On the Practicality of Learning Models for Network Telemetry

Soheil Jamshidi, Zayd Hammoudeh, Ramakrishnan Durairajan, Daniel Lowd, Reza Rejaie
University of Oregon

Walter Willinger
NIKSUN, Inc.

***Abstract*—Today's data plane network telemetry systems enable network operators to capture fine-grained data streams of many different network traffic features (e.g., loss or flow arrival rate) at line rate. This capability facilitates data-driven approaches to network management and motivates leveraging either statistical or machine learning models (e.g., for forecasting network data streams) for automating various network management tasks. However, current studies on network automation-related problems are in general not concerned with issues that arise when deploying these models in practice (e.g., (re)training overhead).**

In this paper, we examine various training-related aspects that affect the accuracy and overhead (and thus feasibility) of both LSTM and SARIMA, two popular types of models used for forecasting real-world network data streams in telemetry systems. In particular, we study the impact of the size, choice, and recency of the training data on accuracy and overhead and explore using separate models for different segments of a data stream (e.g., per-hour models). Using two real-world data streams, we show that (i) per-hour LSTM models exhibit high accuracy after training with only 24 hours of data, (ii) the accuracy of LSTM models does not depend on the recency of the training data (i.e., no frequent (re)training is required), (iii) SARIMA models can have comparable or lower accuracy than LSTM models, and (iv) certain segments of the data streams are inherently more challenging to forecast than others. While the specific findings reported in this paper depend on the considered data streams and specified models, we argue that irrespective of the data streams at hand, a similar examination of training-related aspects is needed before deploying any statistical or machine learning model in practice.

I. Introduction

Recent advances in the programmability of network data plane (e.g., [1]) enable network operators to monitor their desired traffic features at the line rate. For example, individual switches across a campus or data center network can emit a per-second data stream of loss rate [2], flow arrival rate [3] or queue occupancy [4] to a central collector as input to a telemetry task for detecting performance or security-related events (see § II). The availability of network data streams coupled with the increasing complexity of today's networks motivates a data-driven approach for network management and security that can be usually cast as a prediction [5], [6], [7] or a classification [8], [9], [10] problem. In particular, forecasting techniques are used to predict the likely future values of a network data stream based on its past values. The impressive success of deep learning (e.g., recurrent neural network or RNN) techniques in other fields combined with their ability to learn short- and long-term dependencies in data streams make them a promising candidate for forecasting network data streams. This, in turn, has motivated several prior studies to rely on different neural networks (NNs) or statistical models to forecast various data streams in wireless or mobile networks, ranging from the throughput of individual TCP connections [11] and intensity of (per user and aggregate) traffic [12] to aggregate traffic [13] or traffic at a base station [14]. These studies typically consider a significant volume of past data for training which is time-consuming and typically requires significant amounts of computational resources. In addition, they evaluate the overall accuracy of a forecasting model in an off-line manner. To the best of our knowledge, prior studies on forecasting network data streams have not addressed the following important issues regarding the feasibility and deployability of the proposed models in the context of a network telemetry system (e.g., for anomaly detection) that operates in a streaming fashion: *(i)* How can the volume or selection of training data be adjusted to reduce training overhead without degrading forecasting accuracy? *(ii)* How often does a model have to be retrained to maintain a sufficient level of accuracy? *(iii)* Does the accuracy of forecasting models change across different segments of a data stream, and if so, in what manner? *(iv)* How do model selection (i.e., statistical vs. NN) and model casting (i.e., generic vs. per hour) affect the answers to the above questions?

In this paper, we tackle the above issues regarding the feasibility and deployability of forecasting models. In particular, we adopt the following methodology to compare the accuracy of a type of deep learning model (i.e., long short term memory (LSTM) models) with the accuracy of a popular statistical forecasting technique (i.e., seasonal autoregressive integrated moving average (SARIMA) models). We consider two network data streams, namely a per-second flow arrival rate process for all incoming flows (RAF) and all incoming web flows (RWF) to a campus network, respectively. These data streams represent the type and resolution of data that is commonly captured by modern telemetry systems [1], [15]. We show in § IV that these data streams exhibit very different characteristics and thus enable us to demonstrate the effect of these characteristics on the forecasting accuracy. We consider both generic and per-hour versions of both LSTM and SARIMA models for six evenly spaced hours to assess how different characteristics of data streams affect the accuracy of forecasting the next five seconds of data streams. For per-hour models, we explore how the volume, selection, and recency of the training data can affect the accuracy of the resulting model.

978-3-903176-27-0

This, in turn, reveals opportunities to reduce training overhead with minimal or no effect on the forecasting accuracy.

As the main contribution, we report in this paper on a number of empirical findings from our analyses that offer valuable insights for deploying the considered models in practice. First, we observe that for per-hour models, increasing the volume of training data beyond 24 hours of a recent window or similar past instances (i.e., same hour, the same day of the week) does not improve the accuracy of the model but linearly increases the training overhead. Second, we find that per-hour models that are trained with a 24-hour data stream exhibit a comparable accuracy with a generic model that is trained with 30 days of training data. In practical terms, these findings show that we can significantly decrease training overhead without compromising the accuracy of LSTM models. Third, observing that changing the recency of the training data by a few weeks does not affect the accuracy of our per-hour forecasting models suggests that LSTM models do not require frequent (re)training. Fourth, we notice that for our RWF data stream, the prediction accuracy of all per-hour models is lower during the night hours and higher during day hours. This observation shows that certain segments of the data stream are inherently more difficult to forecast than others. Fifth, in the case of our RAF data stream, its more bursty behavior compared to the RWF data stream tends to result in higher accuracy for LSTM models in forecasting RWF compared to SARIMA models. Finally, all the models show wider variations in forecasting accuracy across different samples of our RAF data stream. It is important to note that while the reported findings depend specifically on our considered data streams, the described methodology is applicable to any data stream and should be part of an in-depth assessment of the practical issues that arise when deploying any statistical or learning models.

The rest of this paper is organized as follows. In § II, we present an example telemetry task to illustrate two requirements for a forecasting model in such a setting. § III provides some background on LSTM and SARIMA models. Our empirical approach is described in § IV. We assess the feasibility and accuracy of forecasting models in § V. § VI describes the limitations of this study. § VII discusses some of the closely related prior work. We summarize and outline future work in § VIII.

