ICC 2021 - IEEE International Conference on Communications | 978-1-7281-7122-7/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICC42927.2021.9500858

Deep Learning-Based Forecasting of Cellular Network Utilization at Millisecond Resolutions

Ahmad M. Nagib*, Hatem Abou-zeid†, Hossam S. Hassanein*, Akram Bin Sediq†, Gary Boudreau†
*School of Computing, Queen's University, Kingston, Canada, {ahmad, hossam}@cs.queensu.ca
†Ericsson, Ottawa, Canada, {hatem.abou-zeid, akram.bin.sediq, gary.boudreau}@ericsson.com

***Abstract*—The ability to accurately forecast network resource utilization is vital in next-generation wireless networks. Based on the predicted load, telecom operators can proactively allocate network resources in an efficient way. In this paper, we perform a thorough analysis of a cellular network downlink load dataset collected at millisecond resolution. We first evaluate various statistical metrics of the physical resource block (PRB) utilization data to investigate its predictability. Then, we develop deep learning-based models to forecast PRB utilization in radio access networks (RANs). In particular, we propose univariate and multivariate long short-term memory (LSTM) network-based architectures for the forecasting task and investigate the impact of various prediction horizons and history lengths. When predicting PRB utilization, our approach showed up to 49% improvement in the Coefficient of Determination (r^2 score) and 19.5% decrease in the Root Mean Square Error (RMSE) compared with the baseline methods used.**

***Index Terms*—cellular network, resource utilization, time series forecasting, deep learning, LSTM, LTE.**

I. Introduction

Cellular networks have been growing at various levels to adapt to the continuously evolving user needs. 5G networks are particularly required to be ultra-fast and ultra-reliable. They are highly dense and heterogeneous on several levels. Therefore, the wireless communication environment is very dynamic and depends on constantly changing spatio-temporal conditions. This poses a challenge for many applications such as adaptive multimedia streaming services [1]. Accordingly, networks should be able to inherently adapt to the dynamic network conditions and deal with such heterogeneity.

Anticipating the network conditions is one of the key enablers of smart management in next-generation wireless networks. It is crucial to efficiently enhance the overall network and applications performance. Anticipatory networking attempts to accurately and reliably predict future network state. Thus, it can effectively reduce the uncertainty in network demand. This includes the prediction of future traffic, throughput and channel state information (CSI). Resource utilization forecasting is particularly useful to proactively allocate resources and enhance network functions such as handover and end-to-end network slicing.

The availability of granular real network data is key to building accurate models. This way the models would reflect the dynamic and heterogeneous nature of the network. However, since access to resource utilization in real cellular networks is scarce, researchers often use controlled simulation or testbed data. Such data is based on synthetic traffic models that are not representative of the dynamic traffic patterns experienced in the real world. As such, an analysis and understanding of traffic modeling and predictability at the millisecond (ms) level remains an open research problem that is not well understood. Research toward this will enable researchers to further understand network behavior and improve the design of resource management in wireless cellular networks.

The following points are the key contributions of this paper:

• We performed a thorough analysis on more than 58 million samples of live network cell load data collected at millisecond granularity. The effects of traffic load, number of users, and modulation and coding scheme (MCS) on the PRB utilization were investigated. Moreover, we studied the properties of the utilization data over various averaging levels. To the best of our knowledge, this is the first extensive analysis to be carried out over PRB utilization using real 4G networks data on the Transmission Time Interval (TTI) level.

• We investigated the predictability of PRB utilization. Then, PRB utilization forecasting models were built and compared showing the impact of prediction horizon and history length. In particular, we trained univariate and multivariate long short-term memory (LSTM) deep neural networks to predict future utilization based on its historical values, number of users and MCS. The built LSTM models were compared with baseline models, namely last observation, moving average and simple exponential smoothing.

The rest of the paper is organized as follows. Section II explores the related work. Section III provides an overview of the obtained dataset and the processing carried out on the raw data. An analysis of the resource utilization time series is presented in Section IV. Then, the built models and the related design decisions are described in Section V. In Section VI we present the forecasting pipeline and the evaluation metrics, and discuss the results. Lastly, our work is concluded and some future directions are presented in Section VII.

