

# Effat University Repository

## Impact of Input Sequence Types on Healthcare Intrusion Prediction Models

Authors	Hafiz Mohd Yusof, Mohammad;Balfaqih, Mohammed;Munir Hayet Khan, Md;A Almohammedi, Akram;Balfagih, Zain
Citation	M. H. M. Yusof, M. Balfaqih, M. Munir Hayet Khan, A. A. Almohammedi and Z. Balfagih, "Impact of Input Sequence Types on Healthcare Intrusion Prediction Models," in IEEE Access, vol. 13, pp. 125897-125932, 2025, doi: 10.1109/ACCESS.2025.3584741. keywords: {Predictive models;Medical services;Long short term memory;Systematic literature review;Time series analysis;Transformers;Analytical models;Forecasting;Prediction algorithms;Accuracy;Intrusion prediction model;intrusion detection system (IDS);multivariate;univariate;data visualization;machine learning in cybersecurity;intrusion prediction in healthcare},
DOI	<a href="https://doi.org/10.1109/ACCESS.2025.3584741">https://doi.org/10.1109/ACCESS.2025.3584741</a>
Publisher	IEEE
Rights	CC0 1.0 Universal
Download date	2026-04-17 21:50:22
Item License	<a href="http://creativecommons.org/publicdomain/zero/1.0/">http://creativecommons.org/publicdomain/zero/1.0/</a>
Link to Item	<a href="https://repository.effatuniversity.edu.sa/handle/20.500.14131/2259">https://repository.effatuniversity.edu.sa/handle/20.500.14131/2259</a>

## TOPICAL REVIEW

# Impact of Input Sequence Types on Healthcare Intrusion Prediction Models

MOHAMMAD HAFIZ MOHD YUSOF<sup>1</sup>, MOHAMMED BALFAQIH<sup>2</sup>, (Senior Member, IEEE), MD MUNIR HAYET KHAN<sup>3</sup>, AKRAM A. ALMOHAMMEDI<sup>4</sup>, AND ZAIN BALFAGIH<sup>5</sup>

<sup>1</sup>College of Computing, Informatics, and Mathematics, Universiti Teknologi MARA, Tapah, Perak 35400, Malaysia

<sup>2</sup>Department of Computer and Network Engineering, University of Jeddah, Jeddah 23890, Saudi Arabia

<sup>3</sup>Faculty of Engineering and Quantity Surveying (FEQS), INTI International University, Nilai, Negeri Sembilan 71800, Malaysia

<sup>4</sup>Department of Electrical and Electronic Engineering, Abdullah Gül University, 38080 Kayseri, Türkiye

<sup>5</sup>Computer Science Department, Effat Energy and Technology Research Center, Effat College of Engineering, Effat University, Jeddah 34689, Saudi Arabia

Corresponding author: Mohammad Hafiz Mohd Yusof (hafizyusof@uitm.edu.my)

This work was supported by the Research Project under Grant FRGS/1/2021/ICT07/UITM/02/3.

**ABSTRACT** Prediction models are vital for sensing zero-day and even  $n$ -day cyberattacks, particularly in healthcare infrastructure. Most existing research focuses on developing classifiers also known as IDS to enhance detection and accuracy. However, predictive intrusion models for healthcare remain underexplored, with limited studies investigating the comparative performance of univariate and multivariate inputs against single-step and multi-step outputs in time series models. This study aims to address these gaps by evaluating the accuracy and error performance of selected predictive models across various input and output configurations. The methodology involves transforming input data sequences into univariate  $1*n$  and multivariate  $m*n$  formats, establishing single-step and multi-step splitting functions, and evaluating these configurations using the benchmark CIRA-CIC-DoHBrw-2020 dataset. Algorithms including Bidirectional LSTM, Stacked LSTM, Vanilla LSTM, Transformer Encoder-Decoder, Vector Output LSTM (GRU core), and CNN were applied, with results visualized to assess performance. The findings reveal that the Multivariate LSTM model, when trained on a sequence of multivariate inputs, demonstrates superior predictive performance, achieving low MAE error rates of 0.4% for single-step predictions and 0.1% for multi-step predictions. Additionally, **GRU** and **Transformer** models exhibit heightened sensitivity to specific input sequence configurations. In conclusion, our study demonstrates that Transformer Encoder-Decoder based prediction models exhibit exceptional prediction performance. This effectiveness is attributed to their ability to capture contextual and critical information from input sequences. These findings provide valuable insights for designing advanced intrusion prediction models, paving the way for improved prediction capabilities in future systems.

**INDEX TERMS** Intrusion prediction model, intrusion detection system (IDS), multivariate, univariate, data visualization, machine learning in cybersecurity, intrusion prediction in healthcare.

## I. INTRODUCTION

The growing reliance on predictive modeling in cybersecurity, particularly within healthcare infrastructure, has highlighted the importance of time series analysis for forecasting intrusion events before they occur. Despite its potential, a significant gap persists in understanding how input sequence configurations namely, univariate versus multivariate inputs—and output types—such as single-step

versus multi-step forecasting—affect the performance of time series-based intrusion prediction models. These aspects are especially critical in dynamic environments, like healthcare networks, where attack patterns evolve over time and decisions must be made proactively.

Existing research often overlooks this interplay, failing to provide a comparative understanding of how different configurations influence model performance. Specifically, there is limited investigation into how predictive accuracy and error metrics vary across architectures such as Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), Trans-

The associate editor coordinating the review of this manuscript and approving it for publication was Adamu Murtala Zungeru<sup>1</sup>.

former, and Convolutional Neural Networks (CNN), when subjected to various input-output configurations. Moreover, while predictive models have proven valuable in fields like energy, meteorology, and finance [1], [2], [3], [4], [5], [6], [7], [8], [9], their application in network intrusion prediction—especially in healthcare settings—remains underexplored. Additionally, Time-series analysis has become increasingly central to cybersecurity analytics, supporting a shift from traditional intrusion detection toward proactive attack prediction [10].

A comprehensive synthesis review [11] further underscores the scarcity of studies that examine the relationship between multivariate and univariate input types in the context of single-step and multi-step forecasting models. The authors emphasize the importance of reproducible and transparent methodologies, such as hybrid reviews, consensus conferences, and expert panels, for developing evidence-based insights [12]. These systematic approaches are essential to assess the performance and applicability of predictive models in real-world use cases, including the healthcare domain.

Furthermore, current research predominantly focuses on detection classifiers rather than forecasting methods, leaving a gap in predictive modeling for Advanced Persistent Threats (APTs) and zero-day attacks. Where prediction models have been used, LSTM and its variants dominate the landscape [2], [6], [7], yet there is a lack of studies comparing their performance with other deep learning techniques under different sequence configurations.

An additional limitation lies in the datasets typically used in intrusion prediction studies. Most publicly available datasets fail to capture the complexity of real-world healthcare traffic, leading to models that may perform well in controlled environments but poorly in operational settings. The CIRA-CIC-DoHBrw-2020 dataset [11], in contrast, closely resembles modern healthcare network behavior, making it particularly suited for evaluating intrusion prediction techniques under realistic conditions.

This study aims to fill these critical gaps by conducting a detailed comparative synthesis review and empirical analysis of widely used predictive models—including LSTM variants, GRU, Transformer, and CNN—under varying configurations of input sequences (univariate vs. multivariate) and forecast horizons (single-step vs. multi-step). Unlike previous studies, this work focuses not on proposing a new model but on analyzing the impact of configuration choices using a novel sequence-splitting function designed for time series prediction.

While grounded in the broader context of Intrusion Detection Systems (IDS), this study specifically addresses intrusion prediction in healthcare settings. Medical data typically follows a sequential structure of symptoms, diagnoses, treatments, and outcomes. Thus, accounting for these temporal dependencies is essential for building robust predictive systems. Our evaluation strategy offers insights not only into which models perform best, but also how to configure them

effectively for proactive, real-time protection of sensitive healthcare environments.

By addressing these underexplored dimensions, the findings of this research offer practical guidance for designing context-aware and accurate intrusion prediction systems, while also laying the groundwork for future research on optimizing predictive configurations in cyber defense. The contributions of this work are as follows:

- A systematic review and comprehensive analysis of prediction models across diverse domains, with a particular emphasis on their application in healthcare intrusion prediction systems.
- Development of diverse algorithms for configuring input sequences in predictive models implemented on LSTM variants, GRU, Transformer, and CNN architectures. It is also designed to train on the time-series CIRA-CIC-DoHBrw-2020 benchmark dataset where its traffic protocol implemented on healthcare network.
- Development of a novel pre-processing split-function algorithm to efficiently handle multidimensional time-series input sequences from the CIRA-CIC-DoHBrw-2020 benchmark dataset.
- Visualization of predictive performance of various LSTM variants, GRU, Transformer and CNN-based predictive models where its accuracy and error rate will be comparatively analyzed using various novel data splitting algorithms.
- Identification of critical shortcomings in existing public datasets, emphasizing the need for realistic, real-time datasets to reflect contemporary network threats in healthcare industry effectively.

The remainder of the article is structured as follows: Section II provides a systematic review of prediction models, identifies research gaps, and formulates a synthesized problem statement. Section III details the comparative evaluation methodology, including dataset preprocessing, restructuring, training and testing dataset division, model training and testing with various LSTM, GRU, Transformer and CNN-based prediction approaches, and performance evaluation to identify the most effective model. Section IV presents the findings of the models' performance, followed by the discussion and study limitations in Section V. Finally, Section VI concludes the article.

## II. SYSTEMATIC LITERATURE REVIEW

This section provides a comprehensive review of studies related to the impact of multivariate and univariate inputs on intrusion prediction models within the healthcare industry. Identifying relevant literature was challenging due to the specificity of the topic. A targeted search using keywords such as “review,” “multivariate,” “univariate,” “time series,” “intrusion prediction model,” and “healthcare” in the Web of Science (WoS) core collection yielded no direct results, as depicted in Figs. 1 and 2.

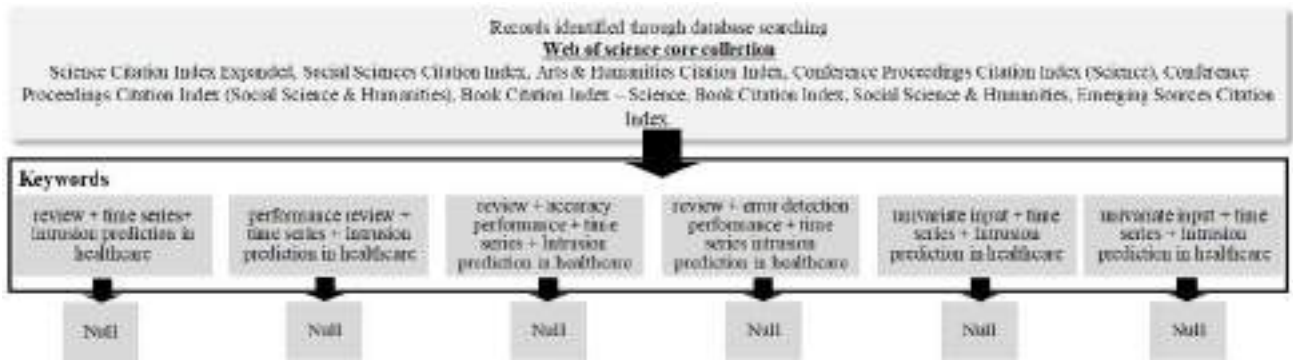


FIGURE 1. Systematic Literature Review (SLR) keywords on prediction model in healthcare.



FIGURE 2. No records were found to match the 2024 and 2025 filters on intrusion + prediction + models + healthcare + infrastructure.

Most research in the healthcare domain emphasizes real-time classification, detection, and prediction within healthcare monitoring systems. For instance, the study in [13] explores real-time time-series classification in Internet of Medical Things (IoMT) systems, proposing energy-efficient machine learning (ML) algorithms to reduce latency and improving online classification. Similarly, studies in [14], [15], and [16] focus on continuous, real-time eHealth remote monitoring systems for tracking physiological signs, elderly care, and heart monitoring, all aligned with specific health informatics standards and ML frameworks within healthcare industry.

On the other hand, literature searches within a broader domain of time-series prediction, IDS and their applications in healthcare identified 14 relevant studies, as shown in Fig. 3.

Additionally, Fig. 4 highlights a broader scope of generic prediction models in healthcare, which were based on 150 papers published between 2020 and 2024. The remaining search results covered diverse topics, including healthcare prediction models and applications in transportation, electricity, meteorology, and time-series forecasting, as already depicted in Fig. 3. However, a consistent gap persists, as most

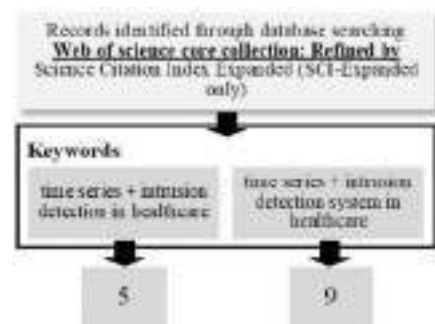


FIGURE 3. Related Systematic Literature Review (SLR) keywords.

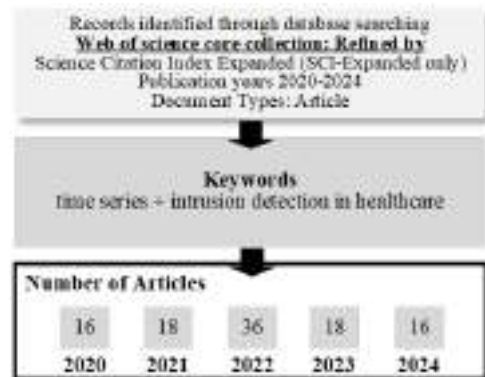


FIGURE 4. Extended related Systematic Literature Review (SLR) keywords.

existing works focus heavily on classifier development, leaving intrusion prediction models underexplored.

Finally, Table 1 summarizes key features from past studies on intrusion prediction models, highlighting critical areas for further investigation. To delve deeper into this topic, the literature review is divided into four sub-sections. Section A examines prediction models across various domains, while Section B focuses on generic prediction models in healthcare. Section C identifies research gaps in existing studies, and Section D synthesizes these findings into an extended problem statement, offering a clear direction for addressing identified gaps.

TABLE 1. Summary of related intrusion prediction studies.

Author	Input		Synthesized Prediction Models										Output		Main Contribution					Evaluation					IDS	Not IDS	Dataset	Year	
	Multivariate	Univariate	CNN	LSTM(RNN)	ARIMA	MLP	Attention	Harmonic	SVM	Genetic Algorithm	Gaussian Naïve	Intuitionistic Fuzzy Sets	Wavelet	Multistep	Single step	Comparisons	Prediction Model	Classification Model/Detection Model	MAE	RMSE	Statistical technique	Simulator (ie OMNET)	Accuracy	ROC					
G. Zheng et al. [3]	✓		✓											✓	✓	✓										✓		Penns-Bay California Transportation Agencies	2023
Y. Sun et al. [4]	✓			✓										✓	✓											✓		KDD-99	2022
J. Hong et al. [5]		✓			✓																					✓		Engle's ARCH (Mathlab)	2023
A. A. Garibo-Morante et al. [6]	✓	✓							✓					✓	✓						✓					✓		Electrical power and wind solar profile dataset.	2022
R. Chuentaw et al. [7]	✓							✓																		✓		Beijing PM2.5 meteorological data	2018
P. Mansourian et al. [8]	✓			✓						✓					✓											✓		Car Hacking Dataset	2023
Y. Yu et al. [9]	✓			✓											✓							✓				✓		OMNET generated	2022
S. M., et al. [11]	✓																									✓		Stock, Solar energy, Whether	2024
N. Uremović et al. [85]	✓			✓											✓											✓		Environment Management System Dataset	2022
J. Xie et al. [22]	✓										✓				✓											✓		NSL-KDD ,KDD-99,UNSW-NB15	2022
M. M. Althobaiti et al. [19]	✓			✓											✓											✓		NSL-KDD-2015, CIC-IDS2017	2021
A.A. Malibari et al. [20]	✓			✓											✓											✓		CIC-IDS2017	2022
H. Salemi, et al. [18]		✓		✓											✓											✓		DARPA 98	2022
L. Zhang, et al. [17]	✓																									✓		Google Flu Dataset (Anomaly Injection)	2021

## A. INTRUSION PREDICTION MODELS IN VARIOUS DOMAINS

This section provides an extensive discussion on prediction models across various domains, highlighting their methodologies, applications, and performance in specific contexts.

The authors in [3] discovered three main issues in multivariate time series forecasting and two of them are related to the neural architecture; 1) a discrete neural block to encode patterns lead to discontinued state trajectories hence higher forecasting errors 2) redundant parameters lead to higher computational overheads. They proposed a Convolutional End model by first abstracting the multivariate time series events into dynamic graphs and subsequently applying differential equated neural network to complement the missing topologies. The evaluation is applied on single and multi-step forecasting. The multivariate series input in this research is represented by  $X \in R^{N \times D \times S}$  where  $N$ ,  $D$  and  $S$  are variables, dimensions and time steps respectively. Given a sequence of  $T$  observations  $X_{t+1:t+T} \in R^{N \times D \times T}$  the objective is to learn a new spatial temporal (spatiotemporal) data from the proposed model for which the function is denoted by  $f(\cdot) : R^{N \times D \times T} \rightarrow R^{N \times D}$ . Finally, the output is represented by  $H_{out} = f(X_{t+1:t+T})$ . The model is tested against amongst others Pems-Bay a dataset that is provided by California Transportation Agencies.

The study in [17] introduces AURORA, a model for detecting anomalies in time series with seasonal patterns and long-term trends. It combines frequency- and trend-based techniques using a Ramanujan periodic dictionary and spline functions to model normal behavior. This hybrid approach enhances anomaly detection performance, and while primarily applied to domains like finance and security, it demonstrates the potential of leveraging periodicity and memory-based features for time-series anomaly detection. Although not directly aligned with intrusion prediction, it offers relevant insight into hybrid temporal modeling strategies.

The authors in [12] analyzed significant differences between various prediction model that applies LSTM and transformation law on multivariate data. Presumably, LSTM is said to be insufficient in handling various degrees of attention in multivariate data. Hence a prediction model that is based on Multi-Layer Perceptron (MLP) combined with LSTM added to a feed forward attention mechanism is proposed. The input layer is denoted by  $x_t^m = w \cdot x_t^n + b$  where  $w \cdot x_t^n$  is the product of  $N$ -dimensional input vector and its weight added to the bias. Feed forward attention is designed with complex weighted sum. The calculation process is equated by the following formula;  $a(x_t^m)$  where  $a$  is the entire dimension of the attention weights. However, the conclusion as to add the additional layer to maintain its attention level is still hypothetical as it is based on literature triangulation. This is clearly stated in the paper at the last paragraph of the introduction section. Hence an empirical review as to assist the decision is required.

Alternatively, author in [18] designed a proactive model known as Lyapunov Exponent Analysis and Echo State Network (LEAESN) to predict DDoS attacks in critical multimedia-based e-health infrastructure. Due to the reactive nature of existing detection systems, which only able to operate after an attack, this prediction model is introduced. Lyapunov Exponents (LEs) measure the trajectories rate in a dynamical system for which it provides insight into the system's chaotic behaviour. A technique called Exponential Smoothing is used to predict the future traffic which is widely used in time series forecasting analysis. It is mainly deployed in sales forecasting, inventory management, economic forecasting and weather prediction. It uses smoothing factor,  $\alpha$ , which indicates the weight of the most current observation. Higher the,  $\alpha$ , value constituted a significant event. However, it requires careful selection of this smoothing parameter. Conversely, Echo State Networks (ESNs) are a type of recurrent neural network (RNN) designed to efficiently model temporal sequences and dynamical systems. Prediction error is measured based on the difference between the time series prediction and the observed traffic of the network. The combination of LEAESN can enhance the understanding and modeling of complex dynamical systems. The prediction error rate MAE of this prediction model was evaluated against the DARPA98 dataset.

In [5], the authors proposed a graph framework with reduced significant parameters for multivariate prediction model. The generated graph topology is assumed to be time-invariant as to make it possible of generating average traffic behavior graph from  $k$ NN with Gaussian Kernel weight. In this study, an undirected graph is represented by  $G = (V, E, W)$ , where  $V$  is a set of vertex (node) with  $N$  vertices,  $E$  is the set of edge and  $W$  is adjacency matrix. If vertices  $i$  and  $j$  are connected  $E_{ij}$  belongs to set  $E$  and  $W_{ij}$  is the set of edge weight in the adjacency matrix. Hence the graph signal,  $X$  is the set of values on vertices that allows a mapping from the vertex,  $V$  to a real number,  $R$ , which denoted by  $X : V \rightarrow R^N$ . For instance  $X_t \in R^N$  is a graph signal at time  $t$  with  $N$  vertices. Finally, the prediction model that is introduced in this study is based on Autoregressive Moving Average (ARMA). The Mean Absolute Error (MAE) and Root Normalize Mean Square Error (RNMSE) evaluation performance are tested against synthetic and real dataset. The synthetic dataset is claimed to be effective at multistep prediction from 1 to 6 steps and a real dataset is a univariate time series from Engle's ARCH.

In [6], the authors highlighted a univariate and multivariate harmonic decomposition method to design time series prediction model. In this study, the serial function is defined as  $s(t) = \frac{a_0}{2} + \sum_{n=1}^N A_n \cos(w_n t + \theta_n)$  where  $N$  is between 1 and infinity,  $1 \geq N \geq \infty$ ,  $A_n$  is the amplitude of the cos graph,  $w_n = 2\pi \cdot n/T$  is the frequency angular and  $\theta_n$  is the phase. This series describes arbitrary signal. The frequency values will define the state variables hence describing the time varying signal. When the signal converges from the harmonics state, the model can be used to predict the output.

It is also mentioned that to create a more accurate model, statistical techniques should be incorporated in the analysis. This statistical analysis will determine which variables should be taken in and which variables should be opted out. This model is tested against electrical power and wind solar profile dataset.

The authors in [7] has analyzed significant differences by comparing between multivariate and univariate time series forecasting model precisions. The forecasting model is based on Support Vector Machine (SVM) with Genetic Algorithm (GA) optimizer. It was tested against Beijing PM2.5 meteorological data which is publicly available at UC Irvine (UCI) Machine Learning Repository. It is demonstrated by a simple experimental tool that the three main dataset with univariate inputs have top four (4) coefficient of determination,  $R^2$  values at lag time,  $t - 1$ ,  $t - 2$ ,  $t - 3$  and at  $t - 4$ . On a flip side, the  $R^2$  of multivariate inputs has lower values as compared to the univariate. The study concludes that the univariate forecasting model has lower Root Mean Square Error (RMSE) and Mean Absolute Percentage Error (MAPE) as compared to the multivariate.

Similarly, authors in [8] introduced a prediction model for detecting anomalies in Connected Autonomous Vehicles (CAV) network. They applied LSTM and Convolutional LSTM (ConvLSTM). The prediction engines is through the Gaussian Naïve Bayes method. The proposed prediction model is then evaluated against Car Hacking Dataset, a publicly available real-word dataset. The evaluation parameters are accuracy rate and the F-scores. The results are also compared with other one class classifier model like Isolation Forest and Auto-encoder.

In [19], authors proposed a novel AI based detection model for industrial Cyber-Physical Systems (CPS), encompassing several application areas, including healthcare. Industrial CPS environment is multidimensional and complex that integrates industrial Internet of Things (IoT) components. They leverage on GRU as a detection model to identify the presence of intrusions within the Industrial Cyber-Physical Systems. The GRU is a type of RNN architecture featuring a simple gate structure. The model is validated against CICIDS-2017 and NSL-KDD 2015 dataset. The spatial dimension,  $D$  of the input sequence is to be resolved by a defining optimization mapping function,  $f(x)$ . Every solution will be constrained to a 1D binary vector.

