal of the Fixed-Share approach is to improve Static Experts' performance, while keeping its simplicity. The basic idea of Fixed-Share is to prevent large differences among experts' weights; to this end, it shares a fixed fraction of the weights of experts that are performing well among the other experts. This additional step, called "Sharing weights" and shown in (3), redistributes evenly a certain fixed fraction of pool, which is the sum of a preset portion of each weight.

$$Pool = \sum_{i=1}^{N} \alpha \times w'_{i,t} \qquad w_{i,t+1} = (1-\alpha) \times w'_{i,t} + \frac{1}{N} \times pool \qquad (3)$$

Although the Fixed-Share algorithm has been shown to perform well when estimating network variables in [2], it exhibits four main weaknesses. First, it must have a priori knowledge of the dataset's range in order to properly set the value of its experts. Likewise, the degree of sharing in the Sharing weights step (3) cannot be appropriately determined unless the number of level shifts is known in advance. Level shift [12] is defined as a significant change in the mean of observed data and is discussed in detail in Section III. Second, the accuracy of the algorithm is quite sensitive to the number of experts whose values determine the granularity over the range of values that the variable in question can assume, and ultimately its accuracy. However, as discussed in Section II.B.1, more experts may also introduce additional errors. Third, since the loss function is pre-determined and not changed considering the target environment and application, it is not always able to exhibit adequate convergence. Finally, if experts perform consistently well for long periods of time, sharing their weight with other experts whose performance is not adequate, compromises the algorithm's overall convergence and performance.

III. SENSE

This section provides a detailed description of our online estimator, SENSE, which employs a combination of Fixed-Share with EWMA. Then, we discuss SENSE's accuracy compared to Multiplicative Weight algorithmic algorithms.

A. SENSE Algorithm

More specifically, SENSE, whose pseudo-code is shown in Algorithm 2, is an enhanced version of the Fixed-Share estimator, where, instead of fixed-value experts, EWMA filters are employed as experts. Table I summarizes SENSE's variables and their descriptions.

More specifically, in the EWMA experts step of Algorithm 2, the prediction of each expert, $x_{i,t}$, is calculated as a weighted sum of the previously observed data item y_{t-1} and the previous prediction $x_{i,t-1}$ where α represents the relative weight between $x_{i,t-1}$ and y_{t-1}. Initially, each expert is assigned a weight, $w_{i,1} = 1/N$, where *N* is the total number of experts; each expert is also assigned an α_i value between 0 and 1 which differentiates experts from each other. In SENSE, EWMA experts replace numeric experts used in the Fixed-Share algorithm, which results in making SENSE's accuracy less sensitive to the number of experts used.

TABLE I
SENSE'S VARIABLES

Variable	Description
x_i	Prediction of expert i
y_t	Observed data at time t
$\hat{y}_t$	SENSE's prediction for time t
w_i	Weight of expert i
NE_i	Normalized error of expert i
N	Total number of experts
$L\,(x_{i,}, t)_{i,t}$	Loss of expert i at time t
y_{max}	Maximum data observed so far
η_i	Determines degree of penalizing expert i
β	Determines how much η_i should be increased or decreased
EL	Error limit (based on user's desired accuracy)
η_{min}, η_{max}	Limit experts' weight
j	Determines the time window to evaluate expert's performance (used to update η_i)

As illustrated in the Prediction step of Algorithm 2, at every trial *t*, SENSE calculates the current prediction $\hat{y}_t$ by adding the weighted predictions from *N* experts. After computing $\hat{y}_t$, the loss function step in Algorithm 2 calculates the absolute difference between the actual outcome, y_t, and each expert's forecast $x_{i,t}$; Then it normalizes this error with the maximum outcome y_{max}, which is updated with the largest outcome observed yet. We have experimented with different loss functions and picked the one shown in Algorithm 2 for its efficiency as well as simplicity. Finally, the loss function, $L\,(x_{i,}, t)_{i,t}$, is set to either the normalized error $NE_{i,t}$ or the *NULL* function depending on $NE_{i,t}$'s value. If $NE_{i,t}$ lies within the satisfactory boundary *EL*, SENSE does not penalize experts differently than the original Fixed-Share algorithm, which constantly adjusts the weight until the prediction equals the outcome. Here, *EL* can be set to any fraction between 0 and 1 according to the accuracy required by the application.

SENSE then runs the META-learning step, which either multiplicatively increases or decreases $\eta_{i,t}$ by β if the normalized error keeps growing or shrinking, respectively, for *j* consecutive trials. Otherwise, it does not change $\eta_{i,t}$. This META-learning step aims at deciding how to adjust the experts' weights based on their recent-past predictions. Clearly, the less accurate an expert's prediction is, the more severe that expert is penalized. In our experiments, we considered a three-trial observation window (i.e., j = 2) to characterize the recent past but larger observation windows

can be used. To prevent each expert's η from becoming too small or too large, $\eta_{i,t}$'s range is specified as $[\eta_{min}, \eta_{max}]$. We explore how $\eta_{i,t}$'s range impacts SENSE's behavior in Section V.B. Recall that the goal of META-learning is to speed up convergence of each expert's prediction to the observed outcome. Thus, the Weight update step updates $w_{i,t}$ with what has been learned, i.e., it multiplies $w_{i,t}$ by e to the power of the product of the loss function $L(x_i, t)_{i,t}$ and learning factor $\eta_{i,t}$.

Algorithm 2. SENSE Algorithm

Initialization:

$$\eta_{min} = \eta_{MIN-INIT} \quad \eta_{max} = \eta_{MAX-INIT} \quad \beta = \beta_{INIT}$$
$$EL = EL_{USER-DESIRED-ACCURACY} \quad w_{1,1} = \cdots = w_{N,1} = \frac{1}{N}$$

EWMA Experts:

$$x_{i,t} = y_{t-1} * \alpha_i + (1 - \alpha_i) * x_{i,t-1}$$

Prediction:

$$\hat{y}_t = \frac{\sum_1^N w_{i,t} * x_{i,t}}{\sum_1^N w_{i,t}}$$

Loss Function:

$$NE_{i,t} = \frac{|x_{i,t} - y_t|}{y_{max}}, L(x_i, t)_{i,t} = \begin{cases} NULL & , |NE_{i,t}| \le EL \\ NE_{i,t} & , Otherwise \end{cases}$$

META Learning:

$$\eta_{i,t} = \begin{cases} \min\left(\eta_{max}, (\eta_{i,t-1} * \beta)\right) & , NE_{i,t} > NE_{i,t-(j-1)} > NE_{i,t-j} \\ \max\left(\eta_{min}, \left(\frac{\eta_{i,t-1}}{\beta}\right)\right) & , NE_{i,t} < NE_{i,t-(j-1)} < NE_{i,t-j} \\ \eta_{i,t-1} & , Otherwise \end{cases}$$

Weight Update:

$$w_{i,t+1} = w_{i,t} * e^{-\eta_{i,t} * l(x_i,t)_{i,t}}$$

Restart Learning:

If Level Shift is detected at nk then,

$$w_{i,t} = w_{i,t} * e^{\sum_{t=-2T}^{t=-T} \eta_{i,t} * l(x_i,t)_{i,t}}$$

Finally, SENSE employs a Level-shift step [12] to detect any significant change in the mean of the observed data. Suppose $\{X_1, X_2, \ldots, X_n\}$ is the sequence of data, where X_1 is the first data after the last detected level shift. The measurement X_k is an increasing (decreasing) level shift if it satisfies the following three conditions:

(1) Data $\{X_1, X_2, \ldots, X_{k-1}\}$ are all lower (higher) than the data $\{X_k, \ldots, X_n\}$,

(2) The median of $\{X_1, X_2, \ldots, X_{k-1}\}$ is lower (higher) than the median of $\{X_k, \ldots, X_n\}$ by more than a relative difference χ, and

(3) $k + 2 \le n$.

