Data Fusion Integrated Network Forecasting Scheme Classifier (DFI-NFSC) via Multi-Layer Perceptron Decomposition Architecture

Erdem Çakan[*,a,b], Volkan Rodoplu[b], Cüneyt Güzeliş[b]

[a]*Akbank, Istanbul, Turkey.*
[b]*Department of Electrical and Electronics Engineering, Graduate School, Yaşar University, İzmir, Turkey*

Abstract

The Massive Access Problem of the Internet of Things stands for the access problem of the wireless devices to the Gateway when the device population in the coverage area is excessive. We develop a hybrid model called Data Fusion Integrated Network Forecasting Scheme Classifier (DFI-NFSC) using a Multi-Layer Perceptron (MLP) Decomposition architecture specifically designed to address the Massive Access Problem. We utilize our custom error metric to display throughput and energy consumption results. These results are obtained by emulating the Joint Forecasting-Scheduling (JFS) system on a single IoT Gateway and distinguishing between ARIMA, LSTM, and MLP forecasters of the JFS system. The outcomes indicate that the DFI-NFCS method plays a notable role in improving performance and mitigating challenges arising from the dynamic fluctuations in the diversity of device types within an IoT gateway's coverage zone.

Keywords: Internet of Things (IoT), emulation, massive access, Medium Access Control (MAC) layer, artificial neural network (ANN), predictive network, joint forecasting-scheduling

[⋆]This work has been supported by TÜBİTAK (Scientific and Technological Research Council of Turkey) under the 1001 program grant no. 118E277.

[⋆⋆]The final version of this preprint is published at Internet of Things https://doi.org/10.1016/j.iot.2024.101341.

[*]Corresponding author

Email addresses: `cakanerdem5@gmail.com` (Erdem Çakan[*]), `volkan.rodoplu@yasar.edu.tr` (Volkan Rodoplu), `cuneyt.guzelis@yasar.edu.tr` (Cüneyt Güzeliş)

1. Introduction

In the next decade, living quarters will be hosting a massive number of devices, e.g., smartphones, computers, tablets, and a swarm of smart city equipment such as smart utility meters and smart grids [1]. According to the foresight of this research, the data traffic load on mobile networks will increase 300 times by connection of each device on the Internet. The IoT Gateways will be hosting a significant spike of load in the Massive IoT section, which will cover 51% of cellular IoT connections in the next 5 years [1]. Hence, new generation communication systems should process this operation by enabling intelligent algorithms, machine learning (ML) models, and other processing techniques. The challenge of ensuring uninterrupted wireless connection for an extensive array of devices is termed the Massive Access Problem.

Previous approaches aimed at addressing the problem have primarily concentrated on resolving collision issues occurring on the Physical Random Access Channel (PRACH), primarily by leveraging random arrival mechanisms [2, 3]. Furthermore, Reference [4] improved the Access Class Barring (ACB) through the use of Markov chains to model preamble status and predict active devices. Meanwhile, Reference [5] developed a recursive ACB algorithm that relies on real-time detection of idle preambles. All these studies addressed similar approaches that model IoT Traffic data by using random arrivals in which each IoT device is assumed to generate a random amount of traffic at random times.

In this respect, predicting the future traffic patterns of IoT devices accurately remains challenging. As a result, the practical approach for managing IoT traffic is to employ MAC-layer protocols that dynamically respond to changing traffic demands, characterized as *reactive*. Conversely, when precise forecasts of individual IoT device traffic become feasible, *proactive* scheduling of their traffic becomes a viable alternative.

References [6, 7] show that machine learning models can predict the M2M traffic generation of individual IoT devices. Reference [8] proposed a novel system called "Joint Forecasting-Scheduling (JFS)". This system is designed to maintain a single IoT Gateway to forecast the future traffic patterns of individual IoT devices in the coverage area of the IoT Gateway, conversely to reactive protocols. Utilizing traffic forecast data, the system assigns MAC-layer resources to these IoT devices in a manner that ensures they do not encounter collisions. Later, References [9] and [10] focus on performance enhancement of JFS system in the network from a multi-scale and multi-channel perspective for throughput and energy consumption of transmitted signal. These improvements are then compared to the performance of reactive benchmark protocols.

Furthermore, Reference [11] proposed a novel framework called Dynamic, Automatic Forecaster Selection (DAFS). The approach is specifically crafted to address situations in which the count of IoT devices within the range of an IoT gateway exhibits dynamic variability. This method aims to imitate the intricate correlation between the quantities of IoT devices within distinct device classes within the gateway's coverage area and the consequent impact on the performance of network, employing an Artificial Neural Network (ANN). Conversely, the DFI-NFSC approach focuses on the comprehensive fusion of multiple data attributes, emulation of Joint Feature Selection (JFS), and the classification of specialized forecasters for JFS. To establish and implement this methodology, we have employed a hybrid model comprising a cascading series of neural networks proficient in performing both regression and classification tasks.

In [6], IoT devices were categorized into four separate device categories according to their traffic generation patterns. These classifications are as follows: Devices belonging to the Fixed-Bit Periodic (FBP) class consistently produce a fixed number of bits at consistent time intervals. Similarly, devices in the Fixed-Bit Aperiodic (FBA) class generate a fixed amount of data, albeit at irregular time intervals. Conversely, devices categorized as Variable-Bit Periodic (VBP) exhibit variable bit generation, occurring at consistent time intervals. Lastly, devices in the Variable-Bit Aperiodic (VBA) class display variable bit generation, but at irregular time intervals.

The primary contribution of this paper involves designing a hybrid approach that combines emulation and classification using a machine learning architecture that can be trained end-to-end. This method is formed by a combination of multiple multi-layer perceptron models to provide a solution method for the Massive Access Problem and to provide performance enhancement for IoT networks. We show that the designed method can analyze and react to the dynamic behavior of devices in the range of an IoT Gateway by imitating the JFS system. Then, we show that the method is applicable to real-time systems by presenting the computation time results.

The second contribution of this paper involves employing a trainable hybrid model that can be trained end-to-end and integrates various features, including the device count within each device category and the corresponding bit generation data for each device class. Via this model, we aim to achieve data fusion (i.e. feature elimination) and emulation of the JFS system. The results show that the proposed method can attain 90% of the maximum achievable throughput and 85% of the highest achievable energy consumption performance within the JFS system.

