Received 4 October 2022, accepted 18 November 2022, date of publication 28 November 2022, date of current version 6 December 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3225401

RESEARCH ARTICLE

Machine Learning to Optimize TCP Communications Over High Frequency Communications Links

ALVIN DMELLO[1], DHAMMIKA JAYALATH[1], (Senior Member, IEEE), ERNEST FOO[2], (Member, IEEE), AND JASON REID[1], (Member, IEEE)
[1]Science and Engineering Faculty, Queensland University of Technology, Brisbane, QLD 4000, Australia
[2]Information and Communication Technology, Griffith University, Southport, QLD 4222, Australia

Corresponding author: Alvin Dmello (alvinthomasarthur.dmello@hdr.qut.edu.au)

ABSTRACT High Frequency communication is a proven method of beyond line of sight (BLOS) communications for decades. With the advent of Wideband HF (WBHF), the ability to communicate data and more specifically TCP over HF is being researched worldwide. HF-TCP, an optimised TCP for HF links provides improved TCP communication over challenging High Frequency (HF) communication systems. It relies on the modification of Forced Retransmission Timeout (fRTO) and Mean Segment Size (MSS) to improve the reliability of communication sessions. Calculating the fRTO for a communication session that would provide the optimum result over a communication link is a complex task due to the timers that manage the session and the HF channel characteristics. In this paper, the use of Machine Learning (ML) techniques to dynamically predict fRTO and MSS to improve the fRTO calculation process for all communication instances is proposed. To achieve this, a Predictor Model is used to predict fRTO and MSS while an Optimiser model optimises the Predictor model's output. Decision Tree Regression was proven to be the most accurate among the various ML algorithms tested with 82 percent prediction accuracy. The performance of HF-TCP with proposed predicted fRTO and MSS is compared with that of standard TCP and the performance of the Predictor and Optimiser models is also analysed. The results show 72 percent of instances have an improvement in link efficiency when using HF-TCP with predicted fRTO and MSS over standard TCP.

INDEX TERMS HF communications, HF-TCP, TCP/IP, wireless communications.

I. INTRODUCTION

High Frequency (HF) communications have been the only communication medium for beyond line of sight wireless communications other than satellite. Satellite denied environments have become a known risk for various capabilities including the most commonly used GPS. Capability to destroy or jam satellites has been demonstrated on many occasions [1]. HF remains the only ground-based communication method over very long distances in cases where satellite communications are not available. The ability of HF to reach beyond line of sight (BLOS) is due to the property of HF signals to bounce off the ionosphere thereby reaching longer distances. While HF provides communication capability over large distances, it is also a medium that is highly susceptible to electromagnetic interference noise due the frequency band it operates in. The noise sources includes man made noise from operating machinery to environmental noise such as solar winds. It is possible to reduce man made noise by operating from quieter locations but environmental noise is beyond human control. Moreover, the HF channel is standardised for 3 kHz channel bandwidth which is appropriate for most voice communications but highly limited in the amount of data that can be encoded as per Nyquist theorem [2]. For this reason, the use of HF has been limited to voice and very low bandwidth data communication like low speed serial data. Additional layer 2 protocols such as STANAG 5066 provide some level of reliability at the cost of bandwidth

The associate editor coordinating the review of this manuscript and approving it for publication was Anandakumar Haldorai.

which is already limited. STANAG 5066 F.12 provides the recommendation based RFC 3135 [3] for a proxy server but has limitations such as interoperability and other constraints highlighted in a study conducted by Isode [4].

Wide Band HF (WBHF) is being trialed and in the process of being standardised across the globe. WBHF promises channel bandwidths of up to 48 kHz which increases prospects of using higher data rates [5]. Higher data rates open up possibilities of utilising HF for other capabilities where most of the communication is data based. This includes Internet Protocol (IP) which is by far the most utilised protocol for wired and wireless communications. Although higher bandwidths are available with WBHF, the problem of reliability remains a challenge due to environmental factors. Transmission Control Protocol over Internet Protocol (TCP/IP) is resistant to errors to a certain degree but cannot cope with HF links that have low reliability and higher delays [6]. In order to overcome the poor error and delay tolerance of TCP/IP, modifications are required to the way TCP operates. DMello et al. [7] have conducted analysis and research on TCP behaviour over HF and proposed modification to TCP in order to improve tolerance while being backward compatible to standard TCP. DMello et, al. proposed HF-TCP as a solution to improve reliability by introducing the fRTO parameter that can improve communications success rates when using TCP. fRTO is a set of two independent parameters, F1 and F2 which are used in HF-TCP to control timeouts. F1 is used to set the RTO timer itself and F2 is the fixed value the RTO timer resets to whenever it expires. F1 and F2 rely on the link condition to set the fRTO parameters that is most suitable for each operating condition. The impacts and benefits of fRTO has been detailed by DMello et al. [7] in their paper.

These conditions are interleaving delay, Bit Error Rate (BER), transmission rate and the Mean Segment Size (MSS) of the TCP/IP session. Of these, interleaving delay, Bit Error Rate (BER) and transmission rate are calculated as per MIL-STD-110D [8]. fRTO is then calculated for different MSS to identify the best value for the transmission that comprises of minimum transmission time and high probability of success.

Due to the numerous permutations a link can operate under, there are a total of approximately eight billion combinations and outcomes. Such computation needs high processing power and algorithms to provide a performance comparable to standard TCP. To over come these shortcomings, this paper proposes the use of Machine Learning (ML) algorithms to predict fRTO and MSS for a given over the air operating condition.

Our contribution provides,

- A Machine Language technique to overcome the over heads of calculating fTRO for HF-TCP; and
- Optimisation of HF-TCP parameters for the best performance.

A ML model has been developed that is capable of predicting the fRTO parameter for successful transmission in the least amount of time. The data size calculation has also been included for prediction to optimise the fRTO values further.

Our proposed solution makes use of two ML models, the Predictor Model and the Optimiser Model. The Predictor Model chooses the fRTO and MSS values for successful transmission and these values are then optimised using the Optimiser Model such that the transmission duration is minimised. Transmission duration is the time taken by HF-TCP to transfer data from the source to destination. This includes the session initiation, maintenance and tear-down time. Transmission duration is referred here because that is the delay experienced at the application layer. The throughput remains constant at layer 1 and 2 of the Open System Interconnection (OSI) model once the link is established whereas transmission duration can vastly vary when retransmission rate is high. Although throughput and transmission duration are interrelated, transmission duration provides a more accurate interpretation of performance. Our proposed solution is able to provide 82 percent prediction accuracy and 72 percent of the time an improvement of up to 81 percent in transmission duration.