The last condition helps to prevent misinterpreting an outlier as a level shift by making sure that a long enough sequence of data is observed to filter out ephemeral fluctuations. Upon the detection of a level shift, SENSE restarts its experts by only considering data after the level shift occurrence and resetting η for each expert. This means that the weight of each expert is determined only by the accuracy of prediction after the last level shift. In other words, the Level-shift step slides its learning window to consider only data after the last level shift in the experts' weight computation.

SENSE's level shift mechanism improves estimation accuracy when compared to Fixed-Share, which keeps weights of poorly performing experts from becoming negligible. It enables SENSE to adapt to persistent conditions as swiftly as Fixed-Share, while allowing poor experts' weight to become as infinitesimal as in Static Experts algorithm.

In summary, SENSE employs three main techniques as follows:

- Smart experts reduce the sensitivity to experts and eliminate the need for a-priori data knowledge. SENSE employs EWMA equations with different weights as its experts and normalizes errors by the maximum observable output.
- META-learning expedites convergence by tracing recent past history and adjusting each expert's penalty accordingly.
- Level-shift improves SENSE's response to sudden data changes by bounding SENSE's learning time window; upon detecting dissimilar data patterns, SENSE reinitializes its tunable parameters and starts to re-learn.

B. SENSE's Accuracy

According to [1], Static Experts' inaccuracy or "loss" is bounded by the sum of two terms, the total loss of the best expert and the total number of experts involved. This is described in (4), where $L(S, A)$ and $L(S, Expert_i)$ represent the loss of Static Experts algorithm *A* over the whole dataset *S* and the loss of the best expert *i* respectively. Furthermore, *c* is a constant determined by the type of loss function employed while *n* is the total number of experts. This upper bound represents how far Static Experts' prediction will deviate from the corresponding real data. Equation (4) confirms that the more experts used, the higher the additional loss (i.e., on top of the best expert's loss) incurred by the Static Experts algorithm.

$$L(S, A) \le L(S, Expert_i) + c \ln n \quad (4)$$

This upper bound on Static Experts' loss is only valid under the assumption that a given expert acts as the best expert over the whole dataset. However, when data patterns change so that experts take turns as the best expert, this upper bound needs to be recalculated as follows. At first we need to count all possible scenarios that can happen under dynamic environments. When all samples (trials) *l* are divided into *k+1* segments, for example, the number of ways to place *k+1* segments over *l* trials is *lCk+1*. Here, a segment refers to a sequence of trials for which a given expert is the best one. Since the number of ways to map n experts to the best expert in each *k+1* segments is *n(n-1)k*, then all possible cases amount to *lCk+1n(n-1)k*. If we consider each case as a "partition" expert, a specific partition expert can act as the best one over the whole trial set so that we can adopt (4) to predict

the accuracy of Static Experts under dynamic environments. Here, the sequence of segments and its associated sequence of best experts are called a partition. Namely under dynamic datasets, the additional loss of Static Experts due to the number of experts becomes *c[(k+1)logn+klogl/k+k]*.

To mitigate the dependence of Static Experts' upper bound on *l* and *k*, the Fixed-Share was introduced in [1]. Its main goal is to lessen the additional loss of the best partition, not the best expert like Static Experts, by introducing the Share Update operation. In Fixed-Share, the loss consists of three components as shown in (5), namely: the loss of the best partition $L(\mathcal{P}_k(S))$, the number of experts, and the final loss incurred by the Share update operation *L(sharing)*. However, Fixed-Share has a problem of inappropriately distributing weights of experts. This is because *L(sharing)* has a term that depends on the number of whole trials *l*, similarly to Static Experts. Note that Fixed Share tends to cut down a relatively large portion of the best expert's weight in preparation for sudden changes in the dataset, whose occurrence times are unpredictable.

$$L(S,A) \le L(P_k(S)) + c\ln n + L(sharing) \quad (5)$$

Unlike Fixed-Share and Static Experts, SENSE's loss upper bound is independent of the length of whole trials. It is dependent on SENSE's Level-shift mechanism, which restarts experts' weights more rapidly without sacrificing of the best expert.

IV. Evaluating SENSE's Performance

We evaluate SENSE using a variety of datasets and compare SENSE's performance against that of the original Fixed-Share algorithm and EWMA.

In the first set of experiments, we use synthetic data that exhibit different periodic patterns. We use both sine and square wave signals with a range of frequencies. These experiments systematically test how well SENSE can track the variation of input data over a wide spectrum of frequencies when compared to Fixed-Share and EWMA with different values of its smoothing factor, α.

TABLE II
SENSE's Parameters

Parameter	Value
β	2
EL	0.01
η_{min}	10
η_{max}	100
N	4
$Expert\ 1's\ \alpha$	0.2
$Expert\ 2's\ \alpha$	0.4
$Expert\ 3's\ \alpha$	0.6
$Expert\ 4's\ \alpha$	0.8
j	2

For a thorough comparative study, we also apply SENSE to the RTT dataset used in [2] and compare its predictions against estimates obtained using: (1) the original Fixed-Share algorithm, (2) Jacobson's TCP RTT estimation algorithm [15] (which is a variant of EWMA), and (3) "pure" EWMA with different smoothing factors.

In addition, we run SENSE over real collision rate data collected from a production Wireless LAN environment where access points (APs) periodically collect traffic and load statistics such as the number of retransmissions, total number of frames transmitted, etc.

Table II lists the default values of SENSE's parameters common to all results presented in this section. The performance impact of SENSE's techniques and parameters is evaluated in Section V.B.

A. Datasets with Periodic Patterns

These first sets of experiments compare SENSE's accuracy with EWMA and Fixed-Share when estimating datasets that follow periodic patterns. We use a dataset consisting of 1,000 samples. For the sine wave pattern, these samples create one period for 0.001 Hz and 200 periods for 0.5 Hz. The amplitude of our sine waves fluctuates between 0.25 and 0.75. For the square wave, these samples generate 40 periods for 0.025 Hz and 200 periods for 0.5 Hz. The amplitude of our square waves fluctuates between 0.1 and 0.7.

