WSN-IoT Forecast: Wireless Sensor Network Throughput Prediction Framework in Multimedia Internet of Things

Rosa Eliviani & Yoanes Bandung[*]

School of Electrical Engineering and Informatics, Institut Teknologi Bandung,
Jalan Ganesha No. 10, Bandung 40132, Indonesia
*E-mail: yoanes.bandung@itb.ac.id

Abstract. Accurate throughput predictions can significantly improve the quality of experience (QoE), where QoE denotes a network's capacity to provide satisfactory service. By increasing the results of good throughput predictions, the best strategy can be planned for managing data transmission networks with the aim of better and faster data transmission, thereby increasing QoE. Consequently, this paper investigates how to predict the throughput of wireless sensor networks utilizing multimedia data. First, we conducted a comparative analysis of relevant prior research on the topic of throughput prediction in Multimedia Internet of Things (Multimedia IoT). We developed a throughput prediction framework for wireless sensor networks based on what we learned from these studies using machine learning. The Throughput Prediction Framework identifies historical throughput data and employs these traits to predict throughput. In the final phase, multiple camera nodes and local servers are utilized to test a framework for throughput prediction. Our analysis demonstrates that WSN-IoT predictions are quite precise. For a 1-second time breakdown, the average absolute percentage error for all investigated scenarios ranges from 1 to 8 percent.

Keywords: *framework; Internet of Things (IoT); multimedia; throughput prediction; wireless sensor network.*

1 Introduction

The Internet of Things (IoT) continues to expand, and with the development of IoT technology, smart cities are becoming more sophisticated. IoT continues to support progress in increasing effectiveness, efficiency, automation, software, and quality connectivity [1]. One of the enablers of IoT is the wireless sensor network (WSNs) technique. WSN gathers sensor data for centralized processing. WSNs are used in a variety of applications that enable integration of the physical world into the computer-based world, yielding benefits and enhancements in remotely managing the physical world, keeping electronic records of physical variables, early detection of potential threats, predictions, and economic benefits. WSNs are desirable for practical IoT implementations due to their low cost and ease of deployment. However, their small size and low cost result in resource

Received February 2nd, 2023, Revised December 13th, 2023, Accepted for publication December 31st, 2023.
Copyright © 2023 Published by IRCS-ITB, ISSN: 2337-5787, DOI: 10.5614/itbj.ict.res.appl.2023.17.3.4

limitations including energy supply, memory size, processing speed, and communication bandwidth. WSNs' limited resources need to be managed effectively to allow them to continue operating for as long as possible [2].

WSN differs from conventional wireless networks in that the resources of each sensor node are severely constrained. The primary objective of a sensor network is to detect a phenomenon and transmit the collected data to a user who is interested. Sensing and the delivery of high-quality data present the greatest difficulty in WSNs [3]. Using a WSN to transmit multimedia IoT data presents greater challenges than transmitting scalar data. Typically, multimedia files such as audio, video, and image contain a greater quantity of data, resulting in a greater degree of delay in the data flow. Currently, multimedia streaming has become one of the internet's most indispensable services. Moreover, due to the vast number of internet users, global network traffic is becoming increasingly congested [4]. Therefore, a significant amount of research is required to address this issue while simultaneously enhancing quality of exerience (QoE).

QoE defines the impact of a service's overall performance in relation to the level of customer satisfaction [5]. Throughput is one of the QoEfactors and is a calculation of the speed of sending appropriate data. Due to two main issues, though, it is very difficult to provide QoS in WSNs. Firstly, the typical severe limitations of WSN nodes, such as those related to their energy, computational, and communication capabilities, as well as the large-scale nature of WSNs. And secondly, the majority of QoS properties are interdependent in a way that improving one of them may degrade others, for example, increasing throughput. These unfavorable circumstances push system designers to seek out the greatest QoS metric compromises possible [6].

One of the QoE metrics is throughput, which shows the reduction in traffic that a destination node successfully receives [6]. Based on a throughput dataset, we can obtain several throughput prediction results. Several studies related to throughput prediction are similar to Elsherbiny's research in [7], which predicts throughput on 4G LTE networks. This throughput prediction uses some machine learning algorithms as in Na's research [8], who used LSTM to predict throughput in LTE networks. The results of this prediction can also be further utilized, as in [9], namely by utilizing throughput prediction results for adaptive bitrate control.

Further research is required into the management of sending this multimedia data to address issues with WSN and multimedia IoT data transmission. When conducting network management, there are several factors to consider before selecting the best method to modify the throughput parameters that determine how strong the network is being used. As a result, this study aimed to develop a throughput prediction framework for WSNs in multimedia IoT. This study started by comparing earlier studies that have addressed the issue of throughput prediction in multimedia IoT. Using machine learning, we then built a throughput prediction framework for WSNS based on the knowledge we gained from the preliminary study. In order to estimate throughput, the Throughput Prediction Framework locates historical throughput data and uses this feature. We tested it with the multimedia IoT framework using many camera nodes and local servers after explaining WSN-IoT predictions. This IoT feature enables multimedia IoT applications, or the use of multimedia (M-IoT). IoT for multimedia is concerned with multimedia data that is connected and engages with other multimedia [10].

The rest of this paper is organized as follows: Section 1 introduces the research; the content is from background research, the research purpose, and a small review. The second section presents related work on the same topic as that of the present research. Section 3 describes the WSN-IoT throughput prediction framework, its phases, and details. Section 4 describes the implementation of the framework, our experiment, the analysis, and the results. Finally, Section 5, provides the conclusions from all the whole study.

1.1 Related Works

This section discusses previous studies related to our research. The following subsections explain the main topics related to this research, i.e., wireless sensor networks (WSNs) in an Internet of Things (IoT) environment and machine learning-based throughput prediction.

