Bandwidth Prediction in 5G Mobile Networks Using Informer

Tahmina Azmin
Department of Computer Science
University of Regina
Regina, SK, Canada
tah437@uregina.ca

Mohamad Ahmadinejad
Department of Computer Science
University of Regina
Regina, SK, Canada
mha808@uregina.ca

Nashid Shahriar
Department of Computer Science
University of Regina
Regina, SK, Canada
nashid.shahriar@uregina.ca

***Abstract*—Fifth-generation (5G) mobile networks aspire to deliver exceptionally high data rates along with ultra-reliable and low-latency connectivity. With the growing popularity of mobile Internet and the increased bandwidth requirements of mobile applications, user Quality of Experience (QoE) is becoming increasingly critical. 5G networks demand predicting the real-time bandwidth of a channel to satisfy the QoE for bandwidth-savvy applications such as video streaming/conferencing, virtual/augmented/mixed reality, and autonomous driving. If future bandwidth can be forecast in advance, the bandwidth-hungry applications may utilize the estimates to adapt their data transmission rates and dramatically enhance user QoE. By analyzing a publicly available 5G dataset comprised of the channel, context, and cell-related metrics with throughput information, existing work has used Long Short Term Memory (LSTM) based mechanisms to predict future bandwidth. We applied the Transformer-based model, namely 'Informer,' to the 5G dataset and found significant improvement of about 95% error decrease for bandwidth prediction. In addition, we combined some new feature analysis approaches (LASSO and Random Forest with new hyper-parameters) in addition to the the existing Random Forest with Informer to find out the most accurate prediction approach.**

***Index Terms*—Bandwidth prediction, 5G Network, Machine Learning, TPA-LSTM, Informer**

I. Introduction

The fifth generation (5G) of the mobile network is expected to provide extremely high data rates through the Enhanced Mobile Broadband (eMBB) class to support bandwidth-savvy applications such as video streaming/conferencing and virtual/augmented/mixed reality [3]. 5G networks are likely to utilize massive MIMO and a higher carrier frequency in the millimeter-wave (mm-wave) band, spanning from 20 to 90 GHz, to provide a larger bandwidth [1]. However, wireless signals are known to be attenuated by interference, path loss, mobile, and static blocking [2]. Due to its wide coverage, mobile access transmissions are more susceptible to attenuation, resulting in bandwidth variance [1]. Prediction for the future bandwidth of a cellular radio link can help many applications to perform dynamic video-rate adoption and on-demand resource allocation [24].

A cellular radio link is meticulously arranged, despite the fact that it is difficult to model the link bandwidth owing to high bandwidth fluctuation. A base station of a mobile network monitors a variety of network variables (which are frequently utilized in the scheduling process) that may be used to estimate the available bandwidth of a radio link in the near future by analyzing historical data (several seconds). In addition, network monitoring provides an operator with a comprehensive view of all devices, their link data, radio spectrum availability, and immediate traffic demand, among other things [25]. Our objective is to make use of this network information in order to predict available bandwidth in the near future and make it available to applications.

For applications which use video communication or streaming, a buffer is often utilized to transfer the payload if the average bandwidth during a certain time period matches the video rate [4]. However, a fundamental issue in delivering real-time video over cellular networks is achieving low-latency and high-rate video transmission on volatile cellular connections with rapidly fluctuating available capacity [5]. The specified video-rate must closely approximate the immediate network bandwidth to avoid video freeze while using a tiny video buffer. For such systems to provide high user quality of experience (QoE), available bandwidth must be carefully analyzed in real-time to guide video-rate adaptation. Overestimation of bandwidth will result in video freezing, while underestimation will result in needless low video quality [6]. Real-time bandwidth prediction helps to integrate the interactions between the user and bandwidth-hungry mobile applications and allows for the adoption of appropriate video rates for video streaming. Existing research has shown that bandwidth prediction is critical for the high performance of the mentioned applications, such as Available Bandwidth Prediction in Realtime [5], Dynamic Adaptive Streaming over HTTP (DASH) [6], HTTP-based Adaptive Video Streaming with FESTIVE [7], Deployable Multipath TCP [8], and FastMPC [9].

In this paper, we focus on the user bandwidth prediction in a 5G network by leveraging a publicly available 5G dataset [10]. The 5G dataset [10] was generated by merging two traffic conditions (static and driving) and multiple applications (file download and video streaming). Client-side cellular key performance indicators (KPIs), such as cell-related metrics,

978-1-6654-7254-8/22/$31.00 ©2022IEEE

context-related metrics, channel-related metrics, and throughput statistics are included as a time series in the dataset. These data are obtained from G-NetTrack Pro, a very well non-rooted Android network tracking app. In this paper, we first analyze the 5G dataset using time series decomposition and stationarity tests and found that user bandwidth is not following any specific trend and seasonal pattern as the bandwidth can be variable over time due to various types of attenuation. This necessitates a robust prediction model that can estimate the bandwidth for the next few seconds depending on the previous bandwidth, channel, and network connection information.

Prior work has leveraged different regression and machine learning-based methods to predict user bandwidth in cellular networks [2], [27]. Among them, the authors in [27] have compared the Autoregressive integrated moving average (ARIMA) with Long Short Term Memory (LSTM) model in predicting user bandwidth in cellular networks using a dataset consisting of packet level metrics and showed the superiority of LSTM. LSTM is basically a Recurrent Neural Network (RNN) and has long been used in network traffic prediction [28]. Very recently, the authors in [2] have applied Recursive Least Squared (RLS), Random Forest (RF), LSTM, and Temporal Pattern Attention LSTM (TPA-LSTM) to predict multivariate time series like the 5G dataset and found the TPA-LSTM model to be the best approach. This is due to the fact that TPA-LSTM [11] is capable of mining the temporal patterns hidden in networking channels and context information for reliable multivariate forecasting and can outperform the LSTM model. In the TPA-LSTM model, time-invariant temporal patterns are typically extracted using a collection of Convolution Neural Network filters.