Choosing the best α value depends on data autocorrelation and is a key factor for EWMA based estimators' performance. Values of α closer to one have less of a smoothing effect and give more weight to recent changes in the data, while values of α closer to zero have a greater smoothing effect and are less responsive to recent changes. Note that, choosing α should be based on how quickly or slowly the dataset change; lower α worsens the accuracy for rapidly changing datasets, while higher α degrades the accuracy when data fluctuations are smoother. We show that SENSE eliminates EWMA's dependency on α.

In these experiments, SENSE uses four EWMA experts with α values evenly spaced between 0 and 1, i.e., 0.2, 0.4, 0.6, and 0.8. We compare SENSE against four EWMA equations with same α values as SENSE, namely: 0.2, 0.4, 0.6, and 0.8 which represent low, medium, and high EWMA smoothing factors. We also compare SENSE with Fixed-Share using 100 experts. In our prior work [2], we used the Fixed-Share algorithm with 100 experts to estimate TCP's RTT and, as expected, observed that beyond 100 experts the resulting improvement in prediction accuracy is not significant given the additional processing cost and convergence time. Each expert was assigned a value in the dataset's range; recall that the expert's value represent its prediction. In the case of sine and square waves, each expert's prediction in the Fixed-Share Expert algorithm is drawn from a uniform distribution from 0 to 1. As for the input data function, we use two patterns: sine-(results plotted in Fig. 2(a)) and square waves (results shown in Fig. 2(b)).

Fig. 2(a) plots the accuracy of the three approaches as measured by their average error as a function of the sine wave frequency. Each point in Fig. 2(a) is calculated by averaging

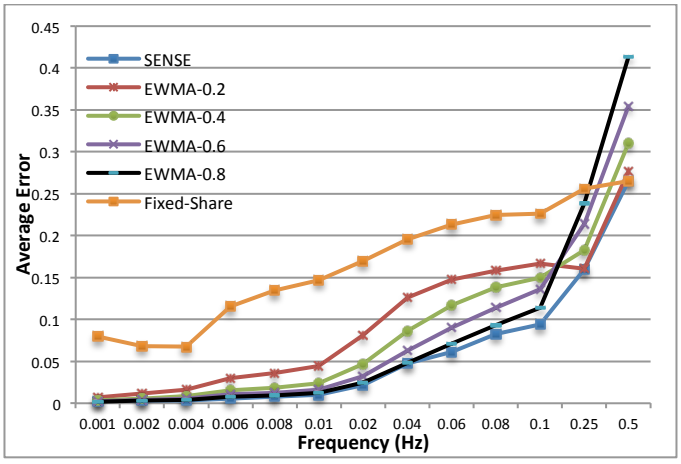
the absolute error of all 1,000 samples. As expected, at higher frequencies, the input's current value tends to be further apart from the last outcome, which makes it harder to accurately predict. Fig. 2(a) confirms that SENSE produces lower average error than any of the four EWMA filters and Fixed-Share over the entire frequency range. As the frequency goes up, errors from EWMA filters rise steeply regardless of the α value. EWMA with higher α tends to exhibit better accuracy over the lower frequency range, while EWMA with lower α performs better for frequencies higher than 0.1 Hz.

The reason for this phenomenon is that at lower frequencies, each sample tends to be similar to its previous one; consequently, tracking the sine wave with higher α by placing more weight on recent trials, yields higher accuracy. At high frequency where recent trials are less correlated to the upcoming trial, however, it is better to stick to previous history that will repeat after a short period of time. Indeed, the higher α causes some constant amount of error at every measurement while the lower alternatively results small and large errors.

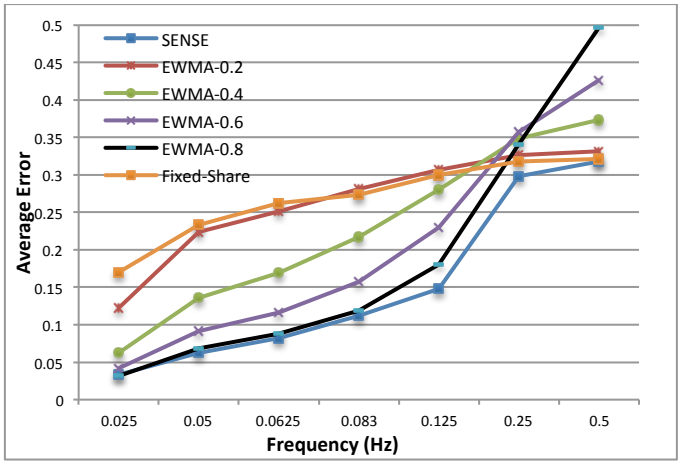
Fixed-Share's average error for various frequencies does not change significantly and its graph has a smaller slope. For sine wave's low frequencies, Fixed-Share shows larger error than other methods. The reason is that it takes longer for Fixed-Share to offset its large number of poor experts' weights and boost few well performing experts' weights. In the case of higher frequencies, it does not follow rapid data fluctuations and rather stays around the average value of sine wave due to a large number of experts and few trials for adaptation.

In contrast to "pure" EWMA and Fixed-Share, SENSE dynamically adapts according to the frequency by choosing an appropriate EWMA expert for a given frequency range. As the frequency increase, SENSE shifts its reliance from EWMA with higher α to EWMA with lower α.

Fig. 2(b) shows the average error of SENSE, four EWMA filters and Fixed-Share method when driven by square waves. This figure exhibits very similar trend as Fig. 2(a) where SENSE outperforms other methods at all frequencies. SENSE's smart experts are able to automatically switch between EWMA with high α value at low frequencies and EWMA with low α over the high frequency range.



(a) Sine waves



(b) Square Waves

Fig. 2. Average error comparison over periodic data

B. *Estimating TCP Round-Trip Times (RTT)*

We also evaluated SENSE's accuracy when applied to real datasets. As discussed in Section I, TCP, one of the most widely deployed Internet protocols, uses round-trip time (RTT) as an indication of network load. TCP employs its RTT estimates to trigger TCP's core functions such as error- and congestion control. Motivated by how critical accurate RTT estimates are for TCP's performance, we evaluate SENSE's accuracy in estimating RTTs in comparison to the Fixed-Share algorithm employed in [2], as well as TCP's original RTT estimator based on Jacobson's well-known EWMA variant [15] as shown in (6), where α is typically set to 0.85.

$$Est\_RTT = \alpha \times Est\_RTT + (1-\alpha) \times RTT \quad (6)$$

For these experiments, we use the RTT dataset in [2]. These RTTs were measured when a 16 MB file was transferred over a real network. As shown in Fig. 3, SENSE is able to keep track of the RTT variations more faithfully than Fixed-Share and Jacobson over the entire observation period.