1.2 Wireless Sensor Networks (WSNs) in an Internet of Things (IoT) Environment

The introduction of WSNs has opened new opportunities for monitoring applications. One of the advances in using WSNs is home monitoring. Home monitoring WSNs can collect sensing data such as temperature, humidity, and the state of other sensors such as magnetic sensors or switches, and are also capable of changing the environment and the physical world through actuators such as servos, motors, or switches. The use of WSNs includes various applications that enable the integration of the physical world into the computer-based world, which results in benefits and an increase in our quality of life. A wide range of wireless sensor devices has also been developed to enable wireless connectivity and the sensing of small objects [11].

However, WSNs have differences compared to other wireless networks. Wireless sensor network is a collective term to define a collection of small, somewhat independent computers whose primary target is sensing some physical property of their environment such as vibration, humidity, or temperature. They consist of several to thousands of sensor nodes, often also referred to as nodes or sensors, which are connected via wireless communication. Typically, there is at least one dedicated node, called a sink or base station, that connects the sensor network to the outside world. Some of the general characteristics of WSNs that differentiate them from other types of wireless networks are that sensor nodes have very limited resources; wireless connections are spontaneous and unplanned; and sensor networks deal with various phenomena and need to transfer data to external users [3].

1.3 Machine Learning based Throughput Prediction

This subsection presents some of the literature study on throughput prediction with machine learning and deep learning methodologies within wireless sensor networks. In [12], the authors conducted research to compare the machine learning algorithms ARIMA and LSTM in forecasting time series datasets. Compared to ARIMA, LSTM consistently achieved error rate reductions between 84 and 87 percent, demonstrating its superiority over ARIMA. Furthermore, it was found that the trained prediction model behaved randomly and that the amount of training cycles, or epochs' as they are called in deep learning, had no impact on its performance. In [13], the authors compare SARIMA, LSTM and CNN to forecast time series data but not to a throughput dataset.

The research in [8] explored the application of machine learning and deep learning methodologies in predicting throughput in wireless sensor networks, explaining previous research efforts that have used learning models for this purpose. Recognizing the importance of accurate throughput predictions, the cited studies aimed to reduce latency and improve the efficiency of time-sensitive services, as reported in their respective research findings. Thus, [8] predicted future throughput using the attention-based LSTM model. The dataset was derived from TCP logs and LTE network throughput. This work mentions that throughput prediction is essential for latency reduction. Moreover, some studied used LSTM in their throughput prediction, for example, [7] and [8]. In [7], besides using an LSTM model, the authors also used the arithmetic mean (AM), harmonic mean (HM), last sample (LS), moving average (MA), a hidden Markov model (HMM), and a stochastic model. In this work [8], regression models such as K-Nearest Neighbor (KNN), Support Vector Machine Regression (SVR), Ridge Regression, Random Forest Regression, Autoregressive Integrated Moving Average (ARIMA), and Long Short-Term Memory (LSTM) were used.

2 WSN-IoT Forecast: Wireless Sensor Network Throughput Prediction Framework in Multimedia Internet of Things

The WSN-IoT Throughput Prediction Framework is a flow framework for predicting future throughput in WSNs in IoT environments based on past throughput data. Figure 1 shows the phases of the WSN-IoT Throughput Prediction Framework. The four stages of predicting throughput in WSNs and M-IoT are shown in Figure 1. These stages are the WSN and Multimedia IoT stage, the dataset stage, the training stage, and finally the throughput prediction stage. The identification stage serves as a pivotal step in analyzing the network and its architecture. Subsequently, the dataset stage is employed to collect and prepare the necessary dataset. Finally, the training and prediction stages are utilized to apply algorithms to predict the throughput dataset based on historical data. Detailed explanations for each stage are given after Figure 1.

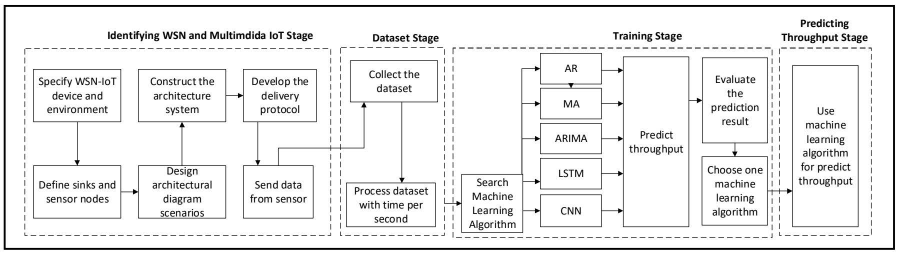
Identifying WSN and Multimdida IoT Stage
Dataset Stage
Training Stage
Predicting Throughput Stage
Specify WSN-IoT device and environment
Construct the architecture system
Develop the delivery protocol
Define sinks and sensor nodes
Design architectural diagram scenarios
Send data from sensor
Collect the dataset
Process dataset with time per second
Search Machine Learning Algorithm
AR
MA
ARIMA
LSTM
CNN
Predict throughput
Evaluate the prediction result
Choose one machine learning algorithm
Use machine learning algorithm for predict throughput

Figure 1 Throughput prediction framework.