Inspired by recent advances of in Transformer [20] based approaches in for predicting multivariate time series, in this paper, we investigate the applicability of Transformer on the user bandwidth prediction problem in a 5G network. Although Transformer can find the time-varying pattern of variables and the impacts on each other rigorously, in terms of Long Sequence Time Series Forecasting (LSTF), Transformers face the problem of consuming much time and memory. This issue has been addressed by the model 'Informer' through ProbSparse self-attention mechanism and the generative style decoder. The self-attention mechanism's atom operation and canonical dot-product have $O(L^2)$ time and memory complexity per layer. ProbSparse replaces canonical self-attention. Dependency alignments take $O(LlogL)$ time and memory. Transformer slows down a long-output prediction, and its dynamic decoding makes inference as slow as RNN. Informer's generative decoder acquires extended sequence output with only one forward step, avoiding cumulative error propagation during inference. Using the Informer on specific well-known datasets such as Electricity Consuming Load and Weather, it has been shown to have increased performance for long-term prediction (horizons), prompting us to apply Informer on the 5G dataset and compare against TPA-LSTM, the best performing model based on the state-of-the-art of 5G user bandwidth prediction [2]. We also combine different feature selection approaches (LASSO and Random Forest with new hyper-parameters) in addition to the Random Forest proposed in [2] with TPA-LSTM and Informer to find out the best performing prediction approach. Our extensive evaluation shows that Informer can improve prediction performance by up to 95% compared to the existing approach of [2].

The rest of the paper is structured as follows. The associated study for real-time bandwidth prediction is included in Section 2. Section 3 presents the problem statement of bandwidth prediction, details regarding the 5G dataset, feature selection methods, and a quick introduction to Informer. Afterwards, Section 4 discusses evaluation settings and results by comparing different combinations of Informer and feature selection methods with the TPA-LSTM approach of [2]. Finally, Section 5 wraps up the paper by discussing future work.

II. Related Work

Real-time bandwidth prediction has been a complex subject for the networking community to solve. When history is available, History-Based (HB) approaches forecast the throughput of TCP flows based on a time series of prior TCP throughput measurements on the same channel. They used simple linear prediction methods such as Moving Average, non-seasonal Holt-Winters, and Exponential Weighted Moving Average [13]. The correlation-based method has been used to improve Network Weather Service (NWS) project [14]. Harmonic Mean of TCP throughput for the past segment downloads has also been used to forecast TCP downloading performance for the future segment [9]. Based on previous and present capacity measurements for video calls over cellular networks, Recursive Least Squares (RLS) were also utilized to anticipate future capacity [5].

Machine learning-based solution PathML was proposed to perform on a large dataset to compare Support Vector Regression (SVR), Kernel Ridge Regression (KRR), Random Forests (RF), and Convolutional Neural Network (CNN) for LTE networks [16]. SVR model has been used to predict TCP throughput [17]. Future bandwidth prediction for video conferencing is derived via probabilistic inference based on the single-server queueing model [18]. To find out the most relevant information from enriched and complex datasets, machine learning-based models perform better. However, the major drawback is traditional statistical or machine learning approaches rely on short sequence histories. Artificial Neural Network (ANN) based prediction for the ON and OFF signal state of optical network unit of wireless access point (ONU-AP) assists in the accuracy of bandwidth demand computation [15].

The 5G bandwidth prediction problem for the multivariate time series is complicated by complex and non-linear interdependencies between variables at different time steps. LSTM and ARIMA models were used to classify and predict user traffic in order to enhance resource use [27]. LSTM [19] has been used for predicting future bandwidth. Compared with LSTM, TPA-LSTM [11] performed better for the 4G/5G dataset in terms of future bandwidth prediction. LSTF is demanded by most real-life time series predicting. LSTF can be effectively used in many real-world applications as Transformer-based models remarkably increase the prediction capacity for multivariate time series. We applied the 5G dataset on a Transformer based model, namely Informer [12] to predict the future bandwidth of the channel. Informer meets the need for the model's high prediction capacity, which can efficiently capture exact long-range dependence coupling between output and input.

III. Methodology

A. Problem Statement

For a random time instance t, suppose the bandwidth is $b(t)$, and a device is measuring the mobile access link every 1 second to attain a discrete-time series. Here, the time series is $X(t), t = 1, 2, 3, \ldots.$, where $X(t) \in \mathbb{R}^n$ is a vector of measured information of n types, including $b(t)$ and other parameters (e.g., signal quality, location, and so on) of the connection. The job of real-time bandwidth forecast at time t is the estimation of the immediately available bandwidth for some future time $t + \tau$, assuming all of the data collected up to t [2]. The equation will be:

$$b(t + \tau) = \mathbf{f}(\{X(k), k = 1, 2, 3, ..., t\}) \tag{1}$$

where τ is the future horizon value and integer number. If we want to predict the future bandwidth with some recent history, $\{X(t - \omega + 1), ..., X(t)\}$ to predict $b(t + \tau)$, where ω is the sliding window size [2]. We utilize past bandwidth measurements to forecast future bandwidth in univariate bandwidth prediction, but in multivariate bandwidth prediction, we employ context and channel-related data in addition to previous bandwidth measurements to make the prediction.

B. Dataset

In this paper, we used the first publicly accessible dataset containing throughput, channel, and context information of 5G networks acquired from a major Irish mobile provider for two application patterns (video streaming and file download) [10]. The collection comprises client-side cellular KPIs that include channel-related metrics, context-related metrics, cell-related metrics, and throughput statistics. These metrics are derived from G-NetTrack Pro, a well-known non-rooted Android network monitoring tool. For both applications, two mobility patterns are available such as static and driving [10].

In this paper, we used history-based prediction, which employs time series forecasting techniques [13]. Since history-based forecasts are path-dependent, we used 83 fixed traces for the time-series data from the 5G dataset [10]. These data have been captured application-wise in two modes (driving and static) for three applications such as- Amazon Prime (Animated Adventure Time and Season3 The Expanse), Download (a random file) and Netflix (Animated Rick and Morty and Season3 Stranger Things). We used these datasets in driving mode to explore the continuous change of bandwidth. We used the dataset in two ways to investigate the difference in prediction. First, we integrated all the data from the application-wise sub-folders and applied it to the models. Second, the sub-folder-based datasets were fed into Informer to see how accuracy differed depending on specific applications.