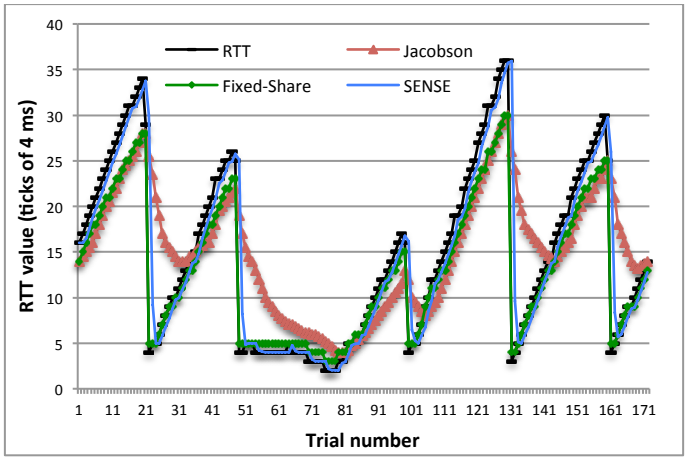


Fig. 3. RTT prediction by SENSE, Fixed-Share and Jacobson for each data sample (represented by a trial number)

Table III summarizes the average normalized error of SENSE, the four different EWMA filters, Fixed-Share and Jacobson's algorithms when applied to the same RTT data of Fig. 3. In order to calculate the average normalized error, we

first divide the absolute error of each sample by the real data it is trying to predict; then, we average these normalized errors. To compute the error ratio, we choose SENSE's average normalized error as baseline. Then, we calculate the other methods' relative error compared to SENSE as the difference between their average normalized error divided by SENSE's average normalized error. The resulting error ratio confirms that SENSE's accuracy outperforms both Fixed-Share and EWMA.

TABLE III
AVERAGE NORMALIZED ERROR COMPARISON FOR RTT DATASET

	AVERAGE NORMALIZED ERROR	ERROR RATIO (%)
SENSE	0.26	-
FIXED-SHARE	0.33	26%
JACOBSON	0.79	191%
EWMA-0.2	0.63	142%
EWMA-0.4	0.44	71%
EWMA-0.6	0.34	32%
EWMA-0.8	0.29	11%

C. *Estimating Collision Rates*

To further evaluate SENSE's ability to forecast network dynamics in real environments, we applied SENSE to collision rate datasets measured in a production Wireless LAN (WLAN) environment. Collision rates were collected at access points (APs) as they send traffic to a node associated with it while other associated nodes concurrently communicate with the AP, as they usually do. Specifically, we transmit 100 Mbps of UDP traffic from the AP to a node for 200 seconds while we simultaneously run different types of traffic between interfering APs and interfering nodes (i.e., located close to the node receiving data from the AP). Collision rates are calculated every second as the ratio of the number of retransmitted packets to the total number of transmitted packets. Since the test AP and the test node are physically close to one another, we assume that retransmitted packets are solely due to collision, and not to noise interference.

Fig. 4 depicts how SENSE and Fixed-Share track a time series of real collision rates gathered from the test network for 200 seconds. Similarly to the previous experiment, we use 100 experts for Fixed-Share and, based on the collision rate data, Fixed-Share's 100 experts are uniformly distributed between 0 and 0.2. We observe from Fig. 4 that, initially, the dataset contains considerable "noise" caused by bursty traffic generated by short-lived flows from applications like the Web. After 100 trials (seconds), longer-lived flows resulting from traffic such as wireless video transmission becomes dominant, yielding "smoother" collision rate variations. Fig. 4 shows that, while SENSE does not exactly follow the sudden jumps in the first half of the time series, its accuracy is significantly higher than Fixed-Share's. And, in the second half of the graph, SENSE is capable of accurately tracking variations in the data.

As shown in Fig. 4, Fixed-Share does not exhibit adequate accuracy and, according to Table VI, results in an unacceptably high error ratio. These results confirmed that Fixed-Share's lack of agility is due to the very short-lived data variations, which do not allow enough time to "train" Fixed-Share's experts.

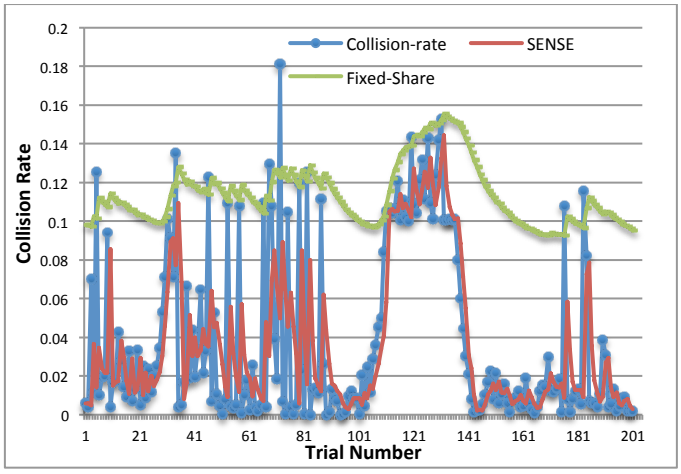


Fig. 4. Trace of SENSE's collision rate prediction vs. Fixed- Share for each data sample (represented by a trial number)

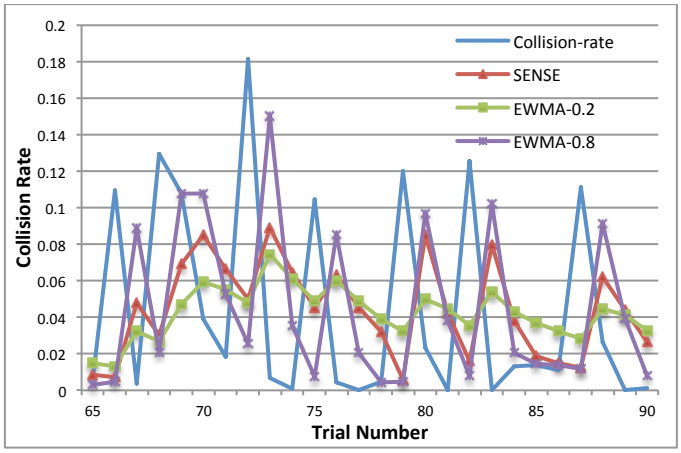


Fig. 5. SENSE vs. EWMA for highly variant portion of collision rate

Fig. 5 shows a closer view of the behavior of SENSE compared against two EWMA filters over a 25-second interval between 65-90 seconds of Fig. 4. Note that in this span of time, data fluctuate significantly, which makes it very difficult for any predictor to predict accurately. In this period, SENSE behaves like a low-pass filter, e.g., EWMA with α set to 0.2, while the curve corresponding to EWMA with α value of 0.8 looks like the real data but delayed by a full trial, which results in the highest error. Table IV summarizes the results shown in Fig. 5 by comparing the average error and error ratio for the first 100 trials of the collision rate dataset. It confirms that, for the first half of the dataset, which is quite "noisy", SENSE acts as an EWMA predictor with lower α and yields highest accuracy.