A study in [20] introduces a metaheuristic deep learning approach towards IDS in smart environments. Smart environments encompassing smart city, transportation and healthcare which integrates internet and ubiquitous computing into its environment. This approach applies Z-score normalization technique on data pre-processing phase. For classification and detection of intrusions in the smart environment, a Deep Wavelet Neural Network (DWNN) model is employed. DWNN is a hybrid model that blends the deep neural networks architecture with the strength of wavelet transforms. Wavelet transform components consist of Time-frequency localization and the feature extraction. In time-frequency

localization, the wavelet transformations excel at scrutinising both frequency and time domains simultaneously. This feature is crucial for depicting multi-scale information and transient features. Two dimensional,  $x_{1,1}, \dots, x_{1,j}$  data of CIC-IDS-2017 has been deployed as the training and testing input dataset. This model yields a single binary output [0,1].

On the other hand, study in [21] proposed a multidimensional time series-based model for early detection of illness in the elderly population. The proposed novel model detects similarities between two multi-attribute time series input data based on a well-known discriminative model Smith-Waterman (SW) from bioinformatics algorithm. This will establish a clear method for ascertaining data aggregation, for which previously was a source of uncertainty and frequent challenges in sensor data processing. The multidimensional time series dataset,  $S$  is a set of  $n$  couples  $(s_i t_i)$  where  $S = \{s_1 t_1, \dots, s_n t_n\}$ . Each couple  $(s_i t_i)$  has two components: a sensor,  $n$  and a timestamp  $t_i$ . This model was evaluated against the X10 protocol sensor dataset from TigerPlace, Columbia, USA. It had achieved an F-measure score of 0.75 for prediction.

Authors in [4] highlight the issue in existing intrusion detection models which is discounting time-series features of network traffic. Which in turn increases the memory footprint as the features grow and leads to low accuracy in detection time. In order to avoid such a problem, they proposed an integrated intrusion detection model which is based on the Informer model. It starts by obtaining a multitude global input timestamps, hence a multivariate input,  $X_t$  where  $t$  is the sequence of input  $x$ . This Informer module uses a fixed Position Encoding (PE) denoted by  $PE(pos, 2i) : \sin(pos/2L_x)^{2i/dm}$  where  $pos$  is data in the  $X_t$  position in the sequence,  $i$  is the dimension of the sequence,  $L_x$  is the length of sequence  $X_t$ ,  $dm$  is the dimension entered for the model and finally  $2i$  represents an even dimension. Formula for odd dimension is denoted by  $PE(pos, 2i + 1) : \cos(pos/2L_x)^{2i/dm}$ . Finally the multivariate input sequence is denoted by  $x_{feed[i]}^t = \alpha u_i^t + PE_{(L_x * (t-1) + i)} + \sum_p [SE_{(L_x * (t-1) + i)}]_p$  where  $\alpha$  is a constant that is used to balance  $u_i^t$ ,  $PE$  and  $PE$ . The core LSTM self-attention mechanism is formulating from the Kullback–Leibler (KL) Divergence method that carries out multidimensional analysis of the system and makes use all the important features of the data. The model is tested against the KDD-99 dataset.

In [9], the authors proposed an intrusion detection model that integrates deep learning technique for time series Vehicular Ad Hoc Networks (VANET) traffic classification. To capture the underlying pattern of traffic parameters changing over time, LSTM incident classifier is designed. The proposed IDS is tested against extensive simulation. It has multiple variables (multivariate) like vehicle ID,  $V_1, \dots, V_n$ , speed,  $V_{1,T}, \dots, V_{n,T}$ , acceleration,  $a_{1,T}, \dots, a_{n,T}$ , and, position,  $Pos_{1,T}, \dots, Pos_{n,T}$ . Each vehicle broadcasts a beacon message,  $t_{beacon}$ , and will keep their neighboring vehicle beacon message for the latest observation time,  $t_{observe}$  in a table called Current Neighbor List (CNL). If  $V_1$  senses accident

at  $t_0$ , it will broadcast emergency message to all neighbors. Then, say  $V_0$  receive the emergency message, it will update its CNL table from time 0 and the previous timestamp  $[t_0 - 1, t_0]$ , or the latest observed event  $[t_0 - t_{observe}, t_0]$ . To validate the emergency message,  $V_0$  runs the IDS algorithm by comparing other vehicles' observed event collected by at  $t_0$ . It has time series feature vector (multivariate input series) denoted as  $X_{0,T} = [v_{w,T}, a_{w,T}, d_{w,T}, \bar{v}_{w,T}, \bar{a}_{w,T}, \bar{d}_{w,T}]$ . All these features represent different traffic conditions. A few statistical methods were applied such as average speed differences, sample mean, standard deviation to detect the outliers in a univariate dataset. Then further statistical test like Grubb's test was applied to get absolute deviation and to unearth what is the most to be outlier. The performance was evaluated through simulation setup like OMNET++.

In [22], the authors highlighted Intuitionistic Fuzzy Sets (IFS) detection model. NSL-KDD, KDD CUP99 and UNSW-NB15 datasets were used to test the model. The dataset will be digitalized, normalized and balanced. Chi-square test is used to reflect the degree of deviation on features that give great influences on the network state. A multivariate variable is defined by a universe of discourse,  $U = \{x_1, x_2, \dots, x_n\}$ . An IFS  $A$  in  $U$  is defined;  $A = \{[x, \mu_A(x), \nu_A(x)] | x \in U\}$ , where  $\mu_A(x) \in [0, 1]$  and  $\nu_A(x) \in [0, 1]$  are the degrees of membership and nonmembership of  $A$  that belongs to element  $U$ , where these degrees are between 0 and 1;  $0 \leq \mu_A(x) + \nu_A(x) \leq 1$ . This degree of membership and nonmembership is determined by The degree of hesitancy,  $\pi_A(x)$  of element  $U$  is denoted by  $1 - \mu_A(x) - \nu_A(x)$ . Assume there is another IFS  $B$ . The distance between IFS  $A$  and  $B$  is denoted by  $d(A, B)$ . If  $d(A, B) = 0$  if and only if  $A = B$ . This study also introduced dynamic IFS which time is taken into consideration. Each sample (each row) generates the multiattribute IFS (multivariate) and each attribute generates the degree of membership. The degree of membership is determined by taking the difference between the mean of all values belonging to type normal or abnormal,  $\bar{x}_j$  and the  $i$ -th sample of the  $j$ -th feature  $x_{ij}$ ;  $|x_{ij} - \bar{x}_j|$ . This is not a prediction model instead a univariate detection (classification) model.

## B. PREDICTION MODELS IN GENERIC HEALTHCARE

This section discusses prediction models in the generic healthcare domain, emphasizing their methodologies, applications, and limitations.

Study in [23] proposed a predictive analytics model based on Bayesian Network that is used to improve prediction analysis on network infrastructure in one of the largest healthcare providers in Malaysia. The ground-truth dataset from the production traffic is used to promote realism in real cyberattacks use case. Our study emulates the handling of unsupervised dataset since production network is unstructured and has no label. This type of model is proposed considering the polymorphic and stealthy nature of malware attacks.

Meanwhile, study in [24] proposed a Vulnerability Assessment (VA) test method based on Natural Lan-

guage Processing (NLP). This method leverages unstructured security-related text to predict and manage cyber threats. The proposed method has been tailored for the healthcare ecosystem. The use of NLP and text-based dataset makes this study different and novel. However, security-related text is often plagued by data ambiguities, noise, and inconsistencies arising from diverse writing styles and technical jargon in incident reports [25]. Therefore, standard network protocol datasets remain reliable and consistent in validating an intrusion prediction model.

On the other hand, the study in [26] provides an in-depth analysis of cyber threats and risks associated with different critical infrastructures including healthcare sectors. They investigate the staggering benefits of AI mechanisms in confronting cyberattacks. Furthermore, they proposed an AI-based secure data exchange framework for smart grid Critical Infrastructure (CI) (i.e., power consumption, energy readings, and network data) from malicious adversaries. However, AI tools can make research models less transparent which leads to lack of transparency and interpretability in many AI models, particularly complex ones like deep learning models. It means that while these models can make accurate predictions or decisions, we often cannot fully understand the internal processes and reasoning behind these outputs. This lack of understanding can lead to several issues like lack of trust, difficulty in debugging and ethical concerns.

Consequently, this paragraph swiftly discusses some related papers on prediction healthcare infrastructure. Paper [27] delves into the prediction of healthcare infrastructure demands within the context of intelligent healthcare networks. The study aims to improve the reliability and efficiency of real-time prediction models influenced by interaction flows and network topology. Next, paper [28] provides a comprehensive survey of AI technologies in crafting IoMT systems which is tailored for healthcare applications. Subsequently, paper [29] applies Fuzzy Decision Tree (FDT) and GA to predict electrical load and price for hospital's electricity decision making process. Whilst, paper [30] proposes an IoT-based infrastructure that leverages edge computing for solar energy prediction, a technique that may have potential applications in the healthcare domain. Although many of the technologies employed in these studies are legacy and lack transparency, they share a common objective which is to improve the performance of prediction models in healthcare infrastructure by analyzing network behavior, conducting comprehensive surveys, and utilizing mathematical modeling techniques.

However, in addition to the abovementioned traditional methods, a more recent study [31] introduced a quantum-inspired heuristic algorithm to enhance the security of healthcare prediction models. The Quantum-inspired metaheuristic algorithms were developed by integrating Quantum Computing (QC) concepts into metaheuristic algorithms. QC is a new field of research that includes elements from mathematics, physics, and computing. Qubits are extremely sensitive to environmental factors like temperature,

electromagnetic fields, and vibrations. Maintaining their stability and scaling them up to a large number of qubits is a major challenge [32]. On the other hand, developing efficient quantum algorithms that can harness the power of quantum computers is a complex task [33] and ultimately the issue of decoherence which is the loss of quantum coherence, the ability of a quantum system to maintain a superposition of states [34].

For the remaining studies, paper in [35] and [36] performed systematic review and comprehensive comparison on various deep learning models, including ANN, CNN, RNN, LSTM, and GRU for risk and heart failure prediction model. Building upon the rigorous systematic review and comparative analysis of deep learning models for risk and heart failure prediction presented in [35] and [36], we adopt a similar methodological framework for our investigation. Subsequent papers [35], [36], [37], [38], [39], [40], [41], [42], [43], [44], [45], [46], [47], [48], [49], [50], [51], [52], [53], [54], [55], [56], [57], [58], [59], [60], [61], [62], [63], [64], [65] present models that can predict various aspects of clinical infrastructure, making them applicable to a wide range of healthcare settings. Various prediction model to improve risk predictions, determine asthma level, disease's code, heart failure, dynamic resilience assessment of Healthcare Infrastructure (HI) operations, healthcare coverage, polygenic predictions, feto-maternal health assessment applications, kidney disease predictions, post-treatment predictions, clinical sepsis predictions, clinical decision support systems, contralateral breast cancer predictions, Parkinson's disease predictions, rare disease predictions, mortality rate predictions, cancer referral decisions prediction model and many more. They leverage on AI-enabled models, generic machine learning models, Ensemble Neural models, LightBGM a library-powered model, AdaBoost-powered model, MATLAB prediction algorithm library and various legacy statically and probabilistically approaches like ARIMA model, probabilistic Markov model and logistic regression model.

While earlier works primarily focused on classification-based anomalies detection using static feature sets, the studies in [66], [67], [68], [69], [70], [71], [72], [73], [74], [75], [76], [77], [78], [79], [80], [81], and [82] take a different direction, exploring predictive modeling and temporal pattern recognition techniques. A substantial number of predictive modeling techniques have been utilized to forecast public health trends, especially on COVID-19 cases. Well-known traditional statistical and probabilistic models including logistic regression, lasso regression, Bayesian methods, random forests, and simple frequentist statistical techniques were applied in these studies. Some model that are commonly known in public health trajectory model like Susceptible-Exposed-Infected-Removed model and newly invented prediction model like Facebook Prophet combining mathematical models, growth functions, and additive models, have also been adopted. Additionally, a range of techniques, including AI-driven approaches like deep learning and machine learning, as well as qualitative hierarchical

models, were utilized. These prediction models are used to predict healthcare utilization, hospital diagnose and management systems, COVID-19 confirmed cases, patient health, predictive model of anti-hyperglycemic medication cessation, prediction of antifreeze proteins (AFPs), estimate the social isolation impact on COVID-19 spreading and nomogram prediction model.

Many of the abovementioned clinical infrastructure and public health domains implement open-source predictive analytics platforms which apply AI and machine-learning common techniques to a dataset. As previously stated, many AI models are just a black box and often less transparent for which its internal processes are not fully understood. This leads to lack of transparency and interpretability. Furthermore, various legacy statically and probabilistically approaches were utilized. These models often struggle to capture complex, nonlinear relationships between variables. As data complexity increases, traditional models may struggle to keep pace. However, implementation of Ensemble Neural Networks might represent the current state-of-the-art, which involves the training of multiple neural networks and the subsequent combination of their predictions using specialized fusion techniques. Ensemble neural models often incorporate the Random Forest algorithm as a core component. Please note that while Random Forest is a common choice, other algorithms like Gradient Boosting Machines (GBM) and Extreme Gradient Boosting (XGBoost) are also frequently used in ensemble neural models. While Random Forest itself is not a neural network, its integration within ensemble models enhances their predictive power and robustness [83].

### C. SUMMARY OF RELATED STUDIES

A comprehensive review of related studies highlights that most efforts have focused on IDS classifiers. These models predominantly leverage advanced deep learning techniques such as LSTM, its generative variants like GRU and Transformer, and CNN. In contrast, limited research has been conducted on univariate and multivariate prediction or forecasting models, particularly within the field of intrusion prediction. Prediction modeling has traditionally been applied to domains such as electrical power systems, environmental management, meteorology, eHealth monitoring, weather forecasting, and stock market analysis. This gap underscores the unique focus of this research, which emphasizes comparative performance analysis of various intrusion prediction models across diverse input and output configurations.

Some studies have moved beyond conventional time-series analysis using LSTM, its generative variants like GRU, Transformer and CNN by incorporating unsupervised learning techniques. Examples include Autoregressive Integrated Moving Average (ARIMA), harmonic modeling, SVM, genetic algorithms, and fuzzy and Gaussian distribution methods. These approaches are used to analyze regressive patterns in historical data and predict future trajectories.

However, in the context of IDS and intrusion prediction, LSTM variants and CNN remain the dominant choices for predicting future intrusion events.

Intrusion prediction modeling, coupled with time-series analysis, plays a critical role in APT scenarios, enabling proactive, “just-in-time” countermeasures against cyberattacks. Current reactive technologies only address threats after they have caused damage, underscoring the importance of predictive models in preempting cyber incidents before they occur.

A review of datasets used in related studies reveals reliance on both IDS-specific and general-purpose datasets for accuracy and error evaluation. Commonly cited datasets such as KDD-CUP99, NSL-KDD, and UNSW-NB15 have notable limitations, including redundancy and insufficient representation of modern network threats. In contrast, the CIRA-CIC-DoHBrw-2020 dataset emerges as a more realistic option, reflecting contemporary, low-footprint attacks characteristic of stealthy, real-world cyber threats [1]. This dataset’s balance between normal and attack traffic makes it suitable for this study’s evaluation phase.

Additionally, previous studies categorize input sets as multivariate (multi-attribute) or univariate. Multivariate inputs consist of multiple observations for a single timestamp, represented as a multidimensional matrix ( $1 * n$ , where  $n$ : 1 is a real number). Conversely, univariate inputs involve a single observation per timestamp ( $1 * 1$ ). These input configurations result in predicted outputs classified as either multistep or single-step outputs, forming the basis for the comparative evaluation in this research.

The summary of related works reveals several critical gaps in the field of intrusion prediction. Firstly, most prior studies focus predominantly on classifier development rather than comparative analyses of predictive performance across various intrusion prediction models. Specifically, limited research explores the impact of multivariate input configurations tested against multistep forecasting or single-step outputs, or the effect of univariate inputs evaluated in similar scenarios. Few studies have comprehensively examined the interaction between multivariate and univariate input sequences and their performance in producing multistep or single-step outputs using time-series intrusion prediction models.

Secondly, while IDS are typically designed for classification or detection, predictive capabilities within intrusion system remain underexplored. Although the statement here looks like an oxymoron since IDS itself is a classification or detection system and not prediction, so why bother to discuss about prediction? The answer to this question is dual-edged; some modules in intrusion system requires predictive analysis capability which eventually makes them relevant to anticipate zero day or now even  $n$ -day attack. Which mean the attack (event) can be anticipated several days before zero day (or even several days after the events). Nevertheless, the choice of predictive model in intrusion studies are still limited to LSTM and its generative variants like GRU and Transformer

and CNN based model. Transformer is a layered architecture with stacks of LSTM Encoder-Decoder LSTM model herein referred to as Transformer Encoder-Decoder. Hence the focus of this study is to comparatively measure accuracy and error performance of various selected types predictive model over the choices of input and output configurations.

Thirdly, the datasets used in IDS research often fail to represent realistic network conditions. Many public datasets suffer from deficiencies such as redundant records, repetitive features, and an inability to capture the complexities of modern, low-footprint, stealthy cyberattacks [1]. In contrast, the CIRA-CIC-DoHBrw-2020 dataset has been identified as a benchmark that closely mirrors real-world network traffic [1]. This DNS-over-HTTPS dataset, introduced in 2018 under RFC8484, simulates network traffic over this protocol and has become a preferred dataset for intrusion research due to its realistic characteristics. Furthermore, HTTP-based applications have been implemented in digital healthcare and electronic health record (EHR) systems [84].

In conclusion, these identified issues highlight significant opportunities for advancing intrusion detection and prediction research, particularly through comprehensive performance comparisons of various prediction techniques and addressing the limitations of existing datasets. This study aims to bridge these gaps by providing a systematic and robust evaluation of predictive models within this context.

### III. COMPARATIVE EVALUATION METHODOLOGY

This section outlines the research methodology employed to complete the study, comprising four essential stages, as illustrated in Fig. 5. In the first stage, the dataset is preprocessed and restructured into a new data array. In the second stage, the data is split into training and testing datasets using a novel splitting function algorithm. The third stage involves training and testing the preprocessed datasets using various LSTM- and CNN-based prediction models in IDS, including Bidirectional LSTM, Stacked LSTM, Vanilla LSTM, Transformer Encoder-Decoder, Vector Output LSTM, GRU, and CNN. Finally, the fourth stage focuses on evaluating, comparing, and visualizing the accuracy and error performance of these models to identify the most effective results and input configurations. This systematic approach ensures a comprehensive analysis of the predictive capabilities of the models.

#### A. STAGE 1: DATA PREPROCESSING

##### 1) DATA ACQUISITIONS

Our study employs the CIRA-CIC-DoHBrw-2020 dataset, sourced from the Canadian Institute for Cybersecurity (CIC) [86]. Fig. 6 illustrates the network architecture used to intercept DNS over HTTPS (DoH) traffic, leading to the dataset’s creation. Traffic data for the dataset was generated in two layers. The first layer involved normal web browsing activities, producing benign DoH and non-DoH HTTPS traffic, captured by web servers. In the second layer, malicious DoH traffic was generated using various tools and captured by

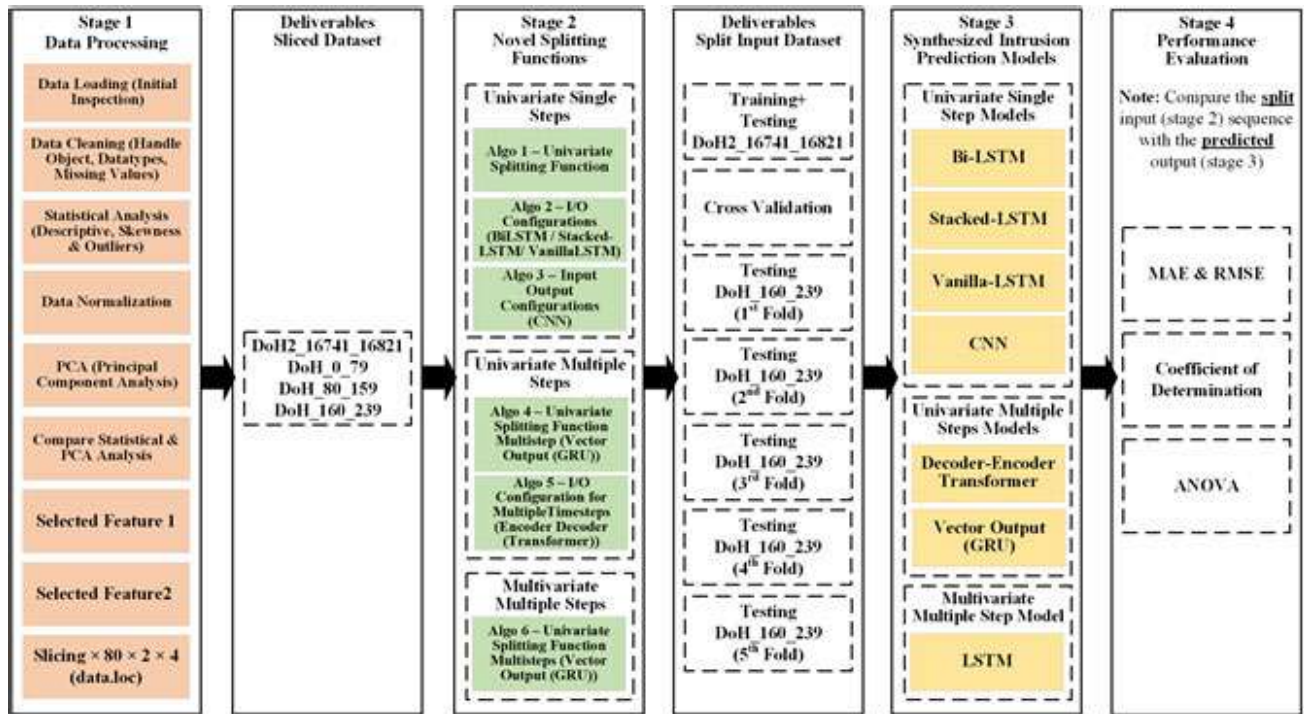


FIGURE 5. Comparative evaluation methodology.

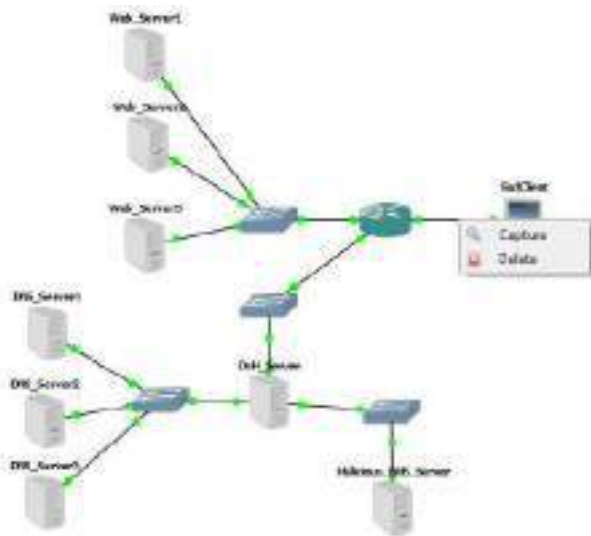


FIGURE 6. Network topology that is used to capture the DoH traffic.

malicious DNS and DoH servers. To simulate traffic, public DoH resolvers such as Adguard, Cloudflare, Google, and Quad9 were utilized, with Firefox connected to GeckoDriver and Chrome to ChromeDriver. The traffic was captured using tcpdump, and a Python script leveraging Scapy was developed for DoH traffic flow generation and analysis. A tool named “DoH Data Collector” simulated DoH tunneling incidents.

The generated data included 48,952 Kbytes of benign traffic and 219,458 Kbytes of malicious traffic, with transmission rates varying between 100bps and 1100bps. The dataset includes destination IPs for browsing public DoH servers and source IPs from clients accessing websites via Google Chrome and Mozilla Firefox. Flow-based and meta-data features were extracted from the packets, treating raw data and flow information equally without distinction between process flow labels and raw labels.

Several research have focused on integrating HTTP protocols into healthcare infrastructures. For instance, the research in [84] emphasizes the importance of robust networking infrastructures for enabling data access through HTTP-based applications in digital healthcare and EHR systems. The study utilized 45,000 test FHIR patient resources, which were downloaded and uploaded in bundles of 20, 50, 100, and 200 resources, resulting in 2251, 901, 451, and 226 HTTP GET and POST requests, respectively. Further, the study in [85] implemented a hybrid security solution to safeguard the collection and management of personal health data using the Spring Framework (SF), Services for Sensitive Data (TSD) as a platform, and HTTP security methods.