2.1 Identifying WSN and Multimedia IoT Stage

The WSN and Multimedia IoT identification stage aims to identify the architectural system used in data transmission. This process is like an observation process, such as node conditions, location, distance, movement, and throughput. The first process is to determine the WSN and Multimedia IoT devices and environments, for example sensor devices with sensing such as capturing pictures and recording audio or video. After that, determine the sink and sensor nodes of the device used are determined. For example, when sending data to the cloud by a camera sensor device, then it is determined which device functions as the sink or base station. Here the sink can be one or more node. Then other sensor devices can have connections with each other and with other devices. After all the points have been determined for their respective functions, the next step is to design a scenario diagram of the implemented system architecture. The system's architectural design can be seen in Figure 2. It is a system architecture for WSN-IoT that can be applied to different system configurations. These nodes are useful as monitoring sensors, for example temperature sensors, image capture sensors, light sensors, and others. Each of these nodes has a relationship with other nodes using one server. Additionally, it is crucial to note that in the image capture process, precise time synchronization is required to obtain accurate data. Furthermore, at the identification stage, it is essential to consider the power and resource requirements of each device to ensure the system's continuity efficiently.

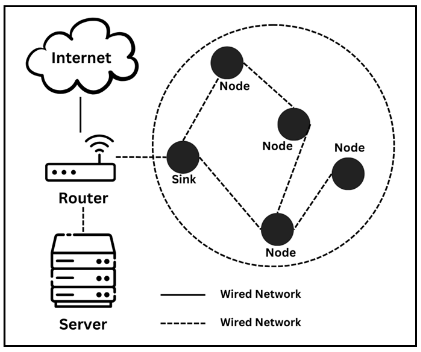

Figure 2 Architecture system.

2.2 Dataset Stage

The dataset stage is the stage of retrieval and preparation of the throughput dataset by processing data according to the applied system architecture. Data retrieval is carried out by recording the process of sending data on multimedia IoT devices. To save the process of sending the data in the form of throughput data, the size of the file sent, the initial time of sending the data in microseconds, and the final time of sending data in microseconds are recorded. This data retrieval aims to determine the throughput history of the WSN and multimedia networks. Then after the data has been collected, for example thousands of data, the dataset is processed. Data processing is done by collecting the data and average them into calculations per second. It is intended that the data sent considers the time between sending data. After that some of these datasets are divided into two parts, namely a training dataset and a testing dataset, where the training dataset can be used to conduct machine learning to study historical throughput patterns. The throughput data is obtained by calculating the number of data bits sent divided by the duration of data transmission. The duration of data transmission is obtained by reducing the value of the final time of sending data with the value of the initial time of sending the data. Based on the results of this dataset processing, predictions are made by testing the dataset and generating future throughput predictions.

2.3 Training Stage

The training stage is the stage for processing the dataset that has been prepared to feed into several appropriate machine learning algorithms because this stage aims to select the best algorithm model for the throughput dataset and train the algorithm. This algorithm is trained so that it can study past throughput historical data and can predict future throughput. According to [7], models are suitable for predicting datasets such as throughput data are based on time series algorithms.

Some of these machine learning algorithms are like Recurrent Neural Networks (RNN), Long short-term memory (LSTM), Gated Recurrent Unit (GRU) or Autoregressive Integrated Moving Average (ARIMA). Based on this, this study used several machine learning time series dataset models, namely AR, MA, ARIMA and LSTM. In addition, a machine learning model, namely deep learning, was added, which is a convolutional neural network (CNN). The machine learning algorithm model with the best predictive results was then selected from the five machine learning models. The selection was based on the results of the mean absolute error (MAE), root mean square error (RMSE), and mean absolute precentag error (MAPE) evaluation matrices. This evaluation used a comparison of the throughput predicted value with the actual throughput data.

2.4 Predicting Throughput Stage

After the training phase has been completed, the machine learning algorithm model is obtained with the closest prediction results. Then the throughput is predicted with the selected machine learning algorithm model using the throughput history with previous data that has been recorded in the dataset stage. For example, there are several machine learning algorithms were used, namely AR, MA, ARIMA, LSTM and CNN. The five machine learning algorithm models are trained and predicted throughput. Then, the algorithm model with the best evaluation result value is used in the prediction stage. Finally, the predicted throughput data can be used for further network management. One such treatment is reported in [9], which used throughput prediction results to adaptively control video bitrates.

3 Experimental Setup and Result

This section presents the setup of the experiment and the experimental result. The experiment started with the throughput prediction process: data collection, preprocessing data, data training, and data prediction. Below, the results are analyzed and discussed.

3.1 System Implementation

For this research, we used the IoT concept for the proposed system architecture. In the experiment, the IoT system consisted of the following components: one ESP32-CAM and one ESP-EYE, a node with sensing capability for taking multimedia data, a router to give access to the internet, and a local server. We used a Raspberry Pi model 3 B. We used wireless fidelity (Wi-Fi) for the connection between devices. The router connected the node device and the local server without modification to that router.

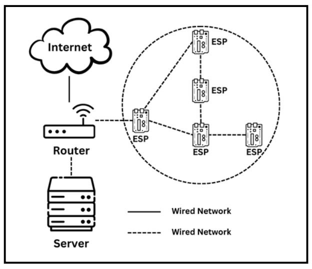

Figure 3 Implemented architecture system.

In this architectural system, several protocols support sending data in IoT systems. The protocols developed in [14] were Message Queuing Telemetry Transport (MQTT), Constrained Application Protocol (CoAP), Hypertext Transfer Protocol (HTTP), and Extensible Messaging and Presence Protocol (XMPP). MQTT transport over TCP/IP, with a tree architecture and application to message IoT applications with the same architecture and usage as before but with CoAP transport via UDP/IP TCP/IP and a client-server architecture were then transported by XMPP and HTTP. However, XMPP is suitable for decentralized IoT systems, and HTTP is suitable for energy management, gateways, and home services. Moreover, [14] compares them by message type, constrained network (GPRS, 2G, 3G), low power, and security. According to [14], with low power, COAP has excellent value. It had the best values compared to the other options with both good and poor values.