II. Illustrative Example

We present an overview of the anomaly detection (telemetry) task that incorporates a forecasting model to illustrate the implications of task requirements and the characteristics of network data streams on training and configuring the model. Consider a programmable switch (e.g., Tofino [16]) that monitors a collection of desired traffic features at the line rate and emits a separate, fine-grained (e.g., per second) data stream for each feature to a remote collector. The availability of high resolution (per second or short timescale) data streams from the data plane telemetry systems enables forecasting models not only to capture finer variations in traffic but also to facilitate faster detection of anomalies. A forecasting model first requires using a history of these data streams for initial training. Then, a trained model can be deployed at the collector and uses the most recent n values of the data stream to forecast the next h values. Then, if the gap between the forecasted and actual h values is larger than the error in the model, this could be viewed as an indication of an anomaly. Subsequently, the telemetry task may trigger further examination of other traffic features and invoke proper actions on forwarding switch pipeline (e.g., dropping or re-routing the relevant packets) to mitigate the problem. Furthermore, depending on the characteristics of the captured data stream, the model may require periodic retraining such that its forecasted values remain sufficiently accurate.

This example illustrates two important requirements for practical deployment of learning models into a telemetry system:

- **R1**: Forecasting models are often trained using a large volume of past data (a few days to weeks or months) which typically takes a long time (hours to days). However, in telemetry systems, a long history of a data stream may not be available and a significant training overhead may not be feasible. To address this requirement, we explore how the training overhead can be reduced by limiting the volume (and selection) of training data without affecting the accuracy of the resulting model.
- **R2**: Since network data streams may evolve (over time), we need to periodically re-train a forecasting model to maintain sufficiently high accuracy. The duration of a retraining should be much shorter than the period for refreshing the model. We examine the effect of the recency of training data on the accuracy of forecasting models to shed light on the required frequency of retraining and its relationship with (re)training overhead.

III. Background

This section provides a brief background on the forecasting models that we consider in this study.

LSTM. Long short-term memory (LSTM) is a recurrent neural network (RNN) architecture [17] that has been widely used for forecasting in various domains, including networking [18], [19], [20], [21]. The key feature of the LSTM model is its ability to capture potential long-range dependencies in the data stream. The LSTM model has several parameters related to its architecture, optimizer, and training approach that should be properly configured such as the number of stacked layers, number of hidden nodes, activation function, dropout, and the number of passes over the data during training (i.e., epochs). To train an LSTM model, the data stream X is divided into two separate sets: M consecutive values of ($X(t_0 - M)$ to $X(t_0 - 1)$) for the training set, and the immediate next N values ($X(t_0)$ to $X(t_0 + N)$) for the testing set. We consider non-overlapping training and testing sets, and further split each set into samples using a rolling window that has been shown to be an effective strategy [22], [23], [24], [25], [26]. Each window (of length (A+B) consecutive values) is considered as

a *sample*. In each sample, the first A values are used as *history* to forecast the next B values. The LSTM output size (S), is one of the LSTM's parameters. If the forecasting horizon H (i.e., the number of data points we expect to forecast) is larger than S, then we have to forecast by rolling the window $\frac{H}{S}$ times and using either the forecasted or actual values of the data stream as history for the next S values. All samples are divided into random and mutually exclusive batches of a certain size where each batch is used for a separate round of training until all samples are utilized. This training process can be repeated multiple times (epochs) to improve the accuracy of the model.

This description reveals several training parameters for an LSTM model: the relative size and selection of training and testing sets, window size, prediction size (LSTM output), window overlap, batch size, and the number of epochs. We found optimal values for each during our tuning process.

(S)ARIMA. Auto-Regressive Integrated Moving Average (or ARIMA) is a popular statistical technique for forecasting stationary data streams [27], [28], [29], [30]. ARIMA has several configurable/tunable components: the auto-regressive component (p) that specifies the number of lags (past values) in the model, the integrated component (d) that represents the degree of differencing, a moving average component (q) that represents the error of the model as a combination of previous error terms. Subsequently, several variants of ARIMA were also proposed. For example, to model time series with periodic characteristics, Seasonal ARIMA (or SARIMA) model [31] was proposed. SARIMA incorporates seasonal auto-regressive (P), differencing (D), and moving average (Q) components as well as a seasonal frequency (s). Given the inherently periodic (daily, weekly) characteristics of most networking data streams, we primarily focus on the SARIMA model in this study.

The process of training a SARIMA model is as follows: SARIMA models assume that the input data is stationary. The stationarity of the time series can be achieved via transformation (e.g., logarithms) to stabilize the time series *variance* and differencing to eliminate the *trend*. Besides, the decomposition can help to de-seasonalize the time series if necessary [32]. We use Augmented Dickey-Fuller (ADF) test [33] as a unit root test to check for deterministic trend stationarity as well as Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test [34] to complement the unit root test. On our data stream, the ADF test confirms the stationary nature of data with 99% confidence while KPSS test indicates that the timeseries are not stationary. In such cases, it is suggested to apply differencing. The KPSS test passes when applied on differenced data.

To identify the best combination of model parameters, we consider the range of values for p, d, and q obtained from the auto-correlation function (ACF) and the partial auto-correlation function (PACF) plots (not shown due to space constraints). Then, we train separate SARIMA models for a different combination of parameters in parallel and select the best model based on their performance on the validation set.

IV. Methodology

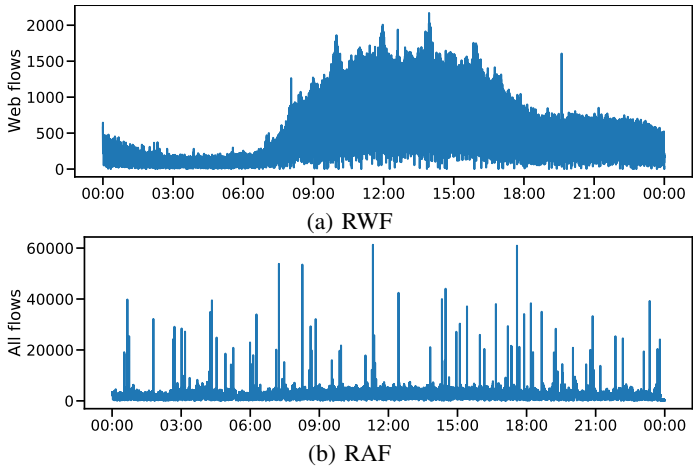
In this section, we discuss our methodology for exploring our motivating questions on incorporating forecasting models into anomaly detection systems. We start by presenting the network data streams and the selection of forecasting models that we consider in this study as well as our training and testing strategies for these models.

