Received 7 June 2023, accepted 19 June 2023, date of publication 23 June 2023, date of current version 28 June 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3289076

RESEARCH ARTICLE

Resource Aware Long Short-Term Memory Model (RALSTMM) Based On-Device Incremental Learning for Industrial Internet of Things

ATALLO KASSAW TAKELE AND BALÁZS VILLÁNYI

Department of Electronics Technology, Faculty of Electrical Engineering and Informatics, Budapest University of Technology and Economics, 1117 Budapest, Hungary

Corresponding author: Atallo Kassaw Takele (atallo.takele@edu.bme.hu)

ABSTRACT The interconnection of instruments (i.e., actuators and sensors) networked together for industrial applications brings about the Industrial Internet of Things (IIoT). This connectivity enables the collection, sharing, and analysis of data to enhance the efficiency and productivity of manufacturing. Machine learning models are popular methods for analyzing massive time-series data collected by industrial control systems. Classical Long Short-Term Memory (LSTM), which is a widely used time-series model, learns patterns by feeding the entire dataset at once, and the model remains fixed. However, real-world industrial control system nodes generate new data. This paper proposes Resource Aware Long Short-Term Memory Model (RALSTMM) based incremental learning for edge devices in the IIoT. In RALSTMM edge devices can collect and analyze data for various predictive applications. The proposed RALSTMM can be deployed on these tiny edge devices and can be updated to enhance existing knowledge using newly generated data. The RALSTMM gradually learns from newly collected data by leveraging crucial information from previously analyzed data, thereby minimizing the resources needed for training. Hence, previous data that has been processed earlier would not undergo further analysis as the model has already extracted the necessary knowledge from it. The performance of RALSTMM has been evaluated with Mean Squared Error (MSE), accuracy, recall, precision, information criteria and processing time using three IoT testbed datasets. A comparative experimental demonstration of the RALSTMM with the existing LSTM proves the effectiveness of the RALSTMM by reducing processing time and maintaining its performance.

INDEX TERMS Industrial IoT, LSTM, incremental learning, on device learning.

I. INTRODUCTION

An Industrial Internet of Things (IIoT) is an aggregate of various sensors, actuators, controlling devices, intelligent wireless communication gateways, and intelligent data analysis tools [1], [2]. Nowadays, industries are supported by IIoT for improving supply chain optimization, plant safety, quality control, predictive maintenance, employee and environmental safety, automated and remote equipment, and monitoring. IIoT supports the manufacturing process through asset monitoring or conditioning, quality control, predictive maintenance, and overall process automation [3]. It is enabled by artificial intelligence, big data, cloud computing, the Internet, the Internet of Things, block chain, and robotics [4]. Recently, modern factories have been systematically supported by tiny devices (i.e., smart embedded devices, sensors, and actuators). These tiny, embedded devices communicate through resource-efficient communication protocols such as 6LoWPAN, CoAP, RPL (Routing Protocol for Low Power and Lossy Network), and MQTT. In the future, factories are supposed to control asset management through intelligent technologies [3].

Machine learning is one of the core parts of the IIoT for enhancing performance and security [7]. The deployment of various sensors and actuators produces enormous amounts of data about the manufacturing process and overall system

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

activities. The collection of this massive data can be used to extract knowledge, insights about the status of instruments and the overall industrial system process. Various machine learning algorithms were proposed and practiced in several domains of Industry 4.0 [10]. Key manufacturing components like asset monitoring or conditioning, quality control, building automation, predictive maintenance, and overall process automation are being advanced and practically supported by machine learning algorithms. Traditional machine learning algorithms like K-nearest neighbor, decision trees, random forests, and regression analysis were common for the maintenance prediction of production assets [10]. Besides these classical machine learning methods, deep learning algorithms have become popular and are preferred by domain experts for their better performance. Deep learning algorithms can learn insights using large volumes of historical data with better prediction accuracy, computational performance, and explainability [9], [10].