The strengths of the DFI-NFSC method are as follows:

- The DFI-NFSC method obviates manual tuning of the forecasting architecture by its automated forecaster selection.
- The execution time maintains a consistent pattern in relation to the number of devices.
- Our modular two-stage design reduces the space complexity of the architecture compared with a monolithic (non-modular) MLP for Network Forecasting Scheme Classifier.

The rest of this paper is organized as follows: In Section 2, we elucidate the relevance of this study within the context of current advancements in the field. Section 3 articulates the foundational assumptions that underscore the significance of our work. Section 4 provides an in-depth description of the method of the proposed architecture. In Section 5, we present the experimental setup and performance evaluation results of the method on real-world data. Lastly, Section 6 presents our research findings and conclusions.

2. Relationship to The State of The Art

In this section, we establish the link between the present article and the contemporary state of the art through two parts: (1) We conduct a comparative analysis between our study and existing literature that explores meta-MAC protocols, encompassing aspects such as switching, selection, configuration, or development of MAC layer protocols. (2) We outline the primary differences between our current article and prior studies in the literature that have centered on the massive access problem using the JFS System.

First, we contrast the current study with the studies in the literature that propose meta-MAC protocols for switching, selecting, re-configuring, or generating MAC-layer protocols (i.e. Time-Division Multiple Access (TDMA) and slotted ALOHA). References [12, 13, 14, 15, 16, 17, 18, 19, 20, 21] configure and optimize the MAC layer by switching protocols. Since these works focus on switching between existing protocols or generating a new one, in contrast, we have made our design to enable dynamic and automated forecaster switching on JFS [8] system forecasters. The DAFS system input is only related to the Gateways coverage area device classes, unlike these articles that have used probabilistic models or feedback systems to update the weights of the existing protocols. In addition, we use machine learning to switch between forecasting schemes while no machine learning algorithm is used for the protocols in this category.

Moreover, References [22, 23] focus on classifying MAC protocols based on the received energy, busy and idle duration as features. References [24, 25] focus on forecasting large, cumulative traffic patterns. Meta-MAC protocols are also designed based on machine learning algorithms, such as Random Forest, Sequential Minimal Optimization (SMO), Naive Bayes, and Support Vector Machine in [26, 27, 28, 29], Convolutional Neural Network (CNN) in [30], and Long-Short Term Memory in [31]. References [32, 33, 34, 35] demonstrated that based on experimental findings, machine-learning techniques are applicable for enhancing the quality of service (QoS) constraints of IoT networks and references [36, 37] proposed access schemes based on proactive methods where these methods predict the traffic of IoT devices to alleviate massive access problem. To this end, References in this part focused on generating a new protocol, reconfiguring the parameters of a protocol, or switching between a set of existing protocols. In contrast, we focus on advancing the network performance based on integrated data fusion of multiple features, emulation of the JFS system, and classification of forecasting schemes (MLP, LSTM, and ARIMA) for JFS.

Second, we compare this current work with the references based on the JFS. Reference [8] proposed the JFS system which performs scheduling by using forecast data of the traffic generation pattern of individual IoT devices in a heuristic manner for a single channel. Reference [9] proposed a multi-scale algorithm, utilizes JFS system work for a massive number of devices, and enables longer scheduling windows compared with [8]. Furthermore, in Reference [10], the JFS system was enhanced for a multi-channel version. In Reference [38], an Application Specific Error Function (ASEF) was introduced, demonstrating how forecasting errors impact the performance of the JFS system. Reference [39] proposed a technique for the heuristic part of JFS where the scheduling is based on queueing theory. Lastly, Reference [11] proposed an innovative framework called Dynamic, Automatic Forecaster Selection (DAFS) for addressing the Massive Access Problem of IoT. The focus is emulation between the number of IoT devices in the range of an IoT gateway and the performance of JFS via ANN. In contrast to previous works in this section, the proposed method works on a meta-level that performs integrated emulation and classification among the forecasters of JFS to enhance the performance. In contrast to the DAFS framework, the DFI-NFSC method utilizes the emulation by data fusion using the number of devices in each device class, the mean of the total number of bits generated by each device class, and the standard deviation of the total number of

Table 1: Contrast of DFI-NFSC against Articles in the Literature

References / Features	Meta-MAC Protocol	Proactive	Probabilistic	ML Based	Hybrid
DFI-NFSC	Yes	Yes	No	Yes	No
[13]	No	No	No	Yes	Yes
[14]	No	No	No	Yes	No
[17]	Yes	Yes	No	No	Yes
[20]	No	Yes	Yes	No	Yes
[22]	No	Yes	No	Yes	No
[8]	Yes	No	No	Yes	No
[23]	Yes	Yes	No	Yes	No
[24]	No	No	Yes	No	No
[25]	No	No	No	Yes	No
[26]	Yes	Yes	No	Yes	Yes
[27]	Yes	Yes	No	Yes	Yes
[28]	Yes	Yes	No	Yes	Yes
[29]	No	Yes	No	Yes	No
[30]	No	Yes	No	No	Yes
[31]	No	Yes	No	No	Yes
[32]	No	Yes	No	Yes	No
[33]	No	No	No	Yes	Yes
[34]	No	Yes	No	Yes	Yes
[35]	No	No	No	Yes	No

bits generated by each device class as features.

Reference [40], a Multi-layer perceptron decomposition architecture for mobile IoT indoor positioning proposed. The main perspective of the reference is high precision indoor positioning based on an end-to-end trainable two stage MLP architecture. In contrast to the reference, our efforts to concentrate on enhancing network performance on the meta-MAC layer by integrating data fusion on features, emulating the Forecaster Scheme, and switching among the forecasters (MLP, LSTM, and ARIMA in our demonstration).

In Table 1, we contrast our DFI-NFSC architecture with key references, focusing on the critical features of the proposed techniques to alleviate the Massive Access Problem. In the first column, we present the references against which we compare DFI-NFSC. In the second column, we show whether the proposed method can switch, select, re-configure, or generate MAC-layer protocols. In the third column, we present, for each reference, whether the technique proposed in that reference is proactive. The fourth and fifth columns display the algorithm details of the proposed techniques. In the sixth column, we display whether the technique in that reference is a hybrid one that is comprised of more than one elementary method.