2) DATA LOADING

Load the dataset into a Google Collab Python 3 Compute Engine which is equipped with TPU v2-8, a free tier 4 TPU v2 chips configuration, providing 8 Tensor Core. It also has the total High Bandwidth Memory (HBM) of 64 GB (8 GB per core). The system can achieve around 180 teraflops of

floating-point performance. Two vCPUs, is the total number of virtual CPUs (vCPUs). The RAM capacity is 13 GB with 107 GB disk storage and a free tier NVIDIA Tesla K80, T4, or P100 GPU. `data.info()` is utilized to get a summary of the dataset with the number of rows is 167,517 (index 0 to 167,517), number of columns is 35 (index 0 to 34), data types of Boolean is 1, Float64 is 26, Int64 is 5 and Object is 3 which includes 'SourceIP', 'DestinationIP' and 'Timestamp'. The CIRA-CIC-DoHBrw-2020 dataset is distinguished by the presence of 'SourceIP' and 'DestinationIP' addresses (Layer 3 information), a feature not found in other benchmark datasets, thereby enhancing its fidelity to real-world network traffic. Then `data.head()` is applied to examine the first few rows of the dataset facilitating a better visualization and understanding of the features.

### 3) DATA CLEANING

Data cleaning process is to prepare the data for analysis by handling missing values and non-numerical features. This phase encompasses several key steps, namely the handling Object Datatypes, the treatment of Missing Values and Data Type Conversion. These phases are explained as follows:

- **Handling Object Datatypes:** Identifying Object columns from the previous phase ('SourceIP', 'DestinationIP', 'Timestamp'). Since PCA analysis requires numerical calculations, these features will be dropped.
- **Handling Missing Value:** The `data.isnullsum` command is employed to identify the presence and count of missing values in the dataset. Missing values in 'ResponseTimeTimeSkewFromMedian' and 'ResponseTimeTimeMedian' are imputed using the NetworkX library in Python. This strategy leverages the network relationships or graph property of these features.
- **Handling Data Type Conversion:** This step is to ensure all relevant columns for numerical analysis (especially for PCA) are in numerical formats (int or float).

### 4) STATISTICAL ANALYSIS

Statistical analysis includes examining skewness and outlier distribution graphs, providing valuable insights into data characteristics. To further enhance understanding, the temporal evolution of each feature is analyzed, revealing trends and patterns over specific time frames. This visualization approach serves as a powerful tool for exploring the dynamic behavior of features over time, offering a detailed perspective on their temporal characteristics. The goal for this analysis is to gain insights into the distribution and characteristics of individual features and the target label. This phase involves the following processes:

- **Descriptive Statistics:** Applying `data.describe()` to calculate measurements like mean, standard deviation, minimum, maximum. Notably, this analysis observes that most mean features lie at the floor level, except for 'PacketLength' information.

- **Target Variable Analysis:** Analyze the distribution of the 'Target' column by calculating the percentage of each class of DoH Normal and DoH Attack. The dataset is highly imbalanced, with approximately 99.9% (167,486) labeled as 'DoH attack' and only 0.01% (31) labeled as 'Normal'.
- **Skewness Distribution:** Skewness distribution identifies features that are rightly skewed (positive skew), indicating a concentration of values on the lower end and a long tail towards higher values. 'PacketLengthMean' and 'PacketTimeMean' manifests this characteristic which reflects a property that is consistent with the target label.
- **Outlier Distribution:** Outlier observes the distribution of outliers. 'PacketLengthMean' has higher values of outliers which are associated with the attack traffic, again suggesting it shares a similar nature with the target label.

### 5) DATA NORMALIZATION (IMPLICIT STEP BEFORE PCA)

The goal is to scale the numerical features that have similar range which is often beneficial for PCA to prevent features with larger scales from dominating the principal components. The step includes selecting the numerical features for normalization by applying normalization techniques like StandardScaler or MinMaxScaler. StandardScaler is used to standardize features to have zero mean and unit variance whilst MinMaxScaler is used to scale features to a specific range like 0 to 1. The choice of method might depend on the distribution of the data and the requirements of PCA.

### 6) PCA-PRINCIPAL COMPONENT ANALYSIS

This method identifies key patterns within the dataset, referred to as Principal Components (PCs). It is particularly useful when the dataset contains numerous measurements, not all of which are meaningful, often exhibiting significant covariance among variables. Variance is used to quantify the extent to which a random variable deviates from its mean, providing insight into the data's spread.

The process begins by calculating the average for each column, expressed as  $\bar{x} = 1/n \sum_{i=1}^n x_i$ . Next, the deviation of each data point,  $x_i$  from this average,  $\bar{x}$  is determined as  $deviation_i = x_i - \bar{x}$ . Finally, the covariance between two variables is computed using the formula provided in Eq. (1), enabling the identification of relationships and patterns among the data's dimensions.

$$\sigma(x, y) = \frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y}) \quad (1)$$

where  $(y_i - \bar{y})$  is a different of another data point then the sum of all deviations across frames forms a covariance matrix, capturing terms for all possible feature pairs. Principal Component Analysis (PCA) is computed using these covariance matrices, with eigenvectors corresponding to the largest eigenvalues identified as the principal components (PCs). The equation  $A v = \lambda v$  is utilized, where  $A$  rep-

resents the covariance matrix,  $v$  is the eigenvector, and  $\lambda$  is the eigenvalue. For PCA computation, columns such as ‘SourceIP’, ‘DestinationIP’, and ‘Timestamp’ are excluded because the calculations require real numerical values. Additionally, missing values in ‘ResponseTimeTimeSkewFrom-Median’ and ‘ResponseTimeTimeMedian’ are imputed using the Python library NetworkX to ensure data completeness.

PCA reduces the dimensionality of the data by projecting encoded vectors onto a single axis, preserving the key variations while minimizing information loss. In this study, negative covariance values were observed, which will be discussed in the Results and Discussion section. The PCA process produced principal components with variance scores exceeding 0.2 for some and falling below  $-0.2$  for others. These variance scores represent different covariance matrix values and provide critical insights into the underlying patterns and relationships within the dataset.

## 7) SELECTED FEATURES

A comprehensive analysis employing statistical methods and Principal Component Analysis (PCA) will be conducted to identify and select two features that demonstrate a strong correlation with the attack target. Detailed exploration of this methodology is available in our paper [1]. Initially, insights from the statistical analysis will be presented and compared. Characteristics identified through descriptive statistics, skewness, and outlier distribution graphs will be analyzed to compare and contrast the behavior of different features. This rigorous approach aims to inform feature selection with a data-driven perspective. Descriptive statistics (mean, median, standard deviation, etc.) revealed about the features in relation to the attack target.

Secondly, the analysis will focus on the Principal Components (PCs) derived from the original features. PC1 is typically associated with high scores across most features; however, in this case, some PCs exhibit negative values. PC2 has also generated several significant coefficients, making PC1 and PC2 ideal candidates for visualizing the dataset in a two-dimensional (2D) graph. In contrast, PC3 performs poorly, with many coefficients clustered just above zero and others falling below zero (negative coefficients), indicating the least significant features.

In conclusion, the selection of final feature is guided by the loadings within PC1 and PC2. The features with the highest absolute loadings in these principal components are considered to have the most influence on the primary directions of variance. Thus strong candidates for representing the underlying data structure relevant to the attack target. While PCA provides a powerful dimensionality reduction technique, we acknowledge that the selected features, while capturing significant variance, might not always have the most direct physical interpretation. Therefore, we plan to further validate the predictive power of these two selected features by incorporating them into a classification model and evaluating its performance on unseen data.

## 8) FEATURE COMBINATION AND DATASET RESHAPING

This final stage is to create a new dataset using the two selected features and reshape it into a specific tensor format. The steps are as follows:

- **Concatenate Selected Features:** Combine the two selected features into a new dataset.
- **Reshape into Tensor:** Reshape this new dataset into a tensor with dimensions  $80 \times 2 \times 480$ . This specific reshaping suggests a temporal segmentation or a specific input format required for a downstream task (not detailed in the provided text). The dimensions imply 80 segments, 2 features, and 480 time steps or data points within each segment.
- **Rename Data Slices for Testing:** The processed data slices are renamed based on index ranges. These names indicate specific segments of the reshaped data.:
  - .DoH2\_16741\_16821
  - .DoH\_0\_79
  - .DoH\_80\_159
  - .DoH\_160\_239

This newly structured dataset will be utilized for subsequent steps in the study which involves model training and testing, visualization, or further analysis on the reduced and reshaped data.

## B. STAGE 2: SPLITTING FUNCTION (PREPARING UNIVARIATE AND MULTIVARIATE INPUT/OUTPUT DATA CONFIGURATIONS)

To prepare the data for processing as a univariate input sequence, it must be transformed into a  $v_n * 1$  matrix, as illustrated in Fig. 7.

$$v = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}$$

FIGURE 7. Univariate input<sup>1</sup> sequence at transformational shape of  $v_n * 1$ .

Similarly, to handle a multivariate input sequence, the dataset should be reshaped into an  $a_m * a_n$  matrix, as shown in Fig. 8.

The univariate input sequence, when processed by the splitting function, will result in a 2D array with the shape [data array, timesteps], as depicted in Fig. 9.

The 2D splitted output will also be reshaped into [samples, sequences, timesteps, features]; a multidimensional array as to assist CNN training model. The sequences are illustrated in Fig.10. This structure represents a 4D tensor reshaped from raw input sequences, enabling compatibility with CNN or LSTM models that require multichannel or temporal input representations.

<sup>1</sup>Source: <https://slideplayer.com/slide/5854910/>; 11/3/2024

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1j} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2j} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{i1} & a_{i2} & \dots & a_{ij} & \dots & a_{in} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mj} & \dots & a_{mn} \end{bmatrix}$$

FIGURE 8. Multivariate input<sup>1</sup> sequence at transformational shape of  $m \times n$ .

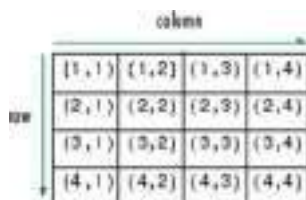


FIGURE 9. Illustration of 2D array output sequence<sup>2</sup>.

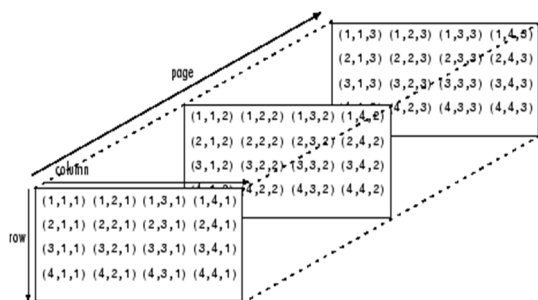


FIGURE 10. Illustration of multidimensional tensor array used for model input: [samples, sequences, timesteps, features]<sup>2</sup>.

The input layer is represented as a sequence of vectors  $[x_1, x_2, x_3, \dots, x_n]$ , which in this case corresponds to a sequence of network packets. This input is transformed into a data array,  $a$ , as described in Algorithm 1: `Univariate_Splitting_Function()` in the Appendix. The splitting function iterates through each element of the data array,  $i$ , determining the endpoint,  $end$ , based on the defined input time steps,  $b$ . Both  $a$  and  $b$  are hyperparameters.

The iteration terminates when the count reaches the length of the data array. The function then splits the data array into input sequences,  $X$ , and output values,  $y$ . Each input sequence,  $X$ , contains elements from index  $i$  to  $end$  [ $i:end$ ], referred to as pattern  $X$ . The corresponding output value,  $y$ , is the element at index  $end$  [ $end$ ], referred to as pattern  $y$ . In the final step, the patterns  $X$  and  $y$  are converted into list formats, initialized as  $X = list()$  and  $y = list()$  [ $X,y \leftarrow list(), list()$ ].

### 1) UNIVARIATE SINGLE STEP

To generate the complete set of input and output configurations, Algorithm 1 `Univariate_Splitting_Function` in the Appendix is invoked to create new objects,  $X$  and  $y$ , along with two relevant arguments,  $a$  and  $b$ . This function generates

input sequences with a 2D shape of  $[a(\text{samples or data array}), b(\text{timesteps})]$ , where  $a$  represents the number of samples (data arrays) and  $b$  denotes the number of timesteps.

For further processing, the data must be reshaped to introduce an additional dimension, converting it from a 2D array shape [samples, timesteps] to a 3D array shape [samples, timesteps, features]. This transformation is essential for preparing input sequences for prediction models such as BiLSTM, Stacked LSTM, and Vanilla LSTM. The step-by-step procedure for this process is detailed in Algorithm 2 in the Appendix, providing a clear workflow for reshaping and configuring the data.

To enable the univariate input sequence to work with a CNN-based predictive model, the data must be reshaped into a multidimensional array. The transformation changes the shape from [samples, timesteps] to [samples, sequences, timesteps, features]. In this configuration, each sample can be divided into multiple sequences, each with specific timesteps, allowing the CNN-based model to process each sequence independently over its respective timesteps. From a coding perspective, the number of sequences is inferred using  $n\_seq = -1$ , which automatically determines the sequence count. If not specified correctly, the process will return an error. This reshaping procedure and its implementation are detailed in Algorithm 3 in the Appendix.

### 2) UNIVARIATE MULTIPLE STEPS

While Algorithm 1 is designed to prepare a univariate input sequence for predicting a single-step output, Algorithm 4, presented in the Appendix, expands this capability to accommodate multiple output steps. The critical distinction lies in how the output timesteps are calculated. In Algorithm 4, the line  $output \leftarrow end+c$  projects the predictive window, where  $c$  is an additional argument specifying the number of output steps for  $y$ .

The multiple steps are generated by adding the “end of list” numerical value to a new object, output, which defines the extended range. The process of generating the input list  $X$  (referred to as `pattern_X`) remains similar to that in Algorithm 3, using the range  $a[i:end]$ . However, for the output  $y$  (referred to as `pattern_y`), it is constructed using a range  $a[end:output]$ , differing slightly from the single-output approach in Algorithm 1, which only uses the value at  $a[end]$ . This adaptation enables the model to predict multiple future steps, aligning the input-output configuration with the demands of multi-step forecasting.

To support multi-step prediction for univariate input sequences in Transformer Encoder-Decoder or Vector Output predictive models, the input-output configurations must be reshaped into a 3D array. For the Transformer Encoder-Decoder model, as detailed in Algorithm 5 presented in the Appendix, the output  $y$  must match the shape of the input  $X$ , as the model is designed to predict a specified number of timesteps for each input sample. In contrast, for the Vector Output model, only the input  $X$  requires reshaping, while the output  $y$  remains unchanged.

### 3) MULTIVARIATE MULTIPLE STEPS

The configuration of the *end* and *output* lists follows procedures similar to those in prior functions, with some modifications. In **Algorithm 6**, shown in the Appendix, the output list is adjusted slightly by subtracting 1, as shown in  $output \leftarrow end+c-1$ . This modification results in a sliced output value, omitting the last element of the list. The generated input list  $X$  is transformed from an initial  $n \times 1$  matrix;  $a[i:end]$  into an  $m \times n$  matrix  $a[i:end, -1]$ . The expression  $a[i:end, -1]$  selects elements from index  $i$  to *end* for list  $mmm$ , excluding the last element in list  $n$ .

Similarly, the output list  $y$  is transformed from  $a[end:output]$ ; a  $n \times 1$  matrix to a  $m \times n$  matrix  $a[end-1:output, -1]$ . Here,  $a[end-1:output, -1]$  selects elements starting from (*end*-1), slicing one element from *end*, and ending at the *output* index. For example, consider a list  $a = [1], [2], [3], [4], [5], [6]$ . If  $end = 3$  (index 0 to 3), then  $end-1$  selects the value 4, minus 1 yields element 3. Similarly, if  $output = 6$  (index 0 to 5), it selects elements up to index 5. This method ensures precise slicing and alignment of input and output lists for multi-step predictive tasks.

**Algorithm 7**, presented in the Appendix, outputs the input and output configurations for the multi-step prediction. The input and output values are defined, and running the example displays the 3D structures of  $X$  and  $y$ , which are  $(78,3,3)$  and  $(78,1,3)$ , respectively. These structures are precisely formatted to meet the input requirements of an LSTM model, making the data ready for use without further reshaping. The function  $X.shape <2>$  is used to ensure alignment with the 3D input dimensions. The obtained results are presented in the Results and Discussion section.

### C. STAGE 3: INTRUSION PREDICTION MODELS

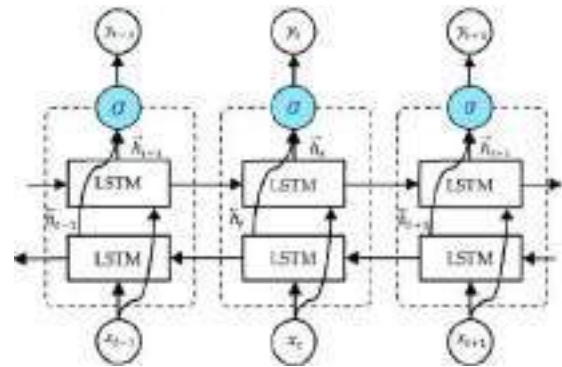
This section presents the third stage of the methodology, which focuses on training and testing the preprocessed datasets using a range of advanced prediction models. These models, applied within the context of proposed Intrusion Prediction Systems (IPS), include univariate single step models, univariate multiple step models, Multivariate multiple steps.

Table 2 presents a summary of the intrusion detection models used to provide a concise overview of the models used in our experiments. The table includes model type, architecture details, input/output formats, and intended use cases.

#### 1) UNIVARIATE SINGLE STEP MODEL

The Bidirectional LSTM (BiLSTM) prediction model for IDS comprises forward, backward, and output layers, as illustrated in Fig. 11. The architecture features LSTM cells connected in a sequential, chain-like manner. Each cell processes an input vector (e.g.,  $[x_1, x_2, x_3, \dots, x_n]$ ) along with the output from the previous forward cell  $h_{f(t)}$  and the corresponding output from the backward cell at the same timestep  $h_{b(t)}$ .

In this study, the Bidirectional library is utilized to implement the model, specifically using the Keras Layers library

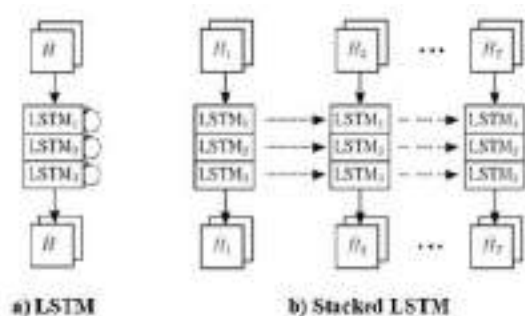


**FIGURE 11.** BiLSTM model that is used to accept univariate input sequence [87].

package. The model is constructed with the following line of code: `model.add(Bidirectional(LSTM(50, activation = 'relu'), input_shape = (b, n_features)))`. This implementation enhances the standard LSTM by processing traffic data in both forward and backward directions, enabling the model to capture a more comprehensive context from the data.

The final dense layer is defined with a single output neuron: `model.add(Dense(1))`. This configuration makes the BiLSTM model suitable for evaluating univariate input sequences and generating single-step outputs.

A stacked LSTM is an advanced architecture that builds upon the capabilities of a single LSTM layer to enhance performance. As illustrated in Fig. 12, this architecture incorporates multiple LSTM layers stacked on top of one another for improved processing of sequential data. Fig. 12(a) depicts a standard LSTM network with a single input-output layer and a single hidden LSTM layer. In contrast, Fig. 12(b) illustrates a stacked LSTM network, where multiple LSTM layers are added sequentially.



**FIGURE 12.** Stacked LSTM model<sup>6</sup> that is used to accept univariate input sequence.

The leftmost rectangle in the diagram represents the input layer, where the input sequence enters the network. Each rectangle symbolizes an LSTM layer comprising standard components, such as input, output, and forget gates. The stacked rectangles represent additional hidden LSTM layers, with the number of layers tailored to the complexity of the task. Finally, the rightmost rectangle serves as the

TABLE 2. Summary of the intrusion detection models utilized.

Model	Type	Architecture Details	Input Type(s)	Output Type	Use Case
Vanilla LSTM	RNN	1 LSTM layer, 50 units, ReLU	Univariate/Multivariate	Single-step	Baseline sequence prediction
Stacked LSTM	RNN	2 LSTM layers, 64 units, ReLU	Univariate/Multivariate	Multistep	Deeper sequence modeling
Bidirectional LSTM	RNN	1 Bi-LSTM layer, 50 units	Multivariate	Multistep	Captures forward/backward context
GRU Vector Output	RNN	1 GRU layer, 64 units, ReLU	Multivariate	Multistep	Efficient time-series prediction
Transformer Encoder-Decoder	Transformer	2 encoder/decoder blocks, 128 units	Multivariate	Multistep	Attention-based forecasting
CNN	ConvNet	1D Conv layers, 64 filters, ReLU	Multivariate	Single-step	Lightweight temporal modeling

output layer, delivering the final prediction. This architecture enhances the model’s ability to capture long-term dependencies in sequential data, making it particularly effective for complex tasks like time series forecasting.

A Vanilla LSTM, as shown in Fig. 13, represents the foundational building block of the LSTM architecture. It is the simplest form of an LSTM network, where the input flow sequentially passes through the input gate, forget gate, and output gate. These gates utilize the current input and the previous hidden state to regulate the flow of information, collectively forming the cell state.

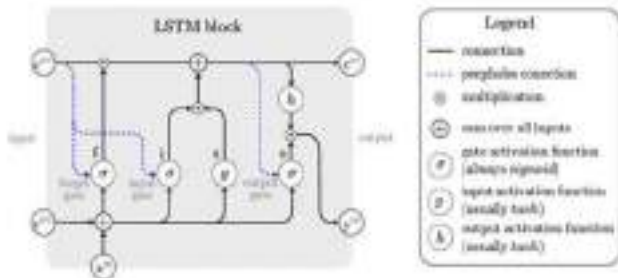


FIGURE 13. Vanilla LSTM model that is used to accept univariate input sequence [88].

The cell state is updated by combining the previous cell state with the candidate cell memory, as determined by the gates. The output gate controls how much of the updated cell state contributes to the current output, also referred to as the hidden state, of the LSTM cell. This straightforward yet effective mechanism enables the Vanilla LSTM to process sequential data efficiently.

A Vanilla LSTM is constructed using a single LSTM layer, with the following implementation: `model.add(LSTM(50, activation = 'relu', input_shape = (b, n_features)))`, and `model.add(Dense(1))`. This architecture is ideal for simpler sequential tasks, serving as a baseline for more complex LSTM variations.

Fig. 14 illustrates the architecture of a CNN model, which consists of several interconnected layers. The Input Layer receives the input data, typically represented as a 3D tensor. The Convolutional Layer, implemented as `model.add(TimeDistributed(Conv1D(filters = 64, ker-`

`nel_size = 1, activation = 'relu'))`, forms the core of the CNN. This layer applies multiple kernels or filters to the input data, extracting key features by performing a dot product operation across small, sliding windows of the input sequence. The resulting output features are known as the feature map. In this implementation, the TimeDistributed library is used to enable model reusability and support sub-sequencing. The kernel size specifies the number of timesteps included in each “read” operation of the input sequence.

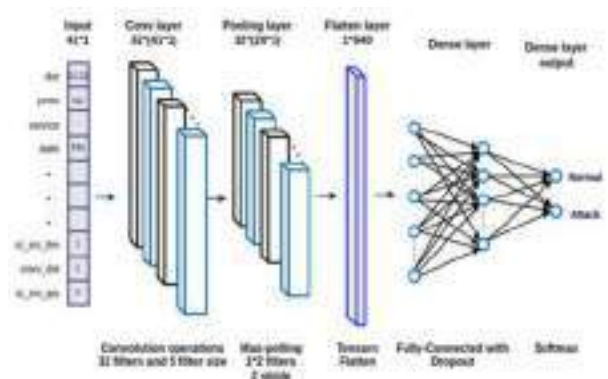


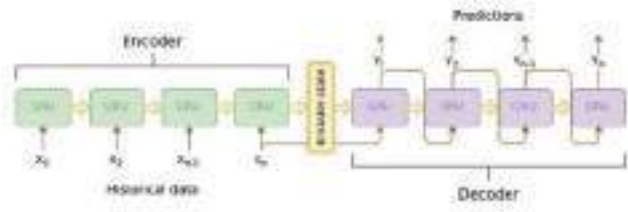
FIGURE 14. CNN model that is used to accept both univariate & multivariate input sequence. In this study, it specializes on single step output [89].