II. Related Work

The number of solutions incorporating forecasting models to anticipate network conditions has been growing [2]. The majority of studies either use statistical methods such as ARIMA [3] or machine learning (ML) methods such as LSTM

This research is supported by a grant from the Natural Sciences and Engineering Research Council of Canada (NSERC) under grant number: STPGP 521432-2018.

neural networks [4]. This includes the prediction of network-related factors such as throughput, traffic, channel state, radio measurements and generally network's Key Performance Indicators (KPIs) [4]–[6]. These KPIs are combined with aspects such as user mobility and area congestion to proactively control network functions [7], [8]. This includes scheduling, handover control, load balancing, network slicing and generally improving end user's Quality of Experience (QoE) [9]. This approach provides network functions with a representative knowledge of the network dynamics and enables them to perform more effectively.

Load prediction can be useful at the individual user level [5], [7], and at the aggregate cell or network level which is the focus of our paper. Most relevantly, both [10] and [11] propose resource utilization forecasting models. The authors in [10] focus on the deployment aspects of such models paying most attention to scalability. They propose to incorporate transfer learning and LSTMs to tackle computation time and data storage limitations. While in [11], the authors propose a Gaussian Process (GP)-based model for the forecasting task. It is good to note that the datasets used in the two studies have a 15-minute and 1-hour granularity respectively. They also have a relatively small number of samples per base station. This might affect both the generalization of the proposed models and their ability to precisely represent the cell load dynamics.

Similarly, the authors of [12] developed a time series forecasting model based on LSTM neural networks (NNs). In a network slicing context, their main goal is to effectively provision resources and make admission control decisions. The dataset used was collected from an experimental testbed based on synthetic loads. In fact, the distribution of utilization is unknown at a low-level granularity. Accordingly, their proposed forecasting model accuracy is expected to be limited as the dataset does not reflect the granular resource utilization dynamics found in a live setting.

III. The Dataset

The dataset consists of 194 trace files, where each trace file includes around five minutes of downlink (DL) interactions of a given base station. A downlink interaction, hereinafter referred to as sample, includes the number of physical resource blocks (PRBs) and the modulation and coding scheme (MCS) allocated by a given Base Station (BS) to a given user at a given timestamp. In total, the processed dataset has more than 58 million samples. The dataset has a millisecond resolution which resembles a single Transmission Time Interval (TTI) in 4G networks. A PRB is the fundamental unit used for resource allocation. In this paper, we refer to PRB utilization as the number of downlink PRBs allocated to users per unit time. The maximum number of PRBs that can be allocated in a given TTI is 100 PRBs, corresponding to a 20 MHz DL carrier.

A. Data Pre-processing

The following steps were applied to the raw data:

TABLE I: Processed Dataset Parameters

Parameter Name	Parameter Description
Timestamp	Sample time in millisecond granularity
PRB utilization	The number of downlink PRBs allocated to users per unit time
Number of users	Number of users who were allocated PRBs per unit time
MCS	The used Modulation and Coding Scheme
Trace File ID	Refers to the original trace file that included a given record

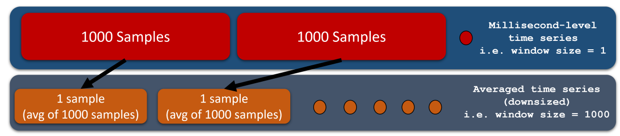


Fig. 1: Averaging Example (window size = 1000).

1) Populating the Time Gaps: Time gaps in the raw data reflect the absence of PRB allocations at some TTIs. The first processing step is to populate such TTIs with zero PRB utilization.

2) TTI Level Data Aggregation: Multiple samples will have the same timestamp if multiple users were allocated PRBs at the same TTI. Therefore, all the data that belongs to a given TTI is aggregated to be reflected in only one record. Other contextual info such as the number of users per TTI is calculated and recorded.