Fig. 6 zooms in the performance of SENSE and two EWMA filters over the interval of 100-145 seconds in Fig. 4. As shown in Fig. 6, SENSE quickly catches up with collision rate changes and behaves similarly to EWMA with α = 0.8 (acting as a high-pass filter). In contrast, EWMA with α value

of 0.2 lags behind and cannot keep up with the collision rate variation. During this period, EWMA with α value of 0.2 exhibits poor performance comparing to the other methods.

TABLE IV
AVERAGE ERROR FOR FIRST 100 TRIALS OF COLLISION RATE DATASET

	AVERAGE ERROR	ERROR RATIO (%)
SENSE	0.0323	-
EWMA-0.2	0.033	2.5%
EWMA-0.4	0.0335	4%
EWMA-0.6	0.0346	7%
EWMA-0.8	0.0365	13%

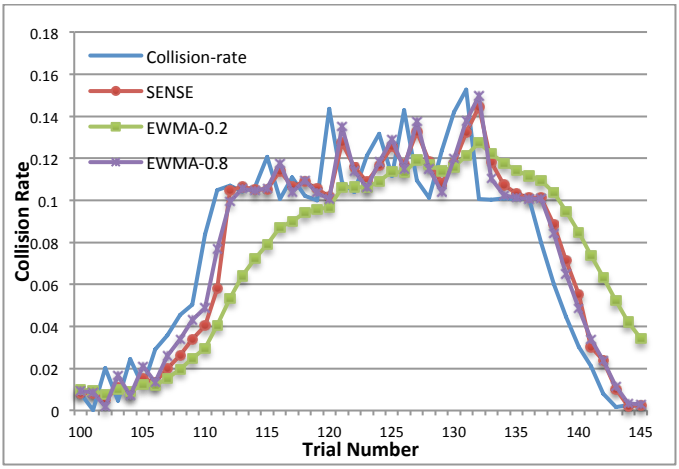


Fig. 6. SENSE vs. EWMA for smooth portion of collision rate

TABLE V
AVERAGE ERROR FOR SECOND 100 TRIALS OF COLLISION RATE DATASET

	AVERAGE ERROR	ERROR RATIO (%)
SENSE	0.0138	-
EWMA-0.2	0.0193	40%
EWMA-0.4	0.0158	14%
EWMA-0.6	0.0148	7%
EWMA-0.8	0.0142	3%

Table V lists the average error and error ratio for the last 100 trials of Fig. 4. During this interval where EWMA with α = 0.8 is clearly a better choice, SENSE behaves as EWMA predictor with high α but achieves slightly higher accuracy.

Table VI summarizes the average error and error ratio of the five different forecast schemes over the whole collision rate dataset depicted in Fig. 4. It confirms SENSE's ability to automatically adapt its performance based on network dynamics. In the case of uncorrelated behavior, SENSE gives more weight to experts with low α and in the case of correlated data, more weight is given to experts with high α value. Since EWMA does not have this capability, for the first half of the dataset, EWMA with α = 0.8 is worse than SENSE by 13% (from Table IV) and for the second half of the dataset, EWMA with α = 0.2 is significantly worse than SENSE (40% from Table V). Table VI clearly evidences that SENSE yields higher accuracy when compared to all the other four methods by at least 8% for the complete dataset. This comparison confirms SENSE's dynamic behavior to selectively and swiftly chooses the best expert according to the observed network dynamics. During noisy periods in the dataset, SENSE picks an expert with low α while during periods when the data changes more smoothly, SENSE prefers an expert with high α value.

TABLE VI
AVERAGE ERROR FOR THE WHOLE COLLISION RATE DATASET

	AVERAGE ERROR	ERROR RATIO (%)
SENSE	0.013	-
EWMA-0.2	0.0257	9%
EWMA-0.4	0.0244	8%
EWMA-0.6	0.0244	8%
EWMA-0.8	0.025	8.5%
FIXED-SHARE	0.0774	466%

D. *Prediction Summary*

The accuracy of Fixed-Share algorithm depends on prior knowledge on the range of dataset, large number of experts, and proper expert distribution that are not usually available in on-line problem. Even though more experts are essential for better accuracy, this requires a lot of trials for Fixed-Share algorithm to settle down to the best expert, resulting in significant inaccuracy under the dynamic situations.

EWMA prediction accuracy is significantly dependent on the value of α. As it is shown in our results, finding the proper α value requires prior statistical information about data, which is again not available in on-line problems.

SENSE eliminates the requirement for this prior knowledge by using smart experts. SENSE dynamically determines appropriate expert values and weights to minimize prediction error. Our results showed that SENSE agility results in best prediction performance under various scenarios and data set.

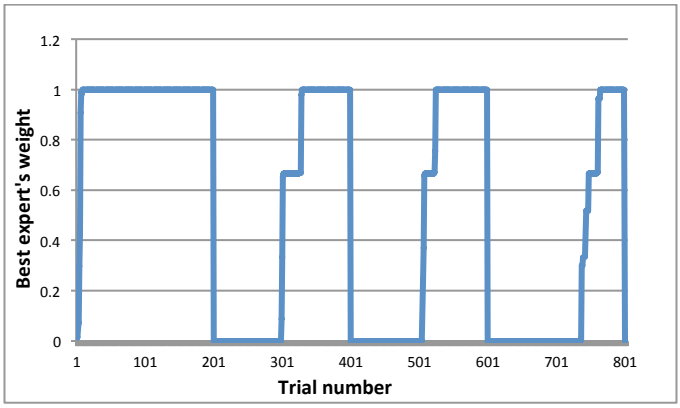
V. IMPACT OF SENSE'S TECHNIQUES AND PARAMETERS

In this section, we examine the impact of SENSE's parameters and techniques, i.e., Level-shift and META-learning. We start by running the same experiments used in [1] which were designed to show how Fixed-Share algorithm tracks the predictions of the best expert; tracking best expert's predictions has been shown to improve prediction accuracy when compared to the Static Experts for rapidly changing data. Our results show that SENSE's best expert weight adjustment mechanism performs as well as Fixed-Share for rapidly changing data.