There are various ML development packages that are either standalone development environments or libraries that can be used by popular programming languages like Python to implement machine learning. Scikit-Learn and Keras are among the popular ML libraries for Python [9], [10]. TensorFlow is a framework developed by Google specialising in neural networks [11]. Other packages include Cloudera Oryx, CUDA-Convent, ConvnetJS and many more. Matlab and opencv too provide support for ML development. A comparison of Matlab and opencv was conducted by Ahmed and Waleed [12]. Each of the libraries and packages are usually targeted to certain types of applications and they differ in the computing resources they utilise for computation like the Mxnet for heterogeneous distributed systems, R for predictive modelling and NLTK for language processing [13], [14], [15].

Scikit-Learn is an open source library for Python used for data mining and machine learning applications [9]. It provides a wide variety of algorithms and a robust ML development pipeline. Scikit-Learn has been extensively used in ML based research [16], [17], [18], [19]. As Scikit-Learn provides access to a variety of algorithms to enable full scaled testing for comparison and also as Scikit-Learn- has been used for other comparative modelling research, Scikit-Learn library for developing ML models in Python has been used.

OMNet++ is a simulation platform used for radio based simulation for research including HF simulation [20], [21]. HF-TCP was also developed with simulations conducted on OMNet++ [22]. To obtain consistent and comparable results, OMNet++ was used for simulations.

Scikit-Learn is used to build the ML models and OMNet++ as the simulation platform.

The rest of the paper is organised as follows, related works is discussed in Section II and then present the proposed solution in Section III which consists for a workflow utilising the two ML models. Section IV provides details on the algorithm and selection criteria used to build the models. Section V describes the simulations conducted on the proposed solution and Section VI presents the results and evaluation. Conclusion is covered in Section VII.

II. RELATED WORKS

Machine Learning is a subset of Artificial Intelligence (AI) studies [23]. ML has been a major research topic due to the large application base where it can be implemented [24]. ML has been used in various situations where the ML algorithms learns from sample data to predict outputs based on inputs that may not part of the sample dataset [25].

Regression and classification are the two main types of ML techniques used depending on what kind of prediction is expected form the ML algorithm [25]. Classification or classifiers are used when prediction is based on class labels. For example facial recognition algorithms that predict similarities between two images. Regression algorithms are used where a value is required to be predicted. For example weather parameters like temperature. Within the two ML techniques there are various algorithm types like decision trees, k-neighbor, Bagging Regressor, etc.

ML has been implemented in various situations in communications with promising results. The application of ML in wireless networks is not new. Jagannath et al. [26] have provided a comprehensive survey of the state of the art in the application of machine learning techniques to address key problems in different aspect of wireless communications in Internet of Things (IoT) implementations. Wang et al. [27] in their paper on HF and its future have indicated the use of ML for future developments of HF systems because of the unpredictable nature of the transmission medium. They propose the use of AI, especially machine learning and deep learning to make HF systems smarter in frequency selection and other aspects of HF communications. Samuel et al. [28] have proposed a technique to implement multiple input multiple output (MIMO) in HF systems using ML for cognitive engine models. They claim their solution provides increased channel capacity in HF communications when using ML. Other interesting works include channel selection and Data Scheduling Approach for High-Frequency Communications in Jamming Environment by Wen et al. [29] in which the authors propose a Q-learning method to enhance communications in difficult jamming conditions resulting in better performance of HF in jamming situations.

ML benefits are recently being explored in research works surrounding Long-Term Evolution (LTE) and 5G networks where end user throughput is predicted in order to provide better service [30]. Supervised ML algorithms for Radio Frequency (RF) interference flagging are also being explored in the field of astronomy [31]. Improvements in RF path loss predictions during design and operations are examined by Zang et al. [32].

For decades, HF has been used for beyond line of sight communications. However, advances in AI technologies allow ML to improve HF communications' performance and efficiency, making it widely available for military and other applications such as HAM radio.

The use of TCP in HF communications has been challenging and various attempts have shown limited success. Some improvements have been presented that improves the possibility of using TCP over HF [20] but a complete solution has not yet been developed. Improving TCP to adapt to an HF environment has been proposed by DMello et al. [22]. Their solution present HF-TCP which is a backward compatible TCP with an innovative method in the way TCP manages Retransmission Timeout (RTO) timers and the Maximum Segment Size (MSS). HF-TCP is further evaluated in their research presenting the improvements over standard TCP [7]. Due to the nature of a HF transmission, HF-TCP faces a challenge for predicting the best fRTO and MSS for a given transmission scenario. Accurate prediction of fRTO is essential in obtaining higher success rates and quicker transmission to provide better data rates.

This paper builds upon the HF-TCP implementation developed in [7] and implement ML to make predictions for fRTO and MSS. Our implementation of ML for predicting the fRTO and MSS will complement HF-TCP and provide a complete HF-TCP solution with the prediction engine supported by two models.

III. MACHINE LEARNING FOR HF-TCP

HF-TCP is an improved version of the standard TCP that is backward compatible [22]. It implements the ability to control the mean segment size (MSS) and the retransmission timeout (RTO) timer. In standard TCP, the RTO is dynamically calculated based on algorithms. In our proposed HF-TCP implementation, the RTO, which is made up of F1 and F2 and called fRTO and the MSS are predicted by a process using two models which are trained on simulated data. Machine learning models were developed to predict fRTO parameters to be used in HF-TCP for data transfers using the minimum possible time. These ML models were built using Scikit-Learn libraries and Python [9].

[7] presents a technique to obtain fRTO and MSS values using simulation and data analysis. However, it was observed that there can be more than one instance where a combination of fRTO and MSS will result in a successful communication for a given transmission scenario. Each of these instances could have different values of transmission duration. Therefore, it is a timely concern to develop methods to select the fRTO corresponding to the minimum duration value.

With only a Predictor Model, it cannot be ensured that the duration for the predicted fRTO and MSS is minimised. Since a conditional prediction cannot be conducted on the ML models, the Predictor and Optimiser Model with an iterative algorithm to find optimum parameters for a given transmission is used. The Predictor Model predicts fRTO and MSS for successful communication. The Optimiser Model then predicts the duration for the predicted fRTO and MSS. The iteration continues by reducing the duration by one seconds for each run until no further reduction in duration can be

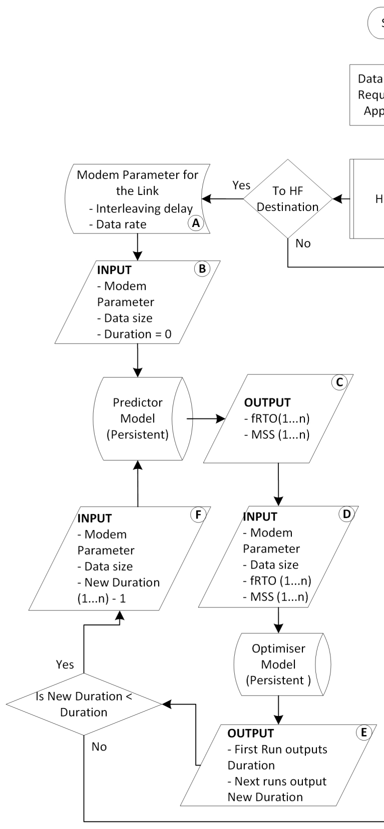

FIGURE 1. **HF-TCP with ML model flowchart.**

obtained. As the TCP duration is rounded to the nearest second, a decision has been made to reduce the predicted transmission duration during the iterations by one-second intervals.