C. Data Analysis

One of the most efficient methods for analyzing historical time series data is time series decomposition. It entails breaking down the data into trend, seasonality, and residuals. The trend is the general movement of time-series data values. Seasonality refers to the behavior of a series across a given seasonal period. It refers to a cyclic event in a time series that happens for a brief period and generates growing or decreasing patterns in the time series. The residuals are the parts of the time series that remain after trend and seasonality have been eliminated from the original signal for additive decomposition. We applied the decomposition to the merged dataset. We found that data is not following any specific trend and seasonal pattern as the bandwidth can be variable over time due to various types of attenuation. Following Fig. (1) shows the decomposition of the folder-wise combined dataset where the y-axis denotes DL_bitrate (which is downlink bitrate in kbps) and the x-axis is for time.

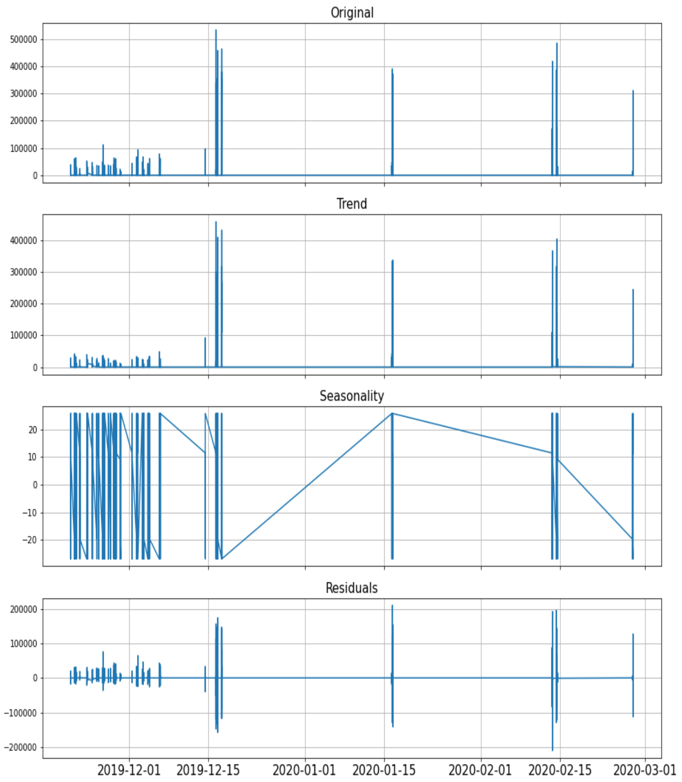


Fig. 1: Decompositon of 5G Data (Folder-wise combined)

The dataset's following analysis is the stationarity test. Data in a time series might be stationary or non-stationary, and the mean and variance of stationary time series are constant, making them easier to forecast. Non-stationary data, on the other hand, is not centered on a single value. For testing the stationarity of the 5G dataset, we used two robust statistical tests: Augmented Dicky-Fuller (ADF) Test [29] and Kwiatkowski-Phillips-Schmidt-Shin (KPSS) [30] Test. Our analysis shows that the 5G dataset is non-stationary in both tests. From this observation, we can consider that it can exhibit odd behavior changes like fluctuations and spikes from one section to the next, demanding a sophisticated prediction mechanism.

D. Feature Selection

We have applied the Random Forest with hyperparameter tuning (using grid search) and Least Absolute Shrinkage and Selection Operator (LASSO) for feature analysis. They are examples of embedded feature selection techniques. Embedded approaches offer the following benefits: high precision, generalizability, and interpretability. For improved forecasting, twelve strongly influenced features on bandwidth from the paper [2] have been applied first into the merged dataset, such as upload and download bitrate, network mode, band, speed, etc., were picked from a set of 23 to be utilized for prediction [10]. The Download (DL) bitrate has the most significant weight, followed by the Upload (UL) bitrate. All features give complementary channel and context information for bandwidth prediction.

Random Forests consist of four to twelve hundred decision trees, each generated using a random extraction of the dataset's observations and characteristics. Since not every tree is exposed to all data attributes, the trees are de-correlated and less susceptible to overfitting. Each tree has a series of yes/no questions depending on a single or several qualities. Using the Random Forest Regressor, we tuned the hyper-parameters further and generated a new set of features as shown in TABLE I.

Furthermore, this study includes a regularized feature selection method known as LASSO. LASSO places a constraint on the sum of the model parameters' absolute values, which must be smaller than a specified number (upper bound). The approach employs a shrinkage (regularisation) procedure that penalizes the regression variables' coefficients, reducing some of them to zero. Only variables with a non-zero coefficient following the shrinking procedure are selected for inclusion in the model during the features selection procedure. The objective of this method is to decrease prediction error and reduce overfitting [26]. The selected features from three different feature selection methods are listed in TABLE I.

E. Prediction Algorithm

We used 'Informer' [12] as the primary bandwidth prediction model, which has been developed based on the concept of Transformer [20]. The transformer enhances the prediction capacity to a greater extent. Still, while applying to LSTF, it undergoes quadratic high memory usage, time complexity, and ingrained limitation of the encoder-decoder architecture. 'Informer' has efficiently handled these problems using three unique features [12]. At first, a self-attention technique named 'ProbSparse' attains $O(LlogL)$ in memory utilization and time complexity while performing comparably well on sequence dependency alignment. Second, by reducing cascading layer input, self-attention distillation reveals dominating attention and effectively manages lengthy input sequences. And the third one is the decoder of generative style, which predicts long sequences of time series in a single forward way, significantly improving the inference speed of long-sequence prediction.