A. Network Data Streams

We focus on flow arrival rate per second (i.e., number of unique incoming network flows that are observed in each second) as our target data streams since it is used as the input of telemetry tasks (e.g., [15] [35] [36]). To this end, we use un-sampled NETFLOW data for all the connections between the University of Oregon campus and the Internet to extract the rate of (incoming) web flows (RWF) and rate of (incoming) all flows (RAF) per second. Our NETFLOW dataset covers a 10 month period from 1/5/2018 till 28/2/2019 where each daily segment of our dataset represents on average 8.8 TB of incoming traffic, associated with 200 million flows, from 3.6 million unique source IPs with 39k unique sources [37].

Fig. 1a and 1b present the variations of our two data streams in a typical day (2018-09-26) and illustrate that these two data streams exhibit very different characteristics as follows. First, RWF shows significantly smaller variations that are dominated by a pronounced diurnal pattern compared to RAF. Therefore, forecasting these data streams is likely to present different challenges. Second, the RWF data stream exhibits a high, low, and moderate degree of variations during the night (0-8), day (8-16) and evening (16-24) hours, respectively. However, the degree of variations in the RAF data stream is very similar across all hours. This observation motivates us to consider *training a separate forecasting model for different hours of the day could lead to a higher accuracy*.

To examine the (dis)similarity of both data streams across different days, we consider 13 scale-invariant attributes of each daily segment for both data streams (proposed by Kang *et al.*



(a) RWF

(b) RAF

Fig. 1: Sample daily variations of our two target flows arrival rates (per second).

[38])[1], apply principal component analysis (PCA) to identify the top two principal components for each data stream. The blue dots and orange crosses in Figure 2 present the values of the top two principal components for RWF and RAF data streams in separate days, respectively. From this figure we make two key observations. First, while both data streams capture flow arrival rates, RAF data stream exhibits wide variations across different days while the characteristics of RWF data stream across different days are more consistent. Therefore these two data streams represent very different input network timeseries for our forecasting models. Second, the two pronounced clusters of blue dots in Figure 2 are related to weekdays (solid black line) and weekends (dotted red line). This evidence indicates that the RWF data streams have distinctly different characteristics on weekdays compared to weekends.

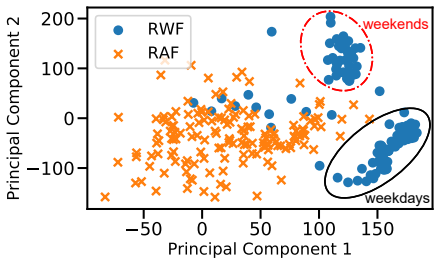


Fig. 2: Comparison of per-second data streams using principal components.

B. Forecasting Models

To examine how the type (i.e., SARIMA vs. LSTM), volume, and selection of training data and other inputs of a model affect its forecasting accuracy, we consider the following four models. Using a short forecasting horizon improves model accuracy but limits the time to react to a detected anomaly and vice versa. To strike a balance between these two opposing requirements, we have examined 5, 10 and 20 seconds (5, 10, 20 values) forecasting horizons and selected 5-second forecasting horizon in all of our models.

LGW(w): This LSTM (L), generic (G) model forecasts any 5-second segment of a data stream by using the recent (w) hours window (W) of a data stream for training. A common practice in training an LSTM model is to use a large volume of a data stream for training and explore whether adding other features improve the accuracy of the models. LGW models represent this common approach for using LSTM models. We use a 30-day recent window of each data stream to train LGW models (i.e., w=30*24) [2].

LHW(x, w): This per-hour (H) LSTM (L) models forecast any 5-second segment of a data stream in a *specific target hour* x using the past w hours of the data stream for training. We train a separate model for six evenly spaced target hours (i.e., x is set to 3, 7, 11, 15, 19, 23) to explore the accuracy of our model for forecasting different parts of the data streams. By setting w to 1, 10, 24, and 48 hours, we also examine how the volume of training data affects the accuracy of these per-hour models. We also consider older windows of training data (from prior weeks) to explore the effect of data recency on the model accuracy.

LHI(x, i): This LSTM (L) model forecasts any 5-second segment in a specific target hour (H) x using the past i instances (I) of the target hour x in the data stream for training. An instance of a specific target hour x is defined by its hour-of-day and its day-of-week. For example, to train a model for 7am hour on a Monday, we use the 7-8am segment of the data stream from i prior Mondays for training. This training strategy is intuitively motivated by the repeating weekly pattern of some network data stream, such as RWF, which suggests that the most relevant training data is the past instances of the same hour.

SHW(x, w): This SARIMA (S) model forecasts any 5-second segment in a *specific target hour* (H) x using the past w hours of the data stream for training (similar to *LHW(x,w)*). By changing w, these models represent a statistical forecasting technique with a different volume of training data. Note that SARIMA model can only use the most recent window of data for training while LSTM can rely on any past window of data for training.

This collection of models enables us to compare generic and per hour models while exploring the effect of the volume, recency, and selection of training data on the overall accuracy of forecasting models.

C. Tuning Models

We take the following steps to properly tune each one of the selected models.

LSTM Models. We tune individual LSTM models by examining the accuracy of the model across thousands of different configurations and training parameters using random search on the validation data that is separate from the training and testing set. We then select the model that exhibits the highest validation accuracy. We leverage the sliding window approach to break both training and testing datasets into samples using the following parameters: window size=150[3], window overlap=135 and forecasting horizons=5 values. These parameters result in 233 samples in each hour of our data streams. The number of epochs is 300 for all models and batch size is 300 for LGW and 150 for other LSTM models. We use checkpoints to identify the model with the lowest validation loss. We utilize a Keras implementation of LSTM with Tensorflow backend. Table I summarizes the main hyperparameters including learning rate (LR), activation function (AF), drop rate (DR), number of hidden layers (LY), number of hidden nodes (NHN), and optimizer (OPT) for our tuned LSTM models.