The Predictor Model developed is used to predict fRTO and MSS given the fixed modem parameters for a particular link condition and the data to be transmitted. The Optimiser Model performs predictions to minimize the transmission duration. To implement the models with HF-TCP, persistent models are created. A persistent model is a model that has already been trained. This persistent model is loaded in memory and can be queried without the need to re-train and run the model. Processing with persistent models is very quick. The persistent models were developed using Python in Scikit-Learn and saved as a file which can then be loaded on demand and retained in memory for multiple predictions. Model development is discussed in Section IV.

Fig. 1 shows the flow diagram of implementation of HF-TCP with ML. The process starts with a data transfer request from the application. This provides the data size to be transmitted. As the HF-TCP is backward compatible, based on the gateway, a decision is made whether the data will be transmitted over HF. If the data is going to be transmitted to the end point over HF, fRTO and MSS needs to be calculated. The modem parameters are based on the link quality. The application provides the data size. With the modem parameters and the data size, the first prediction is made, called PM1 for F1, F2 and MSS. In Fig.1 block B shows the input and block C shows the output for this step.

PM1: Predictor Model (Run = 1)	
Inputs	Data, Delay, Rate, Duration=0
Output	$F1_{PM1}, F2_{PM1}\ and\ MSS_{PM1}$

The outputs obtained is the predicted best values for F1, F2 and MSS where duration value is minimum. Although the values predicted are for minimum duration, there exists some error factor. To reduce this error factor, the F1, F2 and MSS values are optimised by iteration through both models in subsequent runs. In the next run, the Optimiser Model is used.

OM1: Optimiser Model (Run=1)	
Inputs	$Data, Delay, Rate, F1_{PM1}, F2_{PM1}, MSS_{PM1}$
Output	$Duration_{OM1}$

At this stage the F1, F2, MSS and the Duration for the transmission are available. For OM1 the input is shown as block D and output is shown as block E in Fig. 1. The duration obtained from the OM1 is then further optimised. To perform the optimisation both models are used. The reasoning as to why the optimisation process is necessary and effective is discussed in Section III. First, the Predictor Model is run again but this time with a duration one second less than the duration predicted by the Optimiser Model, $Duration_{OM1}$.

PM2: Predictor Model (Run = 2)	
Inputs	$Data, Delay, Rate\ and\ Duration_{OM1} - 1$
Output	$F1_{PM2}, F2_{PM2}\ and\ MSS_{PM2}$

Inputs to the PM2 run are shown as block F while output is shown as block C in Fig. 1. PM2 run predicts $F1_{PM2}, F2_{PM2}\ and\ MSS_{PM2}$ with $Duration_{OM1} - 1$. The predictions are again provided as inputs shown as block D to the Optimiser Model resulting in OM2 which predicts $Duration_{OM2}$ as shown in block E of Fig. 1. Algorithm 1 shows the computation of F1, F2 and MSS.

IV. MODEL DEVELOPMENT

In Machine Learning, model development and maintenance is performed using a model pipeline. The model pipeline is a sequential process of activities that defines how data is collected, processed, modelled and predicted. There has been a lot of research on Machine Learning Pipeline development but for the purpose of this paper, a simpler approach is adopted as noted by Quemy et al. [33] in their paper on ML pipeline optimisation.

Algorithm 1 Computing fRTO and MSS

```
Environment Definition:
    define struc $modem\{delay, rate\}$
    define Predict $Model\{model\ \ query\ \ parameters\}$
    $duration \leftarrow 0$
    $x \leftarrow 0$
Procedure:
 1: Predict $PM1\{modem, data\_size, duration\}$
 2: $PM1 \rightarrow F1_{PM1}, F2_{PM1}, MSS_{PM1}$
 3: Predict $OM1\{modem, data\_size, PM1\}$
 4: $OM1 \rightarrow duration_{OM1}$
 5: Predict $PM2\{modem, data\_size, duration_{OM1} - 1\}$
 6: $PM2 \rightarrow F1_{PM2}, F2_{PM2}, MSS_{PM2}$
 7: Predict $OM2\{modem, data\_size, PM2\}$
 8: $OM2 \rightarrow duration_{OM2}$
 9: while $duration_{OM2} < duration_{OM1}$ do
10:     Predict $PM(x)\{modem, data\_size,$
            $duration_{OM1(x-1)} - 1\}$
11:     $PM2 \rightarrow F1_{PM(x)}, F2_{PM(x)}, MSS_{PM(x)}$
12:     Predict $OM(x)\{modem, data\_size, PM(x)\}$
13:     $OM2 \rightarrow duration_{OM(x)}$
14:     if $duration_{OM(x)} > duration_{OM(x-1)}$ then
15:         break
16:     end if
17:     $x \leftarrow x + 1$
18: end while
19: $F1 = F1_{PM(x-1)}$
20: $F2 = F2_{PM(x-1)}$
21: $MSS = MSS_{PM(x-1)}$
Outputs:
    $F1, F2$ and $MSS$
```

Our model pipeline consisted of five blocks as shown in Fig. 2. Data Gathering, Data Processing, Feature Selection and Algorithm Selection are discussed in the following sections while Model Evaluation is conducted in Section VI

A. DATA GATHERING

HF-TCP has been developed on the OMNet++ [34] simulation platform. OMNet++ has continued to be used to gather data that was used to train our model. The simulation setup used and the parameters are the same as those used in HF-TCP development environment [7], [22]. Due to the large amount of data, the simulation was run for groups of parameter values at a time. All data was collected and stored in a comma separated value (csv) format for processing flexibility. Using the csv format for data gives us the flexibility to process it using a variety of tools including excel, R and Matlab. The dataset consisted of approximately five hundred thousand samples.

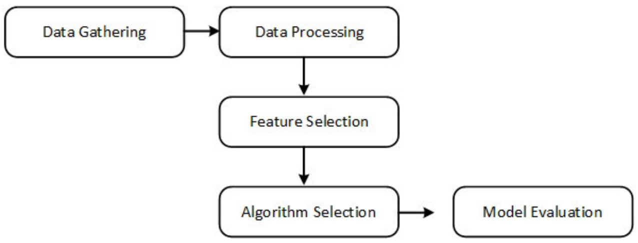

FIGURE 2. ML model development pipeline.