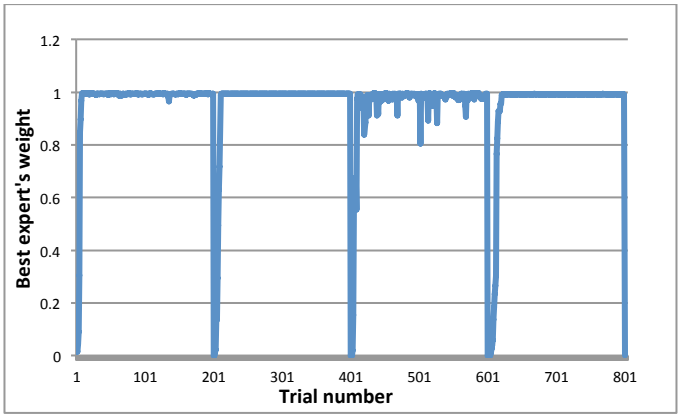
A. *Best Expert's Weight Recovery*

As described in Section II.B, Fixed-Share was proposed to overcome the slow weight recovery of the best expert in the Static Expert algorithm. This feature of Fixed-Share was evaluated in [1], which reports on how Fixed-Share tracks the predictions of the best expert compared to the Static Experts algorithm. We conducted exactly the same experiments described in [1]; our input data consists of a sequence of 800

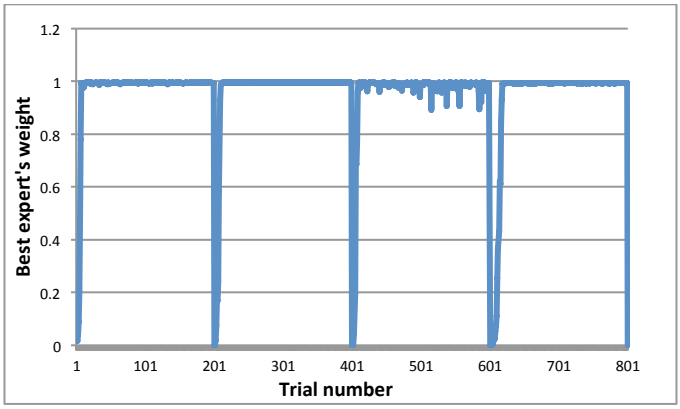
trials with three level shifts at trials 201, 401 and 601. For trials in the range of [1,200], [201,400], [401,600], and [601,800], the outcome y_t, is 0, 10, 5, and 15, respectively. The overall prediction was based on 64 experts. Every group of 200 trials has its own best expert, which changes at every shift. Best expert's prediction comes from a uniform random distribution on a smaller range than typical experts. For example, for trial [0,200], since the outcome is always 0, expert's values are chosen from a uniform random distribution between $(0, \frac{1}{2})$ and $(0, \frac{1}{2}\sqrt{0.1})$ for the typical and best experts, respectively.



(a) Best expert's weight in Static Expert Algorithm



(b) Best expert's weight in Fixed-Share algorithm



(c) Best expert's weight in SENSE

Fig. 7. Best expert's weight recovery test

Fig. 7(a), (b), and (c) plot the weight change of the best expert in the three algorithms during four 200-trial segments. Confirming the results reported in [1], Fig. 7(a) and (b) show that in the first 200-trial segment, Static Experts performed comparably to Fixed-Share, whereas in the remaining three segments, it considerably under-performed. For Static Experts, it takes almost 100 trials for the new best expert's weight to approach 1 from almost 0. In contrast, as shown in Fig. 7(b), Fixed-Share can quickly learn the new best expert for the current segment. We should note that, as in [1], the sharing degree parameter (α) is set to its best value considering the number of shifts in data, which is not always the case. Our results as plotted in Fig. 7(c) confirm that SENSE can learn as quickly as Fixed-Share. SENSE's quick learning ability is accomplished using Level-shift.

B. Impact Of Parameters

In this section, we evaluate the effect of SENSE's tunable parameters such as number of experts, β, η_{min} and η_{max}. Although results presented in this section are from experiments using datasets following sine wave patterns only, we observed similar results when we ran these experiments with our other datasets.

The experiments whose results are shown in Fig. 8 evaluate SENSE's sensitivity to the number of experts. We run SENSE with different numbers of experts on a sine wave input over a range of frequencies. The values of α are uniformly distributed between 0 and 1; for instance, in the case of 4 experts, we use α values of 0.2, 0,4, 0.6 and 0.8. Table VII shows α values used for this experiment. As can be observed in Fig. 8, SENSE's performance changes only slightly when the number of experts increases beyond 2. This is consistent with our observations in [2].

TABLE VII
A VALUES FOR DIFFERENT NUMBER OF EXPERTS

Number of experts	Values of $\boldsymbol{\alpha}$
1	0.5
2	0.25, 0.75
4	0.2, 0.4, 0.6, 0.8
8	0.125, 0.25, …, 0.875, 1
16	0.0625, 0.125, …, 0.875, 0.9375, 1
32	0.03125, 0.0625, …, 0.9375, 0.968, 1
100	0.01, 0.02, 0.03, …, 0.98, 0.99, 1

Fig. 9 shows the impact of META-learning's β parameter on SENSE's behavior by plotting the average error-frequency curves for different β values. We observe that the difference in accuracy is almost indistinguishable for different β. This can be explained by the fact that each expert does its best to keep track of the input data. META-learning is invoked only when errors tend to continuously increase or decrease since it is designed to severely penalize static experts that maintain their prediction regardless of current measurements.

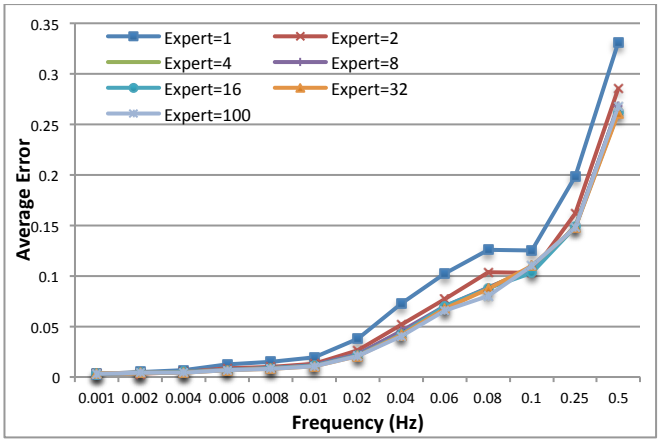


Fig. 8. SENSE's sensitivity to number of experts over sine waves

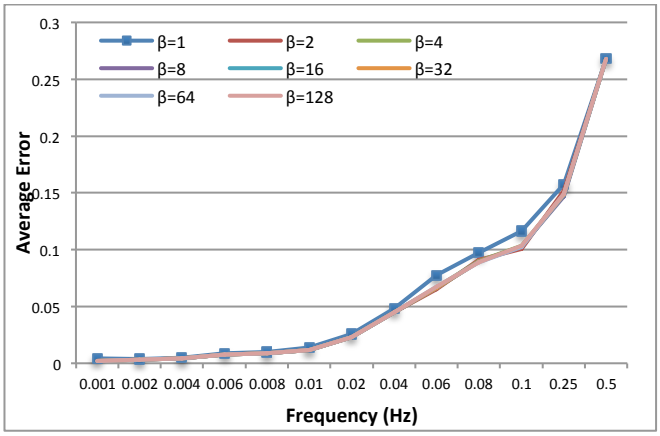


Fig. 9. SENSE's sensitivity to $\boldsymbol{\beta}$ over sine waves

We also evaluated the impact of the META-learning parameters, η_{min} and η_{max} on SENSE's performance. Fig. 10 shows the average error rate for each pair of (η_{min}, η_{max}) using, as input, to sine waves with different frequencies. We observed that different values of η_{min} and η_{max} do not have significant effect on SENSE's performance. In our experiments, we used η_{min} = 10 and η_{max} = 100.