Because of the research paper in [14], data delivery in the proposed architecture system research uses the Constrained Application Protocol (CoAP). The reason for using this protocol is that CoAP supports data delivery with limited resources. The IoT concept with current resources has limited capabilities. It has request and response message types. CoAP works on UDP, so it has high-speed device-to-device communication. It differs from other protocols, like Hypertext Transfer Protocol (HTTP), which works on TCP. Here is a demonstration photo of the ESP32-CAM and the Raspberry Pi.

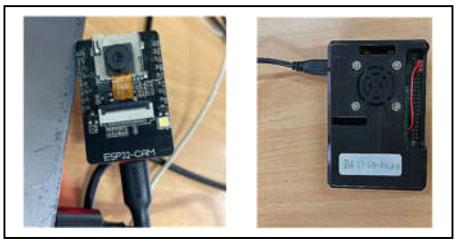

Figure 4 32-CAM and Raspberry Pi.

In Figure 3, there are three scenarios are used for throughput prediction. This scenario can be called Scenario A, Scenario B, and Scenario C. The research used five ESPs that can be seen in the system architecture, namely one ESP-EYE and four ESP32-CAMs. However, the throughput prediction stage uses datasets from the three main ESPs in this architecture system, along with information from the scenarios in Table 1. In utilizing the demonstration scenario, the collected dataset encompasses the size of image files in bits, the initial transmission time, and the successful transmission time.

Table 1 Scenario demonstration.

Scenario	Description
A	ESP-EYE is a sensor node as a sink that is directly connected to the router
B	ESP32-CAM 1 is a sensor node that is connected to one hub
C	ESP32-CAM 2 is a sensor node connected to two hubs

3.2 Throughput Prediction

Throughput is the number of bits per second received by the receiver. This throughput is one of the critical parameters for assessing the service quality of wireless sensor networks [15]. The results of the throughput prediction data assist the next step in dynamically selecting the appropriate multimedia bitrate. This throughput prediction can help ensure the QoE, but throughput prediction alone is not enough. It is not enough because knowing how to predict throughput is only the initial stage of being able to choose the best strategy in network management by adjusting network speed conditions. For example, we can choose a video bitrate by adjusting network speed based on the results of throughput prediction; the better the network conditions, the higher the video bitrate sent. With this strategy, QoE can increase because buffering is avoided.

We set up the components before beginning the throughput prediction process. The IoT system consisted of the following components: one ESP32-CAM and one ESP-EYE, a node with sensing capability for taking multimedia data, a router to give access to the internet, and a local server. For the data programming built with CoAP, in the ESP-32, we used ESP IDF [16], and in the Raspberry Pi, we used AioCoAP [17]. The server program ran with the Raspbian operating system. The camera nodes took the multimedia data and sent it regularly to the server, and then the server received the multimedia data.

In the process of throughput prediction, this work starts with data collection. Data collection is the next step in the delivery of data from the ESP32-Cam and Raspberry Pi. The system should be configured as described in Figure 3 in order to collect this dataset. The amount of data sent by each transmission made by the sensor node is then recorded, along with the initial and last transmissions. The amount of data divided by the delivery duration can be used to get the delivery time duration and throughput value. The dataset is divided into two parts: training data and test data. After the dataset is ready, the next step is preprocessing the data. Data preprocessing prepares the dataset for the machine learning process. The data can be cleaned, normalized, or reduced. The next step is training and predicting the data when it is ready. Training and predicting the data is the process of getting the prediction result. The last step is evaluating the prediction result to know the best prediction from all data used.

Based on the identified characteristics, conditions, and needs, the proposed throughput prediction algorithm in this study encompasses ARIMA, AR, MA, LSTM, and CNN. This assertion is substantiated by the performance results documented in [7], wherein it is revealed that in the time series forecasting model, ARIMA yielded satisfactory outcomes and can be recognized as a suitable algorithm for time-series datasets. Furthermore, [8] concluded that the LSTM model demonstrated robustness against dynamic fluctuations, resulting in the highest prediction accuracy. Lastly, for comparative purposes, CNN was employed because in the paper [13], this algorithm was applied for the analysis and forecasting of time-series data alongside the utilization of ARIMA and LSTM.

This research predicted throughput with a time series dataset. There are many ways to predict time-series datasets. To predict time series data, it is generally recommended to use an autoregressive integrated moving average (ARIMA) and essential models such as autoregressive (AR), moving average (MA), and long-short-term memory (LSTM). Besides that, another deep learning algorithm, a convolutional neural network (CNN), was also used in this research. We utilized the Autoregressive (AR) algorithm model to forecast throughput data based on a dataset comprising 5,050 training data and 500 testing data. The data was transmitted from the ESP32-Cam to the Raspberry Pi through a Wi-Fi connection.

3.2.1 Autoregressive (AR)

The autoregressive algorithm (AR) model is a model that utilizes the dependence between an observation and several observations that are left behind [18]. The equation for this AR model is shown below.

$$y_t = \Phi_1 y_{t-1} + \Phi_p y_{t-2} + \cdots + \Phi_p y_{t-p} \tag{1}$$

Where:

y_t	=	lag term
φ	=	coefficient of the lag term

Visualization results of our throughput predictions use AR algorithms are shown in Figures 5 to 7. The three figures demonstrate that although the anticipated value cannot be predicted with accuracy, it nevertheless complies with the trend that represents the real data.

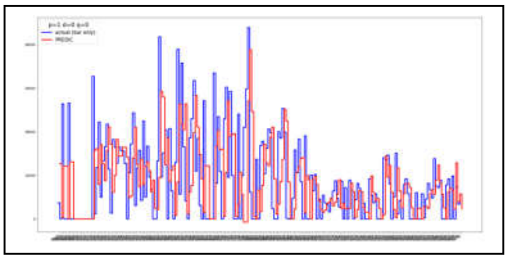

Figure 5 Autoregressive model performance on test set Scenario A.