B. DATA PROCESSING

The data was then processed to make it suitable to be input into a model. To achieve that, the data was parsed to ensure there were no missing data. Outliers were handled by either re-running the data to maintain accuracy or eliminating it. Delay and rate can only have certain fixed values as defined by MIL-STD-110D [8]. The parameter, success, too can have only two values. Since delay, rate and success are not continuous variables, but have pre-defined values, these parameters are considered as categorical values. Categorical values cannot be used in all ML models. Since a regression model has been used, in order to convert our categorical values to numerical values, One-Hot Encoding (OHE) [16] technique is utilised. OHE creates more columns equal to the number of different values a parameter can have. For example, since the delay parameter can have four fixed values of 0.12s, 3.49s, 6.87s and 10.24s as per MIL-STD-110 [8]; Four columns are required to represent the data. The data record row will have an entry of ‘1’ in the respective column for a delay value. All other records for that row will hold a value of ‘0’. Table 1 shows an example of OHE implementation for delay where delay has four values. The row represented by ‘0 0 0 1’ is allocated to one value of delay while ‘0 0 1 0’ is allocated to another value of delay. In this way the four categorical delay values are converted to numerical equivalents for the model.

Using the same OHE technique, rate and success parameters are also converted to numerical values since rate is a defined constant as per MIL-STD-110D [8] and success hold two values, successful or unsuccessful.

C. FEATURE SELECTION & IMPORTANCE

A ML model consists of one or more features and targets. Features are input variable and targets are the resultant output. Different input variables or features have different levels

TABLE 1. OHE used on delay values for modelling.

Delay1	Delay2	Delay3	Delay4
0	0	0	1
0	0	1	0
0	1	0	0
1	0	0	0

TABLE 2. Feature importance score for predictor model.

Feature	F1	F2	MSS
Duration	0.588	0.570	0.638
Data Size	0.128	0.162	0.085
Delay	0.028	0.026	0.022
Rate	0.030	0.020	0.043
Success	0.040	0.055	0.020

TABLE 3. Feature importance score for optimiser model.

Feature	Duration
Data Size	0.194
Delay	0.056
Rate	0.066
Success	0.031
F1	0.029
F2	0.107
MSS	0.177

of impact on prediction of the target. This impact is called feature importance. Feature importance is calculated using permutations of feature values analysed against the model score. Feature importance is a score assigned to features related to the impact it has on predicting the target. Feature importance assists in understanding the data and the model which aids in making optimal feature selection. Certain features can be eliminated in models if the feature does not contribute or contribution is minimal in predicting the target. To keep model processing to a minimum, some features are eliminated. In our case, all features have been used to build the model since our prediction makes use of all features. The same dataset has been used to create a Predictor Model and an Optimiser Model that have been trained differently to provide different predictions. The Predictor Model chooses the fRTO and MSS values for successful transmission and these values are then used on the Optimiser Model to optimise the fRTO and MSS such that the transmission duration is the least. Our models contains eleven features and three targets. While calculating the feature importance of categorical values, the feature importance is averaged across the binary columns. Hence there is one value for delay, rate and success. This effectively gives us five features and three targets.

1) PREDICTOR MODEL

Table 2 shows the feature importance scores for the features in our Predictor Model. Duration and data are observed to be significant contributors to prediction of F1, F2 and MSS. Delay contributes mostly to F1 prediction while rate and success have marginal contribution to prediction.

2) OPTIMISER MODEL

Table 3 shows the feature importance for the Optimiser Model. The data size and MSS are clearly the most important features used to predict the duration of transmission in the Optimiser Model.

D. ALGORITHM SELECTION

The next step in the model pipeline is the selection of the algorithm that will process the data to build our model. Since our targets are numerical continuous valuables, a regression model has been chosen over a classification type model. Since MIL-STD-110D defines a set of operating parameters for any given link condition, the range of operating parameter values is known. Regression analysis is therefore chosen as the ML method for our models.

For the Predictor Model, since there are three target variables, a multi-output regression algorithm with a base estimator has been chosen for use. The multi-output regressor generates multiple models depending on the number of target variables using the base estimator and then combines them to form one model. Two types of multi-output regression models are used, Multioutput Regressor and Regressor Chain [35]. For each of these, experiments with various base estimators are conducted to compare and select the best performing base estimator with each multi-output regression algorithm.

The Model is assessed based on the following key parameters that provides the knowledge of how the models are performing. Each of the parameters are briefly discussed below,

- *Model Training Score:* The model training score is the measure of the fit between the modelling algorithm and the data. It indicates how well the model explains the training data. In our evaluation, it is expressed as a percentage value with higher percentage being better.
- *Accuracy:* Accuracy score is the measure of how accurate the predictions are compared to the true values. Accuracy for the model is calculated on training and test dataset. The dataset used for the model evaluation is split with 80 percent of the data forming the training dataset and the other 20 percent is the test dataset. Accuracy is the resultant score calculated based on the deviation between the predicted and true values in the test dataset

and is represented as a percentage. Higher the percentage, better the accuracy.

- *R-Squared:* Coefficient of determination, R-Squared, is a measure of how well a dependent variable can be predicted based on the independent variable. It has a minimum value of 0 to a maximum value of 1. The higher the value, the better the prediction.
- *Explained Variance Score (EVS):* EVS is the measure of variance that the model accounts for during its prediction. It can be estimated [9], [36] as,

$$(y, \widehat{y}) = 1 - \frac{V_{ar}[y - \widehat{y}]}{V_{ar}[y]}$$

where,
$\widehat{y}$ is predicted output value,
y is the true value, and
V_{ar} is the standard deviation.
EVS has a value between 0 and 1 with 1 being the best possible score.

These attributes were calculated for each of the five base estimators with Multioutput Regressor and Regressor Chain regression algorithms. As seen in Table 4 and Table 5, the Decision Tree Regressor and Bagging Regressor provided best results for each of the attributes when used with Multioutput Regressor and Regressor Chain regression algorithms. Between the Decision Tree Regressor and Bagging Regressor, Bagging Regressor took six times longer to execute than the Decision Tree Regressor. Regressor Chain takes into account relationship between features while Multioutput Regressor treats each feature independently. Comparing the results of Multioutput Regressor and Regressor Chain, Regression Chain took almost twice as long to execute on average and had lower performance as compared to the Multioutput Regressor. Based on the comparison results, Multioutput Regressor with Decision Tree Regressor as the base estimator is chosen to build the Predictor Model.

In the case of Optimiser Model which is a single-output model, the same set of five base estimators are used to compare them against the attributes. Table 6 shows the results of the comparison. The Decision Tree Regressor and Bagging Regressor provided the best results for each of the attributes but because the Bagging Regressor takes almost six times longer to execute on average, the Decision Tree Regressor is chosen as the base estimator to build the Optimiser Model as well.

E. MODEL BUILD

Sci-kit Learn [9] is used to build the models which provides the libraries and a platform for complex machine Learning development. The development and evaluation is done using Python. *sklearn.tree.DecisionTreeRegressor* class is used along with *sklearn. multioutput.MultiOutputRegressor* class to build our models based on the algorithm selection criteria discussed in Section IV-D.