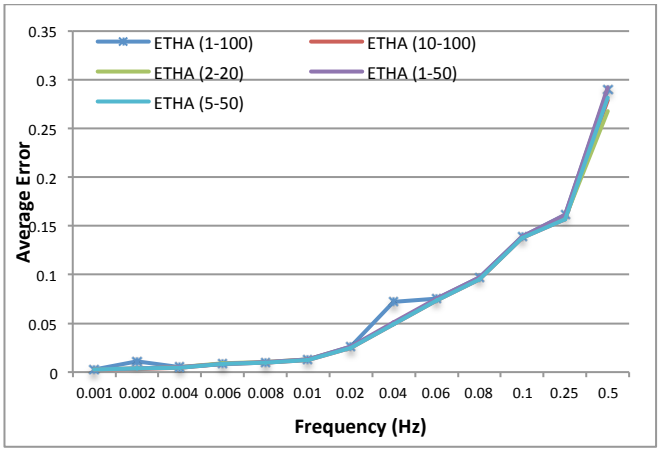


Fig. 10. SENSE's sensitivity to ETHA-min and ETHA-max over sine waves

C. *Impact of Level-shift and META-learning*

We evaluate the effect of the Level-shift and META-learning methods on SENSE's performance. Fig. 11 and 12 show the increase in accuracy (percentage of average error improvement) when SENSE uses: (1) Level-shift only, (2) META-learning only, and (3) Combined Level-shift and META-learning. Both figures confirm that these techniques improve the performance of SENSE. Note that the improvements resulting from Level-shift on RTT are much higher than on collision rate. The reason is that the RTT data has a larger number of level shifts and SENSE's Level-shift mechanism can detect them and adjust the experts' weights to follow the variations in the data. On the other hand, in the collision rate dataset, data fluctuates significantly and does not trigger the Level-shift mechanism.

Similarly to Level-shift, META-learning yields larger contribution to SENSE's performance for the RTT dataset than collision rate. And again, the reason is that the RTT dataset exhibits smoother behavior; therefore, META-learning is able to effectively increase the weight of "good" experts and decrease the weight of "bad" experts, which improves SENSE's performance overall. Consequently, the combined improvement of both techniques for the RTT dataset is almost 25% and just below 10% for the collision rate dataset.

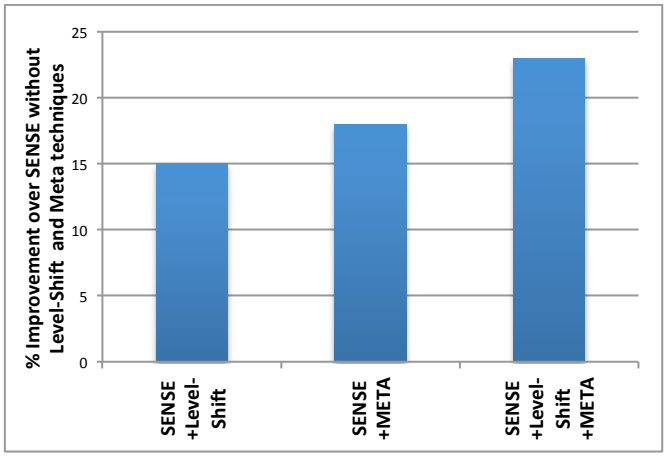


Fig. 11. Impact of Level-shift and META-learning methods on RTT dataset

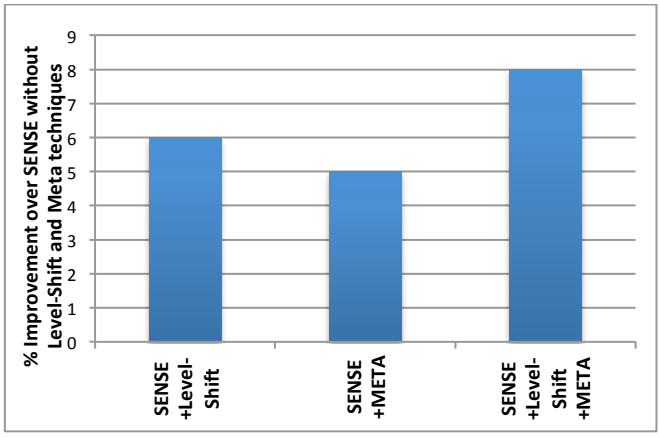


Fig. 12. Impact of Level-shift and META-learning methods on collision rate dataset

VI. Related Work

Several network protocols and applications make use of heuristics to estimate and adapt to the dynamics of the

underlying network. Since the literature on the topic is quite extensive, in this section, we focus on reviewing work that is more closely related to ours.

EWMA is a well-known technique adopted by several communication protocols. As previously pointed out, TCP uses EWMA to estimate near-term round-trip time (RTT), which is used to set TCP's retransmission timeout (RTO). Since, depending on the network environment, RTTs may vary considerably in short timescales, a number of mechanisms have been proposed to either replace or augment EWMA. DualPats, a real time TCP throughput prediction service for distributed applications, was introduced in [13]. It utilizes EWMA to make throughput predictions of large transfers augmented with active probing. In [12], EWMA along with other simple linear predictors was employed to show that; in general, history-based methods predict the throughput of TCP transfers more accurately than formula-based techniques, i.e., mathematical models that express TCP performance as a function of network path characteristics. In [16], collision rate is estimated by an EWMA based mechanism to dynamically adjust the contention window parameters in 802.11 MAC protocol.

More recently, a few efforts have used machine learning techniques to estimate near-term network variables. For instance, the work in [2] proposed a TCP RTT predictor based on a simple yet efficient machine learning algorithm called Fixed-Share [1]. The results presented in [2] show that, for a variety of network scenarios and conditions, the proposed Fixed-Share based predictor was able to improve RTT estimation significantly (thus yielding higher throughput) compared to existing approaches. Support Vector Regression (SVR) [10] also introduced a machine learning method, which can accept multiple inputs to generate accurate predictions. This method was used in [11] to predict the end-to-end TCP throughput for arbitrary file sizes.