3. Assumptions

In this paper, we let G denote a single IoT Gateway that has the JFS System working for all IoT devices that are registered in its coverage area. We let C_G denote the coverage area of G. Next, we let $\mathcal{N}$ denote the set of IoT devices (shortly "device") that are assumed to vary in C_G dynamically. Moreover, it is assumed that every device within the set C_G maintains a direct wireless connection with G at any particular time. In addition, we posit that when a device is situated within C_G, it remains connected to G continuously (regardless of whether it is actively transmitting data to G during that specific time) until the moment it departs from C_G. Next, we let N^1 denote the total number of devices in C_G and we let $\mathbf{N}^2$ denote a vector of IoT devices in C_G. We let N_{FBP}, N_{FBA}, N_{VBP}, and N_{VBA} denote the number of IoT devices that represent device classes FBP, FBA, VBP, and VBA, respectively. Furthermore, we let μ_{FBP}, μ_{FBA}, μ_{VBP}, and μ_{VBA} denote the mean number of bits generated by each of the FBP, FBA, VBP, and VBA classes for a 24-hour window, respectively. We let σ_{FBP}, σ_{FBA}, σ_{VBP}, and σ_{VBA} denote the standard deviation of the number of bits generated by each of the FBP, FBA, VBP, and VBA classes for a 24-hour window, respectively. We examine the mean and the standard deviation of the number of bits generated in addition to the total number of devices for each device class in a given coverage area because the former two give us additional information that will prove to be useful in selecting the forecasting scheme.

We establish the term "throughput" to represent the ratio of successfully transmitted bursts of bits over the total number of bits provided by $\mathcal{N}$ within a scheduling window, with η denoting this throughput. Moreover, we define "energy consumption" as the energy consumed during the transmission over all devices that fall in C_G for each succeeding bit delivery, and we let $\mathcal{E}$ denote this energy consumption. We let η_{ARIMA}, η_{MLP}, and η_{LSTM} denote the throughput of the JFS system under ARIMA, MLP, and LSTM forecasters, respectively. Next, we let $\mathcal{E}_{\text{ARIMA}}$, $\mathcal{E}_{\text{MLP}}$, and $\mathcal{E}_{\text{LSTM}}$ denote the energy consumption of the JFS system under ARIMA, MLP, and LSTM forecasters, respectively. Furthermore, we determine the error metric as a custom metric which is denoted by $f_{\mathcal{X}}$, where $\mathcal{X}$ refers to any desired network performance metric employed within the framework of JFS [11, 8].

[1] $N = N_{\text{FBP}} + N_{\text{VBP}} + N_{\text{FBA}} + N_{\text{VBA}}$

[2] $\mathbf{N} \equiv [N_{\text{FBP}}, N_{\text{VBP}}, N_{\text{FBA}}, N_{\text{VBA}}]$

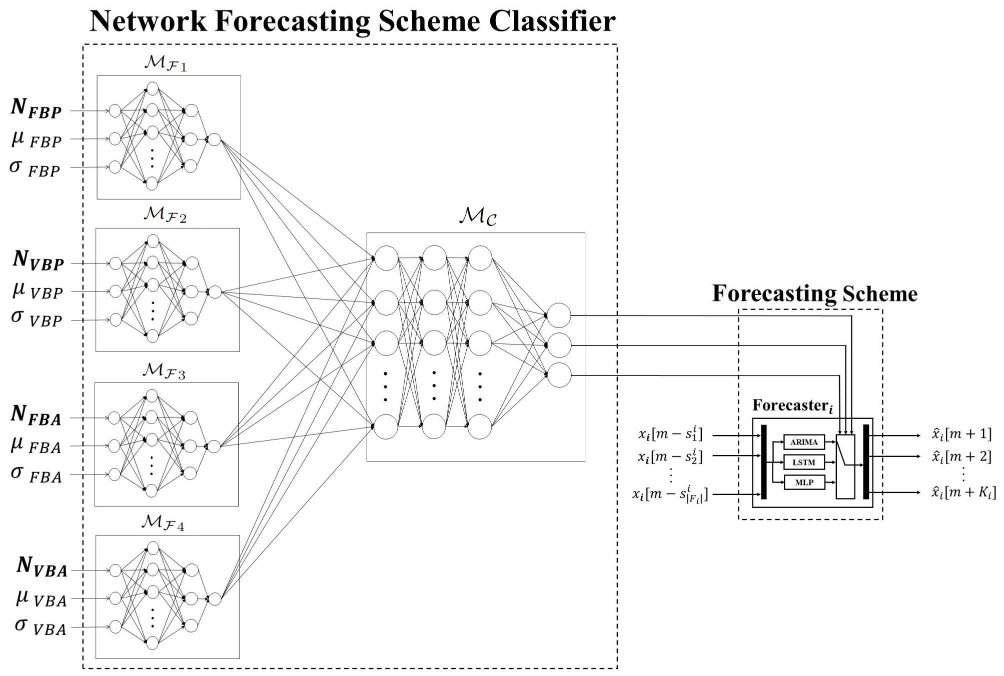

Figure 1: Architecture of the Data Fusion Integrated Network Forecasting Scheme Classifier (DFI-NFSC) based on the MLP Decomposition Model, where the Joint Forecasting-Scheduling (JFS) System embodies a compact Forecaster Scheme (FS) as Forecaster of device *i*

4. Design of the Data Fusion Integrated Network Forecasting Scheme Classifier (DFI-NFSC)

We shall now describe our DFI-NFSC architecture in detail, which we present visually in Fig. 1[3]. As shown in this figure, our architecture is comprised of two stages: The first stage, called the “Network Forecasting Scheme Classifier” (NFSC), is an artificial neural network (ANN) architecture whose purpose is to emulate (that is, mimic as closely as possible) any given system that forecasts and schedules the device traffic at the Medium Access Control (MAC) layer in a given coverage area. (In our demonstration in the next section, we shall use Joint Forecasting-Scheduling (JFS) as a particular example; however, the classifier that we shall build is applicable in general.)

[3]In order to simplify the explanation of the method, the figure is given for the throughput metric; any desired network performance metric can replace throughput in this figure.