3) Annotating Samples with Trace File ID: A record of the trace file number is kept as an identifier in the processed data. This would help in filtering and analyzing the properties of different trace files (e.g. traffic load profiles). Table I describes the main processed dataset parameters.

4) Utilization Time Series Averaging: Several versions of the processed dataset were then created to reflect different prediction horizons. Based on a window size value, a number of successive utilization values are replaced with one value representing their average. As a result, these new versions are downsized (the longer the window size, the smaller the dataset). This fact puts some restrictions on the upper bound of the window size as discussed in Section VI. Fig. 1 shows an example of averaging using a window size of 1000 TTIs.

5) Keeping Auxiliary Meta-data: In addition to the main parameters described in Table I, the data was further processed to keep additional trace files meta-data highlighted in Table II. More specifically, it classifies the trace files into different traffic load profiles based on the average PRB utilization. Moreover, other statistical properties of the individual trace files were calculated.

TABLE II: Trace Files Auxiliary Meta-data

Feature Name	Feature Description
Traffic Load Class	The traffic load profile of the trace file based on the trace file's average downlink PRB utilization
Utilization Mean	The average downlink PRB utilization of a trace file
Utilization Variance	The variance of downlink PRB utilization of a trace file
Utilization Standard Deviation	The standard deviation of downlink PRB utilization of a trace file
Total Number of Timestamps	The total duration of a trace file (including the timestamps in which no PRBs were allocated)
Number of Non-zero Timestamps	The number of timestamps in a trace file in which no PRBs were allocated to users

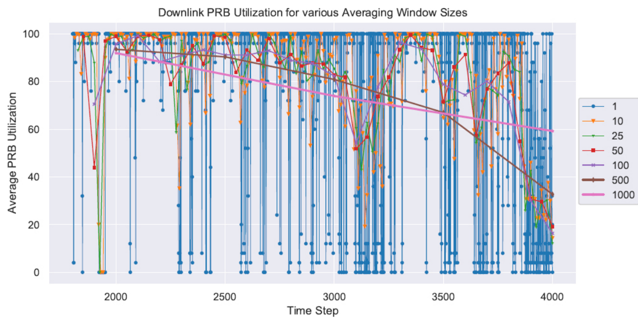


Fig. 2: Sample of Downlink PRB Utilization Time Series.

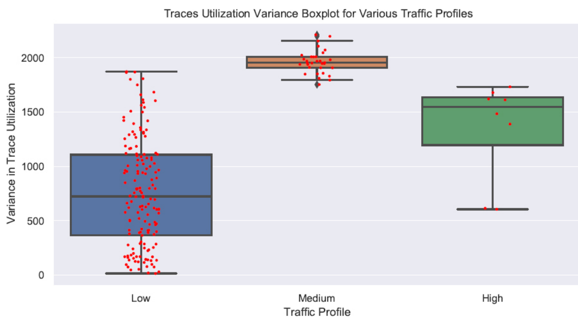


Fig. 3: Utilization Variance Boxplot (Various Traffic Profiles).

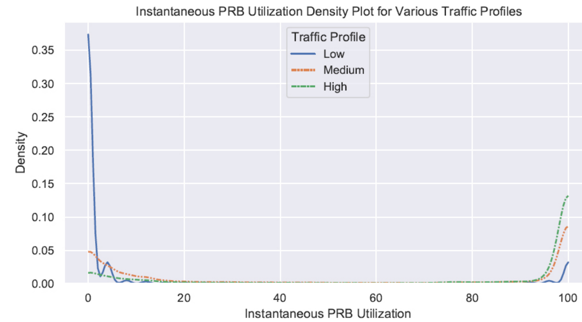


Fig. 4: Instantaneous PRB Utilization Density Plot.