[1]These attributes include trend, spike, linearity, curvature, entropy, skewness, and ACF lags, compared to scale variant attributes such as mean, median, minimum, and maximum

[2]We considered other additional input features such as day-of-week and time-of-day. However, they did not improve the accuracy of LGW models.

[3]Given the selected window size (150) for our samples, having more than 300 samples/hour results in increasing overlap between samples.

TABLE I: Hyperparameters of our LSTM models.

		LR	AF	DR	LY	NHN	OPT
RWF	**LHW/I**	0.005	tanh	0.1	5	64-32(x4)	Adam
	LGW	0.003	tanh	0.2	3	256	Adam
RAF	**LHW/I**	0.005	ReLU	0.2	4	32-16(x3)	Adam
	LGW	0.005	ReLU	0.2	4	64	Adam

We use mean square error ($MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$) as the loss function in our training process where n, y_i and $\hat{y}_i$ denote the forecasting horizon size, the actual and predicted values, respectively.

SARIMA Models. We train a model for 2,500 different combinations of parameters ($5p{*}5q{*}5P{*}5Q{*}2d{*}2D = 2,500$) and perform grid search to identify the parameters of the best model. To make sure that models capture the patterns in the input data stream, we confirmed that the residuals follow the normal distribution and have no auto-correlation (using $Ljung\text{-}Box$ test [39]). Table II presents the final configuration for the SHW models of different hours along with their associated training time.

D. Testing Models

For testing each model, we consider 300 randomly selected points in each test hour and use the model to forecast the next immediate 5 seconds (i.e., our forecasting horizon) of the data stream. We use Root Mean Square Percentage Error (RMSPE) for evaluating the accuracy of our forecasting models across all samples as follows: $RMSPE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\frac{(y_i-\hat{y}_i)}{y_i}^2}$ where n, y_i and $\hat{y}_i$ denote the forecasting horizon, the actual and predicted values, respectively. The normalized nature of RMSPE makes it scale-invariant and interpretable which is more appropriate for our purpose [40]. The overall accuracy of each model is presented with the summary distribution (box-and-whiskers plots where the box shows the quartiles while whiskers show 5^{th} and 95^{th} percentiles) of RMSPE across 300 random samples in each target hour. The variations of error for each model across different hours reveals the effect of temporal characteristics of data streams on the model accuracy.

For each hourly LSTM model LH*(x, w), we train seven separate models for each target hour in seven consecutive days (11/12/18 to 11/18/18)[4], test them on 300 samples in the target hour, and present the summary distribution of RMSPE for all (7*300) samples for each target hour. For LGW model, we train a single model but similarly test it on each target hour across 7 days to present the summary distribution of error for that hour. Therefore, our results are not biased towards a specific day of the week.

V. Assessing the Practicality of Forecasting Models

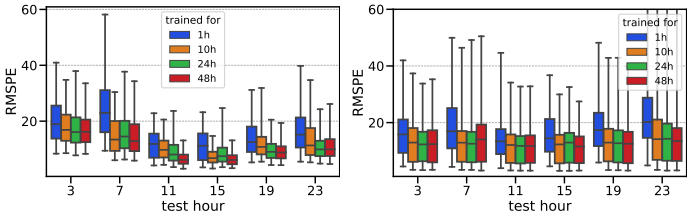
In this section, we assess the practicality of forecasting models in light of the two requirements described in § II.

[4]This is a regular week that school was in session.

A. Impacts of Volume and Selection of Training Data

We evaluate the effect of variations of the data stream as well as the volume, recency, and selection of training data on the accuracy of forecasting models. The goal is to shed light on requirement R1 mentioned in § II.

Volume of Training Data. First, we explore the question of *whether the volume of the training dataset affects the accuracy of a per-hour model?* Figure 3a and 3b present the summary distribution of forecasting error for RWF and RAF data streams across different hours using LHW models. For each hour, we show the error for four models that are trained with 1, 10, 24, and 48-hours of most recent data stream. These two figures show the following points: First, for both data streams, increasing the amount of training data initially improves the accuracy of forecasting for up to 24 hours. However, increasing the training data beyond 24 hours has a diminishing return in the accuracy of the model for most hours (except 11 and 15 hours for RWF). Second, *the accuracy of the best-trained model for RWF varies across different hours* (Figure 3a). In particular, the forecasting error during the night hours is the highest and during the day hours is the lowest. In contrast, the accuracy of models for RAF is very similar across all hours.



(a) RWF (b) RAF

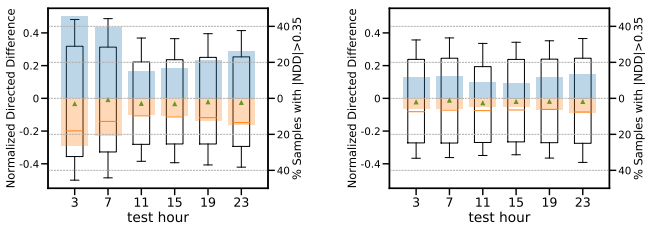
Fig. 3: Effect of the volume of training data on the accuracy of LHW(x, w) model.

To explain the difference in the accuracy of models across different hours, we examine the variations of both data streams during different hours of the day. To this end, we measure the normalized directed difference (NDD) among (150) values in each test sample that is defined as follows $NDD = \frac{max(X)-min(X)}{max(X)} * sign(argmax(X) - argmin(X))$ where X is a test sample, and $argmin(argmax)$ denotes the index of the min and max values, indicating the order in which min and max values are observed (i.e., the positive/negative direction of major change between these values). The box plots in Figure 4 present the summary distribution of NDD values across all samples in each of 6 target hours over 7 days. To further focus on larger NDD values, we also show the fraction of samples with positive (negative) NDD values for each hour that are larger than 0.35 with a blue (orange) bar using the right y-axis. The plots in Figure 4 illustrate that the normalized changes across values of individual samples are larger in all hours of RWF data streams compared to RAF data stream. In particular, hour 3 and 7 of RWF exhibit the largest normalized variations. While it is not trivial to determine which specific

TABLE II: Configuration and training time of SARIMA (SHW(x, 24)) models for both data streams.