TABLE I: Selected Features for Random Forest from paper [2], Hyper-tuned Random Forest and Lasso Rgression

Features	RF from paper [2]	O-RF	LASSO
Longitude		✓	✓
Latitude		✓	
Speed	✓	✓	✓
CellID	✓	✓	✓
NetworkMode	✓	✓	✓
RSRP	✓	✓	✓
RSRQ	✓	✓	
SNR	✓	✓	✓
CQI	✓	✓	
RSSI	✓	✓	
DL_bitrate	✓	✓	✓
UL_bitrate	✓	✓	✓
CELLHEX		✓	
NODEHEX		✓	
RAWCELLID		✓	
NRxRSRP		✓	
NRxRSRQ		✓	✓

The **probSparse** self-attention mechanism maintains a multi-head perspective, and for each head, this attention produces different scattered pairs of query-key, which avoid enormous information loss in response. Self-attention has been defined using query, key, and value from input tuples denoted as Q, K, and V. For the i-th row of Q, K, and V, it will be $\mathrm{q_i}$, $\mathrm{k_i}$, and $\mathrm{v_i}$. The scaled dot product for self-attention mechanism of Transformer is -

$$A(Q,K,V) = Softmax(\frac{QK^{\top}}{\sqrt{d}})V \tag{2}$$

where $Q \in \mathbb{R}^{L_Q \times d}$, $K \in \mathbb{R}^{L_K \times d}$, and $V \in \mathbb{R}^{L_V \times d}$ with the dimension d. Defining the kernel smoother as a probability form, i-th query's attention formula is -

$$A(q_i,K,V) = \sum_j \frac{\kappa(q_i,k_j)}{\sum_l \kappa(q_i,k_l)} v_j = \mathbb{E}_{p(k_j|q_i)}[v_j] \tag{3}$$

Here, the probability is $p(k_j|q_i) = \kappa(q_i,k_j)/\sum_l \kappa(q_i,k_l)$ and $\kappa(q_i,k_j)$ chooses the asymmetric exponential kernel as $e^{q_i k_j^{\top}}/\sqrt{d}$. The significant drawback of this self-attention is that it combines the values and produces an output based on computing the probability $p(k_j|q_i)$, which needs $O(L_Q L_K)$ memory use and quadratic times dot-product computation. The Log-Sum-Exp (LSE) of q_i on all the keys and their arithmetic mean were used to determine the most significant queries and the equation is-

$$M(q_i,K) = \ln \sum_{j=1}^{L_K} e^{\frac{q_i k_j^{\top}}{\sqrt{d}}} - \frac{1}{L_K} \sum_{j=1}^{L_K} \frac{q_i k_j^{\top}}{\sqrt{d}} \tag{4}$$

If the i-th query acquires a bigger $M(q_i,K)$, its attention probability p is more "diverse" and has a greater likelihood of including the dominant dot-product pairings in the header field of the long tail self-attention distribution. **ProbSparse** self-attention is achieved by enabling each key to only attend to the u dominant queries.

$$A(Q,K,V) = Softmax(\frac{\overline{Q}K^{\top}}{\sqrt{d}})V \quad (5)$$

where Q is a sparse matrix of the same size as q and only contains the Top-u queries according to the sparsity measurement $M(q,K)$. Controlled by a constant sampling factor c, $u = c.lnL_Q$, allows the **ProbSparse** self-attention to calculate $O(lnL_Q)$ dot-product for each query-key lookup while maintaining layer memory use $O(L_K lnL_Q)$ and it is the significant compared to Transformer.

The **Encoder** section processes the long sequential inputs for multivariate time series under the constraint of limited memory. The self-attention distillation mechanism efficiently handles highly long sequences of inputs. The encoder's feature map contains redundant permutations of value **V** as a natural result of the self-attention mechanism named ProbSparse. We apply the distillation operation in the following layer to prioritize the best ones with dominant features and create a focused self-attention feature map. It significantly reduces the temporal dimension of the input. Fig.(2) shows the Attention blocks for the n-headed weighted matrix (overlapping red squares).

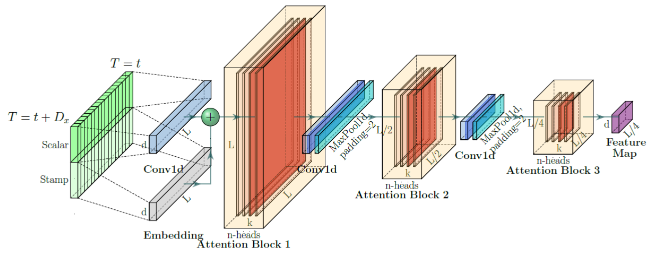
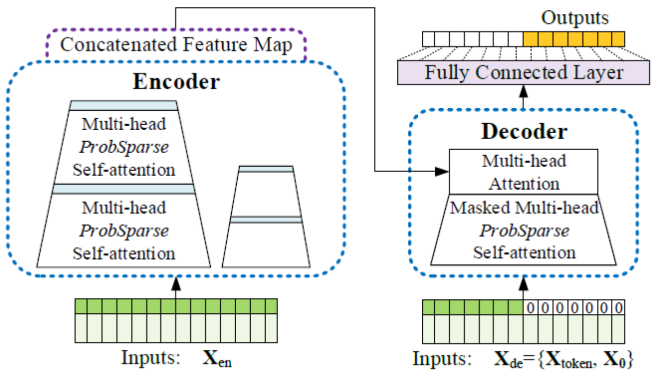


Fig. 2: Informer Encoder Section



Concatenated Feature Map
Encoder
Multi-head ProbSparse Self-attention
Multi-head ProbSparse Self-attention
Inputs: X_{en}
Outputs
Fully Connected Layer
Decoder
Multi-head Attention
Masked Multi-head ProbSparse Self-attention
Inputs: $X_{de}=\{X_{token}, X_0\}$

Fig. 3: Informer Architecture

This distillation process [21] employs the Multi-head ProbSparse self-attention mechanism and the fundamental operations together, where Conv1d(.) implements a 1-D convolutional filter for the kernel width=3 with the ELU(.) activation function [22] on time dimension. After stacking a layer, a max-pooling layer is added with stride two and downsamples the value into its half slice, reducing the overall memory use. To improve the distillation operation's resilience, we create copies of the core stack with halved inputs and gradually reduce the number of distilling layers for self-attention by eliminating one layer at a time, such that their output dimension is aligned. As a result, we concatenate the stack outputs to obtain the encoder's final hidden representation. The absolute confidential encoder representation was constructed by concatenating all stack outputs in Fig.(3).