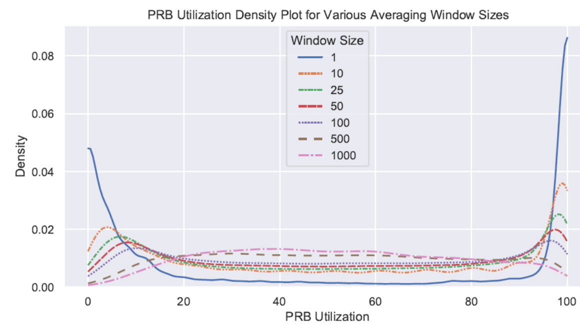


Fig. 5: Medium Traffic Average Utilization Density Plot.

IV. Resource Utilization Time Series Analysis

Fig. 2 shows a sample of around two seconds of the PRB utilization time series. In addition to the millisecond-level utilization, it also shows the averaged versions using different window sizes. It is evident that averaging has a significant effect on smoothing out the time series. As the window size increases, it contributes to the elimination of sudden variation between time steps.

1) Imbalanced PRB Utilization Trace Files: The trace files are classified into 3 even traffic load classes based on the average utilization, namely low, medium and high. Each red point in Fig. 3 represents a trace file. The number of low traffic load trace files is noticeably higher than those having a medium and high average utilization. This suggests that the data is biased towards a specific class of traffic load. Such imbalance can affect the performance of the forecasting models especially the deep learning ones. Accordingly, it might be better to build forecasting models specific to a certain traffic load profile as opposed to generic one model fits all.

2) Extremely Skewed TTI-Level Utilization Distribution: Fig. 4 shows the millisecond-level density plot of PRB utilization for different traffic loads. It is easily noticed that the data has extremely skewed distributions. That is, average utilization of a trace file, and hence the estimated traffic load, is mostly determined by the instantaneous zero or 100 PRB utilization. Fig. 5 visualizes the estimated utilization distribution of the averaged versions of the medium traffic trace files. It can be concluded that the distribution move slowly towards a normal distribution with the increase in the window size.

3) Utilization Variance for Different Traffic Loads: As Fig. 3 suggests, the PRB utilization variance distribution varies significantly from one traffic load to another. It is clear that the medium traffic trace files have the least variation in variance values and are the ones closest to a normal distribution. Accordingly, we will focus on these trace files averaged over the different TTI Levels in the rest of this paper.

4) Additional Cell Load Context: The number of users and the MCS represent additional cell load-related factors. As seen in Fig. 6, the number of users maintain a normal distribution over the various window sizes. The MCS distribution showed a closely similar behavior. Given the utilization distribution in Fig. 5, this may put doubt on the effectiveness of using any of them as an enhancing feature for deep multivariate models.

V. Resource Utilization Forecasting

Exploring the autocorrelation of a time series is one of the main steps prior to building a forecasting model. It can provide

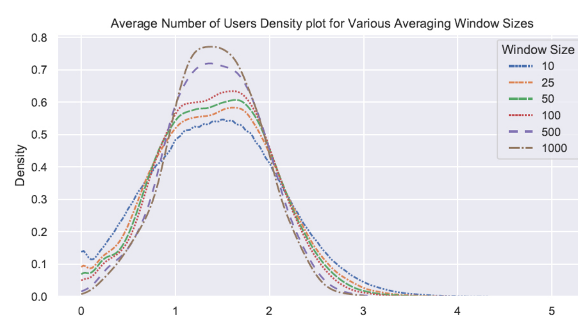


Fig. 6: Number of Users Density Plot (Medium Traffic).

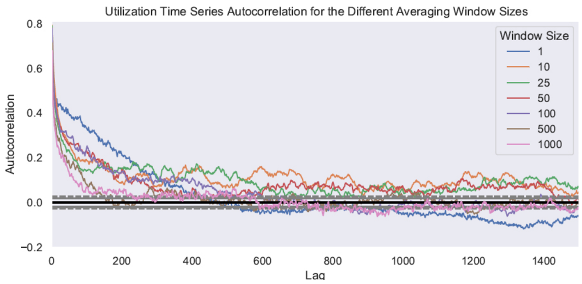


Fig. 7: PRB Utilization Time Series Autocorrelation.

some insight on the time series properties such as periodicity and stationarity [13].