Target hour	Data stream	(p,d,q)x(P,D,Q,s)	Train time (minute)	Data stream	(p,d,q)x(P,D,Q,s)	Train time (minute)
03	RWF	(1, 0, 3)x(3, 1, 3, 15)	62.2	RAF	(3, 1, 1)x(1, 1, 1, 6)	60.42
07	RWF	(3, 0, 2)x(2, 1, 3, 15)	129.52	RAF	(3, 1, 1)x(1, 1, 2, 6)	48.33
11	RWF	(3, 1, 2)x(2, 1, 3, 15)	124.69	RAF	(0, 1, 3)x(2, 0, 3, 6)	16.79
15	RWF	(1, 1, 3)x(2, 1, 3, 15)	156.89	RAF	(0, 1, 3)x(2, 1, 3, 6)	107.24
19	RWF	(3, 0, 3)x(3, 0, 3, 15)	63.81	RAF	(1, 0, 3)x(0, 1, 3, 6)	82.59
23	RWF	(3, 1, 1)x(2, 0, 3, 15)	18.27	RAF	(0, 1, 3)x(2, 1, 3, 6)	122.37

aspects of a data stream affects the accuracy of a forecasting model, we believe that the larger variations in specific hours offer a plausible explanation for lower accuracy of our models for those hours.



(a) RWF (b) RAF

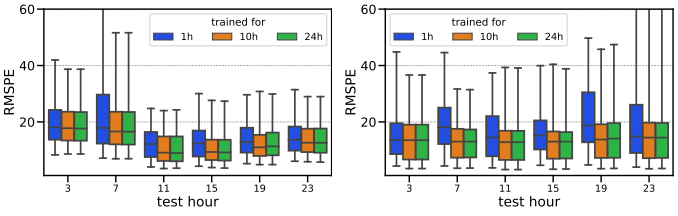
Fig. 4: Distribution of normalized directed difference (NDD) of test samples.

For the rest of the analysis, we consider the LHW models that are trained with the recent 24-hour window of the data stream for these six target hours.

Selection of Training Data. Another important question is *whether the selection of training data affects the accuracy of a model?* We use LHI (x, i) models for each target hour that is trained with 1, 10, and 24 segments of data from prior instances of the same target hour. For example, an LHI (7,10) for a Monday uses the 7-8am segment of the data stream from 10 prior Mondays for training. Figures 5a and 5b present the accuracy of the LHI models for forecasting both data streams across all six target hours that are trained with 1, 10, 24 past instances of the target hour. These results show that *increasing the number of past instances of training segments from 1 to 10 improves the accuracy of both models but adding more training data has no measurable impact.*.

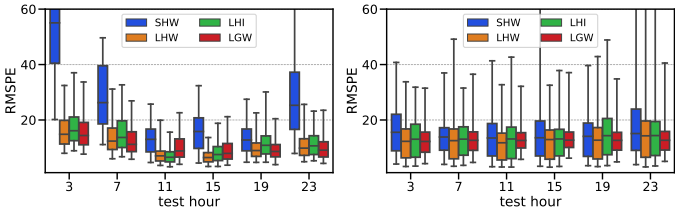
Head-to-Head Comparison of Different Models. We now compare the accuracy of all four models—LGW(30*24), LHW(x,24), LHI(x,24) and SHW(x,24)— for forecasting a 5-second horizon of different target hours of RWF and RAF data streams in Figures 6a and 6b, respectively. This figure illustrates a few important points: First, LHI (x,24), LHW (x,24), and LGW (24*30) exhibit a comparable accuracy across all hours of RWF data streams despite a significantly smaller amount of training data for LHI and LHW models. Note that the LGW model is simply a special case of the LHW model that uses 30 times more training data. Second, the accuracy of SHW models is lower particularly for hours that are difficult to forecast (i.e., 3, 7, and 23). Third, the relative pattern of changes in accuracy across different hours is very similar for RWF models – lowest accuracy in night hours, highest accuracy for day hours, and moderate accuracy in the evening. Fourth, on RAF data stream, SHW has only a slightly higher error compared to different LSTM models. All LSTM models have a very similar accuracy on RAF data stream but LGW exhibits much lower variations in error across different samples. Note that a commonly reported measure of accuracy (mean or median error) does not reveal this difference in the variations of accuracy.

Fifth, comparing all models across both data streams show that LSTM models have a similar accuracy on both data streams during night (and early morning) hours (3, 7) but higher accuracy on forecasting RWF data stream in all other hours. Interestingly, while LSTM models show a similar accuracy for RWF data stream at night hours and RAF data stream at all hours, SHW models have much lower accuracy on RWF data stream at night hours than RAF data stream. This suggests that the LSTM models are more capable to forecast RWF data stream during the night hours (3 and 7)



(a) RWF (b) RAF

Fig. 5: Effect of the selection of training data on the accuracy of LHI(x, i) model.



(a) RWF (b) RAF

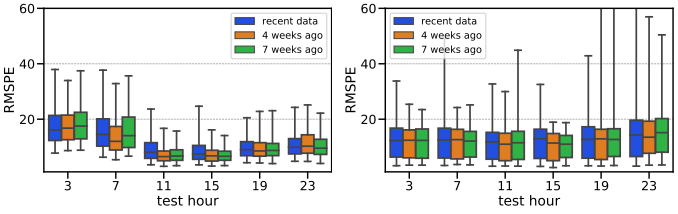
Fig. 6: Comparison of four models on different hours of RWF and RAF data streams.

despite its larger variations (as we reported in Figure 4). Later in this section, our examination of the short term pattern of error offers more insight about this problem with the SARIMA models.

B. Impacts of Recency of Training Data and Training Overhead

We next seek to evaluate the recency of training data and training overheads and how they affect the accuracy of forecasting models to shed light on R2 mentioned in § II.