In the **Decoder** section, Informer uses one forward procedure to generate a long output sequence. The conventional decoder structure used here comprises two attention layers that are similar and multi-head. They stack on top of each other. The generative inference is used to compensate for the speed drop in the extended forecast. Informer uses the concept of start token for Natural Language Processing's "dynamic decoding" [23] slightly differently. It will not select a specific flag as a token; instead, it will take a long sequence sample token into the input. It can be a last slice before the output sequence. For example, if we want to predict bandwidth up to 168th seconds (2.8 minutes of bandwidth prediction), we will take known 2 minutes features as the 'start token' before the target sequence. It will be fed into a generative-style decoder with $X_{de} = X_{2m}.X_0$, where target sequence timestamp is denoted by X_0. Then, instead of "dynamic decoding" in the typical encoder-decoder design, the suggested decoder of Informer predicts outputs using a single forward method which saves time.

IV. Evaluation Results

A. Evaluation Settings

We utilized the dataset in two different methods, and separated the dataset into three sections. They consist of a training set, a test set, and a validation set in the proportions of 70%, 10%, and 20% data, respectively. Maintaining the file count is necessary since they include particular time series. We ran the model five times for each horizon across all datasets for the accuracy calculation and report the average results. All the experiments are performed on an 11th Gen i7 machine with one GeForce RTX 3090 GPU having 24GB memory and 32GB main memory.

B. Compared Methods

- TPA-LSTM and variants: TPA-LSTM [11] is an extension of LSTM that includes the attention mechanism. We first implemented the TPA-LSTM with parameters and selected features from the paper [2]. Afterward, we used grid search to tune the hyper-parameters of the TPA-LSTM and Random Forest Regression and denote them as O-TPA-LSTM and O-RF, respectively. The O-TPA-LSTM neural network structure consists of 1 layer with 128 units, while the TPA-LSTM model from the paper [2] contains three layers, each having 32 units. TABLE II summarizes the different TPA-LSTM variants that are evaluated in our study.

- Informer and variants: The Informer [12] model contains a stack of 1-layer for the encoder and 2-layer for the decoder. The dropout rate for the neural network is 0.05. With proper early stopping, the total epoch number is 6. The Informer and the three separate feature selection procedures have been combined. The combinations that are evaluated in this paper are shown in TABLE III.

TABLE II: Combination of TPA-LSTM and Different Feature Selection Methods

Model + Feature Selection	Explanation
TPA-LSTM + RF	Combination of TPA-LSTM with parameters from the paper [2] with Random Forest feature selection from the paper [2]
O-TPA-LSTM + O-RF	Combination of TPA-LSTM with optimized parameters with optimized Random Forest feature selection
O-TPA-LSTM + LASSO	Combination of TPA-LSTM with optimized parameters with LASSO feature selection

TABLE III: Combination of Informer and Different Feature Selection Methods

Model + Feature Selection	Explanation
Informer + RF	Combination of Informer with Random Forest feature selection from the paper [2]
Informer + O-RF	Combination of Informer with optimized Random Forest feature selection
Informer + LASSO	Combination of Informer with LASSO feature selection

C. Performance Metrics

As primary performance indicators for prediction errors, we use Root Mean Square Error (RMSE) and Mean Absolute Error (MAE). RMSE is a quadratic scoring criterion that quantifies the average error magnitude. It is the square root of the average of squared discrepancies between predicted and observed values. MAE assesses the average size of errors in a series of predictions without considering their direction. It is the average over the test sample of the absolute differences between forecast and actual observation, given that each difference is given equal weight. The equations are as follows:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \hat{x_i})^2} \tag{6}$$

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|x_i - \hat{x_i}| \tag{7}$$

Where N is the total number of observations and x_i and $\hat{x_i}$ are the actual and predicted values for the observation i.

D. Results and Analysis

TPA-LSTM outperformed machine learning models including RLS, RF, and LSTM according to [2]. Thus, we use TPA-LSTM as a baseline to compare the Informer-based approaches. TABLE IV shows the outcomes of comparison between TPA-LSTM and Informer with three feature selection methods - RF from paper [2], O-RF, and LASSO for the combined 5G dataset. As shown in TABLE IV, Informer performs better (**95.54%** RMSE decrease on average) than TPA-LSTM for predicting from smaller to longer time horizons since it was developed with LSTF in mind for all types of feature analysis. The RMSE and MAE values in TABLE IV are increasing rapidly for TPA-LSTM variants in higher horizons, but Informer performs much better across all horizons compared with TPA-LSTM. Informer combines a probabilistic attention method with the learned embedding of relevant temporal aspects to predict long sequence along with the inherited capability of Transformer [12]. The explanation for the ups and downs in the Informer RMSE and MAE in TABLE IV is that when the encoder input has excellent dependencies, the decoder receives relevant local information, which helps to obtain reduced RMSE and MAE [12].

TABLE IV shows that Informer model outperforms TPA-LSTM variants for all feature sets. Let us now discuss why this is so. The problem of disappearing or vanishing gradients plagues traditional RNN, and it forces the model to become biased on the sequence's most recent input. In time series, however, we frequently need to consider previous information that influences the output. LSTM uses an input gate, an output gate, and an extra forget gate to solve this problem. But the information has to go through more steps, which means more calculations have to be done. With Informer, substantially more data can be processed in the same amount of time due to the parallelization capabilities of the Transformer mechanism. That is why Informer gives better accuracy than LSTM-based models.

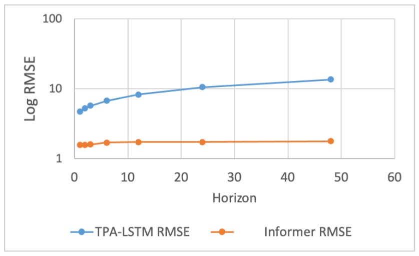
Regarding TPA-LSTM, the O-TPA-LSTM + LASSO is the best among all feature sets related to this model. For the Informer model, combination with our newly applied feature sets (O-RF and LASSO) yielded the best performance. Whenever the comparison arises between O-RF and LASSO, for some horizons (2, 3, 6, and 24), LASSO has the best RMSE. For other horizons (1, 12, and 48), Informer + O-RF performs better. From TABLE IV, we can observe that the feature selection method does not significantly influence the prediction performance of Informer, and any suitable feature selection method can be combined with Informer for the bandwidth prediction.