Fig. 7 shows autocorrelation plots of a subset of the medium traffic time series. The plots show a drop in correlation following an initial high value which indicates short-term correlations. It is obvious that the autocorrelation of the 1000 window size version of the data drops significantly faster than the raw TTI-level version. The plots do not show significant periodic fluctuations, i.e. seasonality. Moreover, the different autocorrelation plots have non-zero values for long duration which is an indication of the non-stationarity of the time series. This affects the decision of choosing a forecasting method significantly.

A. Building Forecasting Models: Decisions Made

The following points are among the main decisions that had to be taken before building our forecasting models.

1) Averaging Levels: We created several averaged versions of the dataset and decided to include 7 versions in this study, corresponding to the following 7 average window sizes: 1, 10, 25, 50, 100, 500, and 1000 ms.

2) Prediction horizon and history length: One of the main decisions to take is how many future steps the model is going to predict. Similarly, the number of past steps to take as an input to the forecasting model should be decided. Such values can be set initially with the help of the autocorrelation plot. The models highlighted in this paper uses the last 10 observations to predict the next one. Note that the actual prediction horizon (and history) also depends on the window size used to create a given version of the dataset.

3) Traffic Load Profile: The whole dataset could be used to train a model. Alternatively, data specific to a given traffic load profile could be used. As mentioned in Section IV-3, we decided to start with the medium traffic trace files that have the lowest variation in variance values among the traffic load profiles.

4) Input Features: Normally a forecasting model uses the past observations of a parameter to predict its future ones. However, deep models such as LSTMs can incorporate additional input features that might enhance the prediction performance. We decided to verify the point made earlier in Section IV-4 regarding the usefulness of using the number of users and MCS as additional input features.

5) Model-specific Hyperparameters: In addition to the mentioned decisions, the model-specific hyperparameter settings will be mentioned in the upcoming subsections.

B. Baseline Models

We used the following models as baseline:

1) Last Observation: This model simply uses the last actual utilization observation as prediction for the next future time step. It can be represented as:

$$\bar{y}_{t+1} = y_t \tag{1}$$

where $\bar{y}_{t+1}$ is the predicted value of PRB utilization at time $t+1$ and y_t is the utilization value observed at time t.

2) Moving Average: The moving average method creates a constantly updated average of the time series. The average is taken over a specific period of time, depending on the decided history length of the forecasting model (10 in our case).

A moving average model with history length of n can be written as

$$\bar{y}_{t+1} = (\frac{1}{n}) \sum_{i=1-n}^{0} y_{t+i} \tag{2}$$

where $\bar{y}_{t+1}$ is the predicted value of PRB utilization at time $t+1$ and $\bar{y}_{t+i}$ is the the utilization value observed previously at time $t+i$ where i ranges from $1-n$ to 0.

3) Simple Exponential Smoothing: This is a forecasting method for univariate data without a trend or seasonality. Simple exponential smoothing can be represented as follows:

$$\bar{y}_{t+1} = \alpha y_t + (1-\alpha)(\frac{1}{n}) \sum_{i=1-n}^{-1} y_{t+i} \tag{3}$$

where α is a constant between 0 and 1. It controls the rate at which the influence of the past observations decay exponentially. Large values of α mean that the model pays attention to the recent observations, while smaller values mean more of the history is considered when making a prediction. We decided to set α to 0.5 in order to be at a state between the two baseline models described earlier.

C. Long Short-Term Memory (LSTM) Networks

An LSTM-based model is a special case of recurrent neural networks (RNNs) that has feedback connections. It can process entire sequences of data which makes it one of the candidates for time series forecasting.

We used two types of LSTMs to predict the future PRB utilization.

1) Univariate LSTM: where only previous PRB utilization observations are used to predict future ones.

2) Multivariate LSTM: where additional features were used alongside with the past PRB utilization to forecast the future PRB utilization. Three cases of additional features were considered: **1-** number of users, **2-** MCS, and **3-** number of users and MCS combined.