The motivation for building such an emulator is that while the actual system behavior is complex, the emulator ANN learns the relationship between particular parameters that are key to the problem (which appear as input on the left hand side of the architecture in the figure) and the output, namely, the selection of the forecasting scheme to be employed, which appears on the right hand side of the figure as "Forecasting Scheme". In this figure, as an example, the forecasting scheme is to be selected among ARIMA, LSTM, and MLP. The output of the NFSC is a vector of three elements, which is 1-hot encoded to select exactly one of the three schemes (in this example).

The NFSC shall be trained on the data obtained from the JFS system (in our demonstration) such that the NFSC learns the relationship between the inputs that appear on the left (which shall be described below) and the highest-performing forecasting scheme obtained by running the JFS on each of the competing forecasting schemes separately. (We shall demonstrate in our results that the training time is reasonable.) Most importantly, once NFSC learns the relationship between the inputs and the highest-performing forecasting scheme, the selection of the forecaster will then be automated via the use of our architecture.

Inside the NFSC block that appears in the figure, we use a Multi-Layer Perceptron Decomposition architecture that is comprised of two stages: The first four subblocks that appear on the left hand side of the NFSC block is the set of processing modules, each of which processes the inputs for a specific device class. We let $\mathcal{M}_{\mathcal{F}j}$ denote the MLP that implements each of these subblocks where j denotes the index of the MLP network for device classes where $j \in \{FBP, VBP, FBA, VBA\}$. Each of these MLPs is identical and each network consists of $L_{\mathcal{M}_\mathcal{F}}$ layers with an output layer that has only a single neuron. By forcing the output layer to consist of only a single neuron, we aim to produce a compact representation output by each processing subblock. In the second stage, these four outputs, which are collected from the individual processing subblocks, are collected and fed into another MLP, denoted by $\mathcal{M}_C$ in the figure. This MLP $\mathcal{M}_C$ fuses the inputs from the prior four individual processing subblocks in order to output a 1-hot encoded vector that indicates the particular choice of one of the forecasting schemes.

We note that during supervised learning, the entire NFSC is end-to-end trained; that is, backpropagation passes through the entire ANN architecture that is comprised of the two stages described above such that the connection weights learn the relationship between the inputs to the NFSC and the choice of the highest-performing forecasting scheme that appears as the output of the

NFSC. (For our demonstration in the next section, we shall give a more detailed description of the particular choices of the hyperparameters in regard to our architecture.)

5. Results

5.1. Experimental Setup

In this section, we will present our methodology to realize a detailed traffic load on G. We use a dataset that is an enriched version of the dataset used in Reference [11]. In Reference [11], the dataset consists of the count of devices within each class that falls in C_G as input. In this work, we use two new features that do not appear in this dataset: the mean over each class for the total bit generation within a 24-hour timeframe, and the standard deviation over each class for the total bit generation within the same 24-hour period. These new features are formed from the online IoT traffic dataset [41, 8]. Recall that the $\mathcal{M}_C$ block in Fig 1 realizes classification; thus, we have transformed the throughput and energy consumption values which were computed for each $\mathbf{N}$ by JFS, into a one-hot encoding representation, where the best forecaster is represented by ”1” and the rest of the forecasters are represented by ”0”.

5.2. Performance Evaluation of the DFI-NFSC

In this section, we will present the performance of the DFI-NFSC method under the JFS system[4]. Furthermore, the DFI-NFSC method outputs the best forecaster, which is represented via one-hot encoding on the vector $\mathbf{N}$ and the bit generation data of devices in C_G. Thus, the application of error metric f_X is implemented for the performance evaluation using the forecaster that corresponds to the output of the DFI-NFSC method and the forecaster that has the highest performance. In contrast to the Reference [11], the DFI-NFSC method directly gives results as the forecaster to be selected.

In Table 2, we provide the network stages of DFI-NFSC and hyperparameters and the search set for each stage. In the first column, we show the name of the model. Next, we show the representation of stages of the model in the second column. Third column displays the hyperparameters of the model. Lastly, the fourth column shows the value/search set of the model.

[4]Recall that the DFI-NFSC method is a meta-level architecture, which emulates any given target system. The JFS system that is used in this work is thus only one example of such a target system whose emulation is performed.

Table 2: Hyperparameters of Stages in DFI-NFSC

Model	Stage of the Model	Hyperparameters	Value / Search Set
MLP	$\mathcal{M}_{\mathcal{F}}$	The number of layers $L_{\mathcal{M}_{\mathcal{F}}}$	4
		The number of neurons $P_l^{\mathcal{M}_{\mathcal{F}}}$ at layer $L_{\mathcal{M}_{\mathcal{F}}}$	1
		The activation function of each neuron at each layer	*tanh*
		The activation function of each neuron at output layer	*tanh*
		The number of layers $L_{\mathcal{M}_{C}}$	4
MLP	$\mathcal{M}_{C}$	The number of neurons $P_k^{\mathcal{M}_{C}}$ at layer $L_{\mathcal{M}_{C}}$	3
		The number of neurons $P_k^{\mathcal{M}_{C}}$ at each layer $k \in \{1, \ldots, L_{\mathcal{M}_{C}} - 1\}$	$\{2 * o\}_{o \in \{1,\ldots,75\}}$
		The activation function of each neuron at the input layer and each hidden layer	*tanh*
		The activation function of each neuron at output layer	*softmax*

5.2.1. Performance Comparison of JFS under the DFI-NFSC Method and DAFS Method

In Fig. 2 and 3, we illustrate the comparative performance of DFI-NFSC and DAFS under the box plots to represent cross-validation errors for both throughput and energy consumption. These comparisons are made concerning the same error metrics, denoted as f_η and $f_\mathcal{E}$, across an incrementally growing number of devices, denoted as N. In both Fig. 2 and Fig. 3, we show the box plot, where the DFI-NFSC is represented by the right box and the DAFS is represented by the left box for each point in x-axis[5]. Each box in the figure represents a fixed N for which the 10-fold cross-validation is performed. Each box represents the maximum value and the minimum value for N. Furthermore, each box has a horizontal line which represents the median of f_η in Fig. 2 and $f_\mathcal{E}$ in Fig. 3.