We have chosen O-TPA-LSTM + LASSO and Informer + LASSO out of six variants and feature selection combination to visualize the result and Fig. 4 reflects the outcome. The comparison graph in Fig. 4(a) and Fig. 4(b) show the corresponding RMSE and MAE for O-TPA-LSTM + LASSO and Informer + LASSO up to horizon 48. Later two graphs of Fig. 4 show the Ground Truth vs Prediction for O-TPA-LSTM + LASSO and Informer + LASSO for horizon 1, respectively.

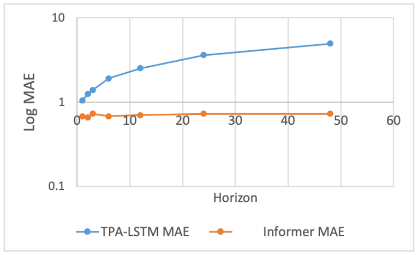
We applied Informer on application-specific datasets to analyze performance improvements. Applications include – Amazon Prime (Animated Adventure Time and Season3 The Expanse), 5G Download (a random file), and Netflix (Animated Rick and Morty and Season3 Stranger Things). Because Informer turns out to be the best model across all feature sets in the combined dataset, it was applied to the subfolder-

TABLE IV: Comparison among different combinations of TPA-LSTM and Informer with Random Forest from paper [2], Optimized Random Forest, and LASSO for the combined 5G dataset

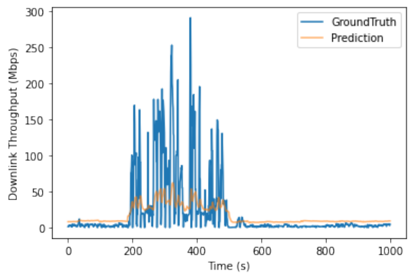
	TPA-LSTM + RF		O-TPA-LSTM + O-RF		O-TPA-LSTM + LASSO		Informer + RF		Informer + O-RF		Informer + LASSO	
Horizon	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
1	24.21	16.29	4.72	1.55	4.69	1.05	**1.73**	**0.89**	**1.55**	**0.7**	**1.57**	**0.68**
2	32.66	19.86	5.82	2.42	5.26	1.25	**1.48**	**0.75**	**1.68**	**0.67**	**1.58**	**0.66**
3	35.63	21.84	6.15	2.16	5.72	1.4	**1.78**	**0.97**	**1.78**	**0.69**	**1.59**	**0.73**
6	38.82	23.88	7.94	3.28	6.73	1.92	**1.67**	**0.81**	**1.72**	**0.69**	**1.7**	**0.68**
12	43.27	28.67	9.55	4.03	8.3	2.52	**1.58**	**0.95**	**1.71**	**0.7**	**1.72**	**0.7**
24	48.16	28.53	12.32	5.67	10.53	3.63	**1.65**	**0.83**	**1.75**	**0.73**	**1.73**	**0.73**
48	51.55	31.55	15.71	7.74	13.6	4.96	**1.67**	**0.85**	**1.74**	**0.71**	**1.78**	**0.73**



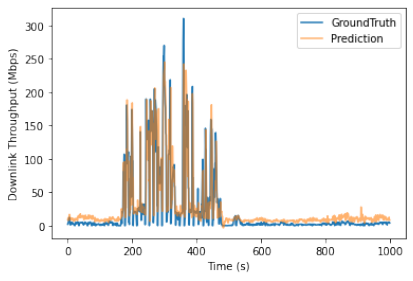
(a)



(b)



(c)



(d)

Fig. 4: (a) and (b) show RMSE and MAE Log for O-TPA-LSTM + LASSO and Informer + LASSO. (c) and (d) show the Ground Truth vs Prediction for O-TPA-LSTM + LASSO and Informer + LASSO in case of Horizon-1

by-subfolder dataset to determine application-centric performance. TABLE V, TABLE VI, TABLE VII, TABLE VIII and TABLE IX summarize the findings for the subfolder-level dataset for all horizons (1–48) with these combination - Informer + O-RF and Informer + LASSO, as well as the mean and standard deviation (in Mbps) for each subfolder. The RMSE and MAE follow a rising trend over longer horizons, and the RMSE can occasionally drop as the input length increases when anticipating longer sequences. The reason is that the more extended encoder input may have more dependencies and the longer decoder token has more local information [12]. We show the training and inference time in these tables to see if the training could be completed online or offline. According to our observations, training should be done offline because training takes several minutes, depending on the size of the dataset. However, inference time is much smaller and hence the trained model can be used to generate real-time prediction outputs without affecting user QoE.

Finally, we have chosen horizon 24 to compare the performance of Informer for the sub-folder-wise dataset with our parameter tuned O-RF and LASSO by changing the window size. Window size has been calculated using the sum of sequence length and prediction length parameter for Informer. Prediction length is the horizon in our case and is kept fixed at 24 for this analysis while changing sequence length. The default sequence length is 96, so we varied the sequence length and observed the impact of changed window size on the prediction. The outcome is displayed in TABLE X for Informer + O-RF and TABLE XI for Informer + LASSO. Our conclusion from these results is that the variable window size has less impact on prediction performance, as the results did not differ significantly when the window size was changed.