A variant of the Fixed-Share approach has also been employed in the context of medium-access control (MAC). More specifically, in [9], a collision-free schedule based MAC that uses Fixed-Share to predict offered traffic load was proposed. Simulations as well as testbed results show the benefits of traffic prediction to schedule flows at the MAC layer in terms of delivery delay and delivery ration when compared to contention based MAC protocols. In [4], a method to predict direct and staggered collision probabilities of each node in WLANs has been introduced. Using information from an access point (AP) about network traffic broadcast as well as the AP's local measurements, each node obtains a spatial picture of the network in order to estimate probabilities of collisions locally. Similar techniques to the one used in [4] have been employed in [5] to improve throughput and link adaption in 802.11 networks with hidden terminals. In particular, a link adaption algorithm, in which nodes estimate the channel conditions by comparing the observed loss statistics to the expected loss statistics based on the estimated collision probability, is employed to select the ideal modulation rate under the estimated network conditions.

VII. Conclusions and Future Works

In this paper, we introduced SENSE (Smart Experts for Network State Estimation) a novel network state predictor based on a simple, yet efficient machine learning technique called Fixed-Share. SENSE improves the Fixed-Share algorithm by employing Exponentially Weighted Moving Average (EWMA)-based "smart" experts, META-learning, and Level-shift techniques. Our experiments on both synthetic and real datasets confirm that SENSE can automatically adapt to fluctuations of different time scales, which sets it apart from "static" techniques such as "pure" EWMA and Fixed-Share. Our experiments over a variety of datasets indicate that SENSE provides higher prediction accuracy over the Fixed-Share algorithm and EWMA.

As future research directions, we plan to apply SENSE to various network protocols such as IEEE 802.11e and X-MAC, which require channel state estimation to achieve better performance. To flexibly manage the degree of differentiation among classes of IEEE 802.11e traffic, we plan to adjust the protocol's contention window based on the collision rate forecast by SENSE. We also plan to use SENSE's collision rate forecast to dynamically enable/disable the RTS/CTS feature in 802.11.

In power-aware MAC protocols such as X-MAC, we will develop an algorithm to dynamically adjust the sleep time of nodes according to the traffic load predicted by SENSE. We will also continue to improve SENSE; for example, we plan to devise a mechanism that allows the experts' smoothing factor, α, to be automatically derived based on the input data.

References

[1] M. Herbster and M. Warmuth, "Tracking the best expert," in Machine Learning. Morgan Kaufmann, 1995, pp. 286–294.

[2] B. Nunes, K. Veenstra, W. Ballenthin, S. Lukin, and K. Obraczka, "A machine learning framework for TCP round-trip time estimation". EURASIP Journal on Wireless Communications and Networking, March 2014.

[3] Y. Edalat, J. S., Ahn and K. Obraczka, "Network state estimation using smart experts". In Proceedings of the 11th International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services, 2014, pp. 11-19.

[4] M. Krishnan and A. Zakhor, "Local estimation of probabilities of direct and staggered collisions in 802.11 WLANs," in Proc. IEEE International Workshops on Global Telecommunications (GLOBECOM), 2009, pp. 1 – 8.

[5] M. Krishnan and A.Zakhor, "Throughput Improvement in 802.11 WLANs using Collision Probability Estimates in Link Adaptation," in Proc. of IEEE Wireless Communications & Networking Conference, Sydney Australia, April 2010.

[6] N. Littlestone and M. Warmuth, "The weighted majority algorithm," in Foundations of Computer Science, 30th Annual Symposium on, oct-1 nov 1989, pp. 256 –261.

[7] N. Cesa-Bianchi, Y. Freund, D. Haussler, D. Helmbold, R. Schapire, and M. Warmuth, "How to use expert advice," Journal of the ACM, Jan 1997, PP. 427-485.

[8] D. Helmbold, D. Long, T. Sconyers, and B. Sherrod, "Adaptive Disk Spin-Down for Mobile Computers," ACM/Baltzer Mobile Networks and Applications, December 2000, 5(4):285–297.

[9] P. Vladislav and K. Obraczka, "Collision-free medium access based on traffic forecasting," In World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2012 IEEE International Symposium on a IEEE 2012, pp. 1-9.

[10] A. J. Smola and B. Scholkopf, “A tutorial on support vector regression” Statistics and Computing , 2004, 14:199–222.
[11] M. Mirza, J. Sommers, P. Barford, and X. Zhu, “A machine learning approach to TCP throughput prediction.” In ACM SIGMETRICS Performance Evaluation Review, ACM, 2007, vol. 35, no. 1, pp. 97-108.
[12] He, Qi, Constantine Dovrolis, and Mostafa Ammar, “On the predictability of large transfer TCP throughput.” In ACM SIGCOMM Computer Communication Review, 2005, vol. 35, no. 4, pp. 145-156. ACM.
[13] D. Lu, Y. Qiao, P. Dinda, and F. Bustamante, “Characterizing and predicting TCP throughput on the wide area network,” in Proc. 25th IEEE ICDCS, Columbus, OH, Jun. 2005, pp. 414–424.
[14] J. S. Hunter, The Exponentially Weighted Moving Average, Journal of Quality Technology 18: 203- 210, 1986
[15] V. Jacobson, “Congestion avoidance and control,” *SIGCOMM Comput. Commun. Rev.*, vol. 25, no. 1, pp. 157–187, 1995.
[16] L. Gannoune and S. Robert, “Dynamic tuning of the contention window minimum (CWmin) for enhanced service differentiation in IEEE 802.11 wireless ad-hoc networks,” in Proc. IEEE Personal, Indoor and Mobile Radio Communications 2004 (PIMRC ’04), Sept. 2004, vol. 1, pp. 311–317.

Yalda Edalat received her B.S. in Software Engineering from Najaf-abad University in Iran and her M.Sc. in Computer Engineering from San Jose State University in 2010. She is currently a Ph.D. candidate in Computer Engineering at UC Santa Cruz. Her research interests center around the application of computational intelligence techniques to computer networks.

Jong-Suk Ahn received the Ph.D. and M.S. degrees in Electrical Engineering from the University of Southern California in 1995 and from the Korean Advanced Institute of Science and Technology in 1985, respectively. He also received the B.S. degree from the Seoul National University in 1983. He was a visiting researcher in ISI/USC in 2001 and in the University of California in Santa Cruz (UCSC) in 2013. He was awarded the Gaheon Prize for the best paper in 2003 from the Korean Information Science Society. He is currently a professor at Dongguk University in Korea. His major interests are sensor networks, dynamic error control and flow control algorithms over the wireless Internet, network simulation techniques, IEEE 802.11, and IEEE 802.15.4.

Katia Obraczka is a Professor of Computer Engineering at the University of California, Santa Cruz. Prof. Obraczka's research interests span the areas of computer networks, distributed systems, and Internet information systems. Her lab, the Internetwork Research Group (i-NRG) at UCSC, conducts research on designing and developing protocol architectures motivated by the internets of the future. She has been a PI and a co-PI in a number of projects sponsored by government agencies (e.g., NSF, DARPA, NASA, ARO, DoE, AFOSR) as well as industry (e.g., Cisco, Google, Nokia). She is a Fellow of the IEEE.