The problem of over-fitting and under-fitting is analysed when using a model. Over-fitting is a problem when the model learns in too much detail thereby learning noise in the data. Under-fitting is when the model does not have enough information to learn. Both over-fitting and under-fitting scenarios impact the accuracy of the model and the predictions it makes. For a decision tree, the depth of the tree can be controlled to avoid over-fitting or under-fitting. When no restrictions are placed on the algorithm, our model generates a depth of 50 to 60 levels. On experimenting with various depths, the decision tree was decided to be pruned at a maximum of 50 levels. At a depth of 50, our model provides the highest accuracy. The reason the maximum depth did over-fit the model was also due to the data processing that was performed earlier as explained in Section IV-B. Supervised learning is used and the evaluation is conducted on a 80-20

TABLE 4. Multioutput regressor - comparison of algorithms for predictor model.

Algorithm	Model Score	Accuracy	R-Squared	EVS-F1	EVS-F2	EVS-MSS
Bagging Regressor	79 %	81 %	0.78	0.69	0.76	0.90
Decision Tree Regressor	80 %	82 %	0.78	0.69	0.76	0.90
Hist Gradient Boosting Regressor	59 %	65 %	0.59	0.53	0.61	0.62
K - Neighbour Regressor	71 %	75 %	0.67	0.60	0.67	0.73
Gradient Boosting Regressor	38 %	53 %	0.38	0.34	0.40	0.41

TABLE 5. Regressor chain - comparison of algorithms for predictor model.

Algorithm	Model Score	Accuracy	R-Squared	EVS-F1	EVS-F2	EVS-MSS
Bagging Regressor	48 %	43 %	0.46	0.69	0.54	0.6
Decision Tree Regressor	42 %	70 %	0.42	0.70	0.63	0.3
Hist Gradient Boosting Regressor	33 %	58 %	0.32	0.53	0.52	0.3
K -Neighbour Regressor	58 %	69 %	0.54	0.60	0.59	0.45
Gradient Boosting Regressor	26 %	49 %	0.26	0.34	0.35	0.15

TABLE 6. **Comparison of algorithms for optimiser model.**

Algorithm	Model Score	Accuracy	R-Squared	EVS-Duration
Bagging Regressor	99 %	98 %	0.98	0.98
Decision Tree Regressor	99 %	99 %	0.99	0.98
Hist Gradient Boosting Regressor	94 %	91 %	0.94	0.94
K - Neighbour Regressor	53 %	22 %	0.35	0.22
Gradient Boosting Regressor	74 %	75 %	0.75	0.74

split of the model data into training and test components. The model is trained on 80 percent of the data and made to predict the other 20 percent. The prediction results are then compared with the real values to estimate the models accuracy and error parameters. For regression models it is important to train the model with feature values that cover the entire range from maximum to minimum for each feature. Prediction beyond the threshold values from what the model is trained on are inaccurate as the model has not learnt about the input and output relationship outside the range.

V. EVALUATION

To evaluate the proposed solution a performance analysis was conducted on the Predictor and Optimiser models and also the performance of the solution using the two models as discussed in Section III. The results of the analysis are then discussed in Section VI.

Simulations are conducted in scenario where over the air characteristics are maintained and controlled by the modem as per MIL-STD-110D. The MIL-STD-110D provides error recovery using modulation type and interleaving such that the Signal to Noise Ratio (SNR) achieved is to a degree acceptable by the model to interpret the signal received. The modulation type and interleaving depth then mandate the data rate the modem is able to transmit on. The details of how this is achieved is referenced in the MIL-STD-110D. The simulations are run for a subset of combinations of allowed data rates and interleaving depth that translates into delay. A random set of data blocks are selected for each simulation.

A. MODEL SUITABILITY

The solution proposed in this paper for predicting fRTO and MSS values for a transmission utilises two models, the Predictor Model and the Optimiser Model. Each of the two models have a different set of inputs and outputs. The performance of each model is examined separately. The performance of the model is evaluated based on the accuracy of prediction. This is achieved by comparing the results of the predicted values against the actual value. In Python using Scikit-Learn, split the dataset randomly into two parts with a ratio of 80:20 where 80 percent of the data is used to train the model called training dataset and 20 percent of the data is used for testing called test dataset. These two data sets are kept separate from each other throughout the performance measuring process.

The training dataset is firstly used to train the model. The features are then separated and extracted from the test dataset and used on the trained model to predict the target values. The predicted targets are then compared with the true target values from the training database. The relationship between the predicted values and the true values is used to calculate the accuracy of the model.

1) PREDICTOR MODEL

The Predictor Model is built to predict three key criteria, namely, F1, F2 and MSS. F1 and F2 values together make up fRTO. F1 is defined as the maximum RTO value. When the RTO timer reaches the F1 value, it is reset to a value, F2. Both F1 and F2 can have a maximum value of 240 seconds. MSS is the mean segment size of the HF-TCP transmission and given the nature of transmission in an HF-TCP implementation, the MSS value remains constant throughout the transmission.

The Predictor Model is a multioutput model as the targets are F1, F2 and MSS. In ML, multioutput models are a combination of three individual models, one for each target. Multioutput Regressor has been used due it its accuracy over Regressor Chain as discussed in Section IV-D. Each of the targets are assessed individually and also the model is assessed as a complete entity. The graphs showing the deviations in values is limited to one thousand samples for legibility. The calculation of accuracy and standard deviation are performed on the entire dataset.

2) OPTIMISER MODEL

The Optimiser Model is used to optimise the F1, F2 and MSS values obtained from the Predictor Model to achieve the lowest transmission time. The model outputs the value *Duration* only and is built using *DecisionTreeRegressor* which is the same base algorithm used for the Predictor Model. The comparison against other models is discussed in Section IV-D.

B. ML FOR HF-TCP

To evaluate the performance of the complete solution, data is collected from a standard TCP run and from HF-TCP when using the ML models as discussed in Section III. The simulations are run on OMNet++ for both scenarios to provide consistent results for comparison. Both simulations are conducted with a different input to as compared to what the models have been trained on. This eliminates the possibility of the models predicting on known scenarios.

1) STANDARD TCP

The first simulation is conducted with a standard TCP. The simulation gathers data on the success of a transmission and the time taken to complete the simulation for each set of inputs. The inputs are recorded in a csv format and processed to identify the success and duration of transmission.

2) HF-TCP WITH ML MODEL

The next simulation is conducted with the same set of inputs as done in the standard TCP run, that is delay, data rate and data size for a network implementing HF-TCP with our ML models providing predicted F1, F2 and MSS values. The prediction is done using the process described in Section III. Again, the data is collected and recorded in a csv format and processed to only include success and duration of transmission for each set of input parameters. On average the time taken to make a prediction for a single transmission is in the order of 0.64 milliseconds. The prediction speed even after multiple iterations in some scenarios, is attributed to the persistent models. It is important to note that the prediction is required only once at the start of every new TCP session. The operational complexity is therefore deemed to be minimum given the frequency of computation and utilisation of persistent models.