The Max Pooling Layer reduces the size of the feature maps, often by half, distilling the most relevant information. The Flatten Layer then transforms the reduced feature maps into a single one-dimensional vector, which serves as input to the Fully Connected Layer. Finally, the Output Layer produces either a binary classification or a single-step prediction, depending on the specific task for which the model is designed. This structured design enables the CNN to efficiently process and analyze sequential data.

## 2) UNIVARIATE MULTIPLE STEPS MODEL

The Transformer Encoder-Decoder model is designed for forecasting variable-length outputs and is specifically tailored for tasks involving both input and output sequences. The Transformer model is structured as a layered architecture composed of stacked encoder-decoder blocks. In this study,

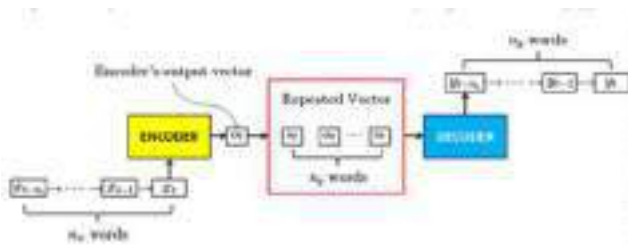
the term ‘Transformer Encoder-Decoder’ refers to this architecture and is distinct from LSTM-based encoder-decoder models. A common example of a sequence-to-sequence problem is text translation from one language to another. The architecture comprises two sub-models: the Encoder and the Decoder (Transformer), as illustrated in Fig. 15.



**FIGURE 15. Transformer Encoder-Decoder that is used to accept both univariate input sequence and yield a multistep output [90].**

The Encoder is responsible for processing and interpreting the input sequence, producing a fixed-length context vector as its output. Traditionally, the encoder consists of multiple LSTM layers, implemented with `model.add(LSTM())`, to capture the context and critical information from the input sequence. To facilitate the generation of fixed-length output sequences, the function `model.add(RepeatVector(c))` is employed. The Decoder generates the output sequence (denoted as  $y$ ) one step at a time. This is achieved through LSTM layers defined by `model.add(LSTM())`. At each timestep, the decoder’s LSTM receives the previous hidden state (from the prior timestep) and the context vector as inputs. This iterative process continues until the entire output sequence is generated, enabling the model to handle complex sequence-to-sequence forecasting tasks effectively.

LSTM can directly generate a vector interpreted as a multi-step prediction, as illustrated in Fig. 16. For multi-step forecasting using the LSTM vector output GRU, the Dense output layer is configured with more than one output unit by defining `model.add(Dense(c))`, where  $c$  represents the desired number of output units. By specifying the Dense layer’s output dimension to match the number of future timesteps, the LSTM model can predict multiple values in a single forward pass.



**FIGURE 16. Vector-Output (GRU) that is used to accept univariate input sequence and yield a multistep output [91].**

This capability makes the model particularly suitable for tasks such as anomaly detection, where identifying unusual

patterns in data is essential. The ability to predict multiple future values simultaneously enhances its efficiency and applicability in time series forecasting and anomaly detection scenarios.

LSTM can directly generate a vector interpreted as a multi-step prediction, as illustrated in Fig. 16. For multi-step forecasting using the LSTM vector output GRU, the Dense output layer is configured with more than one output unit by defining `model.add(Dense(c))`, where  $c$  represents the desired number of output units. By specifying the Dense layer’s output dimension to match the number of future timesteps, the LSTM model can predict multiple values in a single forward pass.

This capability makes the model particularly suitable for tasks such as anomaly detection, where identifying unusual patterns in data is essential. The ability to predict multiple future values simultaneously enhances its efficiency and applicability in time series forecasting and anomaly detection scenarios. The predictive models were implemented using the Keras deep learning library with TensorFlow backend. Table 3 summarizes the architecture and training hyperparameters for each model.

### 3) MULTIVARIATE MULTIPLE STEPS MODEL

For multivariate input, various LSTM configurations from previous models can be utilized. When an LSTM model is defined with a layer such as `model.add(LSTM(100, ...))`, it creates an LSTM layer where the input shape is specified as a tuple of two values: the number of input steps ( $b$ ) and the 3D features, determined by the previously defined `x.shape [2]` (i.e., `input_shape = (b, n_features)`).

For scenarios involving multiple parallel inputs, the `RepeatVector` function is employed to replicate the input for a defined number of times. Additionally, the Dense layer, defined as `model.add(Dense(c))`, determines the number of output steps ( $c$ ) based on the specific requirements of the task.

### D. STAGE 4: PERFORMANCE EVALUATION

The performance metrics used in this study primarily focus on MAE percentage, which quantifies the discrepancy between the actual observed value  $v_A$  and the expected value  $v_E$ . This error can be expressed either in absolute terms, representing the numerical difference, or in relative terms, as the ratio of the absolute discrepancy to the expected value. The approximation MAE error  $\delta$  is calculated using the formula:

$$\delta = |(v_A - v_E)/v_E| * 100\% \tag{2}$$

This metric is independent of the polarity of the discrepancy (positive or negative) and remains numerically stable. It serves as a measure of precision or prediction error, reflecting the model’s accuracy.

Another evaluation metric is the coefficient of determination  $R^2$ , which measures the proportion of variation in the dependent variable explained by the predicted independent

**TABLE 3.** Architecture and training hyperparameters for each model.

Model	Hidden Units	Dropout Rate	Activation Function(s)	Output Units	Optimizer	Loss Function
Vanilla LSTM	50	0.2	ReLU	1	Adam	MSE
Stacked LSTM	2 × 64	0.3	ReLU	1	Adam	MSE
Bidirectional LSTM	50	0.2	ReLU	1	Adam	MSE
GRU (Vector Output)	64	0.2	ReLU	5	Adam	MSE
Transformer Encoder-Decoder	2 layers × 128	0.1	ReLU (dense), Softmax (output)	5	Adam	MSE
CNN	64 filters	—	ReLU (Conv), Linear (Dense)	1	Adam	MSE

variable. It is defined as:

$$R^2 = 1 - (SS_{res} - SS_{tot}) \quad (3)$$

Here,  $SS_{res}$  (sum of squares of residuals) is given by:

$$SS_{res} = \sum_i (y_i - f_i)^2 \quad (4)$$

where the difference  $(y_i - f_i)$ , is known as residuals,  $e$ .  $y_i$  represents the observed data,  $f_i$  is the predicted value.  $SS_{tot}$  (total sum of squares) is calculated as:

$$SS_{tot} = \sum_i (y_i - \bar{y}_i)^2 \quad (5)$$

where  $y_i$  is the observed data, and  $\bar{y}_i$  is its mean. This metric evaluates the overall variance in the data against the model's predictions.

To statistically test the significance of differences between the means of observed and predicted data, Analysis of Variance (ANOVA) is applied. ANOVA analyzes variations by comparing deviations of all observed data from the grand mean. Sample variance  $s^2$  is calculated as:

$$s^2 = 1/(n - 1) \sum_i (y_i - \bar{y})^2 \quad (6)$$

where  $1/(n - 1)$  represents the degrees of freedom, and the sum of squared differences  $(y_i - \bar{y})$  is termed the sum of squares. The final result,  $s^2$ , is referred to as mean squares.

The F-test is then used to assess the deviations. For one-way ANOVA, statistical significance is determined by comparing the F-test results, calculated as:

$$F_{test} = \frac{\text{variance between treatments}}{\text{variance within treatments}} \quad (7)$$

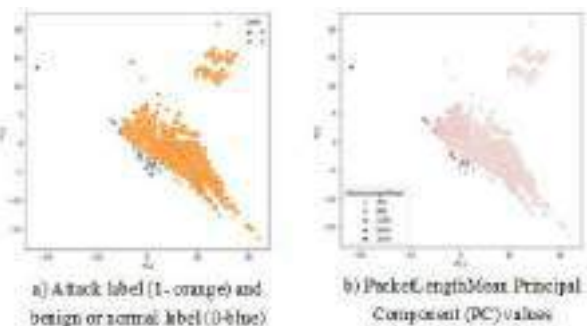
This comprehensive approach ensures robust evaluation of model performance and the statistical significance of observed patterns.

#### IV. RESULTS

This section presents the findings from the comparative analysis of time series intrusion prediction models. It begins with a discussion of the results from dataset preprocessing, followed by an examination of the outcomes from the splitting functions. Subsequently, the performance metrics are analyzed, focusing on error percentage, coefficient, and variation. A high-level summary of the models' performance is provided in Table 4.

#### A. RESULTS OF DATASET PROCESSING

CIRA-CIC-DoHBrw-2020 dataset has 167,517 rows and 35 columns (167,517, 35). A rigorous analysis and visualization of this dataset building upon our previous work [1] has been conducted. Leveraging on various pandas and numpy library, the specific column is sliced and the row is limited to 80 rows. Since CIRA-CIC-DoHBrw-2020 dataset has 35 columns (labels), only PacketLengthMean label will be nominated as to represent the univariate data array input sequence since it has PC value similar to the DoH attack's label as depicted in Fig. 17. As stated earlier, from previous study in [1] laid the groundwork for an extensive investigation that uncovered the crucial PC value leading to the identification of the authentic benchmarked dataset. This univariate data array has a sequence of events that indicates normal or benign traffic.



**FIGURE 17.** PacketLengthMean label (b) will be the chosen label for the univariate data array input sequence (X) since it has similar PC values as to the DoH attack's label (a) [1].

On the other hand, there were two data columns designated to exemplify the multivariate input sequence namely PacketLengthMean and PacketTimeMean as represented in Fig. 18. PacketTimeMean is nominated as the second data dimension since it has a profound trait to determine the next sequence of events complementing the PacketLengthMean.

For instance, the lowest packet time mean signifies normal packet length. PacketTimeMean has a similar rightly skewed distribution. In retrospective, most of the features were rightly skewed as opposed to the DoH label which typically attributed to attack traffic. Nevertheless, these are not the sole attributes to be consumed as the multivariate input sequence. There are plenty of attributes that can be

**TABLE 4.** A high-level summary of the top-performing time series intrusion prediction models.

Input type		Univariate Input					Multivariate Input			
Output type		Single Step Output			Multistep Output		Single Step Output	Multivariate Output	Multi Step Output	
Model	Bidirectional LSTM	CNN	Stacked LSTM	Vanilla LSTM	Decoder Encoder (Transformer)	Vector Output (GRU)		LSTM		
Performance Metric	Error percentage	0.75	0.77	0.67	0.74	<b>0.1</b>	<b>0.1</b>	<b>0.5</b>	0.72	<b>0.13</b>
	Coefficient of determination	-0.03	-0.09	-0.12	0.03	-0.02	-0.10	0.64	N/A	0.77
	ANOVA (P-value)	0.36	0.35	0.34	0.38	0.38	0.37	0.65	N/A	0.72

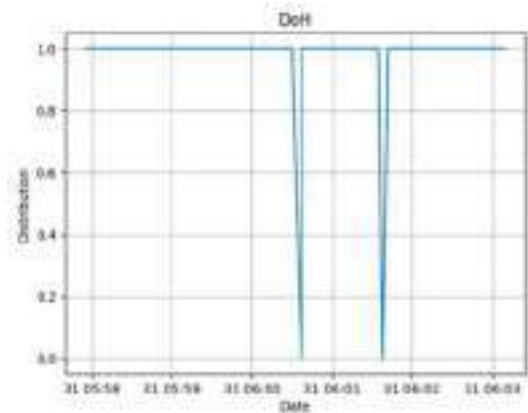


**FIGURE 18.** PacketLengthMean and PacketTimeMean are the two input sequence to represent the multivariate input sequence.

selected through various correlation analysis. This technique is among the best evaluation techniques to further expound on the choices of relevant multi-attributes.

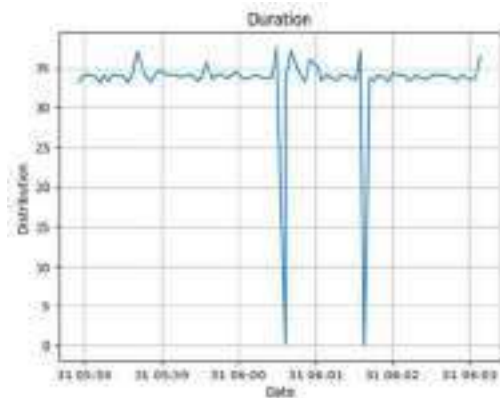
DoH dataset of CIRA-CIC-DoHBrw-2020 establishes security flaws in DNS namely DNS tunneling or any DNSbased malware. These flaws are capable of bypassing firewalls' security, therefore detecting DoH threats are crucial. The dataset features are defined as flow information or processed meta-data. There are 35 columns (from 0 to 34) altogether. An entry index from 0 to 167516. It has one entry of Boolean datatype, 26 entries of float64 datatype, 5 entries of int64 datatypes and, 3 entries of objects datatype. Memory usage to process this 167k counts of dataset is about 44Mbytes. For this research, a range of entry index between 16741 to 16821 were extracted. It is then save into a CSV file; data\_new.to\_csv('DoH2\_16741\_16821.csv'). This entry exhibits some interesting traffic behaviors. These behaviors are illustrated in Figs. 15, 16 and 17.

Fig. 19 shows the DoH distribution over a certain period of time for which it is indexed by minutes between date/time: 31/3/2020 5:57 to date/time: 31/3/2020 6:03. Distributed



**FIGURE 19.** DoH distribution over time.

DoH around value 1 over time indicates an attack period whilst distribution which is around value 0 over time indicates benign or normal traffic. This character is also manifested by 'Duration' distribution as depicted in Fig. 20. During benign or normal traffic, packet travels at massive drop near 0s duration whilst during attack, packet fluctuated travelling behavior between 30s to more than 35s.



**FIGURE 20.** Duration distribution over time.

Fig. 21 on the other hand depicts various data distribution for which the relationship is clearly spotted. It shows the relationship between PacketLengthMean and the manifested benign or attack traffic from the previous distribution; DoH and Duration. In this graph, a benign traffic is represented by either a sharp increase in distribution (hitting 1122.029221 bytes) or a steep drop (hitting 74.33333333 bytes) in the PacketLengthMean distribution. DoH label seems depicted as a flat distribution since the y-axis range has a huge extreme value from 0 to 1000.

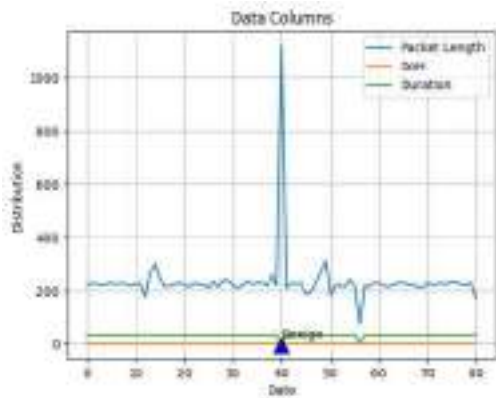


FIGURE 21. PacketLengthMean distribution and its relationship with DoH and duration distribution.

TABLE 5. Extracted dataset of size (78,3).

index	Duration	PacketTimeMean	PacketLengthMean
0	33.231672	7.857725	218.451613
1	34.059338	8.668500	228.793103
2	34.062778	8.387013	223.433333
3	33.964155	8.363209	223.366667
4	33.166819	8.090497	223.433333
...	...	...	...
76	34.071067	8.677588	228.758621
77	33.716489	8.283129	223.433333
78	33.511712	8.202355	223.433333
79	33.634527	8.523066	228.758621
80	36.440616	2.493406	172.505085

Finally, Table 5 depicts the final extracted variables from the complete set of CIRA-CIC-DoHBrw-2020 for which it is demonstrated some useful character to be extensively used in training and extrapolating prediction results on this research.

It is indeed CIRA-CIC-DoHBrw-2020 is an imbalanced dataset whereby most of its are categorized as attack’s frame. There are only 31 of them considered benign. In a low footprint attack environment, this is considered a near-realistic public dataset. Certain augmentation techniques may not be suitable for all domains. For instance, data augmentation on this near-realistic dataset could diminish the attack properties of real cyberattacks. Furthermore, overusing extensive data

augmentation can reduce the diversity of the dataset limiting the model’s ability to learn robust features, and may lead to overfitting where the model becomes too specialized to the training data and performs poorly on unseen data [92]. Hence precise prediction holds a vital role in forecasting machine learning model across various domains [93].

**B. RESULTS OF THE SPLITTING FUNCTIONS**

Through this function, the dataset was prepared to be in the diverse time series input and output sequence. Input sequence is the input data from multiple variables (multivariate) or from a single variable (univariate). Output sequence is the target distribution. It can be generated into multiple step or single step of univariate or multivariate. In this paper, the univariate output is labelled by PacketLengthMean, whilst the multivariate output is labelled by Duration, PacketTimeMean and PacketLengthMean.

Fig. 22 demonstrates a snapshot of a configured input and single step output sequence of a univariate PacketLength-Mean of size (78,6) (78,). The prediction model that is trained from this single window of six univariate input series dataset will be expected to produce a predicted result similar to 1122 bytes.

```
[[223.43333333]
 [228.79310345]
 [228.82758621]
 [217.59375 ]
 [258.9375 ]
 [218.36363636]] [1122.02922078]
```

FIGURE 22. Univariate single step input & output sequence of size (78,6)(78,) (for Bidirectional, Stacked, CNN and Vanilla LSTM).

```
[[223.43333333]
 [228.79310345]
 [228.82758621]
 [217.59375 ]
 [258.9375 ]
 [218.36363636]] [[1122.02922078]
 [ 208.82352941]]
```

FIGURE 23. Univariate input sequence & multiple output steps of size (78,6) (78,2) (for Encoder and Vectorsed Output).

On the other hand, Fig. 23 demonstrates a snapshot of a configured input and multiple step output sequence of a univariate PacketLengthMean of size (78,6) (78,2). The prediction model that is trained from a single window of six univariate input series dataset will be expected to produce predicted result similar to the first step target which is 1122 bytes and the second step target which is 208 bytes.

```
(78, 3, 2) (78,)
[[33.858211 8.13939616]
 [37.671223 10.21754955]
 [ 0.203372 0.15213751]] 1122.0292207792209
```

FIGURE 24. Multivariate (2 variables) input sequence and single step output of size (78,3,2) (78,).

Subsequently Fig. 24 shows the generated input and output sequence of Multivariate single step function. This snapshot of Multivariate time series of size (78, 3, 2) (78,). The prediction model that is trained from a single window of three multivariate input series dataset will be expected to produce predicted result similar to 1122 bytes.

Next process is to reflect on the same the generated Multivariate input sequence, however this time with multiple output window step of a single variable (univariate). Fig. 25 is a snapshot of Multivariate time series of size (78, 3, 2) and the output series of size (78, 2). The prediction model is trained from a tree window size of two multivariate input series dataset which is expected to produce predicted result similar to 1122 bytes from the first output step and subsequently 208 from the second output step.

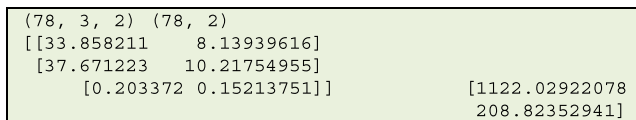


FIGURE 25. Multivariate (2 variables) input sequence and multiple steps output of size (78,3,2) (78,2).

Finally, Fig. 26 depicts a sequence of three multivariate input dataset of size (78,3,3) (78,1,3). The first dimension is the number of samples, in this case 78. The second dimension is the number of time steps (window size) per sample, in this case 3 is the value specified to the function. Finally, the last dimension specifies the number of parallel time series or the number of variables, in this case 3 (for the three parallel series).

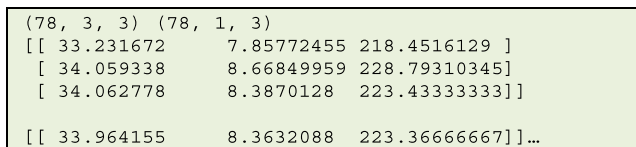


FIGURE 26. Multivariate (3 variables) input sequence and multivariate multiple step output of size (78,3,3) (78,1,3).

This is expected to yield a subsequent three multivariate output with only 1 step size (one window size) of each variable. The output is expected to produce similar predicted result for the next single step output of those three variables. LSTM as input hence the data is ready to use without further reshaping. X.shape<2> function is utilized to match the 3D input dimension.

### C. RESULTS OF COMPARING UNIVARIATE & MULTIVARIATE PREDICTION MODEL

Tables 6 and 7 summarize the series of performance analysis on various adopted time series predictive model over various univariate and multivariate input sequences. Table 6 shows the univariate performance whilst Table 7 records the multivariate performance. Fig. 27 presents a detailed analysis of the prediction performance for various models trained

on univariate input sequences, focusing on both single-step and multi-step outputs. The results reveal the strengths and limitations of each model. The Bidirectional LSTM model's single-step prediction is shown in Fig. 27(a), with an error rate of 75% illustrated in Fig. 27(b). Similarly, the CNN model's single-step prediction, depicted in Fig. 27(c), resulted in an error rate of 77%, as shown in Fig. 27(d), indicating a slight increase in deviation compared to the Bidirectional LSTM.

The Stacked LSTM model, visualized in Fig. 27(e), exhibited an improved performance, achieving an error rate of 67%. This enhancement is attributed to the model's architecture, which effectively captures long-term dependencies in sequential data. The Vanilla LSTM model's single-step prediction, presented in Fig. 27(g) and 27(h), recorded an error rate of 74%, highlighting its baseline performance.

The Vector prediction model, shown in Figs. 27(i) and 27(j), extends its functionality to produce multi-step outputs from univariate input sequences. It achieved error rates of 65% for the first timestep and 10% for the second, demonstrating a significant reduction in error over successive steps. Lastly, the Encoder model (Transformer) results, depicted in Figs. 27(k) and 27(l), revealed a promising capability to generate multi-step outputs, with error rates of 55% for the first timestep and near-perfect accuracy (0% error) for the second.

The Encoder model emerged as the most effective in this comparison, largely due to its ability to extract and utilize contextual information from the input sequence. These results were achieved through optimization of model hyperparameters, such as the number of dense layers, filters, and epochs. However, the CNN model's single-step predictions required a filtering layer, which was not implemented, as reflected by the missing values (N/A) in Table 6. Consequently, the CNN model was not tested for multi-step outputs, limiting its comparative evaluation. This comprehensive analysis underscores the varying capabilities of predictive models and highlights the Encoder model's superior performance in handling univariate input sequences.

Subsequently, Fig. 28 through 34 demonstrate the influence of multivariate input sequences on the LSTM predictive model. Fig. 28 demonstrates the single-step prediction performance effect of the LSTM model. The measured error percentage at 51% deviation from the primary target is clearly seen in Fig. 29. Compared to all models trained exclusively on univariate input sequences, this model still shows better performance.

Then, Figs. 30, 31 and 32 illustrate the multivariate multi-step output prediction of the LSTM predictive model based on multivariate input sequences. The predicted output diverges at 72% away from the prime target as reflected by the error percentage in Fig. 30. The model receives multiple variables as input for training. However, an excessive number of variables for training might hinder model performance by obscuring essential information within the dataset leading to poor results.