V. CONCLUSION

In this paper, we compared the performance of TPA-LSTM and Informer in predicting future bandwidth in a publicly available 5G dataset. Prior work shows that TPA-LSTM outperformed machine learning algorithms including RLS, RF, and LSTM to predict multivariate time series data like the 5G dataset. We have shown that the model “Informer” has beaten TPA-LSTM in terms of the multivariate time series prediction with more than **95%** RMSE decrease for a more extended period which can be efficiently used for real-time video-rate adoption for many kinds of mobile applications. In this paper, we used fixed-route data and compared the results

TABLE V: Comparison between O-RF and LASSO with Informer for 5G Download

Mean: 28.574 Mbps SD: 57.697								
	Informer + O-RF				Informer + LASSO			
Horizon	RMSE	MAE	Training Time(s)	Inference Time(ms)	RMSE	MAE	Training Time(s)	Inference Time(ms)
1	0.75	0.37	179.21	0.99	0.72	0.35	178.42	1.32
2	0.84	0.41	179.71	0.98	0.84	0.42	178.39	1.28
3	0.95	0.49	180.95	0.99	0.97	0.49	180.25	1.30
6	1.21	0.66	181.50	1.00	1.19	0.63	175.77	1.31
12	1.28	0.70	151.81	1.01	1.27	0.69	160.77	1.14
24	1.33	0.77	148.81	0.87	1.33	0.73	175.27	1.20
48	1.37	0.80	135.28	2.35	1.36	0.75	178.84	1.28

TABLE VI: Comparison between O-RF and LASSO with Informer for AP Animated Adventure Time

Mean: 0.623 Mbps SD: 2.167								
	Informer + O-RF				Informer + LASSO			
Horizon	RMSE	MAE	Training Time(s)	Inference Time(ms)	RMSE	MAE	Training Time(s)	Inference Time(ms)
1	0.90	0.33	69.32	1.26	0.81	0.31	74.37	0.54
2	0.93	0.32	61.93	1.24	0.92	0.33	71.11	0.55
3	0.93	0.31	67.52	1.27	0.92	0.33	70.77	0.54
6	0.93	0.34	75.35	1.24	0.93	0.34	64.91	0.52
12	0.95	0.35	80.15	1.27	0.94	0.36	53.99	0.82
24	0.99	0.42	76.27	1.25	0.94	0.35	55.21	0.82
48	1.01	0.42	85.07	0.57	0.95	0.37	68.52	0.77

TABLE VII: Comparison between O-RF and LASSO with Informer for AP Season3 The Expanse

Mean: 0.763 Mbps SD: 2.529								
	Informer + O-RF				Informer + LASSO			
Horizon	RMSE	MAE	Training Time(s)	Inference Time(ms)	RMSE	MAE	Training Time(s)	Inference Time(ms)
1	0.80	0.35	143.04	2.30	0.75	0.33	121.59	1.55
2	0.86	0.37	137.62	2.16	5.26	1.25	121.46	1.56
3	0.86	0.36	125.09	2.22	5.72	1.4	113.48	1.57
6	0.88	0.35	146.10	2.21	6.73	1.92	106.74	1.57
12	0.87	0.36	132.74	2.22	8.3	2.52	126.78	1.33
24	0.93	0.39	156.73	2.33	10.53	3.63	122.79	1.30
48	0.88	0.36	166.3	2.42	13.6	4.96	109.01	1.41

TABLE VIII: Comparison between O-RF and LASSO with Informer for Netflix Animated RickandMorty

Mean: 0.592 Mbps SD: 2.973								
	Informer + O-RF				Informer + LASSO			
Horizon	RMSE	MAE	Training Time(s)	Inference Time(ms)	RMSE	MAE	Training Time(s)	Inference Time(ms)
1	0.67	0.23	80.90	1.15	0.66	0.24	78.05	1.11
2	0.71	0.27	81.04	1.18	0.68	0.25	77.48	1.11
3	0.74	0.33	81.06	1.17	0.71	0.32	72.47	1.10
6	0.74	0.31	81.11	1.17	0.73	0.29	58.97	1.15
12	0.77	0.32	78.08	1.15	0.73	0.29	58.97	1.15
24	0.76	0.29	82.22	1.16	0.74	0.32	59.79	1.16
48	0.75	0.30	80.65	1.20	0.74	0.33	68.54	1.18

TABLE IX: Comparison between O-RF and LASSO with Informer for Netflix Season3StrangerThings

Mean: 0.873 Mbps SD: 3.850								
	Informer + O-RF				Informer + LASSO			
Horizon	RMSE	MAE	Training Time(s)	Inference Time(ms)	RMSE	MAE	Training Time(s)	Inference Time(ms)
1	0.83	0.31	170.15	1.06	0.75	0.33	143.08	1.28
2	0.91	0.39	171.99	1.08	0.85	0.37	129.12	1.27
3	0.91	0.36	165.77	1.72	0.85	0.36	117.44	1.18
6	0.97	0.46	161.79	1.71	0.84	0.38	128.69	1.19
12	1.04	0.52	143.82	1.39	0.85	0.37	101.03	1.16
24	0.96	0.43	137.74	1.40	0.85	0.36	125.02	1.17
48	1.01	0.48	155.25	1.46	0.86	0.38	148.87	1.23

of these two models since history-based machine learning approaches use time-series forecasting. We found that Informer performed very well in the experiment. For future research, we can investigate some recent machine learning models in the comparison. As the dataset has not been updated since the COVID-19 pandemic, we will try to collect the latest dataset for 5G to ensure better QoE for the users.

TABLE X: Comparison for Sub-Folder wise Dataset with Different window size for Random Forest

Horizon-24	Window Size-48		Window Size-96		Window Size-144		Window Size-192	
Dataset	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
5Gdownload	1.30	0.70	1.33	0.77	1.37	0.82	1.38	0.83
AP AnimatedAdventureTime	0.94	0.37	0.99	0.42	0.94	0.38	0.97	0.41
APTheExpanseSeason3	0.81	0.35	0.93	0.39	0.81	0.32	0.81	0.32
NetflixAnimatedRickandMorty	0.75	0.28	0.76	0.29	0.74	0.28	0.77	0.32
NetflixStrangerThingsSeason3	1.20	0.59	0.96	0.43	0.95	0.42	1.09	0.60

TABLE XI: Comparison for Sub-Folder wise Dataset with Different window size for LASSO

Horizon-24	Window Size-48		Window Size-96		Window Size-144		Window Size-192	
Dataset	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
5Gdownload	1.29	0.71	1.33	0.73	1.35	0.82	1.3	0.83
AP AnimatedAdventureTime	0.94	0.37	0.94	0.35	0.93	0.35	0.93	0.35
APTheExpanseSeason3	0.89	0.40	0.89	0.41	0.84	0.40	0.85	0.41
NetflixAnimatedRickandMorty	0.73	0.31	0.74	0.32	0.74	0.32	0.74	0.30
NetflixStrangerThingsSeason3	0.87	0.37	0.85	0.36	0.88	0.40	0.86	0.37

References

[1] Al-Falahy, N., and Alani, O. Y. (2019). Millimetre wave frequency band as a candidate spectrum for 5G network architecture: A survey. Physical Communication, 32, 120-144.