VI. RESULTS

The results obtained from the simulations are placed into two groups. Firstly, individual model performance and suitability is discussed by examining the predicted and test values of target outputs for each of the models. The results presented are obtained from simulations as discussed in Section V.

A. MODEL PERFORMANCE

The Predictor and Optimiser models show the following performance.

1) PREDICTOR MODEL PERFORMANCE

Fig. 3 shows the deviation in values for F1 between the test and predicted data of the Predictor Model. The y-axis in seconds shows the deviation with a mean of zero and the scale going towards positive and negative. For F1, the Predictor Model predicts with an accuracy of up to 70 percent. The calculated standard deviation is 28 seconds for F1 in the range of 0 to 240 seconds. The accurate prediction of F1 and F2 results in the algorithm choosing the best possible values of MSS for the quickest transmission time.

Fig. 4 shows the deviation in values for F2 between the test and predicted data sets of the Predictor Model. This graph too shows the deviation in seconds on the y-axis from a mean of zero and scaling towards negative and positive values of seconds. For F2 and accuracy of up to 75 percent is achieved. The calculated standard deviation being 25 seconds in the range of 0 to 240 seconds.

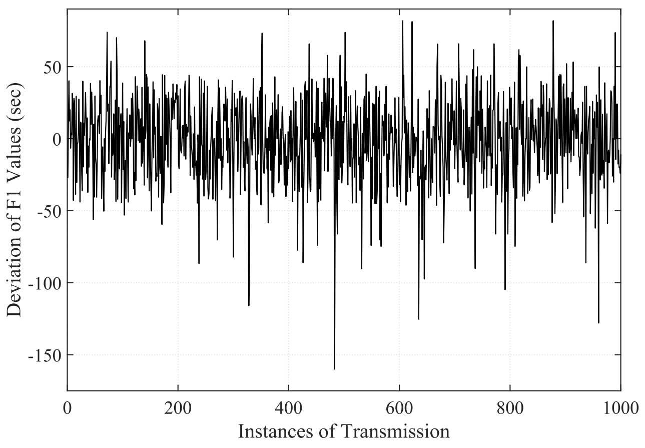

FIGURE 3. Deviation of F1 between test and predicted values.

Fig. 5 shows the deviation in values for MSS between the test and predicted data sets of the Predictor Model. The y-axis plots the number of bytes showing deviation from a mean of zero to positive and negative values. For MSS an accuracy of up to 90 percent is achieved and the calculated deviation is 46 bytes in the range of 0 to 580 bytes.

2) OPTIMISER MODEL PERFORMANCE

Fig. 6 shows the deviation in values for the value of Duration between the test and predicted data sets in the Optimiser Model. The y-axis shows the number of seconds from a mean of zero and moving towards positive and negative seconds. To make the graph legible, only one thousand samples from the entire dataset are shown. For Duration, an accuracy of up to 98 percent is achieved from the model with a standard deviation of 647 seconds.

B. ML PREDICTION FOR HF-TCP

The results obtained from the simulations in Section V are presented in Fig. 7. The x-axis plots the different instances of successful transmission and y-axis plots the duration of a successful transmission. The data for the graph has been sorted on the value duration for ease of analysis and interpretation. The continuous line indicates recorded duration when using our models with HF-TCP and the dotted line indicated standard TCP duration for transmission instances. From the graph it can be observed that that the model predicted F1, F2 and MSS values for a transmission instance is significantly lower towards the end of the graph. There is little or no improvement for the initial instances which are higher speed and lower delays like any good transmission characteristics. When the transmission characteristics degrade with lower speeds and higher delays required for error recovery using interleaving, significant improvement in the transmission duration is seen when using the models with HF-TCP. Standard TCP performance is significantly degraded and this is due to the inability of standard TCP to sustain long delays and slow speeds in transmission. In such scenarios RTO calculations using Karn's algorithm as in RFC

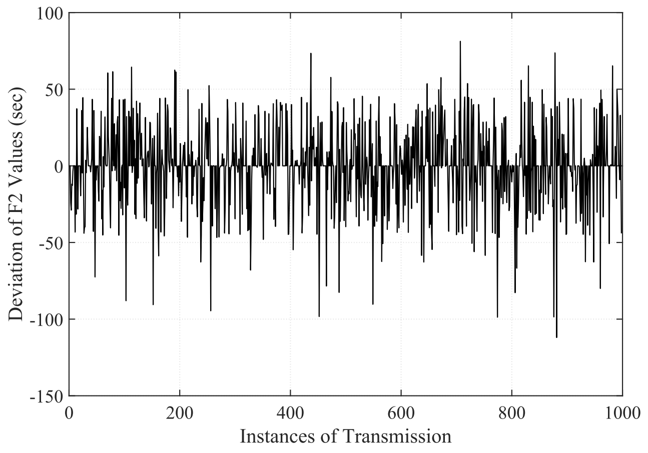

FIGURE 4. Deviation of F2 between test and predicted values.

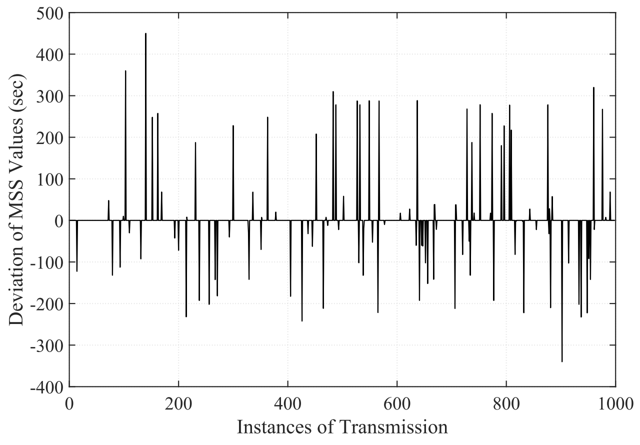

FIGURE 5. Deviation of MSS between test and predicted values.

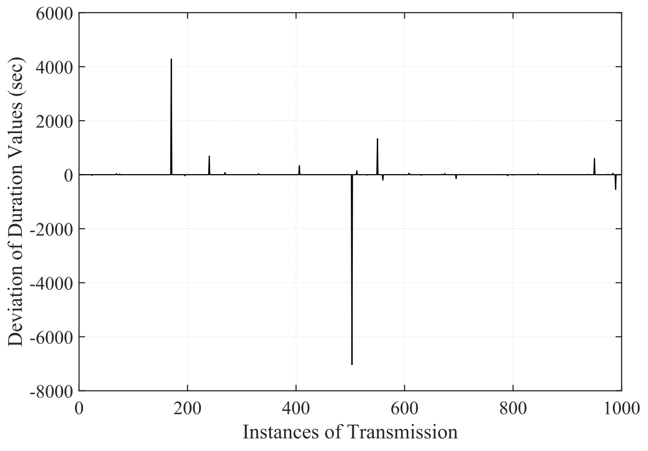

FIGURE 6. Deviation of duration between test and predicted values.