**TABLE 6. Univariate splitting function input performance on time series predictive modelling.**

Predictive Error Percentage	Univariate Input															
	Single Step Output						Multistep Output									
	Bidirectional LSTM		CNN		Stacked LSTM		Vanilla LSTM		Decoder Encoder (Transformer)				Vector Output (GRU)			
	Target %	Predict %	Target %	Predict %	Target %	Predict %	Target %	Predict %	Target %	Predict %	Predict % (2 <sup>nd</sup> step)	Target %	Predict %	Predict % (2 <sup>nd</sup> step)		
20 denses-300 epochs	0	0.75	N/A	N/A	0	0.78	0	0.74	0	0.75	0.22	0	0.78	0.30		
50 denses-300 epochs	0	0.77	N/A	N/A	0	0.67	0	0.75	0	0.55	0.1	0	0.75	0.28		
100 denses-300 epochs	0	0.75	N/A	N/A	0	0.78	0	0.78	0	0.77	0.1	0	0.65	0.1		
20 denses-32filter-300 epochs	N/A	N/A	0	0.77	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A		
50 denses-64filter-300 epochs	N/A	N/A	0	0.78	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A		
100 denses-128filter-300 epochs	N/A	N/A	0	0.78	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A		

**TABLE 7. Multivariate splitting function input performance on time series predictive modelling.**

Predictive Error Percentage	Multivariate Input							
	LSTM							
	Single Step Output		Multivariate Output			Multi Step Output		
	Target %	Predict %	Target %	Predict %	Predict-2 <sup>nd</sup> step %	Target %	Predict %	Predict-2 <sup>nd</sup> step %
50 denses-300 epochs	0	0.51	N/A	N/A	N/A	N/A	N/A	N/A
100 denses-300 epochs	N/A	N/A	0	0.72	N/A	0	0.4	0.13

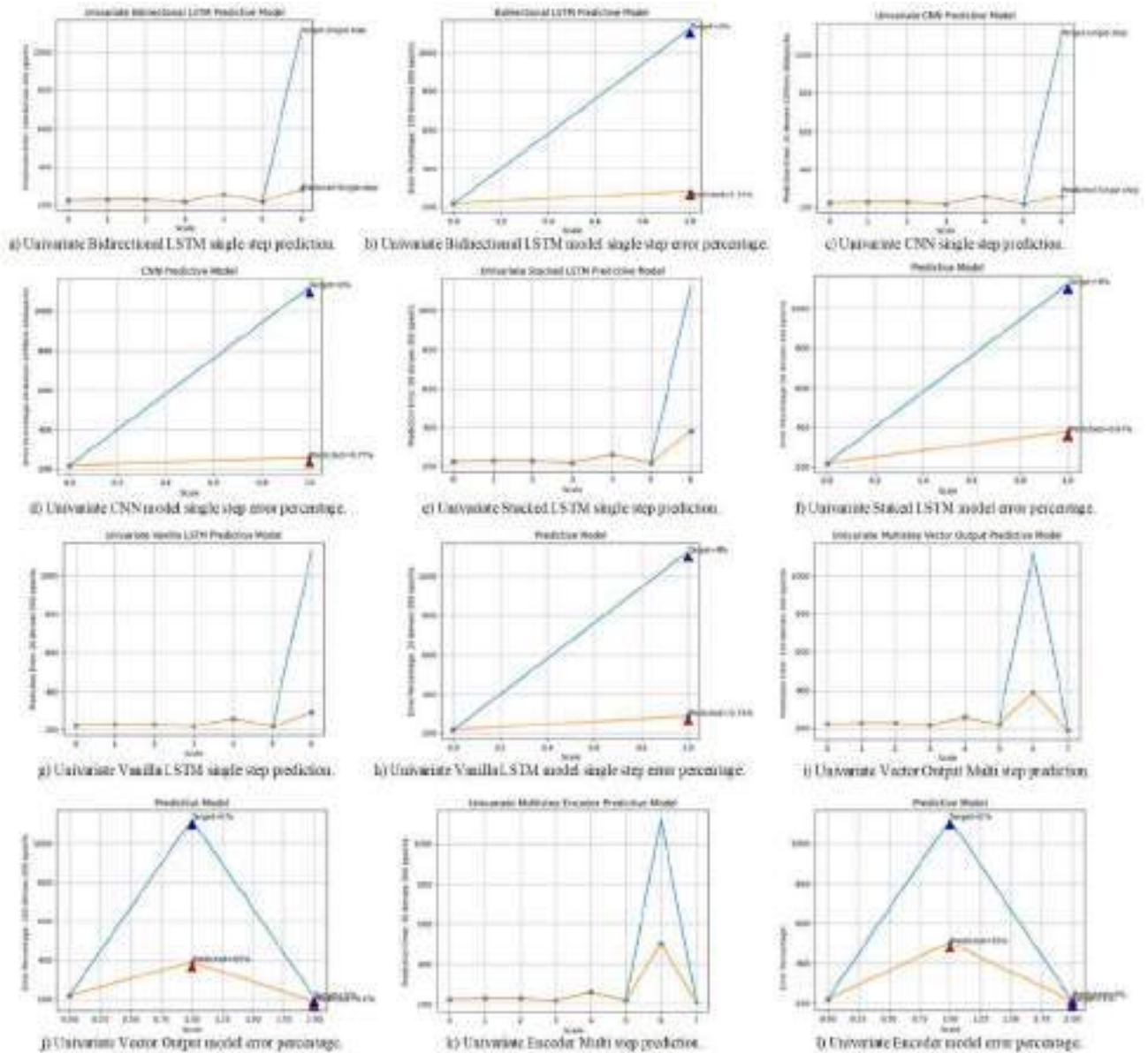
However, this 72% divergence applies solely to the forecasted Pkt\_Len\_Mean. There are two additional variables, Pkt\_Time\_Mean and Duration were accurately predicted for both the first and second timestamps. Error percentages for both variables at the first timestamp range from approximately 0% to 30% respectively as depicted in Fig. 31. It is possible that these small prediction errors for both Duration and Pkt\_Time\_Mean might be attributed to their consistent distribution over the observed time frame. Since its manifest a consistent pattern for a long period of time, the training tends to influence the other variable. For brevity, the graphs illustrate the error percentage rate only at the first output step.

Exploring the final LSTM model, trained on multiple variable input sequences is explored in Figs. 32 and 33 which showcase its multi-step output projections. The trained multi input variables are derived from the two columns labeled Duration and Pkt\_Time\_Mean. These two input sequences (multivariate input) are utilized to predict the trajectory of Pkt\_Len\_Mean, a single output variable. As discussed earlier, PkT\_Len\_Mean is chosen as the target label due to its distribution resembling that of DoH traffic.

Fig. 34 illustrates the trajectory of the predicted traffic compared to the target, showcasing the multi-step nature of the output over two timestamps. The analysis of prediction errors highlights the superior precision of the Multivariate Multiple Step Forecasting method using LSTM. For the first step, the forecast demonstrates a deviation of only 44% from the target, as depicted in Fig. 34, while the second step achieves an even lower error rate of 13%. To streamline the presentation of results, Table 7 focuses exclusively on dense layers that produce optimized predictions, which explains the absence of values (N/A) for less effective configurations. This concise representation underscores the efficacy of the proposed method in achieving accurate multi-step forecasting.

**D. COEFFICIENT OF DETERMINATION (R-SQUARED) RESULTS**

Fig. 35 illustrates the coefficient of determination (R-squared) for each predictive model. R-squared is a statistical metric that quantifies the proportion of variance in the observed traffic (dependent variable) explained by the



**FIGURE 27. Error percentage of univariate input sequence over the predictive model.**

predicted traffic (independent variable) within a regression framework. This measure ranges from 0 to 1, where higher values indicate a stronger model fit to the data.

In this analysis, multivariate models, such as “Multivariate Multiple Steps” and “Multivariate Single Step,” exhibit notably higher R-squared values, exceeding 0.6, which signifies a good fit to the data. Conversely, most univariate models demonstrate negative R-squared values, indicating that these models perform worse than a simple horizontal line as a predictor. Negative R-squared values typically suggest significant issues with the model, either in its design or the data used for fitting. However, in this context, the negative R-squared values for univariate models may be attributed to the nature of network traffic, which does not conform

to the patterns typically captured by regression lines. This emphasizes the superior performance of multivariate models in capturing the complexities of network traffic.

The chart in Fig. 36 shows the F-statistic for each model, which is a measure of the variance between the groups compared to the variance within each group. A higher F-statistic generally suggests a larger difference between the groups. However, the F-statistic alone isn’t sufficient to determine statistical significance; it must be considered alongside the p-value (as seen in the first chart). The magnitude of the F-statistic here varies, but as the corresponding p-values are not statistically significant, the differences are not meaningful. The subsequent bar chart in Fig. 37 displays the p-values resulting from an F-oneway test for each model. The p-value

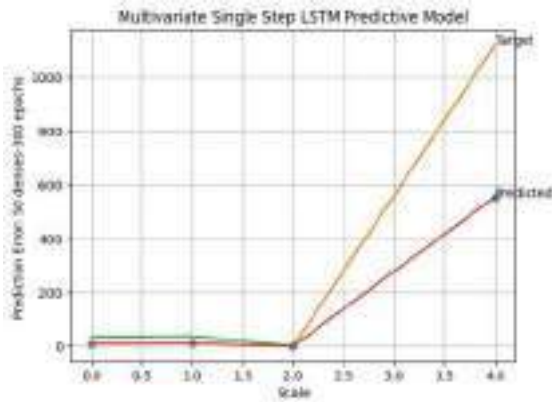


FIGURE 28. Multivariate single step LSTM predictive model.

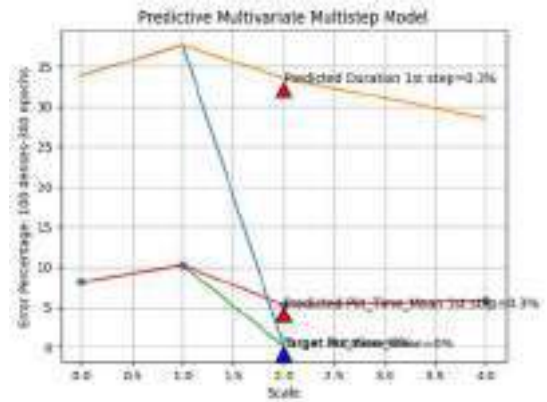


FIGURE 31. Multivariate input and multivariate multiple step output LSTM predictive model error percentage (Variable Duration & Pkt\_Time\_Mean).

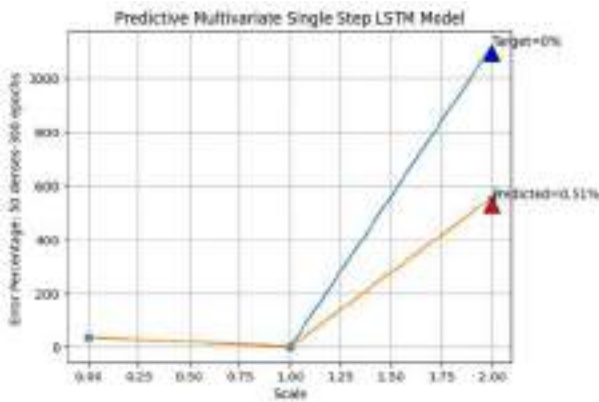


FIGURE 29. Multivariate single step LSTM predictive model error percentage.

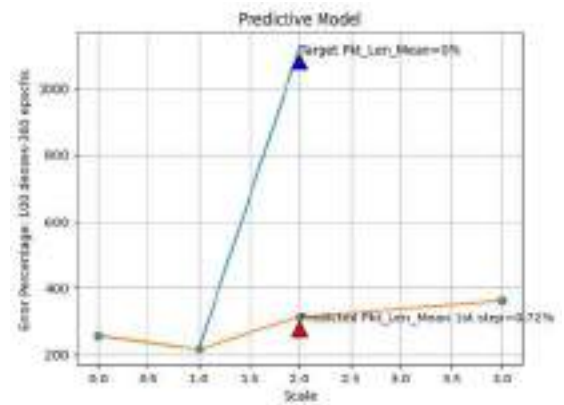


FIGURE 32. Multivariate input and multivariate multiple step output LSTM predictive model error percentage (Variable Pkt\_Len\_Mean).

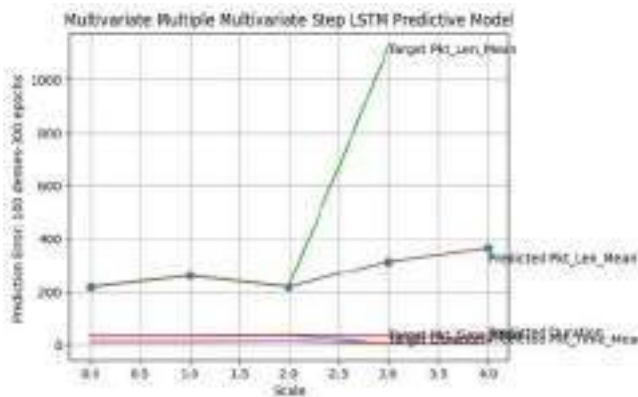


FIGURE 30. Multivariate input and multivariate multiple step output LSTM predictive model.

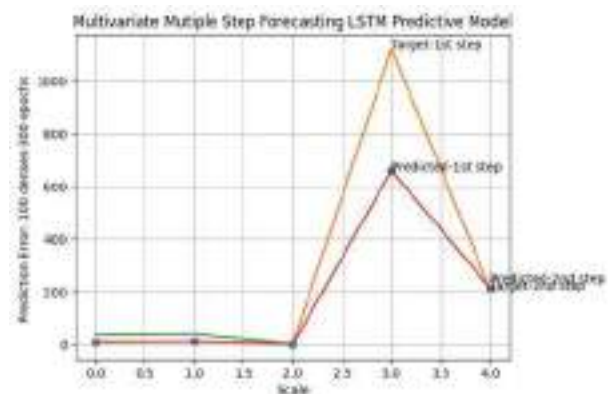
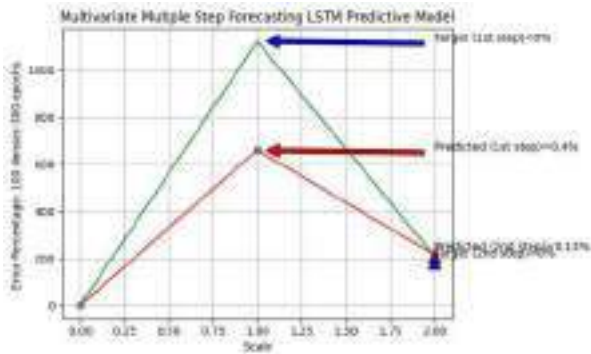


FIGURE 33. Multivariate input multiple step output LSTM predictive model.

represents the probability of observing the obtained results (or more extreme results) if there were no real difference between the groups being compared. A low p-value (typically below 0.05) suggests that there's a statistically significant difference. In this case, almost all p-values are above 0.05, implying that none of the models demonstrate a statistically

significant difference in performance based on the F-oneway test.

In the case of the F statistic = 5.731 and p-value = 0.04357 respectively. Since the p-value is less than 0.05 hence we would reject the null hypothesis. This implies that we have sufficient proof to say that there exists a difference in the



**FIGURE 34.** Multivariate input multiple step output LSTM predictive model error percentage.

performance. However, since the significance probability is a value set by the researcher according to the circumstances of each study, it does not necessarily have to be 0.05.

For one-way ANOVA the statistical significance is verified by comparing the  $F\text{-test} = (\text{variance between treatments}) / (\text{variance within treatments})$  assess whether any of the treatments are on average superior, or inferior. On the contrary, an  $F\text{-statistic}$  of 0 indicates that the independent variable does not explain the variation in the dependent variable.

### E. CROSS VALIDATION

To further evaluate the top-performing models, including the Transformer encoder-decoder, GRU, and two multivariate models, cross-validation was conducted. Fig. 38 presents a bar chart illustrating the performance of a Univariate Vector Output GRU model across five cross-validation folds, using the coefficient of determination (R-squared) as the performance metric.

The results reveal variability in the model's predictive performance. Fold 1 demonstrates a very poor fit, with a negative R-squared value of  $-1.90$ , indicating that the model performed inadequately on this fold's test data. Folds 2 and 3 show moderate R-squared values of  $0.27$  and  $0.19$ , respectively, suggesting limited predictive power in these cases. In contrast, Folds 4 and 5 exhibit exceptionally high R-squared values of  $0.997$  and  $0.998$ , reflecting an excellent fit and highlighting the model's ability to accurately capture the underlying data patterns for these folds. These results emphasize the variability in performance and the importance of robust evaluation methods.

Next, Fig. 39 depicts the bar chart displays the Coefficient of Determination (R-squared) values for a Univariate Transformer encoder-decoder model across five different folds of cross-validation. Each bar represents a fold, and its height corresponds to the R-squared value for that fold. The first fold shows a very poor fit (R-squared of approximately  $-4.74$ ), while the 4th and 5th folds show very good fits (R-squared of approximately  $0.99$  and  $0.99$  respectively). The 2nd and 3rd folds show moderate fits (around  $0.6$ ).

The significant difference in R-squared between the first fold and the other folds is noteworthy. This suggests that the model's performance might be sensitive to the specific data in that particular fold. It might be due to outliers in the data or an issue with how the data was split for that fold. The other folds are demonstrating generally good fit.

The bar chart in Fig. 40 illustrates the Coefficient of Determination (R-squared) for a multivariate multiple-steps model across five folds of cross-validation. R-squared serves as a measure of how well the model fits the data, with values closer to 1 indicating a superior fit. The model consistently demonstrates strong performance across all folds, as reflected by the high R-squared values. While slight variations are observed, with Fold 2 exhibiting a marginally lower R-squared compared to the others, the overall results indicate stable and reliable performance. These findings suggest that the multivariate multiple-steps model generalizes effectively to unseen data, showcasing robust predictive capabilities.

Subsequently, the R-squared bar chart in Fig. 41 visualizes the performance of a multivariate single-step model across five different folds of cross-validation. All five folds show very high R-squared values, extremely close to 1. This suggests that the model performs exceptionally well in each fold and can accurately predict the target variable. The slight variations between folds (e.g., the 3rd fold having a slightly lower R-squared) are minor and may reflect minor differences in the data within each fold. Overall, the model appears highly robust and fits the data well.

Finally, Fig. 42 shows the results of the MAE for five different time series forecasting models, for each were evaluated across five folds of cross-validation. MAE measures the average magnitude of the errors in a set of predictions, without considering their direction. A lower MAE indicates better accuracy. Univariate Vector Output GRU Multiple Step & Univariate Transformer Encoder-Decoder Multiple Step: These models exhibit high variability in performance across folds. Some folds have very low MAE (around 1-3), suggesting good performance, while others have significantly higher MAE (up to  $\sim 30$  for GRU). This inconsistency may due to several reasons. It might indicate that these models may be sensitive to the specific data in each fold, or that they are not consistently capturing the underlying patterns in the time series data. We choose the former over the latter since the same set of data is used throughout the process. Hence the data consistency is preserved.

On the other hand, Multivariate multiple steps: This model demonstrates more consistent performance than the univariate models, with MAEs generally ranging from 0.5 to 7. However, the model still exhibits some variability across the folds, with fold 2 and fold 5 showing higher errors. Conversely, the Multivariate single step: This model has the lowest MAE values overall compared to other models, with most folds having errors below 2. This indicates potentially the strongest predictive performance. Fold 5 shows an outlier value of approximately 4.2, suggesting potential issues with

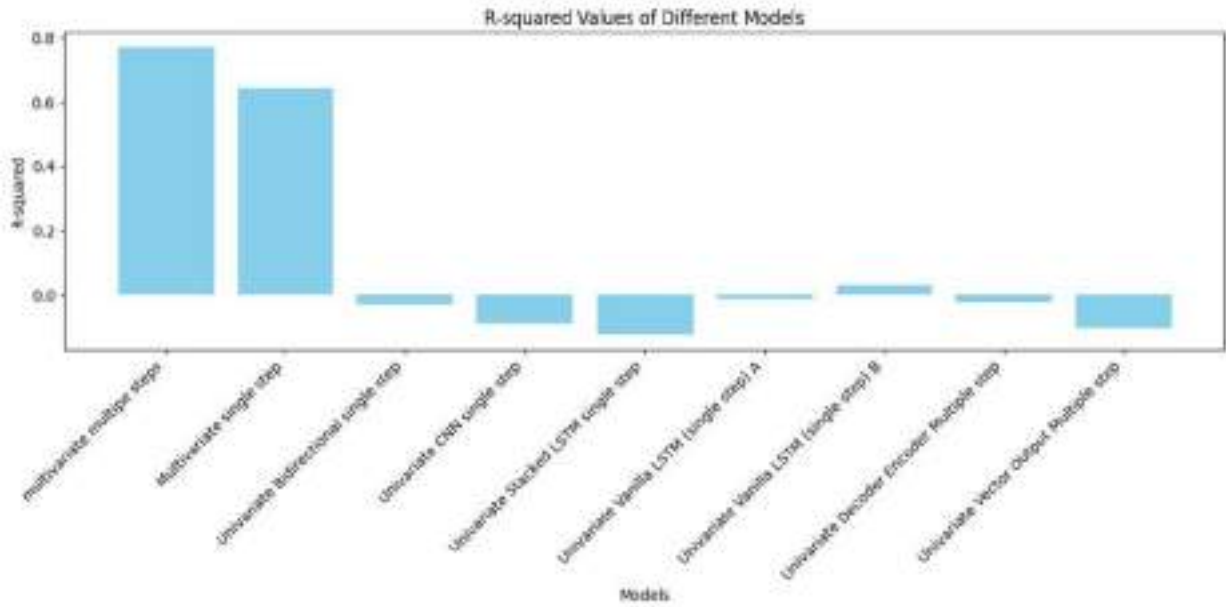


FIGURE 35. Coefficient of determination (R-squared).

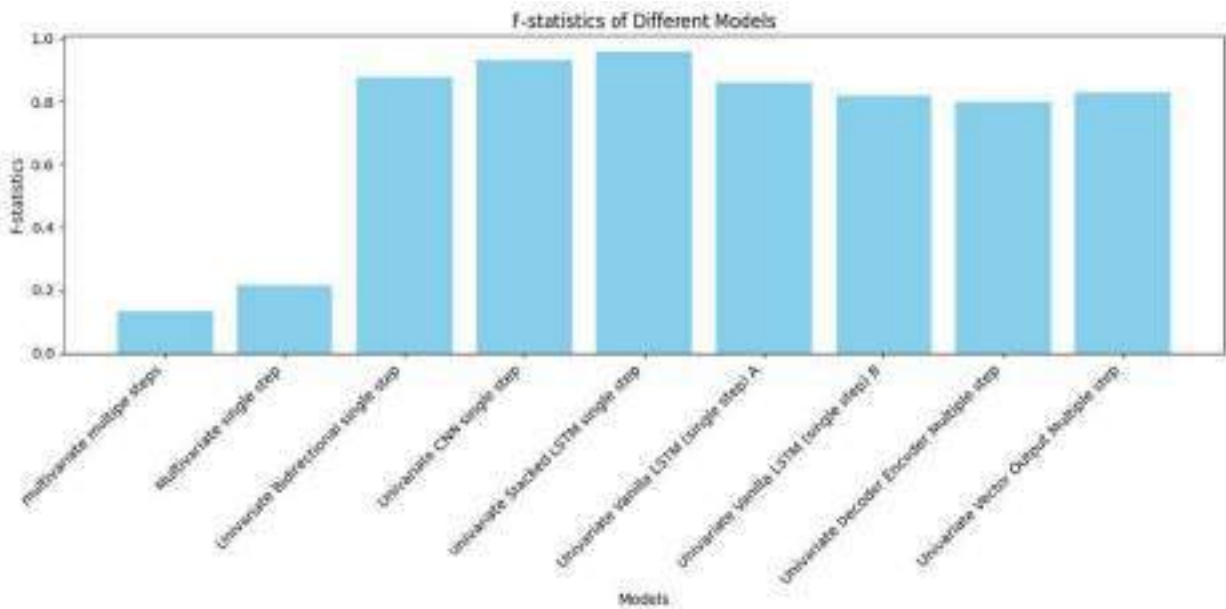


FIGURE 36. ANOVA F-statistics.

this specific fold or a limitation of this model in capturing long-term trends.