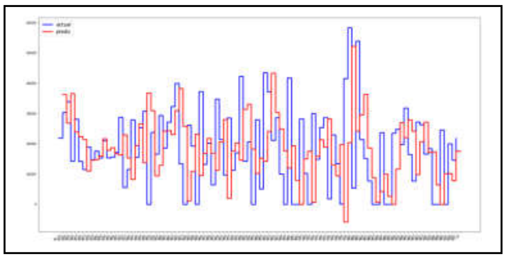

Figure 6 Autoregressive model performance on test set Scenario B.

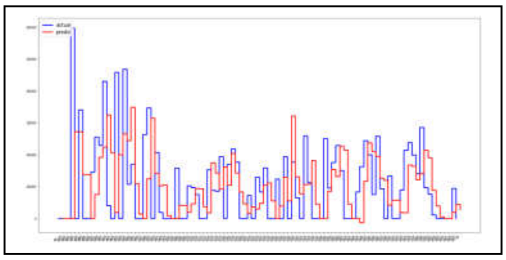

Figure 7 Autoregressive model performance on test set Scenario C.

3.2.2 Moving Average (MA)

The moving average (MA) algorithm model is a model that predicts the next data based on the previous data so that the data has a dependency between the observed values and the consecutive error values [18].

$$y_t = \theta_0 a_t + \theta_1 a_{t-1} + \cdots + \theta_q a_{t-q} \qquad (2)$$

Where:

y_t	=	moving average process
θ	=	constants
e_t	=	error terms

Figures 8 to 10 show visualization results from our throughput prediction using a moving average model.

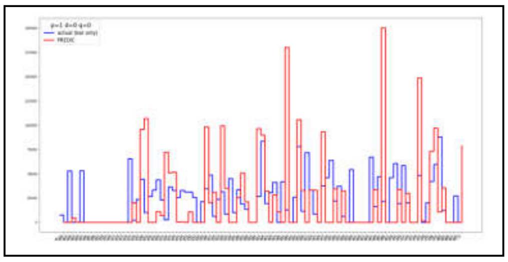

Figure 8 Moving average model performance on test set Scenario A.

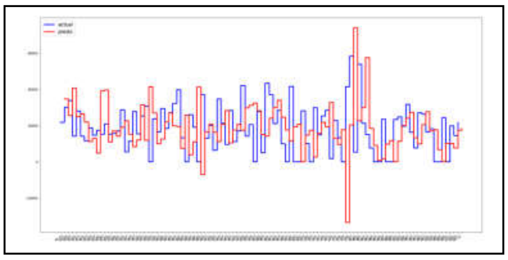

Figure 9 Moving average model performance on test set Scenario B.

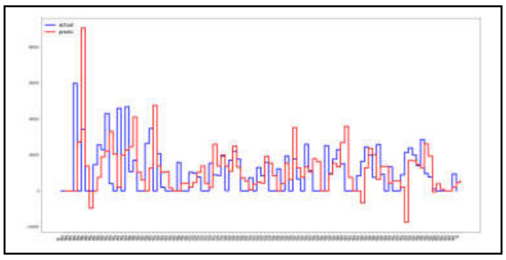

Figure 10 Moving average model performance on test set Scenario C.

3.2.3 Autoregressive Integrated Moving Average (ARIMA)

Autoregressive Integrated Moving Average (ARIMA) is a model that utilizes the forecasting time of pre-existing values [19]. ARIMA is a general model of the autoregressive moving average (ARMA) model. This algorithm model combines the processes of the AR and MA models, then creates a composite model from the time series. Usually, ARIMA is defined in three terms, namely p, d, and q, to capture the critical elements of the model [18]. P is the autoregressive process, d is the differentiation process, and q is the moving average process. It is a form with order (p, q) on the ARIMA model.

$$x_t = c + \sum_{i=0}^{p} \Phi_i x_{t-i} + \epsilon_t + \sum_{i=1}^{q} \theta_i a_{t-i} \tag{3}$$

where:

φ_i	$\neq$	0
θ_i	$\neq$	0
σ	$>$	0
p	=	AR orders
q	=	MA orders

Figures 11 to 13 show visualization result from our throughput prediction use ARIMA algorithms. The same as with AR, three figures demonstrate that although the anticipated value cannot be predicted with accuracy, it nonetheless complies with the trend that represents the real data.

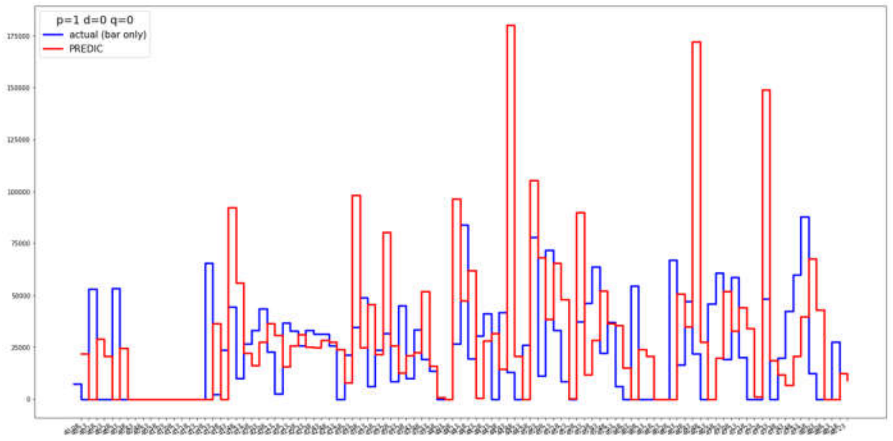

Figure 11 ARIMA model performance on test set Scenario A.