Fig. 2 shows that DFI-NFSC outperforms DAFS for all N with respect to f_η, and it is less than 0.1 for all N except $N = 700$, $N = 900$, and $N = 1000$, where the minimum value and the median are still lower than the DAFS method. In detail, for these samples, DFI-NFSC performs the selection of forecasters by 90% of its highest throughput performance for JFS. In addition, the increasing trend in error can be observed in the figure for larger N as emulation. However, the margin between the maximum and minimum value of error is lower than the results for DAFS. In contrast to the relatively small error margin and the increasing trend in error for larger N, the DFI-NFSC method has the largest error of 0.12 at $N = 700$, but it performs 88% of its highest performance in these samples.

In Fig. 3, we see a similar downtrend in the energy consumption box plot where DFI-NFSC outperforms DAFS except at $N = 500$ and $N = 600$. For

[5]Recall that the architecture and parameters of these methods are entirely different: The DAFS method performs emulation whereas the DFI-NFSC method performs end-to-end data fusion, emulation, and classification

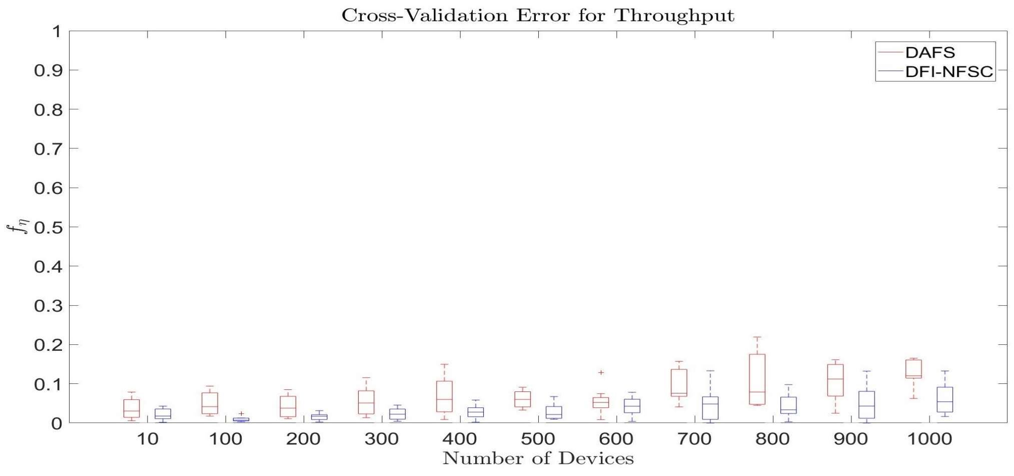

Figure 2: Box plot of f_η for the DFI-NFSC method under 10-fold cross-validation with respect to the total number of devices N in the C_G.

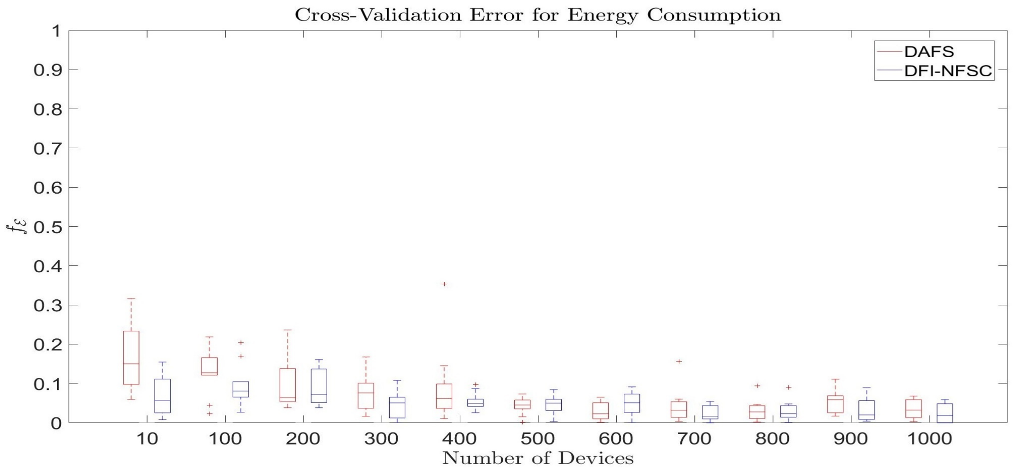

Figure 3: Box plot of $f_\mathcal{E}$ for the DFI-NFSC method under 10-fold cross-validation with respect to the total number of devices N in the C_G.

all N, the median of $f_\mathcal{E}$ is lower than 0.1 and the highest error $f_\mathcal{E}$ is 0.15. In particular, the highest error for DFI-NFSC occurs at $N = 10$, which equals the median of DAFS, and the highest error for DAFS at $N = 10$ is 0.3. Moreover, the median fluctuates on the same value of error, even though the result is obtained by using cross-validation.

In addition, the caveats in using our approach are as follows: First, there exists additional computational load incurred due to the offline search for the optimal selection of the forecaster. Second, the results show that the performance of the proposed method for energy consumption cannot achieve the same level of performance that is obtained for throughput. Recall that, for the throughput, the DFI-NFSC method performed 90% of the highest attainable performance whereas it can achieve 85% of the highest attainable performance on energy consumption of the JFS system.

5.2.2. Generalizaiton Ability of DFI-NFSC

We present the cross-validation error for throughput via a histogram in order to demonstrate the generalization ability of our DFI-NFSC method. To this end, we randomly create 100 DFI-NFSC architectures in order to analyze the robustness of forecaster selection for JFS. Subsequently, we present the results with respect to the metrics f_η and $f_\mathcal{E}$.