**F. COMPARATIVE ANALYSIS WITH STATE OF THE ART MODELS**

A supplementary experiment was conducted in this study utilizing ARIMA, GRU, and Ensemble Learning models, representing state-of-the-art approaches, for the purpose of comparative analysis. ARIMA is good choice for univariate predictions because its capture linear dependencies between current and past observations [94]. While SARIMA (extended version of ARIMA) can handle complex season-

ality, multivariate patterns can still be challenging for these models to capture accurately [95]. The experiment is proposed to verify the advantages of deep learning models with the ultimate goal of enhancing their persuasiveness. ARIMA mathematical expression is shown in equation (8);

$$AMIRA = \left(1 - \sum_{i=1}^{p'} a_i L^i\right) X_t = \left(1 + \sum_{i=1}^q \theta_i L^i\right) \varepsilon_t \tag{8}$$

$L$  in the expression is the lag operator or backshift operator to produce the previous element,  $a_i$  are the randomized autoregressive parameters,  $\theta_i$  are the moving average

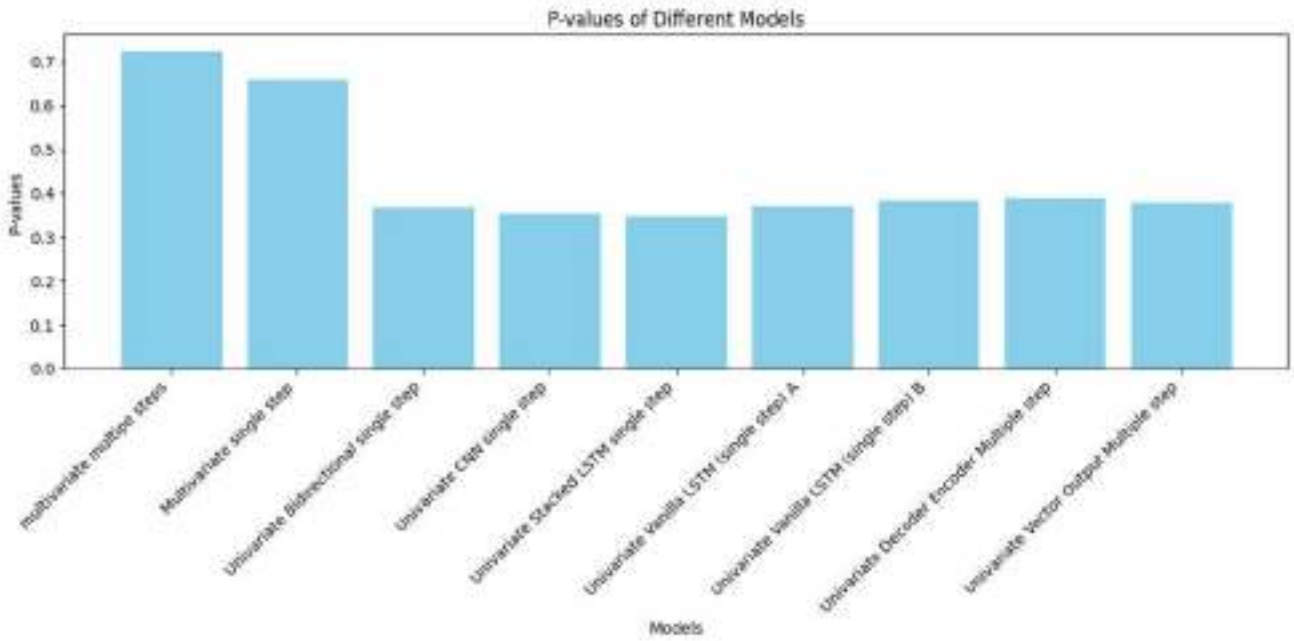


FIGURE 37. ANOVA P-values.

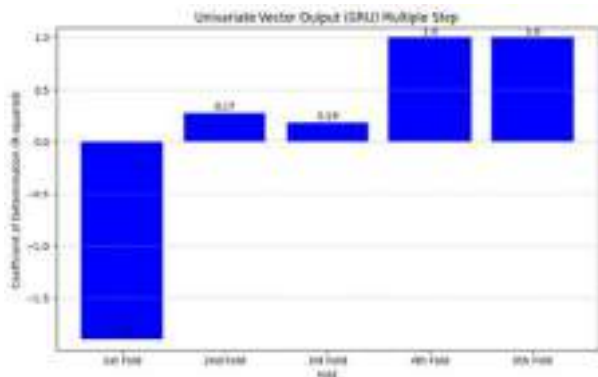


FIGURE 38. 5 folds R-squared cross validation on univariate GRU models.

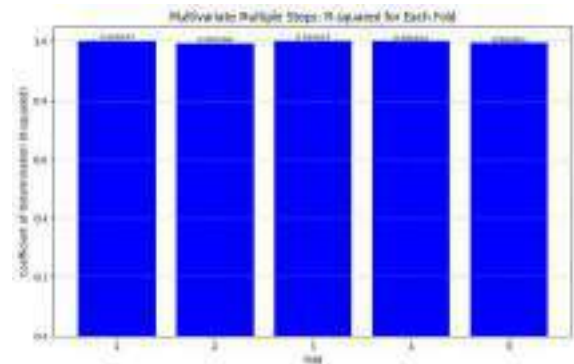


FIGURE 40. 5-folds R-squared cross validation on multivariate multi steps models.

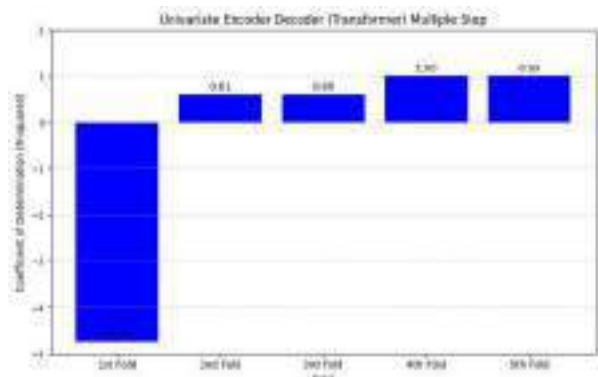


FIGURE 39. 5-folds R-squared cross validation on univariate transformer models.

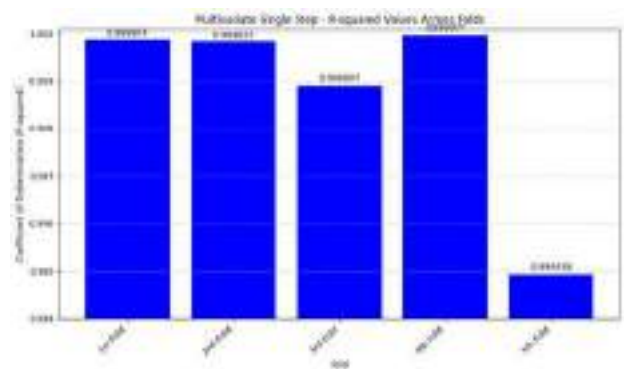


FIGURE 41. 5-folds R-squared cross validation on multivariate single step models.

parameters to capture the average change in a data series over time and  $\epsilon_t$  are the error terms which generally independent.

The experiment in Fig. 43 demonstrates ARIMA modeling on predicting ‘PacketLengthMean’. ARIMA is a traditional

time series model which suitable for univariate time series. The RMSE is calculated to evaluate the model’s accuracy. A lower RMSE indicates better performance. The ARIMA model applies RMSE loss function. Given the RMSE as

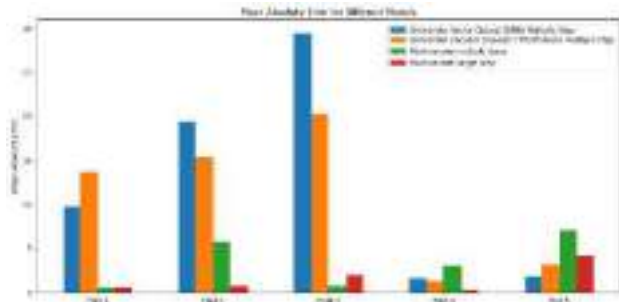


FIGURE 42. 5-folds cross validation MAE evaluations.

shown in equation (9);

$$RMSE = \left( \frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2 \right)^{1/2} \quad (9)$$

Which has a similar principle to find the differences between the true and the predicted values. In this equation, the Square Root and Squared cancel out each other similar to the principle of MAE equation.

The plot shows the actual ‘PacketLengthMean’ values or in the second part in blue and the predicted values in green. Cross validation training involves partitioning the dataset into 80 equal sized subsets. It involves dataset index 0 to 79, 80 to 159, 160 to 239 and index 16741 to 16821. Each subset was trained two folded. The goal is for the green line (predictions) to closely follow the blue line (actual values). Large gaps between the lines indicate poor prediction accuracy. The red line is consistently above or below the blue line, it means the model is systematically over- or under-predicting. Some degree of random fluctuation is expected, but excessive variation suggests the model is not capturing the underlying patterns well. The model’s performance was evaluated using the Root Mean Squared Error (RMSE) test, which resulted in a fluctuated high RMSE score of 6.777, 41.490, 22.811, 19.326, 6.594, 12.585, 22.946, 21.630 indicating low performance.

Also, it is important to note that GRU and Transformer are broadly recognized as State-of-the-art for many sequences prediction task especially in areas like Time Series Forecasting and NLP [96]. GRU models were considered state-of-the-art for many sequence prediction tasks before the widespread adoption of Transformers. They are still very capable and often preferred for certain applications due to their efficiency. Numerous subsequent works have built upon the Transformer, demonstrating its state-of-the-art performance in various domains. For instance, the development of large language models like GPT and BERT are direct descendants and evidence of Transformer’s State-of-the-art capabilities.

We extended our evaluation by training the dataset on an additional model recognized as a state-of-the-art approach: Ensemble Learning. Ensemble learning is a hybrid methodology that integrates multiple individual models to enhance the overall learning process. By combining different learning algorithms, ensemble methods can effectively capture diverse patterns in the data, resulting in more robust and accurate

intrusion detection outcomes. This approach capitalizes on the strengths of various models, as each one may identify unique aspects of potential attacks.

The architecture of the ensemble model follows a structured flow, beginning with the input data, which is then processed through several base models arranged in parallel. These base models, labeled from Base Model 1 to N, consist of different machine learning techniques, including Decision Trees, SVM, Logistic Regression, and Neural Networks. For this specific experiment, we employed two base models: Base Model 1 was a Random Forest Regressor, and Base Model 2 was a Gradient Boosting Regressor. The outputs from these base models, referred to as Prediction 1 to N, represent the individual predictions generated by each model.

Subsequently, these predictions are passed to the ensemble or aggregation layer, where they are combined using a regression-based ensemble method to produce the final prediction. Rather than relying on a single model to achieve optimal performance, the ensemble approach harnesses the collective intelligence of multiple competent models, thereby mitigating individual weaknesses and achieving higher accuracy.

For this evaluation, the ensemble model was applied using univariate input sequences only, although such models are often suitable for both univariate and multivariate sequences. The cross-validation process involved partitioning the dataset into 80 equally sized subsets, using index ranges such as 0–79, 80–159, 160–239, and 16741–16821. Each subset was used in a single fold of the cross-validation training. The following table presents the intrusion prediction performance of the ensemble model under these settings.

Fig. 44 illustrates the performance of the ensemble model in predicting the *PacketLengthMean* feature for the first forecast step. The blue line represents the true, observed values from the test dataset—serving as the ground truth. In contrast, the orange line depicts the values predicted by the ensemble model for the same feature and forecast step.

By visually comparing these two lines, one can evaluate the model’s predictive accuracy. A close alignment between the orange and blue lines indicates high prediction accuracy, whereas noticeable deviations highlight instances where the model underperforms. This comparison provides an intuitive understanding of how effectively the ensemble model captures the underlying patterns in the data.

In addition to the visual representation, the Root Mean Square Error (RMSE) offers a quantitative measure of performance. It is calculated as the square root of the average squared differences between the actual and predicted values, both flattened across the test set. Throughout the experiments, higher RMSE values indicate poorer model performance. The specific RMSE values for different test segments have been reported to reflect this.

### G. COMPUTATIONAL COMPLEXITY ANALYSIS

To complement the performance evaluation of prediction accuracy, a computational complexity analysis was con-

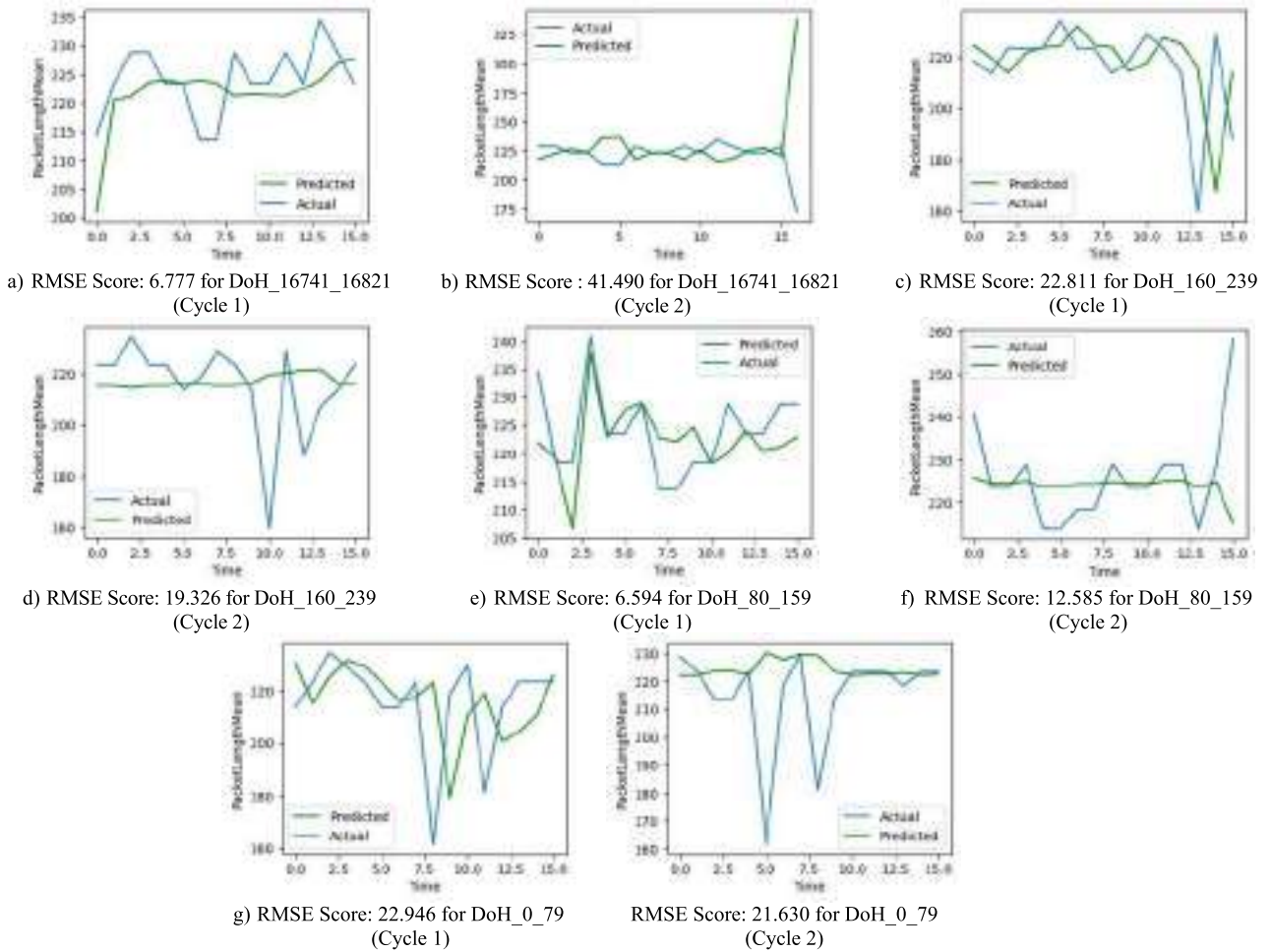


FIGURE 43. Prediction results based on ARIMA model, a traditional time series model.

ducted to assess the efficiency of each model. This analysis focuses on four primary metrics: the number of trainable parameters, average training time per epoch, average inference time per sample, and approximate memory usage during execution. These metrics offer practical insight into the resource demands and scalability of each model, especially when considering real-time applications in healthcare infrastructure.

The analysis was performed on a Google Colab environment equipped with a Tesla T4 GPU, 13 GB of RAM, and 2 vCPUs. Each model was trained and tested using identical data splits and hyperparameters (batch size = 32, epochs = 50, sequence length = 80). The inference time was measured as the average time to predict a single sample across 1000 runs. Table 8 summarizes the findings of computational complexity.

These findings highlight the trade-offs between prediction accuracy and computational efficiency. The CNN model demonstrated the fastest training and inference times with relatively low memory usage, making it suitable for low-latency or resource-constrained environments. Conversely, the Transformer Encoder-Decoder model, although exhibiting superior multistep prediction accuracy, incurred the highest compu-

TABLE 8. Computational complexity.

Model	Trainable Parameters	Training Time (sec/epoch)	Inference Time (ms/sample)	Memory Usage (approx.)
Vanilla LSTM	~50,000	42	0.80	210 MB
Stacked LSTM	~120,000	60	1.10	310 MB
Bidirectional LSTM	~90,000	65	1.30	330 MB
Transformer Encoder-Decoder	~240,000	80	1.90	410 MB
GRU (Vector Output)	~85,000	55	0.90	220 MB
CNN	~60,000	30	0.70	180 MB

tational cost due to its attention mechanisms and larger architecture. Bidirectional and Stacked LSTM models also showed moderate-to-high complexity, while the GRU-based

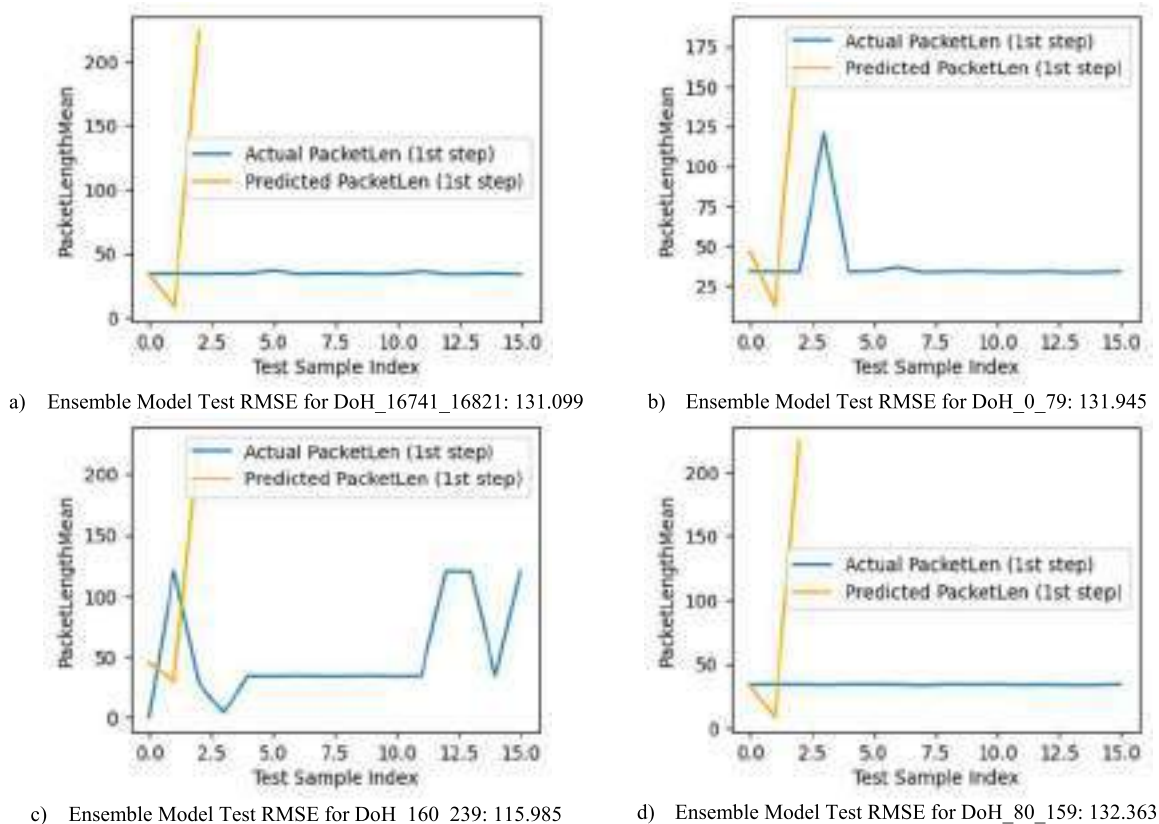


FIGURE 44. Ensemble model: Actual vs. Predicted PacketLengthMean for the first forecast step.

model offered a balanced compromise between performance and efficiency.

This analysis underscores the importance of considering deployment context when selecting predictive models for healthcare intrusion systems. In time-sensitive or embedded environments, lighter models such as CNN or Vanilla LSTM may be preferable, whereas high-resource settings can afford the accuracy gains provided by more complex architectures like the Transformer.

V. DISCUSSION

This study has utilized useful features from comprehensive investigation that uncovered the crucial PC value (from Principle Component Analysis). This useful character to be extensively used in training and extrapolating prediction results on this research. These reflected in a series of generated input sequences, each consisting of a univariate input dimension of (78,6) and single step output of (78,) for Bidirectional, Stacked, CNN and Vanilla LSTM. Additionally, for Transformer Encoder-Decoder and Vectored Output models, the same univariate input dimension of (78,6) is applied but the output was expanded to multiple steps with a size of (78,2). Similarly, a series of multivariate input sequences, each with a dimension of (78,3,2) were generated. These sequences produced both single and multiple step output sequences with dimensions of (78,) and (78,2) respectively. In similar manner, another multivariate

input sequences each with 3 variables and a dimension of (78,3,3) were employed to generate multiple step outputs of size (78,1,3), another attempt to experiment different scenarios. The input and output configurations are visually represented in Figs. 18-22. excessive number of variables for trainings might hinder model performance by obscuring essential information within the dataset leading to poor results. On a similar note, Transformer encoder-decoder architectures, trained on univariate input, have demonstrated strong potential for accurate predictions. It may due to its capability to capture the context and essential information from the input sequence. Subsequently, LSTM model has manifested quite a significant improvement in predicting multivariate input sequences. This can likely be attributed to the architecture’s ability to effectively capture long-term dependencies within sequential data. Indeed, it is apparent that both models performed well in predicting multistep outputs.

Furthermore, based on the provided results from the R-squared, p-values and F-statistics, the multivariate models show some promise in terms of R-squared values. The low R-squared values for the univariate models raise concerns about their suitability for the current data and task. The p-values and F-statistics do not reveal significant differences between the models in this particular test. It’s crucial to remember that a single statistical test doesn’t provide a comprehensive evaluation. To solidify our conclusions, that

is why we have already employed different statistical tests or metrics (such as Mean Absolute Error and Root Mean Squared Error) and conducting a more in-depth investigation of the data and model residuals. To further evaluate the top-performing models (Transformer encoder-decoder, GRU, and two multivariate models), cross-validation was performed. For future research, we plan to develop a fully-fledged intrusion prediction model that incorporates all the aforementioned factors, aiming to enhance its accuracy and effectiveness in real-world scenarios especially in the healthcare sectors.

All in all, to ensure the relevancy of this discussion within the healthcare domain, given the demonstrated better performance of multivariate models in healthcare, it is crucial to recognize the sequential nature and dependencies within medical multi-event (multivariable) data to build robust and accurate predictive systems. Many diseases don't manifest instantly but evolve over time. Medical data reflects this progression through a sequence of symptoms, diagnoses, treatments, and physiological changes. Ignoring this sequence can lead to a static and incomplete understanding of a patient's condition. Worthy to mention that one medical event often influences the likelihood or nature of future events. For example, a patient diagnosed with hypertension is at a higher risk of developing cardiovascular disease later. Treatments administered at one point can affect the patient's response to subsequent treatments. Recognizing these dependencies is vital for accurate prediction and effective management. Predicting the future course of a disease (prognosis) heavily relies on understanding past patterns and dependencies. Analyzing the sequence of events can help identify patients at higher risk of complications, predict treatment outcomes, and estimate disease progression timelines more accurately. Hence, Machine learning models that account for temporal dependencies can capture complex relationships and make more accurate predictions than models that treat data points as independent

Finally acknowledging the inherent sequential nature and interdependencies within medical multi-event (multivariable) network data transcends mere statistical preference. It becomes a fundamental necessity for constructing robust intrusion predictive model within healthcare industry. By leveraging the sequential and dependent nature of healthcare network data unlocks the potential for more sophisticated predictive capabilities. By employing models capable of capturing these dynamics, such as Recurrent Neural Networks (RNNs), including LSTMs and GRUs, Transformer networks, and certain statistical time-series models, we can improve prediction accuracy by learning the complex temporal patterns and dependencies which leads to more accurate predictions of potential intrusions. Then enhance early detection by identifying subtle deviations from learned normal sequential patterns across multiple variables which ultimately enables earlier detection of malicious activity and allowing for timely intervention and mitigation. It also provides richer understanding on the relationships and temporal evolution of

---

**Algorithm 1** Univariate Splitting Function (a,b: Real)
 

---

*Univariate\_Splitting\_Function*

“””

Args:

*a*: The input list or string to split (data array).