The size of the input feature vector of the used LSTM models is illustrated in Fig. 8. The LSTM architecture is almost

TABLE III: LSTM Hyperparameter Settings

Parameter Name	Parameter Settings
Number of Layers	2 stacked LSTM layers + 1 Dense layer
Number of LSTM Units	10-50
Loss Function	Mean Squared Error (MSE)
Optimizer	Adam
Activation Function	Hyperbolic Tangent for LSTM, Linear for Dense layer
Learning Rate	0.001
Batch Size	25-40
Number of Epochs	45-170

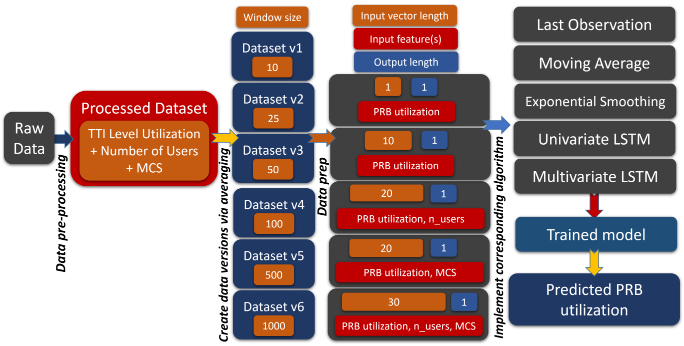


Fig. 8: Pipeline Used to Build the Forecasting Models.

the same for the 3 cases. It consists of 2 stacked layers of LSTM units sequentially followed by 1 fully dense layer that produces the predicted utilization value. Table III shows the hyperparameter settings used to train the various LSTM models.

VI. Results

A. Forecasting Pipeline

The main steps followed to build the described forecasting models are highlighted in Fig. 8. Initially, the raw data is processed to capture the main features that would be used as input to the forecasting models. Then, several versions of the data are created via averaging over different window sizes to reflect different prediction horizons. The third step includes preparing the data to have the shape required by a supervised regression predictor. This should be based on the decisions discussed in Section V-A. Finally the chosen model is built and deployed to predict the future mean PRB utilization value.

The experiments were carried out on a Windows VM having 32 CPUs, 64 GB of RAM and NVIDIA GeForce RTX 2080Ti GPU. statsmodels and Keras, with TensorFlow as backend, are the Python packages used to implement the simple exponential smoothing and LSTM respectively.

B. Model Evaluation Metrics

The following metrics are used to evaluate the performance of the built forecasting models. The test split of the dataset is used to compare the models predictions against the ground truth to calculate both the r^2 scores and the $RMSE$.

1) Coefficient of Determination: r^2 score indicates the proportion of the variance in utilization that is explained by the model. It is normally a number between zero and one. That is, the closer the value to one, the better the performance of the forecasting model. r^2 score can be calculated as follows:

$$r^2 = 1 - \frac{SS_{residual}}{SS_{total}} \tag{4}$$

where $SS_{residual}$ is the residual sum of squares, and SS_{total} is the total sum of squares associated with the outcome variable.

2) Root Mean Square Error (RMSE): RMSE is a measure of accuracy used to compare forecasting errors of different models for a particular dataset. RMSE represents the quadratic mean of the differences between predicted values and observed values. In general, a lower RMSE is better, and it can be calculated as follows:

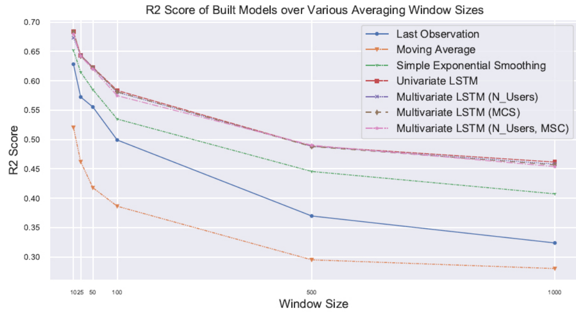
$$RMSE = \sqrt{(\frac{1}{n})\sum_{i=1}^{n}(\bar{y}_i - y_i)^2} \tag{5}$$

where $\bar{y}_i$ represents the predicted values, while y_i represents the ground truth values and n is the number of observations.