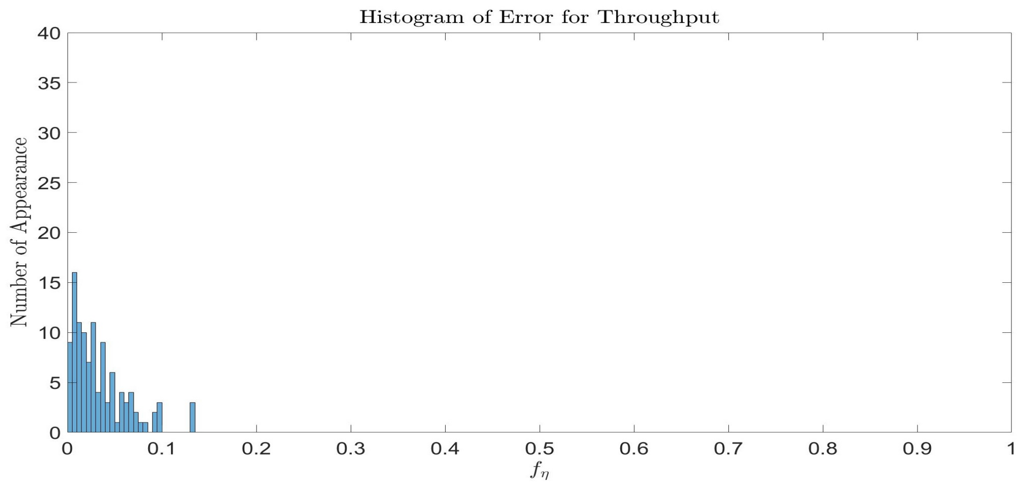

Figure 4: Histogram of the 10-fold cross-validation for f_η, forecaster selection performance of DFI-NFSC.

In Fig. 4, we see that f_η is less than 0.1 except at the outliers where f_η is 0.14. In addition, the f_η is skewed towards the left which is observable that the majority of the samples have error less than 0.05. The peak occurrence of the f_η is in the range of 0.01-0.04 and it drops through this range. Furthermore, in Fig. 5, we see that $f_\mathcal{E}$ is less than 0.2, and again $f_\mathcal{E}$ skewed towards left and the majority of samples have error less than 0.06. The highest formation of $f_\mathcal{E}$ realized in the range of 0-0.02 with 0.05-0.07 intervals, and it drops significantly over this level. This shows that selection of the low performing forecaster is uncommon. DFI-NFSC performs highly robust performance in selecting the

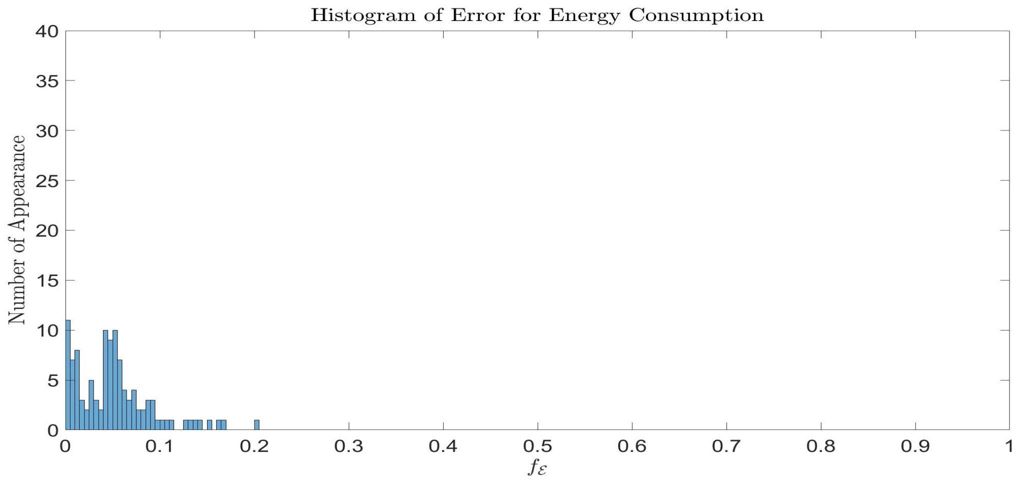

Figure 5: Histogram of the 10-fold cross-validation for $f_{\mathcal{E}}$, selection performance of DFI-NFSC.

Forecasting Scheme even when the parameters of the architecture are randomly generated.

5.3. Training and Execution Time

In this section, we present the training and execution time of DFI-NFSC in order to assess its practical applicability. We have used Tensor Processing Unit (TPU) accelerator of Google Colab to obtain these test results. The results were obtained by calculating the mean of the training time and standard deviation of training time across each cross-validation fold. In Fig. 6, training time is between 6 and 10 seconds. In Fig. 7, execution time is between 0.17 and 0.2 seconds. The training time and execution results do not change with respect to *N* since the input, output size and the number of samples are fixed. Thus, the results show that the computational complexity of DFI-NFSC scales well with the number of devices.

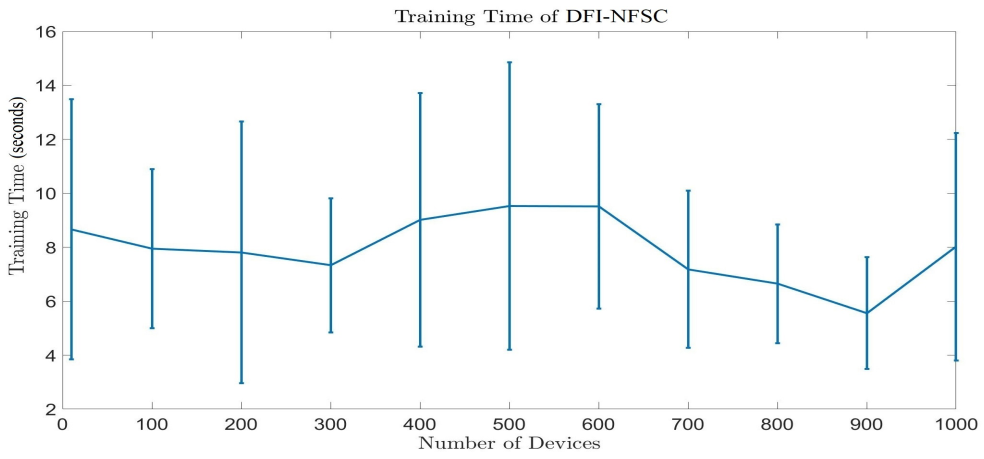

Figure 6: The mean training time of the DFI-NFSC, along with error bars representing the standard deviation, is depicted with respect to the total number of devices N in the C_G.