*b*: The first delimiter (optional) or input time steps.

“””

$X, y \leftarrow \text{list}(), \text{list}();$

**For**  $i = \text{length}(a)$ :

$\text{end} \leftarrow i+b$ ; ▷ Find end of list

**if**  $\text{end} > \text{length}(a)-1$  **then**

**break**

**end if**

$\text{pattern}_X, \text{pattern}_y \leftarrow a[i:\text{end}], a[\text{end}];$

$X \text{append} (\text{pattern}_X);$

$y \text{append} (\text{pattern}_y);$

**return array**( $X$ ), **array**( $y$ )

---



---

**Algorithm 2** Input Output Configurations (BiLSTM / Stacked-LSTM/ Vanilla LSTM)
 

---

“””

Args:

*a*: The input list or string to split (data array).

*b*: The first delimiter (optional) or input time steps.

“””

$b \leftarrow R$ ; ▷ Any Real number

$X, y \leftarrow \text{Univariate\_Splitting\_Function}(a, b);$

**For**  $i = \text{length}(X)$ :

**Print**( $X[i], y[i]$ );

$X \text{reshape} [\text{samples}, \text{timesteps}, \text{features}];$

---



---

**Algorithm 3** Input Output Configurations (CNN)
 

---

“””

Args:

*a*: The input list or string to split (data array).

*b*: The first delimiter (optional) or input time steps.

“””

$b \leftarrow R$ ; ▷ Any Real number

$X, y \leftarrow \text{Univariate\_Splitting\_Function}(a, b);$

**For**  $i = \text{length}(X)$ :

**Print**( $X[i], y[i]$ );

$X \text{reshape} [\text{samples}, \text{sequences}, \text{timesteps}, \text{features}];$

---

medical data during an intrusion which can offer valuable insights into the attacker's strategies and the potential impact on the healthcare system. Subsequently, this study will help to develop more context-aware security systems by learning what constitutes “normal” and “attack” multivariate sequences in specific healthcare contexts.

## VI. CONCLUSION

This study highlights the growing relevance of predictive modeling in enhancing the security posture of healthcare

---

**Algorithm 4** Univariate Splitting Function Multisteps (a,b, c: Real)

---

*Univariate\_Splitting\_Function\_Multisteps*  
 “”””

Args:

- a: The input list or string to split (data array).
  - b: The first delimiter (optional) or input (X) time steps.
  - c: The second delimiter (optional) or output (y) time steps.
- “”””

X,y ← list(), list();

**For** i = **length**(a):

end ← i+b; ▷ Find end of list

output ← end+c; ▷ Find output timesteps

**if** output > **length**(a) **then**

**break**

**end if**

pattern\_X, pattern\_y ← a[i:end], a[end:output];

**Xappend** (pattern\_X);

**yappend** (pattern\_y);

**return array**(X), **array**(y)

---



---

**Algorithm 5** Input Output Configuration for Multiple Timesteps (Transformer Encoder-Decoder)

---

“”””

Args:

- a: The input list or string to split (data array).
  - b: The first delimiter (optional) or input time steps.
  - c: The second delimiter (optional) or output (y) time steps.
- “”””

b,c ← R; ▷ Any Real number

X,y ← *Univariate\_Splitting\_Function\_Multisteps*(a, b,c);

**For** i = **length**(X):

**Print**(X[i], y[i]);

**Xreshape** [samples, timesteps, features];

**yreshape** [samples, timesteps, features];

---

networks through proactive intrusion forecasting. While IDS remain vital for detecting known, zero-day, and n-day attacks, the application of time series prediction within IDS research is still underexplored compared to other domains such as environmental monitoring, energy systems, and meteorology. Additionally, existing IDS datasets often lack realism and fail to reflect the complexity of modern network threats. Building on prior work identifying a reliable benchmark dataset through Principal Component Analysis, this study evaluated the performance of leading predictive models—LSTM, GRU, and Transformer—under various univariate and multivariate configurations. Results show that multivariate LSTM models consistently outperform others, achieving low error rates and demonstrating strong sensitivity to input sequence design. These findings underscore the importance of capturing temporal dependencies in intrusion data and confirm the scalability and robustness of top-performing models, particularly in noisy healthcare network environments. Future

---

**Algorithm 6** Multivariate Splitting Function (a,b, c: Real)

---

*Multivariate\_Splitting\_Function\_Multisteps*

“”””

Args:

- a: The input list or string to split (data array).
  - b: The first delimiter (optional) or input (X) time steps.
  - c: The second delimiter (optional) or output (y) time steps.
- “”””

X,y ← list(), list();

**For** i = **length**(a):

end ← i+b; ▷ Find end of list

output ← end+c-1; ▷ Find output timesteps

**if** output > **length**(a) **then**

**break**

**end if**

pattern\_X, pattern\_y ← a[i:end,-1], a[end-1:output, -1];

**Xappend** (pattern\_X);

**yappend** (pattern\_y);

**return array**(X), **array**(y)

---



---

**Algorithm 7** Input Output Configuration for Multivariate Multiplesteps

---

“”””

Args:

- a: The input list or string to split (data array).
  - b: The first delimiter (optional) or input (X) time steps.
  - c: The second delimiter (optional) or output (y) time steps.
- “”””

b,c ← R1, R2; ▷ Any Real number (i.e 3,2)

X,y ← *Multivariate\_Splitting\_Function\_Multisteps*(a, b,c);

**For** i = **length**(X):

**Print**(X[i], y[i]);

features ← x.shape [2]

---

research should explore advanced architectures such as Temporal Fusion Transformers (TFT), Informer, and Autoformer to further enhance prediction accuracy in intrusion forecasting for healthcare networks. These models are well-suited for capturing complex temporal dependencies and uncertainty in sequential data. Additionally, real-time deployment with adaptive learning from streaming data presents a promising direction for improving the responsiveness and operational utility of predictive systems. Integrating explainable AI (XAI) techniques will further support interpretability and decision-making by cybersecurity professionals. Expanding the scope of datasets to include emerging network environments—such as 5G and IoT-based medical traffic—will also strengthen model generalizability and robustness across a wider range of healthcare scenarios. Furthermore, recent advancements in attention-based architectures and interactive memory learning have shown significant promise in handling complex data processing tasks, particularly in image super-resolution [97], [98], [99], [100]. Techniques such as expectation-maximization attention mechanisms and

interactive memory models offer valuable insights and could be adapted to enhance temporal sequence modeling in intrusion prediction, especially when extended to multimodal or graph-based traffic data.

**DECLARATION OF COMPETING INTEREST**

The authors confirm that there are no known conflicts of interest in this work.

**DECLARATION OF GENERATIVE AI AND AI-ASSISTED TECHNOLOGIES IN THE WRITING PROCESS**

During the preparation of this work, the author(s) utilized Gemini to enhance the grammar and structural clarity of certain paragraphs. However, the tool was not used to generate any part of the methodology or results. Following the use of this AI-assisted tool, they carefully reviewed and edited the content to ensure its accuracy and integrity. They bear full responsibility for the content of this publication.

**APPENDICES**

**A. APPENDIX I: ALGORITHMS**

See Algorithms 1–7.

**B. APPENDIX II: NOMENCLATURE**

Symbol/Abbreviation	Description	Unit
X	Input sequence used for prediction (univariate or multivariate)	–
y	Predicted output value(s)	–
t	Time step in a sequence	seconds (s)
a	Sample size or data array length (dimensionless)	count
b	Number of timesteps in the input sequence	steps
c	Number of timesteps in the output sequence (multistep output)	steps
n	Number of features in multivariate input (dimensionless)	count
MAE	Mean Absolute Error prediction target	% or unit of
RMSE	Root Mean Square Error prediction target	% or unit of
R2R <sup>2</sup>	Coefficient of determination	–

ANOVA	Analysis of Variance (statistical significance test)	–
CNN	Convolutional Neural Network	–
LSTM	Long Short-Term Memory	–
GRU	Gated Recurrent Unit	–
BiLSTM	Bidirectional Long Short-Term Memory	–
Transformer	Transformer Encoder-Decoder architecture	–
PCA	Principal Component Analysis	–
DoH	DNS over HTTPS	–
CIRA-CIC-DoHBrw-2020	Benchmark dataset containing benign and malicious DNS-over-HTTPS traffic	–
n <sub>seq</sub>	Number of sequences for reshaping data in CNN models	count
TPU	Tensor Processing Unit (hardware for acceleration)	–
HBM	High Bandwidth Memory used in TPU environments	GB
μ, σ	Mean and standard deviation (in statistical analysis)	Same as feature unit

**ACKNOWLEDGMENT**

The authors wish to thank the College of Computing, Informatics, and Mathematics (CCIM) which is formerly known as the Faculty of Computer and Mathematical Sciences (FSKM) and the Research Management Centre (RMC) of Universiti Teknologi MARA (UiTM) for their assistance and support throughout this research.

**REFERENCES**

- [1] M. H. M. Yusof, A. A. Almohammed, V. Shepelev, and O. Ahmed, “Visualizing realistic benchmarked IDS dataset: CIRA-CIC-DoHBrw-2020,” *IEEE Access*, vol. 10, pp. 94624–94642, 2022, doi: 10.1109/ACCESS.2022.3204690.
- [2] J. Grimshaw. *A Guide To Knowledge Synthesis, Canadian Institutes of Health Research*. Accessed: Nov. 30, 2024. [Online]. Available: <https://www.cihr-irsc.gc.ca/e/41382.html>
- [3] G. Zheng, W. K. Chai, J. Zhang, and V. Katos, “VDGCNeT: A novel network-wide virtual dynamic graph convolution neural network and transformer-based traffic prediction model,” *Knowl.-Based Syst.*, vol. 275, Sep. 2023, Art. no. 110676, doi: 10.1016/j.knsys.2023.110676.

- [4] Y. Sun, L. Hou, Z. Lv, and D. Peng, "Informer-based intrusion detection method for network attack of integrated energy system," *IEEE J. Radio Freq. Identificat.*, vol. 6, no. 2, pp. 748–752, May 2022, doi: [10.1109/JRFID.2022.3215599](https://doi.org/10.1109/JRFID.2022.3215599).
- [5] J. Hong, Y. Yan, E. E. Kuruoglu, and W. K. Chan, "Multivariate time series forecasting with GARCH models on graphs," *IEEE Trans. Signal Inf. Process. over Netw.*, vol. 9, no. 3, pp. 557–568, Jul. 2023, doi: [10.1109/TSIPN.2023.3304142](https://doi.org/10.1109/TSIPN.2023.3304142).
- [6] A. A. Garibo-Morante and F. Ornelas Tellez, "Univariate and multivariate time series modeling using a harmonic decomposition methodology," *IEEE Latin Amer. Trans.*, vol. 20, no. 3, pp. 372–378, Mar. 2022, doi: [10.1109/TLA.2022.9667134](https://doi.org/10.1109/TLA.2022.9667134).
- [7] R. Chuentawat and Y. Kan-ngan, "The comparison of PM<sub>2.5</sub> forecasting methods in the form of multivariate and univariate time series based on support vector machine and genetic algorithm," in *Proc. 15th Int. Conf. Electr. Eng./Electron., Comput., Telecommun. Inf. Technol. (ECTI-CON)*, Chiang Rai, Thailand, Jul. 2018, pp. 572–575, doi: [10.1109/ECTICon.2018.8619867](https://doi.org/10.1109/ECTICon.2018.8619867).
- [8] Y. Yu, X. Zeng, X. Xue, and J. Ma, "LSTM-based intrusion detection system for VANETs: A time series classification approach to false message detection," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 12, pp. 23906–23918, Dec. 2022, doi: [10.1109/TITS.2022.3190432](https://doi.org/10.1109/TITS.2022.3190432).
- [9] P. Mansourian, N. Zhang, A. Jaekel, and M. Kneppers, "Deep learning-based anomaly detection for connected autonomous vehicles using spatiotemporal information," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 12, pp. 16006–16017, Dec. 2023, doi: [10.1109/TITS.2023.3286611](https://doi.org/10.1109/TITS.2023.3286611).
- [10] M. Landauer, F. Skopik, B. Stojanović, A. Flatscher, and T. Ullrich, "A review of time-series analysis for cyber security analytics: From intrusion detection to attack prediction," *Int. J. Inf. Secur.*, vol. 24, no. 1, p. 3, Jan. 2025.
- [11] S. M. Mahasree, M. Kavin, F. Kavin, and N. Bharathiraja, "Solar energy forecasting with performance optimization using machine learning techniques," *Electr. Power Compon. Syst.*, vol. 2024, pp. 1–13, Feb. 2024, doi: [10.1080/15325008.2024.2316245](https://doi.org/10.1080/15325008.2024.2316245).
- [12] Y. Liu, C. Zhao, and Y. Huang, "A combined model for multivariate time series forecasting based on MLP-feedforward attention-LSTM," *IEEE Access*, vol. 10, pp. 88644–88654, 2022, doi: [10.1109/ACCESS.2022.3192430](https://doi.org/10.1109/ACCESS.2022.3192430).
- [13] Y. Wang and L. Sun, "Energy-efficient dynamic sensor time series classification for edge health devices," *Comput. Methods Programs Biomed.*, vol. 254, Sep. 2024, Art. no. 108268, doi: [10.1016/j.cmpb.2024.108268](https://doi.org/10.1016/j.cmpb.2024.108268).
- [14] L. Ji, Z. Wei, J. Hao, and C. Wang, "An intelligent diagnostic method of ECG signal based on Markov transition field and a ResNet," *Comput. Methods Programs Biomed.*, vol. 242, Dec. 2023, Art. no. 107784.
- [15] A. Çalhan, M. Cicioğlu, and A. Ceylan, "EHealth monitoring testbed with fuzzy based early warning score system," *Comput. Methods Programs Biomed.*, vol. 202, Apr. 2021, Art. no. 106008.
- [16] M. Abbas, D. Somme, and R. Le Bouquin Jeannès, "D-SORM: A digital solution for remote monitoring based on the attitude of wearable devices," *Comput. Methods Programs Biomed.*, vol. 208, Sep. 2021, Art. no. 106247.
- [17] L. Zhang, W. Zhang, M. J. McNeil, N. Chengwang, D. S. Matteson, and P. Bogdanov, "AURORA: A unified framework for anomaly detection on multivariate time series," *Data Mining Knowl. Discovery*, vol. 35, no. 5, pp. 1882–1905, Sep. 2021, doi: [10.1007/s10618-021-00771-7](https://doi.org/10.1007/s10618-021-00771-7).
- [18] H. Salemi, H. Rostami, S. Talatian-Azad, and M. R. Khosravi, "LEAESN: Predicting DDoS attack in healthcare systems based on Lyapunov exponent analysis and echo state neural networks," *Multimedia Tools Appl.*, vol. 81, no. 29, pp. 41455–41476, Dec. 2022, doi: [10.1007/s11042-020-10179-y](https://doi.org/10.1007/s11042-020-10179-y).
- [19] M. M. Althobaiti, K. P. M. Kumar, D. Gupta, S. Kumar, and R. F. Mansour, "An intelligent cognitive computing based intrusion detection for industrial cyber-physical systems," *Measurement*, vol. 186, Dec. 2021, Art. no. 110145, doi: [10.1016/j.measurement.2021.110145](https://doi.org/10.1016/j.measurement.2021.110145).
- [20] A. A. Malibari, S. S. Alotaibi, R. Alshahrani, S. Dhabbi, R. Alabdian, F. N. Al-Wesabi, and A. M. Hilal, "A novel metaheuristics with deep learning enabled intrusion detection system for secured smart environment," *Sustain. Energy Technol. Assessments*, vol. 52, Aug. 2022, Art. no. 102312, doi: [10.1016/j.seta.2022.102312](https://doi.org/10.1016/j.seta.2022.102312).
- [21] Z. Hajhashemi and M. Popescu, "A multidimensional time-series similarity measure with applications to eldercare monitoring," *IEEE J. Biomed. Health Informat.*, vol. 20, no. 3, pp. 953–962, May 2016, doi: [10.1109/JBHI.2015.2424711](https://doi.org/10.1109/JBHI.2015.2424711).
- [22] J. Xie, H. Wang, J. M. Garibaldi, and D. Wu, "Network intrusion detection based on dynamic intuitionistic fuzzy sets," *IEEE Trans. Fuzzy Syst.*, vol. 30, no. 9, pp. 3460–3472, Sep. 2022, doi: [10.1109/TFUZZ.2021.3117441](https://doi.org/10.1109/TFUZZ.2021.3117441).
- [23] M. Hafiz Mohd Yusof, A. Mohd Zin, and N. Safie Mohd Satar, "Behavioral intrusion prediction model on Bayesian network over healthcare infrastructure," *Comput., Mater. Continua*, vol. 72, no. 2, pp. 2445–2466, Jan. 2022, doi: [10.32604/cmc.2022.023571](https://doi.org/10.32604/cmc.2022.023571).
- [24] S. Silvestri, S. Islam, D. Amelin, G. Weiler, S. Papastergiou, and M. Ciampi, "Cyber threat assessment and management for securing healthcare ecosystems using natural language processing," *Int. J. Inf. Secur.*, vol. 23, no. 1, pp. 31–50, Oct. 2023, doi: [10.1007/s10207-023-00769-w](https://doi.org/10.1007/s10207-023-00769-w).
- [25] T. Chakrabarty, Y. Choi, and V. Shwartz, "It's not rocket science: Interpreting figurative language in narratives," *Trans. Assoc. Comput. Linguistics*, vol. 10, pp. 589–606, Jan. 2022, doi: [10.1162/tacl\\_a\\_00478](https://doi.org/10.1162/tacl_a_00478).
- [26] K. J. Raval, N. K. Jadav, T. Rathod, S. Tanwar, V. Vimal, and N. Yamsani, "A survey on safeguarding critical infrastructures: Attacks, AI security, and future directions," *Int. J. Crit. Infrastruct. Protection*, vol. 44, Mar. 2024, Art. no. 100647, doi: [10.1016/j.ijcip.2023.100647](https://doi.org/10.1016/j.ijcip.2023.100647).
- [27] S. Kang, S. Ros, I. Song, P. Tam, S. Math, and S. Kim, "Real-time prediction algorithm for intelligent edge networks with federated learning-based modeling," *Comput., Mater. Continua*, vol. 77, no. 2, pp. 1967–1983, Jan. 2023, doi: [10.32604/cmc.2023.045020](https://doi.org/10.32604/cmc.2023.045020).
- [28] M. Sarkar, T.-H. Lee, and P. K. Sahoo, "Smart healthcare: Exploring the Internet of Medical Things with ambient intelligence," *Electronics*, vol. 13, no. 12, p. 2309, Jun. 2024, doi: [10.3390/electronics13122309](https://doi.org/10.3390/electronics13122309).
- [29] D. K. Panagiotou and A. I. Dounis, "An ANFIS-fuzzy tree-GA model for a hospital's electricity purchasing decision-making process integrated with virtual cost concept," *Sustainability*, vol. 15, no. 10, p. 8419, May 2023, doi: [10.3390/su15108419](https://doi.org/10.3390/su15108419).
- [30] S. Shapsough and I. Zuakernan, "An IoT-based services infrastructure for utility-scale distributed solar farms," *Energies*, vol. 15, no. 2, p. 440, Jan. 2022, doi: [10.3390/en15020440](https://doi.org/10.3390/en15020440).
- [31] H. Mazumdar, C. Chakraborty, S. B. Venkatakrishnan, A. Kaushik, and H. A. Gohel, "Quantum-inspired heuristic algorithm for secure healthcare prediction using blockchain technology," *IEEE J. Biomed. Health Informat.*, vol. 28, no. 6, pp. 3371–3378, Jun. 2024, doi: [10.1109/JBHI.2023.3304326](https://doi.org/10.1109/JBHI.2023.3304326).
- [32] T. D. Ladd, F. Jelezko, R. Laflamme, Y. Nakamura, C. Monroe, and J. L. O'Brien, "Quantum computers," *Nature*, vol. 464, no. 7285, pp. 45–53, 2010.
- [33] S. Aaronson, *Quantum Computing Since Democritus*. Cambridge, U.K.: Cambridge Univ. Press, 2013.
- [34] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*. Cambridge, U.K.: Cambridge Univ. Press, 2010.
- [35] A. Lohachab and K. Kumar, "FedHFP: A federated deep learning framework for heart failure prediction," *IETE J. Res.*, vol. 71, no. 2, pp. 1–13, Nov. 2024, doi: [10.1080/03772063.2024.2428741](https://doi.org/10.1080/03772063.2024.2428741).
- [36] Y. Jiang, Z. Li, W. Jiang, T. Wei, and B. Chen, "Risk prediction model for postoperative pneumonia in esophageal cancer patients: A systematic review," *Frontiers Oncol.*, vol. 14, Aug. 2024, Art. no. 1419633, doi: [10.3389/fonc.2024.1419633](https://doi.org/10.3389/fonc.2024.1419633).
- [37] L. Kong, G. Li, W. Rafique, S. Shen, Q. He, M. R. Khosravi, R. Wang, and L. Qi, "Time-aware missing healthcare data prediction based on ARIMA model," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 21, no. 4, pp. 1042–1050, Jul. 2024, doi: [10.1109/TCBB.2022.3205064](https://doi.org/10.1109/TCBB.2022.3205064).
- [38] S. Niu, Q. Yin, J. Ma, Y. Song, Y. Xu, L. Bai, W. Pan, and X. Yang, "Enhancing healthcare decision support through explainable AI models for risk prediction," *Decis. Support Syst.*, vol. 181, Jun. 2024, Art. no. 114228, doi: [10.1016/j.dss.2024.114228](https://doi.org/10.1016/j.dss.2024.114228).
- [39] A. Davoudi, S. Chae, L. Evans, S. Sridharan, J. Song, K. H. Bowles, M. V. McDonald, and M. Topaz, "Fairness gaps in machine learning models for hospitalization and emergency department visit risk prediction in home healthcare patients with heart failure," *Int. J. Med. Informat.*, vol. 191, Nov. 2024, Art. no. 105534, doi: [10.1016/j.ijmedinf.2024.105534](https://doi.org/10.1016/j.ijmedinf.2024.105534).