Recency of Training Data. We now explore the question of *whether the recency/freshness of training data affects the accuracy of the forecasting?* More specifically, does it make any difference if we train a LHW model with different 24-hour segments of the data stream? Figures 7a and 7b depict the accuracy of LHW (x,24) models for forecasting the six target hours of both data streams using three different training datasets for each model: *(i)* the most recent 24 hours of the data stream (labeled *recent data*), *(ii)* the same 24 hours of the data stream from *4 weeks ago*, and *(ii)* the same 24 hours of the data stream from *7 weeks ago*. Surprisingly, we observe that *the recency of a (sufficiently long) training dataset has a rather minor (or no) effect on the accuracy of the model for both data streams*. This finding suggests that *a LHW model that is trained with 24 hours of the data stream has observed a sufficiently rich set of variations and does not need to be retrained frequently in the absence of any major data drift.*



(a) RWF (b) RAF

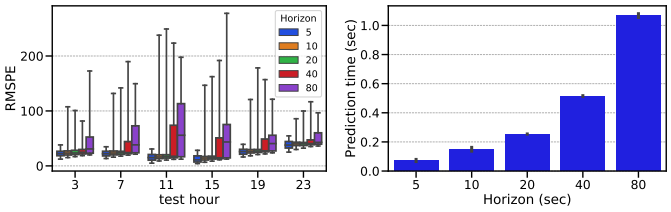
Fig. 7: Effect of the recency of training data on the accuracy of LHW model for RAF data stream.

Training Overheads. Table III presents the total training time (with 300 epochs) for LGW and LHW/LHI (with different volume of training data) forecasting models of both data streams on both CPU (using two Intel Xeon Gold 5218) and GPU (using GeForce RTX 2080 Ti with 11GB GDDR6 memory). Table III shows that (i) training time linearly increases with the volume of training data and it is 7–26x faster on a GPU than a CPU, (ii) training a model for RWF takes 2–3x longer than RAF data stream, and (iii) it is feasible to retrain a new LHW/LHI model on a daily basis using GPU or CPU whereas LGW model can be retrained only on a daily (weekly) basis using GPU (CPU). The training times for hourly SARIMA models (i.e., SHW$(x, 24)$) on CPU are reported in Table II. We observe that the training time for both data stream varies between 16-160min across different hours. This indicates that these model can be (re)trained on a daily basis.

TABLE III: Total training time of LSTM models.

Model	Volume of Training Data (hour)	RWF training time (minute)		RAF training time (minute)	
		GPU	CPU	GPU	CPU
LHW/I	1	0.4	10	0.25	6.5
LHW/I	10	5	50	1.2	20
LHW/I	24	10	105	5	40
LHW/I	48	20	200	11	76
LGW	30*24	667	8,550	190	3,850

Longer Forecasting Horizon. The results presented so far are based on the forecasting horizon of 5-second horizons of the corresponding data stream by each model. If a telemetry task requires a longer forecasting horizon, we have two options: *(i)* "re-anchoring" the model every 5 seconds by using the past 150-second values of the data stream, or *(ii)* "rolling over" the model to forecast multiple 5-second intervals by feeding the forecasted values back to the model Figure 8a presents the accuracy of the latter option, namely rolling the model over, for forecasting longer horizons of RWF stream using the LHW model for different target hours. Note that the range of the Y-axis for this figure is much larger than our prior plots. Figure 8a shows that the typical accuracy of this roll-over strategy in all hours is clearly degraded as we increase the forecasting horizon. To complement this result, Figure 8b shows the effect of forecasting horizon on inference latency (prediction time). We can observe that the prediction time linearly grows with the forecasting horizon but remains generally low, e.g., roughly 1.1 seconds to predict the next 80 seconds of the data stream.



(a) On accuracy (b) On prediction time

Fig. 8: Effect of longer forecasting horizon (with roll over strategy) on model accuracy for RWF data stream

Short Term Pattern of Error. Our analyses have primarily considered the overall notion of error based on the RMSPE measure. In this subsection, we explore the temporal pattern of the forecasting error by SHW and LHW models to examine how closely it tracks sudden changes in the original data stream. Figure 9 depicts a 120-second segment of data stream along with the forecasted values by both LHW and SHW models (with 5-second forecasting horizon) using re-anchoring and roll-over strategies. This figure clearly demonstrates that the LHW model generally tracks the variations in the data stream, especially by the re-anchoring strategy. In particular, forecasting longer horizons (beyond 20 seconds in this example) even with the LSTM model can lead to a large error when

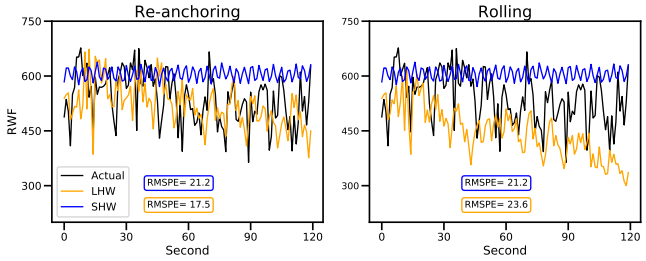


Fig. 9: Effect of prediction strategy on accuracy.

the rolling strategy is used to extend the forecasting horizon. The forecasted values by the SARIMA model simply represent average values of the data stream and do not seem to track its variations. *In summary, the LSTM models coupled with re-anchoring strategy offers the most accurate forecasting results for our data streams.*

VI. Limitation of this Study

The main limitation of our study is its focus on two specific network data streams from a particular campus network. While our methodology is certainly applicable to forecasting any data stream from any network, our trained models and findings cannot be generalized. More importantly, we argue that such a generalization of ML models—as it is done in other domains (e.g., image classification)—may not necessarily be feasible in networking. This observation is motivated by the fact that network data streams are likely to exhibit diverse (short term) characteristics across different networks. This, in turn, suggests that the training and deployment of ML models should be customized for a specific data stream from a particular network to ensure high accuracy. In short, any modeling study in networking is likely to be specific to its target setting.