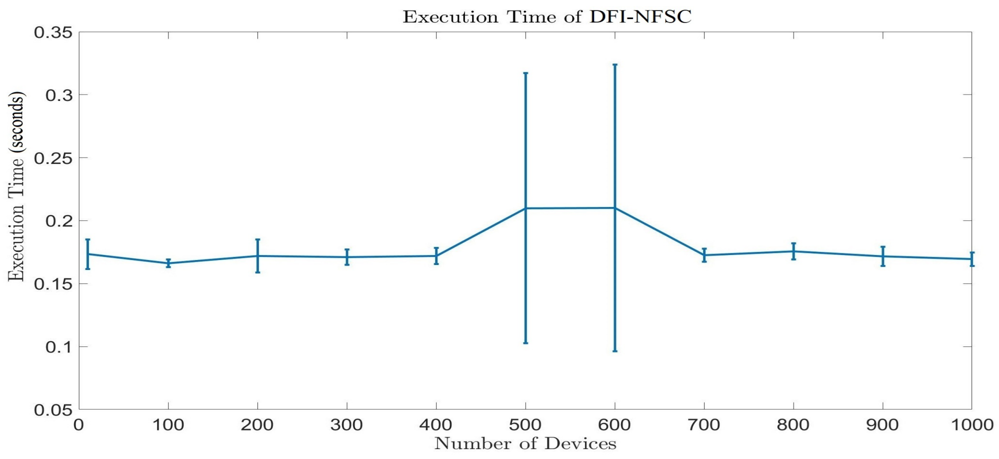

Figure 7: The mean execution time of the DFI-NFSC, along with error bars representing the standard deviation, is depicted with respect to the total number of devices N in the C_G.

6. Conclusions

In this study, we have developed a hybrid method based on cascaded data fusion, emulation, and classification, which is called the “Network Forecaster Scheme Classifier” for IoT networks. The proposed approach has multiple Multi-Layer Perceptron (MLP) model structures, where the data fusion/feature elimination is realized. We have designed the model in two stages. In the first stage, the model takes inputs consisting of the number of devices in each class and the corresponding bit-generation data for each class. This is done to accomplish data fusion and feature selection. Thus, in the second stage, the classification model switches between the forecasting schemes by using the output of the first stage.

We present the results of this current article by utilizing the method on JFS system. The results are based on 10-fold cross-validation; the DFI-NFSC method performs the forecaster scheme selection by succeeding 90% of the highest attainable performance on throughput and achieving 85% of the highest attainable performance on energy consumption for the JFS System. In addition, the DFI-NFSC method outperforms the DAFS method by lowering the error by almost 50% for each N.

In future work, the DFI-NFSC method can be modified to multi-channel JFS. Second, the methods in this paper can be extended to different application domains, such as the optimization of electricity consumption on a smart grid. Third, the methods in this paper can be applied to models, which makes real-time forecaster selection. We note that the methods that have been developed in this paper can be applied to contexts outside its immediate domain and thus serve as a springboard for research into a rich variety of applications.

The method can learn the dynamic environment of the IoT Gateway, regardless of whether the devices are static or mobile, and perform the network forecaster scheme switching through the classification within real-time and based on real-world data. Thus, the method significantly enhances the QoS constraints of IoT networks that has dynamic environment.

References

[1] R. Saksena, D. Lu, C. Zaidi, Ilyas, Ericsson mobility report november 2021 (Nov. 2021). URL `https://www.ericsson.com/en/reports-and-papers/mobility-report/reports/november-2021`

[2] L. Tello-Oquendo, D. Pacheco-Paramo, V. Pla, J. Martinez-Bauset, Reinforcement learning-based ACB in LTE-A networks for handling massive M2M and H2H communications, in: 2018 IEEE International Conference on Communications (ICC), IEEE, 2018, pp. 1–7.

[3] A. O. Almagrabi, R. Ali, D. Alghazzawi, A. AlBarakati, T. Khurshaid, A Poisson process-based random access channel for 5G and beyond networks, Mathematics 9 (5) (2021) 508.

[4] J. Liu, L. Song, et al., A novel congestion reduction scheme for massive machine-to-machine communication, IEEE Access 5 (2017) 18765–18777.

[5] H. Jin, W. T. Toor, B. C. Jung, J.-B. Seo, Recursive pseudo-Bayesian access class barring for M2M communications in LTE systems, IEEE Transactions on Vehicular Technology 66 (9) (2017) 8595–8599.

[6] M. Nakip, B. C. Gül, V. Rodoplu, C. Güzeliş, Comparative study of forecasting schemes for

IoT device traffic in machine-to-machine communication, in: Proceedings of the 2019 4th International Conference on Cloud Computing and Internet of Things, 2019, pp. 102–109.

[7] M. Nakip, K. Karakayali, C. Güzeliş, V. Rodoplu, An end-to-end trainable feature selection-forecasting architecture targeted at the Internet of Things, IEEE Access 9 (2021) 1–1. doi:10.1109/ACCESS.2021.3092228.

[8] M. Nakip, V. Rodoplu, C. Güzeliş, D. T. Eliiyi, Joint forecasting-scheduling for the internet of things, in: 2019 IEEE Global Conference on Internet of Things (GCIoT), IEEE, 2019, pp. 1–7.

[9] V. Rodoplu, M. Nakıp, D. T. Eliiyi, C. Güzelis, A multi-scale algorithm for joint forecasting-scheduling to solve the massive access problem of IoT, IEEE Internet of Things Journal 7 (9) (2020) 8572–8589. doi:10.1109/JIOT.2020.2992391.

[10] V. Rodoplu, M. Nakip, R. Qorbanian, D. T. Eliiyi, Multi-channel joint forecasting-scheduling for the internet of things, IEEE Access 8 (2020) 217324–217354.

[11] M. Nakip, E. Çakan, V. Rodoplu, C. Güzeliş, Dynamic automatic forecaster selection via artificial neural network based emulation to enable massive access for the internet of things, Journal of Network and Computer Applications 201 (2022) 103360.

[12] V. Petkov, K. Obraczka, The case for using traffic forecasting in schedule-based channel access, in: 2011 IEEE Consumer Communications and Networking Conference (CCNC), 2011, pp. 208–212.

[13] V. Petkov, K. Obraczka, Collision-free medium access based on traffic forecasting, in: 2012 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2012, pp. 1–9.

[14] M. Ateeq, F. Ishmanov, M. K. Afzal, M. Naeem, Predicting delay in IoT using deep learning: a multiparametric approach, IEEE Access 7 (2019) 62022–62031.