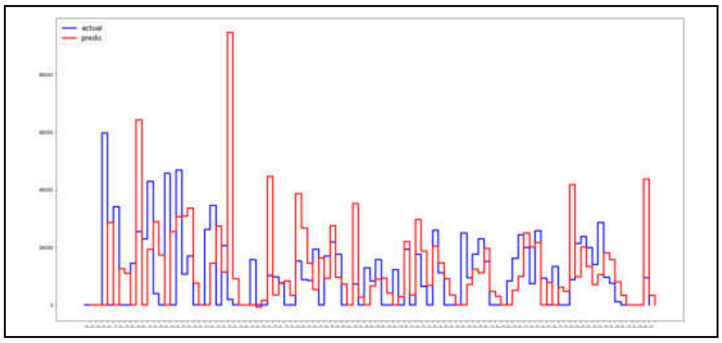

Figure 12 ARIMA model performance on test set Scenario B.

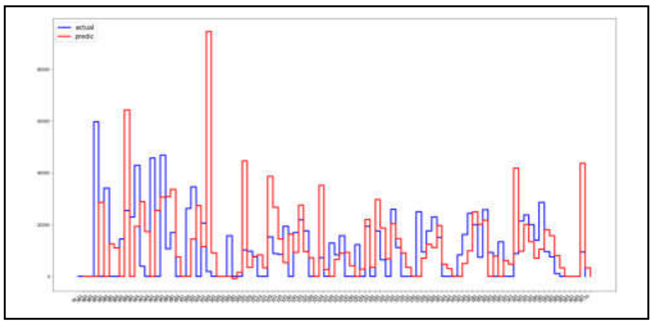

Figure 13 ARIMA model performance on test set Scenario C.

3.2.4 Long-Short Term Memory (LSTM)

This long-short-term memory (LSTM) model is an upgrade from the existing recurrent neural network (RNN) algorithm. RNN needs to be improved. RNN has a weakness in learning because it is long-term [18]. The upgrade of the LSTM creates a model that can remember data or information for a longer period of time. This upgrade allows the model to read, write, and delete data and information. LSTM generally has three gates: forget, input, and output. The function of the forget gate is to set the required or deleted information. The input gate determines the information that enters the memory, and the last output gate considers the value of the output result [18].

In this research, using the LSTM algorithm model, we predicted the throughput of data delivery from the ESP32-Cam to the Raspberry Pi using a Wi-Fi connection as shown in Figure 3. Figure 14, Figure 15, and Figure 16 show visualization results from our throughput prediction. The three figures show that the predictions had constant values, did not fluctuate, and still corresponded to the actual data trends.

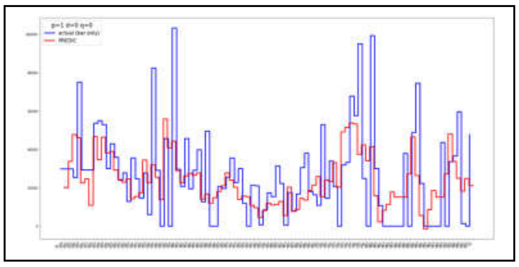

Figure 14 LSTM model performance on test set Scenario A.

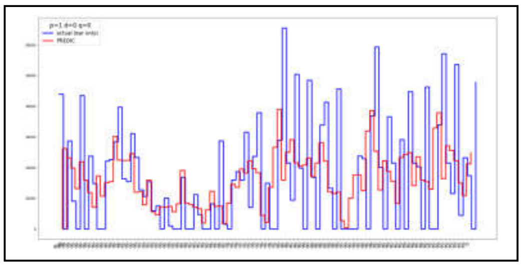

Figure 15 LSTM model performance on test set Scenario B.

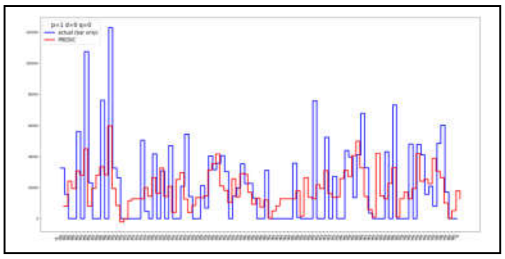

Figure 16 LSTM model performance on test set Scenario C.

3.2.5 Convolutional Neural Network (CNN)

The CNN method is a deep learning method that can carry out the learning process. CNN consists of four layers: a convolutional layer, a pooling, being layer, a fully connected layer, and a striding layer. This convolution layer embeds a filter so that it helps perform convolution operations by looking at the output and its dimensions. Pooling helps reduce the spatial size of the representation gradually. The fully connected layer is a valuable layer for connecting inputs and all neurons; it can activate the sigmoid. Finally, some strides help to show the number of pixels that pass through each execution [13]. From the three figures, the throughput predictions were still inaccurate and below the actual values, especially in Scenario C, as can be seen from the three figures (Figures 17 to 19), but they could be made to follow the actual data trend.

The model is defined as a sequential model, representing a linear sequence of layers. The first layer is the Conv1D layer with 16 filters, a kernel size of 3, and a ReLU activation function. The second layer is the MaxPooling1D layer with a pool size of 2. Next, there is the Flattening layer, utilized to flatten the output from the previous layer into a one-dimensional vector. The fourth layer is the Dense layer with 10 units and ReLU activation. The fifth layer is the Dense layer with the number of units matching n_outputs and without activation function (linear). This design is intended to effectively extract and comprehend features from the input data for prediction or analysis purposes.

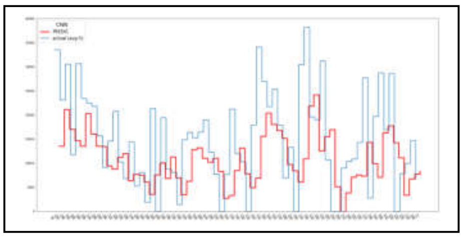

Figure 17 CNN model performance on test Scenario A.