VII. Related Work

In this section, we primarily focus on prior studies that apply forecasting techniques to networking data streams/time series. Several prior studies have relied on different neural networks (NNs) or statistical models to forecast various data streams in wireless or mobile networks, ranging from throughput of individual TCP connections [11] and intensity of (per user and aggregate) traffic [12] to aggregate traffic [13] or traffic at a base station [14]. These studies have focused on inherently different data streams with a coarser time scale (e.g., per few minutes or per hour) compared to our work.

There are a few other studies that use statistical and NN models to forecast backbone traffic (e.g., [41], [18], [42], [6]), using the available data stream of aggregate traffic often at a resolution of 5-15 minutes. Ramakrishnan *et al.* [43] presents the closest prior study to ours by comparing the accuracy of Recurrent NN (RNN) architectures to predict traffic volumes, packet distributions and protocols at per-second granularity, and show that RNN-based solutions outperform the statistical models in all three tasks. They present a very limited evaluation using MSE which makes the reported error specific to their non-representative per-second dataset.

These previous studies have primarily focused on the "off-line" evaluation of forecasting method on network data streams with per-minute (or coarser) resolution due to the limited availability of representative streams with per-second granularity. Therefore, the characteristics of their data streams were very different. More importantly, to our knowledge, none of the prior studies have explored the effect of the following practical issues on incorporating forecasting model into telemetry systems using network data streams with different characteristics, (i) the variations in characteristics of different segments of the data stream that is often observed in network data streams, and (ii) the effect of volume, selection, and recency of training data on different models.

VIII. Conclusion & Future Work

The sheer volume of network data stream coupled with the increasing complexity of today's network and innovation in switch data planes motivate forecasting techniques as the key ingredient to automate network management and security tasks. While great progress has been made by prior efforts in applying AI/ML to network automation, the practicality of deploying ML models such as training strategies (e.g., the volume, selection and recency of training data; and having separate models for different hours of a data stream) have not received enough attention. To shed light on this issue, this paper explores the forecasting per-second flow arrival rate for all incoming flows and incoming web flows using LSTM. Our results provide valuable insights into the ability of the forecasting models for short-term forecasting of the two data streams and elucidate the effects of training strategies, input features, among others, on the accuracy of models.

We plan to extend this study along the following directions. For one, the speed-accuracy trade-offs associated with automated inference determine how well a given telemetry task as a whole can be performed with no human operator in the loop. While the idea of exploiting the diversity in available compute and communication resources and programmability capabilities among the different hardware components to achieve the "best case" scenario is already being explored [1], how to get the network to recognize such "best case" scenarios and then operate at such "sweet spots" remains an open problem. Moreover, other open problems include creating theoretical bounds on the time required and accuracy desired to perform a certain network telemetry task, and architectural designs that are needed to enable such "sweet spot-seeking" network telemetry at scale; that is, executing hundreds or thousands of highly diverse network telemetry tasks concurrently and as fast and accurately as possible despite the uncertainties in the environment (e.g., traffic load, application mix, failure scenarios).

ACKNOWLEDGEMENT

We thank the anonymous reviewers and our shepherd, Daphne Tuncer, for their insightful feedback. This work is supported by NSF grants CNS-1320977, CNS-1719165, and CNS-1850297, and access to Intel AI DevCloud platform for training and testing of our models. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of NSF or Intel.

REFERENCES

[1] A. Gupta, R. Harrison, M. Canini, N. Feamster, J. Rexford, and W. Willinger, "Sonata: Query-driven streaming network telemetry," in *ACM Special Interest Group on Data Communication*, 2018, pp. 357–371.

[2] Y. Li, R. Miao, C. Kim, and M. Yu, "Flowradar: A better netflow for data centers," in *USENIX Symposium on Networked Systems Design and Implementation NSDI*, 2016, pp. 311–324.

[3] Z. Liu, A. Manousis, G. Vorsanger, V. Sekar, and V. Braverman, "One sketch to rule them all: Rethinking network flow monitoring with UnivMon," in *ACM SIGCOMM Conference*, 2016, pp. 101–114.

[4] X. Chen, S. L. Feibish, Y. Koral, J. Rexford, and O. Rottenstreich, "Catching the microburst culprits with snappy," in *Afternoon Workshop on Self-Driving Networks*, 2018, pp. 22–28.

[5] B. Yang, S. Sun, J. Li, X. Lin, and Y. Tian, "Traffic flow prediction using LSTM with feature enhancement," *Neurocomputing*, vol. 332, pp. 320–327, 2019.

[6] A. Lazaris and V. K. Prasanna, "An LSTM framework for modeling network traffic," in *IFIP/IEEE Symposium on Integrated Network and Service Management*, 2019, pp. 19–24.

[7] M. Mirza, J. Sommers, P. Barford, and X. Zhu, "A machine learning approach to TCP throughput prediction," *IEEE/ACM Transactions on Networking (TON)*, vol. 18, no. 4, pp. 1026–1039, 2010.

[8] A. K. J. Michael, E. Valla, N. S. Neggatu, and A. W. Moore, "Network traffic classification via neural networks," University of Cambridge, Computer Laboratory, Tech. Rep., 2017.

[9] J. Zhang, X. Chen, Y. Xiang, W. Zhou, and J. Wu, "Robust network traffic classification," *IEEE/ACM Transactions on Networking (ToN)*, vol. 23, no. 4, pp. 1257–1270, 2015.

[10] B. Yamansavascilar, M. A. Guvensan, A. G. Yavuz, and M. E. Karsligil, "Application identification via network traffic classification," in *Computing, Networking and Communications (ICNC)*, 2017, pp. 843–848.

[11] L. Mei, R. Hu, H. Cao, Y. Liu, Z. Han, F. Li, and J. Li, "Realtime mobile bandwidth prediction using lstm neural network," in *International Conference on Passive and Active Network Measurement*. Springer, 2019, pp. 34–47.

[12] A. Azari, P. Papapetrou, S. Denic, and G. Peters, "Cellular traffic prediction and classification: a comparative evaluation of LSTM and ARIMA," *arXiv preprint arXiv:1906.00939*, 2019.

[13] A. Y. Nikravesh, S. A. Ajila, C.-H. Lung, and W. Ding, "Mobile network traffic prediction using MLP, MLPWD, and SVM," in *Big Data (BigData Congress)*, 2016, pp. 402–409.