[15] Y. Liu, C. Yuen, X. Cao, N. U. Hassan, J. Chen, Design of a scalable hybrid MAC protocol for heterogeneous M2M networks, IEEE Internet of Things Journal 1 (1) (2014) 99–111.

[16] M. El Tanab, W. Hamouda, A scalable overload control algorithm for massive access in machine-to-machine networks, in: 2017 IEEE International Conference on Communications (ICC), IEEE, 2017, pp. 1–6.

[17] N. Flick, D. Garlisi, V. R. Syrotiuk, I. Tinnirello, Testbed implementation of the meta-MAC protocol, in: 2016 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2016, pp. 580–585.

[18] M. Sha, R. Dor, G. Hackmann, C. Lu, T.-S. Kim, T. Park, Self-adapting MAC layer for wireless sensor networks, in: 2013 IEEE 34th Real-Time Systems Symposium, IEEE, 2013, pp. 192–201.

[19] J. Ansari, X. Zhang, A. Achtzehn, M. Petrova, P. Mähönen, A flexible MAC development framework for cognitive radio systems, in: 2011 IEEE Wireless Communications and Networking Conference, IEEE, 2011, pp. 156–161.

[20] J. Zhen, V. Rodoplu, Automated MAC protocol generation for dynamic topologies, in: 2012 IEEE Global Communications Conference (GLOBECOM), IEEE, 2012, pp. 439–444.

[21] S. Hu, Y.-D. Yao, Z. Yang, MAC protocol identification approach for implement smart cognitive radio, in: 2012 IEEE International Conference on Communications (ICC), IEEE, 2012, pp. 5608–5612.

[22] Y. Chu, P. D. Mitchell, D. Grace, Aloha and q-learning based medium access control for wireless sensor networks, in: 2012 International Symposium on Wireless Communication

Systems (ISWCS), IEEE, 2012, pp. 511–515.

[23] R. V. Kulkarni, G. K. Venayagamoorthy, Neural network based secure media access control protocol for wireless sensor networks, in: 2009 international joint conference on neural networks, IEEE, 2009.

[24] Z. Chen, J. Wen, Y. Geng, Predicting future traffic using hidden markov models, in: 2016 IEEE 24th international conference on network protocols (ICNP), IEEE, 2016, pp. 1–6.

[25] S. Chinchali, P. Hu, T. Chu, M. Sharma, M. Bansal, R. Misra, M. Pavone, S. Katti, Cellular network traffic scheduling with deep reinforcement learning., in: AAAI, 2018, pp. 766–774.

[26] M. Qiao, H. Zhao, S. Wang, J. Wei, MAC protocol selection based on machine learning in cognitive radio networks, in: 2016 19th International Symposium on Wireless Personal Multimedia Communications (WPMC), 2016, pp. 453–458.

[27] M. Qiao, H. Zhao, S. Huang, L. Zhou, S. Wang, An intelligent MAC protocol selection method based on machine learning in wireless sensor networks., KSII Transactions on Internet & Information Systems 12 (11) (2018).

[28] S. Hu, Y.-D. Yao, Z. Yang, MAC protocol identification using support vector machines for cognitive radio networks, IEEE Wireless Communications 21 (1) (2014) 52–60.

[29] E. Eldeeb, M. Shehab, H. Alves, A learning-based fast uplink grant for massive IoT via support vector machines and long short-term memory, IEEE Internet of Things Journal (2021).

[30] X. Zhang, W. Shen, J. Xu, Z. Liu, G. Ding, A MAC protocol identification approach based on convolutional neural network, in: 2020 International Conference on Wireless Communications and Signal Processing (WCSP), IEEE, 2020, pp. 534–539.

[31] H. Li, S. Peng, Z. Chen, X. Qin, MAC protocol recognition based on LSTM network in cognitive radio, Journal of Signal Processing 35 (5) (2019).

[32] Z. Guan, Z. Wang, Y. Cai, X. Wang, Deep reinforcement learning based efficient access scheduling algorithm with an adaptive number of devices for federated learning IoT systems, Internet of Things 24 (2023).

[33] L. N. CheSuh, R. Á. Fernández-Diaz, J. M. Alija-Perez, C. Benavides-Cuellar, H. Alaiz-Moreton, Improve quality of service for the internet of things using blockchain & machine learning algorithms, Internet of Things 26 (2024).

[34] A. Akbas, H. U. Yildiz, A. M. Ozbayoglu, B. Tavli, Neural network based instant parameter prediction for wireless sensor network optimization models, Wireless Networks 25 (2019).

[35] S. Khan, J. Gomes Jr, M. H. ur Rehman, D. Svetinovic, Dynamic behavior assessment protocol for secure decentralized federated learning, Internet of Things 24 (2023).

[36] L. Ruan, M. P. I. Dias, E. Wong, Machine learning-based bandwidth prediction for low-latency h2m applications, IEEE Internet of Things Journal 6 (2) (2019).

[37] M. Shehab, A. K. Hagelskjær, A. E. Kalør, P. Popovski, H. Alves, Traffic prediction based fast uplink grant for massive iot, in: 2020 IEEE 31st annual international symposium on personal, indoor and mobile radio communications, IEEE, 2020.

[38] M. Nakip, A. Helva, C. Güzeliş, V. Rodoplu, Subspace-based emulation of the relationship between forecasting error and network performance in joint forecasting-scheduling for the internet of things, in: 2021 IEEE 7th World Forum on Internet of Things (WF-IoT), IEEE, 2021, pp. 247–252.

[39] M. Nakip, E. Gelenbe, Randomization of data generation times improves performance of predictive IoT networks, in: 2021 IEEE World Forum on Internet of Things (WF-IoT),

2021, In Press.

[40] E. Çakan, A. Şahin, M. Nakip, V. Rodoplu, Multi-layer perceptron decomposition architecture for mobile iot indoor positioning, in: 2021 IEEE 7th World Forum on Internet of Things (WF-IoT), IEEE, 2021, pp. 253–257.

[41] IoT Traffic Generation Pattern Dataset (Jan 2021).
URL `https://www.kaggle.com/tubitak1001118e277/iot-traffic-generation-patterns`