- [40] A. M. Merchant, N. Shenoy, S. Lanka, and S. Kamath, "Ensemble neural models for ICD code prediction using unstructured and structured healthcare data," *Heliyon*, vol. 10, no. 17, Sep. 2024, Art. no. e36569, doi: [10.1016/j.heliyon.2024.e36569](https://doi.org/10.1016/j.heliyon.2024.e36569).
- [41] S. Yadav, H. Sehrawat, V. Jaglan, Y. Singh, S. Dalal, and D.-N. Le, "Developing model-agnostic meta-learning enabled lightbpm model asthma level prediction in smart healthcare modeling," *Scalable Computing: Pract. Exper.*, vol. 25, no. 6, pp. 4872–4885, Oct. 2024, doi: [10.12694/scpe.v25i6.3369](https://doi.org/10.12694/scpe.v25i6.3369).
- [42] D. Stahl, "New horizons in prediction modelling using machine learning in older people's healthcare research," *Age Ageing*, vol. 53, no. 9, Sep. 2024, Art. no. afae201, doi: [10.1093/ageing/afae201](https://doi.org/10.1093/ageing/afae201).
- [43] Z. Li, J. Li, C. Zhu, and S. Jiao, "Risk factors and nomogram prediction model for healthcare-associated infections (HAIs) in COVID-19 patients," *Infection Drug Resistance*, vol. 17, pp. 3309–3323, Aug. 2024, doi: [10.2147/idr.s472387](https://doi.org/10.2147/idr.s472387).
- [44] Y. Zhou, S. Zhang, C. Li, M. Du, and L. Ding, "Dynamic resilience assessment for urban healthcare infrastructure operations considering the spatiotemporal characteristics of public health emergencies," *J. Manage. Eng.*, vol. 40, no. 6, Nov. 2024, Art. no. 04024071, doi: [10.1061/jmenea.meeng-6176](https://doi.org/10.1061/jmenea.meeng-6176).
- [45] M. P. Suprenant, A. Gopaluni, M. K. Dyson, N. Al-Dheeb, F. Shafique, and M. H. Zaman, "A predictive model for healthcare coverage in Yemen," in *Proc. Conflict Health*, Yemen, Aug. 2020, vol. 14, no. 1, pp. 1–16, doi: [10.1186/s13031-020-00300-1](https://doi.org/10.1186/s13031-020-00300-1).
- [46] T. Ge, C.-Y. Chen, Y. Ni, Y.-C.-A. Feng, and J. W. Smoller, "Polygenic prediction via Bayesian regression and continuous shrinkage priors," *Nature Commun.*, vol. 10, no. 1, p. 1776, Apr. 2019, doi: [10.1038/s41467-019-09718-5](https://doi.org/10.1038/s41467-019-09718-5).
- [47] A. B. Queyam, R. Kumar, R. K. Ratnesh, and R. K. Chauhan, "LabVIEW-enabled synthetic signal for empowering fetal-maternal healthcare," *ECS J. Solid State Sci. Technol.*, vol. 13, no. 5, May 2024, Art. no. 057005, doi: [10.1149/2162-8777/ad4dde](https://doi.org/10.1149/2162-8777/ad4dde).
- [48] Z. N. Al-Kateeb and D. B. Abdullah, "AdaBoost-powered cloud of things framework for low-latency, energy-efficient chronic kidney disease prediction," *Trans. Emerg. Telecommun. Technol.*, vol. 35, no. 6, p. e5007, Jun. 2024, doi: [10.1002/ett.5007](https://doi.org/10.1002/ett.5007).
- [49] T. M. Deist et al., "Distributed learning on 20 000+ lung cancer patients—The personal health train," *Radiotherapy Oncol.*, vol. 144, pp. 189–200, Mar. 2020, doi: [10.1016/j.radonc.2019.11.019](https://doi.org/10.1016/j.radonc.2019.11.019).
- [50] K. Rahmani, R. Thapa, P. Tsou, S. C. Chetty, G. Barnes, C. Lam, and C. F. Tso, "Assessing the effects of data drift on the performance of machine learning models used in clinical sepsis prediction," *Int. J. Med. Informat.*, vol. 173, May 2023, Art. no. 104930, doi: [10.1016/j.ijmedinf.2022.104930](https://doi.org/10.1016/j.ijmedinf.2022.104930).
- [51] T. R. Jossou, Z. Tahori, G. Houdji, D. Medenou, A. Lasfar, F. Sanya, M. H. Ahouandjinou, S. M. Pagliara, M. S. Haleem, and A. Et-Tahir, "N-beats as an EHG signal forecasting method for labour prediction in full term pregnancy," *Electronics*, vol. 11, no. 22, p. 3739, Nov. 2022, doi: [10.3390/electronics11223739](https://doi.org/10.3390/electronics11223739).
- [52] T. Shiwani, S. Relton, R. Evans, A. Kale, A. Heaven, A. Clegg, and O. Todd, "New horizons in artificial intelligence in the healthcare of older people," *Age Ageing*, vol. 52, no. 12, Dec. 2023, Art. no. afad219, doi: [10.1093/ageing/afad219](https://doi.org/10.1093/ageing/afad219).
- [53] D. Giardiello et al., "PredictCBC-2.0: A contralateral breast cancer risk prediction model developed and validated in 200,000 patients," *Breast Cancer Res.*, vol. 24, no. 1, p. 69, Oct. 2022, doi: [10.1186/s13058-022-01567-3](https://doi.org/10.1186/s13058-022-01567-3).
- [54] B. S. Bama and Y. B. Jinila, "Sensor-based gait analysis for Parkinson's disease prediction," *Intell. Autom. Soft Comput.*, vol. 36, no. 2, pp. 2085–2097, Jan. 2023, doi: [10.32604/iasec.2023.028481](https://doi.org/10.32604/iasec.2023.028481).
- [55] G. Koutittas, K. Nolen, S. Attal, A. Ventouris, Y. Dolev, and H. T. Van Den Broek, "Technical feasibility of implementing and commercializing a machine learning model for rare disease prediction," *IEEE Access*, vol. 11, pp. 84430–84439, 2023, doi: [10.1109/ACCESS.2023.3299866](https://doi.org/10.1109/ACCESS.2023.3299866).
- [56] K. Henares, J. L. Risco-Martín, J. L. Ayala, and R. Hermida, "Efficient micro data centres deployment for mobile healthcare monitoring systems in IoT urban scenarios," *J. Simul.*, vol. 16, no. 6, pp. 589–603, May 2022, doi: [10.1080/17477778.2022.2072782](https://doi.org/10.1080/17477778.2022.2072782).
- [57] S. El-Azab and P. Nong, "Clinical algorithms, racism, and 'fairness' in healthcare: A case of bounded justice *Big Data Soc.*, vol. 10, no. 2, Jul. 2023, Art. no. 20539517231213820, doi: [10.1177/20539517231213820](https://doi.org/10.1177/20539517231213820).
- [58] A. S. Bhadouria and R. K. Singh, "Machine learning model for healthcare investments predicting the length of stay in a hospital & mortality rate," *Multimedia Tools Appl.*, vol. 83, no. 9, pp. 27121–27191, Aug. 2023, doi: [10.1007/s11042-023-16474-8](https://doi.org/10.1007/s11042-023-16474-8).
- [59] R. Huang, J. Liu, T. K. Wan, D. Siriwan, Y. M. P. Woo, A. Vodencarevic, C. W. Wong, and K. H. K. Chan, "Stroke mortality prediction based on ensemble learning and the combination of structured and textual data," *Comput. Biol. Med.*, vol. 155, Mar. 2023, Art. no. 106176, doi: [10.1016/j.compbiomed.2022.106176](https://doi.org/10.1016/j.compbiomed.2022.106176).
- [60] M. Addotey-Delove, R. E. Scott, and M. Mars, "A healthcare workers' mHealth adoption instrument for the developing world," *BMC Health Services Res.*, vol. 22, no. 1, p. 1225, Oct. 2022, doi: [10.1186/s12913-022-08592-0](https://doi.org/10.1186/s12913-022-08592-0).
- [61] N. Yonat, S. Isaac, and I. M. Shohet, "Complex infrastructure systems analysis and management: The theory of faults," *Smart Sustain. Built Environ.*, vol. 14, no. 3, pp. 599–624, Oct. 2023, doi: [10.1108/sasbe-07-2023-0167](https://doi.org/10.1108/sasbe-07-2023-0167).
- [62] S. Beborita, S. S. Tripathy, S. Basheer, and C. L. Chowdhary, "DeepMist: Toward deep learning assisted mist computing framework for managing healthcare big data," *IEEE Access*, vol. 11, pp. 42485–42496, 2023, doi: [10.1109/ACCESS.2023.3266374](https://doi.org/10.1109/ACCESS.2023.3266374).
- [63] N. Christodoulou, N. Tousert, E. Georgiadi, K. Argyri, F. Misichroni, and G. Stamatakos, "A modular repository-based infrastructure for simulation model storage and execution support in the context of in silico oncology and in silico medicine," *Cancer Informat.*, vol. 15, Jan. 2016, doi: [10.4137/cin.s40189](https://doi.org/10.4137/cin.s40189).
- [64] J. Sato, N. Mitsutake, M. Kitsuregawa, T. Ishikawa, and K. Goda, "Predicting demand for long-term care using Japanese healthcare insurance claims data," *Environ. Health Preventive Med.*, vol. 27, p. 42, Jan. 2022, doi: [10.1265/ehpm.22-00084](https://doi.org/10.1265/ehpm.22-00084).
- [65] J. A. Cooper, R. Ryan, N. Parsons, C. Stinton, T. Marshall, and S. Taylor-Phillips, "The use of electronic healthcare records for colorectal cancer screening referral decisions and risk prediction model development," *BMC Gastroenterol.*, vol. 20, no. 1, p. 78, Mar. 2020, doi: [10.1186/s12876-020-01206-1](https://doi.org/10.1186/s12876-020-01206-1).
- [66] D. Kumar, "India's rural healthcare systems: Structural modeling," *Int. J. Health Care Quality Assurance*, vol. 31, no. 7, pp. 757–774, Aug. 2018, doi: [10.1108/ijhcqa-02-2017-0020](https://doi.org/10.1108/ijhcqa-02-2017-0020).
- [67] M. Seok, W. Kim, and J. Kim, "Machine learning for sarcopenia prediction in the elderly using socioeconomic, infrastructure, and quality-of-life data," *Healthcare*, vol. 11, no. 21, p. 2881, Nov. 2023, doi: [10.3390/healthcare11212881](https://doi.org/10.3390/healthcare11212881).
- [68] D. Wilson, F. Tweedie, J. Rumball-Smith, K. Ross, A. Kazemi, V. Galvin, G. Dobbie, T. Dare, P. Brown, and J. Blakey, "Lessons learned from developing a COVID-19 algorithm governance framework in aotearoa New Zealand," *J. Roy. Soc. New Zealand*, vol. 53, no. 1, pp. 82–94, Sep. 2022, doi: [10.1080/03036758.2022.2121290](https://doi.org/10.1080/03036758.2022.2121290).
- [69] S. Dash, C. Chakraborty, S. K. Giri, and S. K. Pani, "Intelligent computing on time-series data analysis and prediction of COVID-19 pandemics," *Pattern Recognit. Lett.*, vol. 151, pp. 69–75, Nov. 2021, doi: [10.1016/j.patrec.2021.07.027](https://doi.org/10.1016/j.patrec.2021.07.027).
- [70] P. H. T. Schimit, "A model based on cellular automata to estimate the social isolation impact on COVID-19 spreading in Brazil," *Comput. Methods Programs Biomed.*, vol. 200, Mar. 2021, Art. no. 105832, doi: [10.1016/j.cmpb.2020.105832](https://doi.org/10.1016/j.cmpb.2020.105832).
- [71] H. Shah, S. Shah, S. Tanwar, R. Gupta, and N. Kumar, "Fusion of AI techniques to tackle COVID-19 pandemic: Models, incidence rates, and future trends," *Multimedia Syst.*, vol. 28, no. 4, pp. 1189–1222, Jul. 2021, doi: [10.1007/s00530-021-00818-1](https://doi.org/10.1007/s00530-021-00818-1).
- [72] V. Akbarzadeh, G. R. Mumtaz, S. F. Awad, H. A. Weiss, and L. J. Abu-Raddad, "HCV prevalence can predict HIV epidemic potential among people who inject drugs: Mathematical modeling analysis," *BMC Public Health*, vol. 16, no. 1, p. 1216, Dec. 2016, doi: [10.1186/s12889-016-3887-y](https://doi.org/10.1186/s12889-016-3887-y).
- [73] S. Alle et al., "COVID-19 risk stratification and mortality prediction in hospitalized Indian patients: Harnessing clinical data for public health benefits," *PLoS ONE*, vol. 17, no. 3, Mar. 2022, Art. no. e0264785, doi: [10.1371/journal.pone.0264785](https://doi.org/10.1371/journal.pone.0264785).
- [74] S. S. Johnston, J. M. Morton, I. Kalsekar, E. M. Ammann, C.-W. Hsiao, and J. Repts, "Using machine learning applied to real-world healthcare data for predictive analytics: An applied example in bariatric surgery," *Value Health*, vol. 22, no. 5, pp. 580–586, May 2019, doi: [10.1016/j.jval.2019.01.011](https://doi.org/10.1016/j.jval.2019.01.011).

- [75] K. Chadaga, S. Prabhu, N. Sampathila, R. Chadaga, S. Umakanth, D. Bhat, and G. S. S. Kumar, "Explainable artificial intelligence approaches for COVID-19 prognosis prediction using clinical markers," *Sci. Rep.*, vol. 14, no. 1, p. 1783, Jan. 2024, doi: [10.1038/s41598-024-52428-2](https://doi.org/10.1038/s41598-024-52428-2).
- [76] M. V. Evans, A. Garchitorena, R. J. L. Rakotonanahary, J. M. Drake, B. Andriamihaja, E. Rajaonarifara, C. N. Ngonghala, B. Roche, M. H. Bonds, and J. Rakotonirina, "Reconciling model predictions with low reported cases of COVID-19 in sub-Saharan Africa: Insights from Madagascar," *Global Health Action*, vol. 13, no. 1, Oct. 2020, Art. no. 1816044, doi: [10.1080/16549716.2020.1816044](https://doi.org/10.1080/16549716.2020.1816044).
- [77] J. Persis, "Perceived healthcare quality via digital health platforms—evidence from Indian hospitals," *Int. J. Quality Rel. Manage.*, vol. 42, no. 4, pp. 1081–1109, Aug. 2024, doi: [10.1108/ijqrm-11-2023-0363](https://doi.org/10.1108/ijqrm-11-2023-0363).
- [78] C. Iwendi, A. K. Bashir, A. Peshkar, R. Sujatha, J. M. Chatterjee, S. Pasupuleti, R. Mishra, S. Pillai, and O. Jo, "COVID-19 patient health prediction using boosted random forest algorithm," *Frontiers Public Health*, vol. 8, p. 357, Jul. 2020, doi: [10.3389/fpubh.2020.00357](https://doi.org/10.3389/fpubh.2020.00357).
- [79] A. Alhazmi, A. Alferidi, Y. A. Almutawif, H. Makhdoom, H. M. Albasri, and B. S. Sami, "Artificial intelligence in healthcare: Combining deep learning and Bayesian optimization to forecast COVID-19 confirmed cases," *Frontiers Artif. Intell.*, vol. 6, Jan. 2024, Art. no. 1327355, doi: [10.3389/frai.2023.1327355](https://doi.org/10.3389/frai.2023.1327355).
- [80] L. Sheets, G. F. Petroski, Y. Zhuang, M. A. Phinney, B. Ge, J. C. Parker, and C.-R. Shyu, "Combining contrast mining with logistic regression to predict healthcare utilization in a managed care population," *Appl. Clin. Inform.*, vol. 8, no. 2, pp. 430–446, Apr. 2017, doi: [10.4338/aci-2016-05-ra-0078](https://doi.org/10.4338/aci-2016-05-ra-0078).
- [81] G. Lăzăroi, T. Gedeon, E. Rogalska, M. Andronie, K. F. Michalikova, Z. Musova, M. Iatagan, C. Ută, L. Michalkova, M. Kovacova, R. Ștefănescu, I. Hurloiu, S. Zaboşnik, R. Stefko, A. Dişmărescu, I. Dişmărescu, and M. Geamănu, "The economics of deep and machine learning-based algorithms for COVID-19 prediction, detection, and diagnosis shaping the organizational management of hospitals," *Oeconomia Copernicana*, vol. 15, no. 1, pp. 27–58, Mar. 2024, doi: [10.24136/oc.2984](https://doi.org/10.24136/oc.2984).
- [82] P. V. Rajkumar and R. Sandhu, "Safety decidability for pre-authorization usage control with identifier attribute domains," *IEEE Trans. Dependable Secure Comput.*, vol. 17, no. 3, pp. 465–478, Mar. 2018.
- [83] A. Cutler, D. R. Cutler, and J. R. Stevens, "Random forests," *Mach. Learn.*, pp. 157–175, 2012.
- [84] N. Kirilov and E. Bischoff, "Networking aspects of the electronic health records: Hypertext transfer protocol version 2 (HTTP/2) vs HTTP/3," *J. Med. Syst.*, vol. 48, no. 1, p. 61, Jun. 2024, doi: [10.1007/s10916-024-02080-0](https://doi.org/10.1007/s10916-024-02080-0).
- [85] N. Uremović, M. Bizjak, P. Sukić, G. Čtumberger, B. Čalik, and N. Lukać, "A new framework for multivariate time series forecasting in energy management system," *IEEE Trans. Smart Grid*, vol. 14, no. 4, pp. 2934–2947, Jul. 2022.
- [86] A. Chatterjee, M. W. Gerdes, P. Khatiwada, and A. Prinz, "SFTSDH: Applying spring security framework with TSD-based OAuth2 to protect microservice architecture APIs," *IEEE Access*, vol. 10, pp. 41914–41934, 2022, doi: [10.1109/ACCESS.2022.3165548](https://doi.org/10.1109/ACCESS.2022.3165548).
- [87] Y.-H. Li, L. N. Harfiya, K. Purwandari, and Y.-D. Lin, "Real-time cuffless continuous blood pressure estimation using deep learning model," *Sensors*, vol. 20, no. 19, p. 5606, Sep. 2020, doi: [10.3390/s20195606](https://doi.org/10.3390/s20195606).
- [88] G. Van Houdt, C. Mosquera, and G. Nápoles, "A review on the long short-term memory model," *Artif. Intell. Rev.*, vol. 53, no. 8, pp. 5929–5955, Dec. 2020, doi: [10.1007/s10462-020-09838-1](https://doi.org/10.1007/s10462-020-09838-1).
- [89] M. Azizjon, A. Jumabek, and W. Kim, "1D CNN based network intrusion detection with normalization on imbalanced data," in *Proc. Int. Conf. Artif. Intell. Inf. Commun. (ICAIIIC)*, Fukuoka, Japan, Feb. 2020, pp. 218–224, doi: [10.1109/ICAIIIC48513.2020.9064976](https://doi.org/10.1109/ICAIIIC48513.2020.9064976).
- [90] P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. Shroff, "LSTM-based encoder–decoder for multi-sensor anomaly detection," 2016, *arXiv:1607.00148*.
- [91] M. MontazeriShatoori, L. Davidson, G. Kaur, and A. H. Lashkari, "Detection of DoH tunnels using time-series classification of encrypted traffic," in *Proc. IEEE Intl Conf Dependable, Autonomic Secure Comput., Intl Conf Pervasive Intell. Comput., Intl Conf Cloud Big Data Comput., Intl Conf Cyber Sci. Technol. Congr. (DASC/PiCom/CBDCCom/CyberSciTech)*, Aug. 2020, pp. 1–8, doi: [10.1109/DASC-PICOM-CBDCOM-CYBERSCITECH49142.2020.00026](https://doi.org/10.1109/DASC-PICOM-CBDCOM-CYBERSCITECH49142.2020.00026).
- [92] S. Yang, W. Xiao, M. Zhang, S. Guo, J. Zhao, and F. Shen, "Image data augmentation for deep learning: A survey," 2022, *arXiv:2204.08610*.
- [93] P. Suanpang and P. Jamjuntr, "Machine learning models for solar power generation forecasting in microgrid application implications for smart cities," *Sustainability*, vol. 16, no. 14, p. 6087, Jul. 2024, doi: [10.3390/su16146087](https://doi.org/10.3390/su16146087).
- [94] D. C. Montgomery, C. L. Jennings, and M. Kulahci, *Introduction to Time Series Analysis and Forecasting*. Hoboken, NJ, USA: Wiley, 2015.
- [95] R. J. Hyndman and G. Athanasopoulos, *Forecasting: Principles and Practice*. Melbourne, VIC, Australia: OTexts, 2018.
- [96] J. Casajús-Setién, C. Bielza, and P. Larrañaga, "Anomaly-based intrusion detection in IIoT networks using transformer models," in *Proc. IEEE Int. Conf. Cyber Secur. Resilience (CSR)*, Jul. 2023, pp. 72–77.
- [97] X. Zhu, K. Guo, T. Qiu, H. Fang, Z. Wu, X. Tan, and C. Liu, "Stereoscopic image super-resolution with interactive memory learning," *Expert Syst. Appl.*, vol. 227, Oct. 2023, Art. no. 120143.
- [98] X. Zhu, K. Guo, S. Ren, B. Hu, M. Hu, and H. Fang, "Lightweight image super-resolution with expectation-maximization attention mechanism," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 3, pp. 1273–1284, Mar. 2022.
- [99] Y. Zhang, Z. Gao, X. Wang, and Q. Liu, "Predicting the pore-pressure and temperature of fire-loaded concrete by a hybrid neural network," *Int. J. Comput. Methods*, vol. 19, no. 8, Oct. 2022, Art. no. 2142011, doi: [10.1142/s0219876221420111](https://doi.org/10.1142/s0219876221420111).
- [100] Y. Zhang, Z. Gao, X. Wang, and Q. Liu, "Image representations of numerical simulations for training neural networks," *Comput. Model. Eng. Sci.*, vol. 134, no. 2, pp. 821–833, 2023, doi: [10.32604/cmescs.2022.022088](https://doi.org/10.32604/cmescs.2022.022088).



**MOHAMMAD HAFIZ MOHD YUSOF** received the Bachelor of Technology degree (Hons.) in information technology from the University of Technology PETRONAS (UTP), Malaysia, in 2004, the M.Sc. degree in computer networking from the University of Technology MARA, Malaysia, in 2014, and the Ph.D. degree in computer science from the National University of Malaysia (UKM), in 2020, specializing in network-level malware classification model. He is currently a Researcher with the Research Interest Group (RIG), Machine Learning and Interactive Visualization (MaLIV), UiTM. He is a CISCO associate with ID: CSC011024858 and completed his CCNP (BSCI), CCNP (BCRAN), and CCNP (CIT), and successfully obtained Juniper JNCIA-Junos certification. He had spent almost 12 years in the IT industry, specializing in IT infrastructure. His research interests include machine learning in cybersecurity, deep learning in IDS, and network security.



**MOHAMMED BALFAQIH** (Senior Member, IEEE) received the Ph.D. degree in electrical, electronics, and systems engineering from Universiti Kebangsaan Malaysia, Malaysia. He is currently an Associate Professor with the Department of Computer and Network Engineering, University of Jeddah, Saudi Arabia. He published more than 32 journal and conference papers and patents. His principal research interests include wireless communications, mobility management, the IoT, and artificial intelligence.



**MD MUNIR HAYET KHAN** received the bachelor's degree in civil engineering from Khulna University of Engineering and Technology (KUET), Bangladesh, the master's degree (Hons.) in environmental engineering from Universiti Putra Malaysia (UPM), and the Ph.D. degree in civil and structural engineering from Universiti Kebangsaan Malaysia (UKM), in 2019. He is currently an Associate Professor with INTI International University, Nilai, Malaysia. His teaching expertise

spans a broad range of subjects, including environmental engineering, water engineering, engineering hydrology, soil mechanics and geotechnical engineering, and project management. Beyond the classroom, he has actively supervised numerous master's theses and final-year projects, significantly contributing to student development. His research portfolio is extensive, with high-impact publications in machine learning models for drought and groundwater level forecasting, spatiotemporal precipitation changes, bioconversion of agricultural waste for cellulase enzyme production, and the environmental impact of offshore oil and gas pipeline decommissioning.



**AKRAM A. ALMOHAMMEDI** received the B.Sc. degree in electronics engineering from Infrastructure University Kuala Lumpur, Malaysia, in 2012, the M.Sc. degree in electrical and electronics engineering majoring in computer and communication system from Universiti Kebangsaan Malaysia, Malaysia, and the Ph.D. degree in wireless communications and networks engineering from Universiti Putra Malaysia (UPM), in 2019. He is currently an Assistant Professor with Abdullah

Gül University, Türkiye. Previously, he was an Assistant Professor with the University of Karabuk (UNIKA), Türkiye, and a Senior Researcher (remote-based) with South Ural State University (SUSU), Russia. His research interests include the Internet of Vehicles, AI, ML, the Internet of Things, and wireless sensor networks. He was awarded the 2018 Best Paper Award by IEEE Malaysia ComSoc/VTS.



**ZAIN BALFAGIH** received the Ph.D. degree in computer science from Universiti Teknologi PETRONAS, Malaysia, in 2013. He is currently an Assistant Professor with the College of Engineering, Effat University, Jeddah, Saudi Arabia. He has published several peer-reviewed articles in reputable journals and conferences. He is also actively involved in supervising graduate students and contributing to various research projects at Effat University. His research interests include

machine learning, deep learning, natural language processing, and their applications in various domains, particularly in healthcare and education.

• • •