[14] Y. Zang, F. Ni, Z. Feng, S. Cui, and Z. Ding, "Wavelet transform processing for cellular traffic prediction in machine learning networks," in *Signal and Information Processing (ChinaSIP), IEEE China Summit and International Conference on*, 2015, pp. 458–462.

[15] J. J. Santanna, R. van Rijswijk-Deij, R. Hofstede, A. Sperotto, M. Wierbosch, L. Z. Granville, and A. Pras, "Booters—an analysis of DDoS-as-a-service attacks," in *IFIP/IEEE International Symposium on Integrated Network Management*, 2015, pp. 243–251.

[16] Tofino, https://www.barefootnetworks.com/products/ brief-tofino-2/, 2020, [Online; accessed 10-May-2020].

[17] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[18] A. Azzouni and G. Pujolle, "A long short-term memory recurrent neural network framework for network traffic matrix prediction," *arXiv preprint arXiv:1705.05690*, 2017.

[19] X. Wang, Z. Zhou, Z. Yang, Y. Liu, and C. Peng, "Spatio-temporal analysis and prediction of cellular traffic in metropolis," in *International Conference on Network Protocols (ICNP)*, 2017, pp. 1–10.

[20] A. A. Adebiyi, A. O. Adewumi, and C. K. Ayo, "Comparison of ARIMA and artificial neural networks models for stock price prediction," *Journal of Applied Mathematics*, 2014.

[21] D. Janardhanan and E. Barrett, "Cpu workload forecasting of machines in data centers using LSTM recurrent neural networks and ARIMA models," in *International Conference for Internet Technology and Secured Transactions*, 2017, pp. 55–60.

[22] F. Ordóñez and D. Roggen, "Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition," *Sensors*, vol. 16, no. 1, p. 115, 2016.

[23] K. Saleh, M. Hossny, and S. Nahavandi, "Driving behavior classification based on sensor data fusion using LSTM recurrent neural networks," in *International Conference on Intelligent Transportation Systems (ITSC)*, 2017, pp. 1–6.

[24] R. Yu, Y. Li, C. Shahabi, U. Demiryurek, and Y. Liu, "Deep learning: A generic approach for extreme condition traffic forecasting," in *SIAM International Conference on Data Mining*, 2017, pp. 777–785.

[25] A.-r. Mohamed, F. Seide, D. Yu, J. Droppo, A. Stoicke, G. Zweig, and G. Penn, "Deep bi-directional recurrent networks over spectral windows," in *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2015, pp. 78–83.

[26] B. J. Radford, L. M. Apolonio, A. J. Trias, and J. A. Simpson, "Network traffic anomaly detection using recurrent neural networks," *arXiv preprint arXiv:1803.10769*, 2018.

[27] A. A. Ariyo, A. O. Adewumi, and C. K. Ayo, "Stock price prediction using the ARIMA model," in *UKSim-AMSS International Conference on Computer Modelling and Simulation*, 2014, pp. 106–112.

[28] J. Contreras, R. Espinola, F. J. Nogales, and A. J. Conejo, "ARIMA models to predict next-day electricity prices," *IEEE transactions on power systems*, vol. 18, no. 3, pp. 1014–1020, 2003.

[29] V. Ş. Ediger and S. Akar, "ARIMA forecasting of primary energy demand by fuel in Turkey," *Energy policy*, vol. 35, no. 3, pp. 1701–1708, 2007.

[30] K. Gilbert, "An ARIMA supply chain model," *Management Science*, vol. 51, no. 2, pp. 305–310, 2005.

[31] G. Box and G. Jenkins, "Time series analysis: forecasting and control rev," *Oakland California Holden-Day.*, 1976.

[32] R. J. Hyndman and G. Athanasopoulos, *Forecasting: principles and practice*. OTexts, 2018.

[33] W. Enders, *Applied econometric time series*. John Wiley & Sons, 2008.

[34] D. Kwiatkowski, P. C. Phillips, P. Schmidt, and Y. Shin, "Testing the null hypothesis of stationarity against the alternative of a unit root: How sure are we that economic time series have a unit root?" *Journal of econometrics*, vol. 54, no. 1-3, pp. 159–178, 1992.

[35] S. Narayana, A. Sivaraman, V. Nathan, P. Goyal, V. Arun, M. Alizadeh, V. Jeyakumar, and C. Kim, "Language-directed hardware design for network performance monitoring," in *ACM Special Interest Group on Data Communication*, 2017, pp. 85–98.

[36] K. Sutiene, G. Vilutis, and D. Sandonavicius, "Forecasting of grid job waiting time from imputed time series," *Elektronika ir Elektrotechnika*, vol. 114, no. 8, pp. 101–106, 2011.

[37] B. Yeganeh, R. Rejaie, and W. Willinger, "A view from the edge: A stub-as perspective of traffic localization and its implications," in *2017 Network Traffic Measurement and Analysis Conference (TMA)*. IEEE, 2017, pp. 1–9.

[38] Y. Kang, R. J. Hyndman, and F. Li, "Gratis: Generating time series with diverse and controllable characteristics," *arXiv preprint arXiv:1903.02787*, 2019.

[39] G. M. Ljung and G. E. Box, "On a measure of lack of fit in time series models," *Biometrika*, vol. 65, no. 2, pp. 297–303, 1978.

[40] R. J. Hyndman and A. B. Koehler, "Another look at measures of forecast accuracy," *International journal of forecasting*, vol. 22, no. 4, pp. 679–688, 2006.

[41] P. Cortez, M. Rio, M. Rocha, and P. Sousa, "Multi-scale internet traffic forecasting using neural networks and time series methods," *Expert Systems*, vol. 29, no. 2, pp. 143–155, 2012.

[42] R. Vinayakumar, K. Soman, and P. Poornachandran, "Applying deep learning approaches for network traffic prediction," in *Advances in Computing, Communications and Informatics (ICACCI)*, 2017, pp. 2353–2358.

[43] N. Ramakrishnan and T. Soni, "Network traffic prediction using recurrent neural networks," in *IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2018, pp. 187–193.