[2] Mei, L., Gou, J., Cai, Y., Cao, H., and Liu, Y. (2021). Realtime Mobile Bandwidth and Handoff Predictions in 4G/5G Networks. arXiv preprint arXiv:2104.12959.

[3] Sutton, A. (2018). 5G network architecture. J. Inst. Telecommun. Professionals, 12(1), 9-15.

[4] Mei, L., Hu, R., Cao, H., Liu, Y., Han, Z., Li, F., and Li, J. (2020). Realtime mobile bandwidth prediction using LSTM neural network and Bayesian fusion. Computer Networks, 182, 107515.

[5] Kurdoglu, E., Liu, Y., Wang, Y., Shi, Y., Gu, C., and Lyu, J. (2016, May). Real-time bandwidth prediction and rate adaptation for video calls over cellular networks. In Proceedings of the 7th International Conference on Multimedia Systems (pp. 1-11).

[6] Tian, G., and Liu, Y. (2012, December). Towards agile and smooth video adaptation in dynamic HTTP streaming. In Proceedings of the 8th international conference on Emerging networking experiments and technologies (pp. 109-120).

[7] Jiang, J., Sekar, V., and Zhang, H. (2012, December). Improving fairness, efficiency, and stability in http-based adaptive video streaming with festive. In Proceedings of the 8th international conference on Emerging networking experiments and technologies (pp. 97-108).

[8] Raiciu, C., Paasch, C., Barre, S., Ford, A., Honda, M., Duchene, F., and Handley, M. (2012). How hard can it be? designing and implementing a deployable multipath TCP. In 9th USENIX symposium on networked systems design and implementation (NSDI 12) (pp. 399-412).

[9] Yin, X., Jindal, A., Sekar, V., and Sinopoli, B. (2015, August). A control-theoretic approach for dynamic adaptive video streaming over HTTP. In Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication (pp. 325-338).

[10] Raca, D., Leahy, D., Sreenan, C. J., and Quinlan, J. J. (2020, May). Beyond throughput, the next generation: a 5g dataset with channel and context metrics. In Proceedings of the 11th ACM Multimedia Systems Conference (pp. 303-308).

[11] Shih, S. Y., Sun, F. K., and Lee, H. Y. (2019). Temporal pattern attention for multivariate time series forecasting. Machine Learning, 108(8), 1421-1441.

[12] Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., and Zhang, W. (2021, May). Informer: Beyond efficient transformer for long sequence time-series forecasting. In Proceedings of AAAI.

[13] He, Q., Dovrolis, C., and Ammar, M. (2005). On the predictability of large transfer TCP throughput. ACM SIGCOMM Computer Communication Review, 35(4), 145-156.

[14] Swany, M., and Wolski, R. (2002, November). Multivariate resource performance forecasting in the network weather service. In SC'02: Proceedings of the 2002 ACM/IEEE Conference on Supercomputing (pp. 11-11). IEEE.

[15] Ruan, L., Dias, M. P. I., and Wong, E. (2019). Machine learning-based bandwidth prediction for low-latency H2M applications. IEEE Internet of Things Journal, 6(2), 3743-3752.

[16] Sato, N., Oshiba, T., Nogami, K., Sawabe, A., and Satoda, K. (2017, July). Experimental comparison of machine learning-based available bandwidth estimation methods over operational LTE networks. In 2017 IEEE Symposium on Computers and Communications (ISCC) (pp. 339-346). IEEE.

[17] Mirza, M., Sommers, J., Barford, P., and Zhu, X. (2007). A machine learning approach to TCP throughput prediction. ACM SIGMETRICS Performance Evaluation Review, 35(1), 97-108.

[18] Winstein, K., Sivaraman, A., and Balakrishnan, H. (2013). Stochastic forecasts achieve high throughput and low delay over cellular networks. In 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13) (pp. 459-471).

[19] Gers, F. A., Schmidhuber, J., and Cummins, F. (2000). Learning to forget: Continual prediction with LSTM. Neural computation, 12(10), 2451-2471.

[20] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., and Polosukhin, I. (2017). Attention is all you need. In Advances in neural information processing systems (pp. 5998-6008).

[21] Yu, F., Koltun, V., and Funkhouser, T. (2017). Dilated residual networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 472-480).

[22] Clevert, D. A., Unterthiner, T., and Hochreiter, S. (2015). Fast and accurate deep network learning by exponential linear units (elus). arXiv preprint arXiv:1511.07289.

[23] Devlin, J., Chang, M. W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

[24] Riad, M. (2020). Machine Learning QoE Prediction for Video Streaming over HTTP (Doctoral dissertation).

[25] Zou, X. K., Erman, J., Gopalakrishnan, V., Halepovic, E., Jana, R., Jin, X., and Sinha, R. K. (2015, February). Can accurate predictions improve video streaming in cellular networks?. In Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications (pp. 57-62)

[26] Fonti, V., and Belitser, E. (2017). Feature selection using lasso. VU Amsterdam research paper in business analytics, 30, 1-25.

[27] Azari, A., Papapetrou, P., Denic, S., and Peters, G. (2019, October). Cellular traffic prediction and classification: A comparative evaluation of LSTM and ARIMA. In International Conference on Discovery Science (pp. 129-144). Springer, Cham.

[28] Hassan, M.K., Syed Ariffin, S.H., Ghazali, N.E., Hamad, M., Hamdan, M., Hamdi, M., Hamam, H. and Khan, S., 2022. Dynamic Learning Framework for Smooth-Aided Machine-Learning-Based Backbone Traffic Forecasts. Sensors, 22(9), p.3592.

[29] Lopez, J. H. (1997). The power of the ADF test. Economics Letters, 57(1), 5-10.

[30] Shin, Y., and Schmidt, P. (1992). The KPSS stationarity test as a unit root test. Economics Letters, 38(4), 387-392.