6298 [37] fails to provide effective communications. At the peak, there is an improvement of 81 percent in the time it takes to complete the transmission.

DMello et. al [7] used manual prediction on a subset of transmission conditions and present a result where on an average 15 percent of the cases resulted in improvement of transmission time. Using ML helps achieve 72 percent of cases with improved transmission time on a much larger subset of cases.

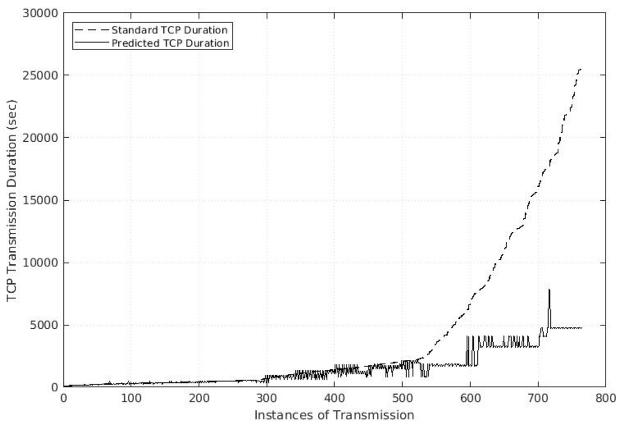

FIGURE 7. Simulated standard TCP transmission duration vs Model predicted transmission duration.

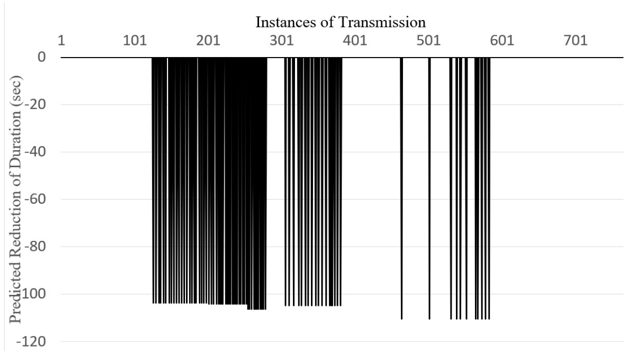

FIGURE 8. Improvement in transmission duration between Predictor and Optimiser Model predictions.

Another important finding is that, although the models provide improvement over standard TCP in 72 percent of the cases with a magnitude of 40 percent overall improvement. The standard TCP performs better in 28 percent of the cases but with only a 0.3 percent overall improvement in magnitude.

The benefits of the Optimiser Model are also examined. The intent of the Optimiser Model is to optimise the F1, F2 and MSS values obtained from the Predictor Model such that the transmission duration is the lowest that can be achieved. Fig. 8 shows the number of instances and the magnitude of reduction of duration predicted by the Optimiser Model for successful transmission instances over what is being predicted by the Predictor Model. An improvement of up to 110 seconds is achieved in certain instances when using the Optimiser Model over the Predictor Model for F1, F2 and MSS values. It is also important to note that the Optimiser Model always provides either improvement or no change in what the Predictor Model had predicted. There is never a deterioration of duration when using the Optimiser Model over the Predictor Model for F1, F2 and MSS values.

VII. CONCLUSION

The Predictor and Optimiser models together provide F1, F2 and MSS values that enables the transmission of successful data in least amount of time. Although the Predictor Model can be utilised by itself, the Optimiser Model provides further improvement to the final solution. Better transmission duration translates into improved link utilisation and improved throughput. Although this paper provides experimental results for HF environments and discusses TCP in the context of HF communications, the solution provided can be easily adapted for similar networks with low bandwidth and or longer delays.

Future improvement in WBHF and modulation to reduce errors and improve bandwidth will further complement the proposed solution in the paper. As discussed, erroneous nature of HF links is a natural phenomenon and even with higher bandwidths, the delay introduced to overcome the errors will continue to pose a challenge for standard TCP. Modifications of standard TCP timers to suit a transmission condition as proposed in this paper with HF-TCP will be required if TCP communications is to be achieved on links with challenges such as those experienced with HF links.

REFERENCES

[1] E. Tepper. (May 2020). *Space Force One: The Complex Laws of Space Warfare*. [Online]. Available: https://ssrn.com/abstract=3672841

[2] S. Aramvith, C.-W. Lin, S. Roy, and M.-T. Sun, "Wireless video transport using conditional retransmission and low-delay interleaving," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 6, pp. 558–565, Jun. 2002, doi: 10.1109/TCSVT.2002.800326.

[3] J. Border, M. Kojo, J. Griner, G. Montenegro, and Z. Shelby. *Performance Enhancing Proxies Intended to Mitigate Link-Related Degradations*, document RFC 3135, Internet Requests for Comments, Jun. 2001. [Online]. Available: https://tools.ietf.org/html/rfc3135

[4] Isode. (2020). *Measuring and Analysing STANAG 5066 F.12 IP Client*. Accessed: Jul. 1, 2020. [Online]. Available: https://www.isode.com/whitepapers/measuring-stanag5066-f12-ip-client.html

[5] J. W. Nieto and W. N. Furman, "Improved data rate robustness of U.S. MIL-STD-188–110C appendix D wideband HF waveforms," in *Proc. 12th IET Int. Conf. Ionospheric Radio Syst. Techn. (IRST)*, May 2012, pp. 1–5.

[6] T. Chowdhury and M. J. Alam, "Performance evaluation of TCP Vegas over TCP Reno and TCP NewReno over TCP Reno," *JOIV, Int. J. Informat. Visualizat.*, vol. 3, no. 3, pp. 275–282, Aug. 2019.

[7] A. D. Mello, E. Foo, J. Reid, and D. Jayalath, "On the enhancement of TCP/IP protocol for use in high frequency communication systems," in *Proc. Mil. Commun. Inf. Syst. Conf. (MilCIS)*, Nov. 2020, pp. 1–7.

[8] *Interoperability and Performance Standards for Data Modems*, Dept. Defense Interface Standard, Dec. 2017.

[9] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Nov. 2011.

[10] N. Ketkar, "Introduction to Keras," in *Deep learning with Python*. Berlin, Germany: Springer, 2017, pp. 97–111.

[11] M. Abadi, P. Barham, J. Chen, and Z. Chen, "Tensorflow: A system for large-scale machine learning," in *Proc. 12th USENIX Symp. Operating Syst. Design Implement.*, 2016, pp. 265–283.

[12] A. A. Elsayed and W. A. Yousef, "MATLAB vs. OpenCV: A comparative study of different machine learning algorithms," 2019, *arXiv:1905.01213*.