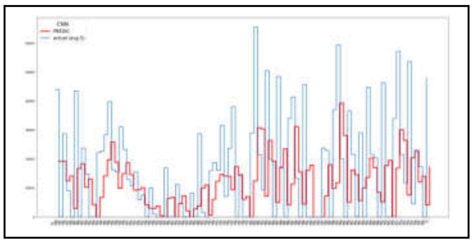

Figure 18 CNN model performance on test set ESP 2.

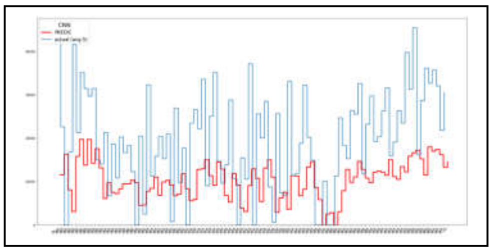

Figure 19 CNN model performance on test set ESP 3.

3.3 Evaluation

From throughput prediction results, there are evaluations for assigning value to the actual and prediction data. In this research, we used three performance metrics to analyze the results of the throughput prediction. These three performance-metrics were mean absolute error (MAE), root mean square error (RMSE), and mean absolute percentage error (MAPE). The MAE calculates the typical value difference between the prediction and the actual value in the final form. The range of values for this MAE is 0 to infinity. An inaccuracy in the actual value is represented by MAE. The forecasting model used performs better the lower the MAE value. If the outliers are evidence of faulty data, use MAE. If the test set additionally contains a lot of outliers, the model's performance will be subpar [20]. The formula for MAE is given in Eq. (4), where n is the number of datasets, P denotes the prediction, and A denotes the actual value [21].

$$\mathrm{MAE} = \frac{1}{\mathrm{n}}\sum_{\mathrm{i}=1}^{\mathrm{n}}|\mathrm{P_i} - \mathrm{A_i}| \tag{4}$$

The root mean square error (RMSE) compares the predicted result to the actual value. The smaller the RMSE value, the more accurate the forecast. A fitted linear regression model displays a value of 0 when the model has perfect data [7]. Eq. (5) presents the formula for RMSE, where n denotes several datasets, ŷ denotes the prediction, and denotes the actual value.

$$\mathrm{RMSE} = \sqrt{\frac{1}{\mathrm{n}}\sum_{\mathrm{i}=1}^{\mathrm{n}}(\hat{y}_\mathrm{i} - y_\mathrm{i})^2} \tag{5}$$

The mean absolute precentage error (MAPE) is a percentage of the average absolute value of the forecasting error. MAPE has a very intuitive interpretation in terms of relative error. The smaller the MAPE value, the more accurate the forecasting model is implemented. The equation for calculating the MAPE value is shown in the following equation [20].

$$\mathrm{MAPE} = \frac{100\%}{\mathrm{n}}\sum_{\mathrm{i}=1}^{\mathrm{n}}\left|\frac{y - x_\mathrm{i}}{y_\mathrm{i}}\right| \tag{6}$$

where:

n	=	number of datasets
y	=	average value
x	=	predicted value

This research used five algorithm models: AR, MA, ARIMA, LSTM, and CNN. Moreover, from the results, we evaluated every result with three performance metrics: MAE, RMSE, and MAPE. MAE serves as a performance metric by calculating the average difference between the forecasted and the actual values in their absolute form. It ranges from 0 to infinity, with lower MAE values indicating better performance of the applied forecasting model. MAE specifically measures errors in the actual values and is particularly useful when outliers represent data anomalies [7]. On the other hand, the RMSE evaluates the residuals between the predicted and the actual results, providing insights into the model's overall performance. A smaller RMSE value signifies a more accurate forecast [20]. Meanwhile, the MAPE represents the percentage of the average absolute value of the forecasting errors. MAPE offers an intuitive interpretation in terms of relative error, where a decreased MAPE value indicates enhanced accuracy in the implemented forecasting model [7].

Table 2 Summary of throughput prediction evaluation.

ESP	Model	MAE	RMSE	MAPE
A	AR	13277.624	19491.594	**1.034**
	MA	14751.351	22853.587	1.301
	ARIMA	16395.375	29056.982	7.814
	LSTM	**11518.082**	**15942.046**	1.403
	CNN	17744.811	24095.251	2.514
B	AR	9902.120	14201.272	8.962
	MA	11237.650	17399.863	1.149
	ARIMA	10924.456	16003.699	**1.130**
	LSTM	9740.015	**12506.501**	1.729
	CNN	**8974.625**	12919.303	7.393
C	AR	14015.330	22171.242	1.618
	MA	15748.343	25410.757	1.943
	ARIMA	18631.169	50125.583	**1.247**
	LSTM	**12846.562**	**18249.030**	1.919
	CNN	21853.083	36403.502	2.524

The bold type in Table 2 is the best evaluation value compared to the values of the other algorithms. LSTM had the best value in the MAE and RMSE evaluations in Scenarios A and C, while in Scenario B it was only for the RMSE values. In addition, in terms of MAPE scores, ARIMA had the best evaluation scores in Scenarios B and C, while Scenario A it was not ARIMA but AR. We can conclude that LSTM was the optimal algorithm in this study, followed by ARIMA. In many instances of this throughput prediction experiment, LSTM was appropriate for most of the data. In addition, CNN was capable of predicting throughput, which is the time series data, but this is insufficient. Utilizing recurrent neural networks (RNN) and their development, such as LSTM, provides a more accurate prediction of throughput for time series data. Consequently, the next step for this research is to utilize the LSTM throughput prediction results to assist in dynamically selecting the optimal bitrate for multimedia.