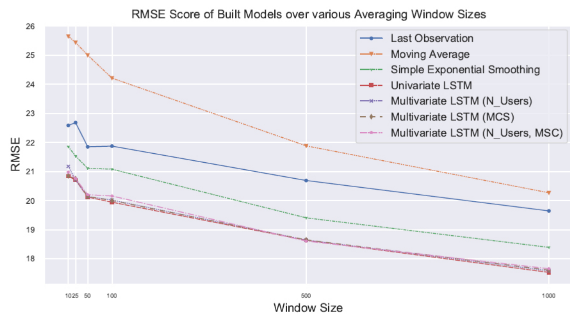
C. Model Performance Discussion

In Fig 9, we illustrate the performance of the built forecasting models using the evaluation metrics described in VI-B. We can observe the following:

- Our LSTM models perform better than all the other built models. Our results show that LSTM models have the highest r^2 scores and the lowest $RMSE$ given all window sizes. However, simple exponential smoothing still leads to well performing models. Thus, it is a candidate predictor that has a relatively low complexity.
- Interestingly, the moving average models perform worse that the simpler last observation baseline. This implies that the utilization is more correlated to its recent observations as the autocorrelation plot previously suggested.
- The r^2 score decreases significantly for all model types with the increase in window size. That is to say, the fitted models fail to capture the variance in the training data when a longer window size is used. In case of LSTM, this might be due to the fact that the training split size decreases when the window size increases. Depending on the original dataset size, this might put a limitation on the maximum averaging window size to be used.
- The $RMSE$ of all the built models decreases with the increase in window size. In other words, as the window size increases, the difference between the predicted values and observed values decreases on average. This also reflects a lower variance in a model residuals with the increase in window size.
- There is an apparent trade-off between r^2 scores and $RMSE$ when it comes to choosing the averaging window size. Such choice implicitly affects both the prediction horizon and the history length.
- Consistent with the point we made in Section IV-4, adding extra input features does not seem to enhance the LSTM model performance. Adding MCS and number of users to the input feature vector resulted into a comparable performance to that of the univariate LSTM model in terms of r^2 scores and $RMSE$.
- Both single-feature and double-feature multivariate LSTM models show more error consistency compared with the univariate LSTM as seen in Fig. 10. The figure shows the estimated residuals distribution of LSTM models given a window size of 100 and 1000 TTI. Residuals can be treated as a measure of



(a) r^2 Score.



(b) RMSE.

Fig. 9: The r^2 Score and RMSE of Built Models.

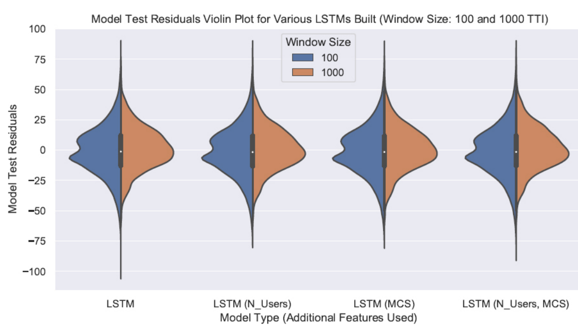


Fig. 10: Test Residuals for Various LSTMs Built.

variation unexplained by the fitted model. They are expected to be roughly normal and independently distributed. The figure generally suggests that longer window sizes achieve more consistent residuals.

In addition to the models highlighted in this paper, other classic ML-based regression models were built to predict future resource utilization. These predictors, such as random forests, resulted in a performance comparable to LSTMs but still require further hyperparameter tuning.

VII. Conclusions and Future Work

In this paper, we carried out a large-scale analysis on PRB utilization data that has a millisecond granularity and built deep learning-based forecasting models to predict the future PRB utilization. We believe this is the first study analyzing resource utilization predictability at such minute timescales using real network data. Our findings show that the LSTM models resulted in the best forecasting performance among the built models. They acheieved up to 49% improvement in the Coefficient of Determination (r^2 score) and 19.5% decrease in the RMSE compared with the baseline methods used.