[13] T. Chen, M. Li, Y. Li, M. Lin, N. Wang, M. Wang, T. Xiao, B. Xu, C. Zhang, and Z. Zhang, "MXNet: A flexible and efficient machine learning library for heterogeneous distributed systems," 2015, *arXiv:1512.01274*.

[14] B. Lantz, *Machine Learning With R: Expert Techniques for Predictive Modeling*. Birmingham, U.K.: Packt, 2019.

[15] E. Loper and S. Bird, "NLTK: The natural language toolkit," 2002, *arXiv: cs/0205028*.

[16] G. Varoquaux, L. Buitinck, G. Louppe, O. Grisel, F. Pedregosa, and A. Muller, "Scikit-learn: Machine learning without learning the machinery," *GetMobile, Mobile Comput. Commun.*, vol. 19, no. 1, pp. 29–33, Jun. 2015.

[17] O. Kramer, *Scikit-Learn*. Cham, Switzerland: Springer, 2016, pp. 45–53.

[18] J. Hao and T. K. Ho, "Machine learning made easy: A review of scikit-learn package in Python programming language," *J. Educ. Behav. Statist.*, vol. 44, no. 3, pp. 348–361, Jun. 2019.

[19] E. Bisong, "More supervised machine learning techniques with scikit-learn," in *Building Machine Learning and Deep Learning Models on Google Cloud Platform*. Berlin, Germany: Springer, 2019, pp. 287–308.

[20] J. Weston and E. Koski, "High frequency radio network simulation using OMNeT++," 2015, *arXiv:1509.03176*.

[21] E. Koski and J. Weston, "Efficient high-fidelity simulation of HF communications systems and networks," in *Proc. IEEE Mil. Commun. Conf. (MILCOM)*, Oct. 2015, pp. 1460–1466.

[22] A. D. Mello, E. Foo, and J. Reid, "Characterizing TCP/IP for high frequency communication systems," in *Proc. Mil. Commun. Inf. Syst. Conf. (MilCIS)*, Nov. 2018, pp. 1–7.

[23] X.-D. Zhang, "Machine learning," in *A Matrix Algebra Approach to Artificial Intelligence*. Berlin, Germany: Springer, 2020, pp. 223–440.

[24] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science*, vol. 349, no. 6245, pp. 255–260, 2015.

[25] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of Machine Learning*. Cambridge, MA, USA: MIT Press, 2018.

[26] J. Jagannath, N. Polosky, A. Jagannath, F. Restuccia, and T. Melodia, "Machine learning for wireless communications in the Internet of Things: A comprehensive survey," *Ad Hoc Netw.*, vol. 93, Oct. 2019, Art. no. 101913.

[27] J. Wang, G. Ding, and H. Wang, "HF communications: Past, present, and future," *China Commun.*, vol. 15, no. 9, pp. 1–9, 2018.

[28] S. Spillane, K. H. Jung, K. Bowers, T. Peken, M. H. Marefat, and T. Bose, "Machine learning based MIMO equalizer for high frequency (HF) communications," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2020, pp. 1–8.

[29] W. Li, Y. Xu, Q. Guo, Y. Zhang, D. Liu, Y. Li, and W. Bai, "A Q-learning-based channel selection and data scheduling approach for high-frequency communications in jamming environment," in *Machine Learning and Intelligent Communications*, X. B. Zhai, B. Chen, and K. Zhu, Eds. Cham, Switzerland: Springer, 2019, pp. 145–160.

[30] D. Minovski, N. Ogren, C. Ahlund, and K. Mitra, "Throughput prediction using machine learning in LTE and 5G networks," *IEEE Trans. Mobile Comput.*, early access, Jul. 26, 2021, doi: 10.1109/TMC.2021.3099397.

[31] K. Harrison, "Machine learning for radio frequency interference flagging," Ph.D. dissertation, Faculty Eng. Built Environ., Dept. Elect. Eng., 2021. [Online]. Available: http://hdl.handle.net/11427/33777

[32] Y. Zhang, J. Wen, G. Yang, Z. He, and J. Wang, "Path loss prediction based on machine learning: Principle, method, and data expansion," *Appl. Sci.*, vol. 9, no. 9, p. 1908, May 2019. [Online]. Available: https://www.mdpi.com/2076-3417/9/9/1908

[33] A. Quemy, "Two-stage optimization for machine learning workflow," *Inf. Syst.*, vol. 92, Sep. 2020, Art. no. 101483.

[34] A. Varga, "OMNet++," in *Modeling and Tools for Network Simulation*. Berlin, Germany: Springer, 2010, pp. 35–59.

[35] D. Xu, Y. Shi, I. W. Tsang, Y.-S. Ong, C. Gong, and X. Shen, "Survey on multi-output learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 7, pp. 2409–2429, Jul. 2020.

[36] A. DeMARIS, "Explained variance in logistic regression: A Monte Carlo study of proposed measures," *Sociol. Methods Res.*, vol. 31, no. 1, pp. 27–74, Aug. 2002.

[37] V. Paxson, M. Allman, J. Chu, and M. Sargent. *Computing TCP's Retransmission Timer*, document 6298, Internet Requests for Comments, Jun. 2011.

ALVIN DMELLO is currently pursuing the Ph.D. degree with the Science and Engineering Faculty, Queensland University of Technology (QUT). He has more than 20 years of experience in the communications industry on various technologies. He worked with many large companies and is a Chartered Profession Engineer (C.P.Eng.) in Australia. His research interests include understanding the limits of communications technologies while integrating them with newer protocols. He has also been exploring cyber security capabilities in networks.

DHAMMIKA JAYALATH (Senior Member, IEEE) received the B.Sc. degree in electronics and telecommunications engineering from the University of Moratuwa, Sri Lanka, the M.Eng. degree in telecommunications from the Asian Institute of Technology, Thailand, and the Ph.D. degree in wireless communications from Monash University, Australia, in 2002. He was a fellow with the Australian National University and a Senior Researcher with the National ICT Australia. He has been an Academician with the Science and Engineering Faculty, Queensland University of Technology, since 2007. His research interests include the general areas of communications and signal processing. He has published significantly in these areas. His current research interests include applying machine learning principles in communication systems, physical layer security, and signal design for robust communications.

ERNEST FOO (Member, IEEE) is currently an Associate Professor. His research can be broadly grouped into the field of applied cyber security. National critical infrastructure systems, such as electricity substations and water treatment plants are becoming more connected and are susceptible to cyber-attacks. His research work looks for new ways to detect cyber attackers in critical infrastructure systems and prevent those attackers from being effective. The mechanisms he has employed in his research include machine learning and data mining as well as cryptographic protocols and network simulations. He has published over 90 refereed articles, including 20 journal articles.

JASON REID (Member, IEEE) received the Ph.D. degree in the area of trusted computing and smart card security. He is currently an Adjunct Associate Professor with the Science and Engineering Faculty, Queensland University of Technology, having over 20 years of experience in information security research and consultancy. His research interests include access control, authentication and identity management, and network security.