4 Conclusions

The purpose of this paper was to create a framework for predicting throughput, especially for WSN and multimedia IoT. In this paper, we present WSN-IoT Forecast: Wireless Sensor Network Throughput Prediction Framework in Multimedia Internet of Things, a machine learning framework that makes use of past throughput data in wireless sensor networks (WSN) and the Internet of Things (IoT) from an environmental perspective. We utilized the autoregressive (AR), moving average (MA), ARIMA, LSTM, and CNN. According to the results of the evaluation, throughput prediction was acceptable in terms of mean absolute percentage error, with low but acceptable accuracy. For a time breakdown of one second, the average absolute percentage error for all investigated scenarios fell between 1 and 8 percent. The next challenge is utilizing the predicted results for adaptive bitrate control in real-time.

References

[1] Saad, W., Bennis, M. & Chen, M., *A Vision of 6G Wireless Systems: Applications, Trends, Technologies, and Open Research Problems*, IEEE Network, pp. 134-142, 2020.

[2] Jamalipour, A., *Wireless Sensor Network: A Networking Perspective*, Hoboken, New Jersey: John Wiley & Sons, Inc., 2009.

[3] Förster, A., *Introduction to Wireless Sensor Networks*, New Jersey: John Wiley & Sons, 2016.

[4] Krawiec, P. & Sosnowski, M., *DASCo: Dynamic Adaptive Streaming over CoAP*, Multimedia Tools and Applications, pp. 4641-4660, 2018.

[5] Bouraqia, K., Sabir, E., Sadik M. & Ladid, L., *Quality of Experience for Streaming Services: Measurements, Challenges and Insights*, IEEE Access, **8**, pp. 13341-13361, 2020.

[6] Koubâa, A., Severino, R., Alves, M. & Tovar, E., *Improving Quality-of-Service in Wireless Sensor Networks by Mitigating Hidden-Node Collisions*, IEEE Transactions on Industrial Informatics, **5**(3), pp. 299-313, 2009.

[7] Elsherbiny, H., Abbas, H.M., Abou-zeid, H., Hassanein H.S. & Noureldin, A., *4G LTE Network Throughput Modelling and Prediction*, IEEE Global Communications Conference, Canada, 2020.

[8] Na, H., Shin, Y., Lee, D. & Lee, J., *LSTM-Based Throughput Prediction for LTE Networks*, ICT Express, 2021.

[9] Wei, B., Song, H., Wang, S., Kanai, K. & Katto, J., *Evaluation of Throughput Prediction for Adaptive Bitrate Control using Trace-Based Emulation*, IEEE Access, **7**, pp. 51346-51356, 2019.

[10] Wang, Q., Zhao, Y., Wang, W., Minoli, D., Sohraby, K., Zhu, H. & Occhiogrosso, B., *Multimedia IoT Systems and Applications*, 2017 Global Internet of Things Summit (GIoTS), 2017.

[11] Jurado-Lasso, F.F., Marchegiani, L., Jurado, J.F., Abu-Mahfouz, A.M. & Fafoutis, X., *A Survey on Machine Learning Software-Defined Wireless Sensor Networks (ML-SDWSNs): Current Status and Major Challenges*, IEEE Access, **10**, pp. 23560-23592, 2022.

[12] Siami-Namini, S., Tavakoli, N. & Siami-Namini, A., *A Comparison of ARIMA and LSTM in Forecasting Time Series*, in 17th IEEE International Conference on Machine Learning and Applications, 2018.

[13] Dwivedi, S.A., Attry, A., Parekh, D. & Singla, K., *Analysis and Forecasting of Time-Series Data using SARIMA, CNN and LSTM*, International Conference on Computing, Communication, and Intelligent Systems (ICCCIS), 2021.

[14] Nikolov, N., Research of MQTT, CoAP, *HTTP and XMPP IoT Communication protocols for Embedded Systems*, Proc. XXIX International Scientific Conference Electronics ET2020, Sozopol, Bulgaria, 2020.

[15] Vatti, R. & Gaikwad, A.N., *Throughput Improvement of Wireless Networks using Collision Control Approach*, SRATE-International Journal of Research in Application Technologies, **6**(2), pp. 15-23, 2016.

[16] Hoang, T.N., Van S.T. & Nguyen, B., *ESP-NOW based Decentralized Low Cost Voice Communication Systems for Buildings*, International Symposium on Electrical and Electronics Engineering (ISEE), Ho Chi Minh City, Vietnam, 2019.

[17] Viola, F., Turchet, L., Antoniazzi, F. & Fazekas, G., *C Minor: A Semantic Publish/Subscribe Broker for the Internet of Musical Things*, 23rd Conference of Open Innovations Association (FRUCT), Bologna, Italy, 2018.

[18] Verma, P., Reddy, S.V., Ragha D.L. & Datta, D.D., *Comparison of Time-Series Forecasting Models*, International Conference on Intelligent Technologies (CONIT) , Karnataka, India, 2021.

[19] Siami-Namini, S., Travakoli, N. & Namin, A.S., *A Comparative Analysis of Forecasting Financial Time Series using ARIMA, LSTM and BiLSTM*, arXiv, 2019.

[20] Chicco, D., Warrens, M.J. & Jurman, G., *The Coefficient of Determination R-Squared Is More Informative Than SMAPE, MAE, MAPE MSE and RMSE in Regression Analysis Evaluation*, Peerj Computer Science, 2021.

[21] Khan, M. & Noor, S., *Performance Analysis of Regression-Machine Learning Algorithms for Predication of Runoff Time*, Agrotechnology, **8**(1), pp. 1-12, 2019.