Further research on developing even more accurate models for traffic prediction at the millisecond resolution is needed, and will open the door to new directions in low latency predictive resource management. Our suggestions for next steps are to tune the built LSTM models and investigate hybrid LSTMs combined with other architectures such as convolutional neural networks (CNNs) to extract hidden traffic patterns. The use of additional LSTM input features such as statistical properties of the recent utilization observations is another avenue for exploration. AutoML-based tools may also be incorporated to effectively perform model and NN architecture search, and hyperparameter tuning [14].

References

[1] H. Abou-Zeid, H. S. Hassanein, and S. Valentin, "Energy-efficient adaptive video transmission: Exploiting rate predictions in wireless networks," *IEEE Transactions on Vehicular Technology*, vol. 63, no. 5, pp. 2013–2026, 2014.

[2] N. Bui, M. Cesana, S. A. Hosseini, Q. Liao, I. Malanchini, and J. Widmer, "A survey of anticipatory mobile networking: Context-based classification, prediction methodologies, and optimization techniques," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1790–1821, 2017.

[3] M. F. Iqbal, M. Zahid, D. Habib, and L. K. John, "Efficient prediction of network traffic for real-time applications," *Journal of Computer Networks and Communications*, vol. 2019, 2019.

[4] A. Azari, P. Papapetrou, S. Denic, and G. Peters, "Cellular traffic prediction and classification: a comparative evaluation of lstm and arima," in *International Conference on Discovery Science*. Springer, 2019, pp. 129–144.

[5] H. Elsherbiny, H. M. Abbas, H. Abou-zeid, H. S. Hassanein, and A. Noureldin, "4g lte network throughput modelling and prediction," in *GLOBECOM 2020-2020 IEEE Global Communications Conference*. IEEE, 2020, pp. 1–6.

[6] A. Kulkarni, A. Seetharam, A. Ramesh, and J. D. Herath, "Deepchannel: Wireless channel quality prediction using deep learning," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 1, pp. 443–456, 2019.

[7] H. Abou-Zeid and H. S. Hassanein, "Predictive green wireless access: Exploiting mobility and application information," *IEEE wireless communications*, vol. 20, no. 5, pp. 92–99, 2013.

[8] H. Gebrie, H. Farooq, and A. Imran, "What machine learning predictor performs best for mobility prediction in cellular networks?" in *2019 IEEE International Conference on Communications Workshops (ICC Workshops)*. IEEE, 2019, pp. 1–6.

[9] G. L. Santos, P. T. Endo, D. Sadok, and J. Kelner, "When 5g meets deep learning: a systematic review," *Algorithms*, vol. 13, no. 9, p. 208, 2020.

[10] C. Parera, Q. Liao, I. Malanchini, D. Wellington, A. E. C. Redondi, and M. Cesana, "Transfer learning for multi-step resource utilization prediction," in *2020 IEEE 31st Annual International Symposium on Personal, Indoor and Mobile Radio Communications*, 2020, pp. 1–6.

[11] Y. Xu, W. Xu, F. Yin, J. Lin, and S. Cui, "High-accuracy wireless traffic prediction: A gp-based machine learning approach," in *GLOBECOM 2017-2017 IEEE Global Communications Conference*. IEEE, 2017, pp. 1–6.

[12] C. Gutterman, E. Grinshpun, S. Sharma, and G. Zussman, "Ran resource usage prediction for a 5g slice broker," in *Proceedings of the Twentieth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, 2019, pp. 231–240.

[13] C. Chatfield, *The analysis of time series: theory and practice*. Springer, 2013.

[14] Q. Yao, M. Wang, Y. Chen, W. Dai, H. Yi-Qi, L. Yu-Feng, T. Wei-Wei, Y. Qiang, and Y. Yang, "Taking human out of learning applications: A survey on automated machine learning," *arXiv preprint arXiv:1810.13306*, 2018.