Most of the data generated by industrial control system devices is time dependent [11]. Sensors and actuators targeted for different purposes observe the environment based on a time sequence. Therefore, classical classification and regression methods are not sufficient to learn the pattern of most of the data collected from industrial sensors. Nowadays, sequence models are popular for understanding time series patterns and are mostly used for predicting financial time series data, text processing, speech data processing, sensors, sequence value prediction, etc. [9]. Recurrent Neural Networks (RNN), Convolutional Neural Networks (CNN), and Autoregressive Integrated Moving Average (ARIMA) are some of the common sequential algorithms for learning sequential patterns [21]. LSTM (Long Short-Term Memory) is among the variants of RNN, which is the most popular for time series analysis. Vanilla RNN is susceptible to vanishing and exploding gradient problems while updating weights and biases for a long time [9]. LSTM was proposed to overcome the issue of gradient vanishing and exploding in RNN by maintaining long-term dependency [13].

Classical LSTM models learn patterns by feeding the whole dataset at once, and the model remains fixed [12]. However, real-world industrial control system nodes generate new data over time. Hence, the model should be updated with new data to enhance its performance. Incremental learning promises to update new knowledge in the existing trained model based on previous time steps [30]. Through incremental learning, continuous lifetime learning and knowledge accumulation can be achieved [14], [15], [35]. It continuously accepts new inputs from the environment and executes the model to extract better results.

The data analytics process was mostly accomplished by cloud services using MLaaS (Machine Learning as a Service) through the transmission of collected data from edge devices to the server [6]. Edge devices' self-collected confidential data should be shared with the central server for aggregation and analysis [5]. However, recent industrial systems require on-device learning to enhance the security of the system and communication costs [8], [17]. On-device learning enables faster, real-time and more flexible decisions making for time critical applications. Recently, edge devices' computational power and storage space becomes advanced to handle machine learning tasks [16], [17]. However, the demands for storage space and computational power becomes higher for training advanced machine learning algorithms in order to achieve better performance.

IIoT nodes generate data continuously over time, which increases the volume of the dataset. Edge devices can utilize data processing to make predictions for various purposes. A large volume of data enables better performance of the prediction, but computational resource demand gets higher. The machine learning model deployed on these devices can be updated to enhance existing knowledge with newly generated data. However, analyzing the whole dataset, which is collected continuously for a long time, is a challenging task for resource deficient devices [16], [20]. A number of works have been done in this aspect and some of them proposed a federated learning approach [8], [18]. Other studies such as forming weak learners and aggregating them to form a strong learner [12], [34], and various types of incremental learning approaches [30], [31], [33] were investigated. However, these approaches are not sufficient enough in terms of resource utilization, complexity and performance.

In this paper, we proposed a modularized incremental learning approach named Resource Aware Long Short-Term Memory Model (RALSTMM) for reducing computational resource. It is a typical on-device incremental learning in which edge devices collect their own data and analyze it by themselves. Over time, RALSTMM acquires knowledge by leveraging newly generated data and transferring insights from previous data. It is executed modularly by taking input data over a certain range of time. In this manner, only essential information can be transferred from previously executed models with datasets collected over a certain range of time. This reduces resource consumption without affecting the performance as it transfers essential information. The comparative experimental evaluation with the existing LSTM using three different datasets proves that the RALSTMM reduces computational time by maintaining performance.

The contribution of this paper is presented as follows

- Proposes Resource Aware Long Short-Term Memory Model (RALSTMM) based on-device incremental learning for minimizing resource by maintaining performance
- The proposed RALSTMM accepts self-collected data continuously for a certain range of time as an input
- The RALSTMM extracts essential information from the previously analyzed dataset and transfers knowledge to the next time range
- We conducted considerable experiments using three real world datasets for demonstrating the feasibility of the RALSTMM

The remaining part of the paper is presented as follows: Section II details the background and related works in the proposed specific area, and Section III presents the methodology of the proposed RALSTMM. Section IV discusses the experimental demonstration process and results, and Section V presents the conclusion of the proposed study.

II. LETRATURE REVIEW

Liu et al. [8] present a federated based anomaly detection model using deep LSTM. It is a collaborative training model that involves multiple edge devices for the IIoT. The model includes both edge devices and cloud aggregators. It also consists of a gradient compression mechanism for minimizing bandwidth overload and an anomaly detection mechanism. This approach uses gradient compression and applies federated learning to preserve privacy. However, it is an offline learning approach, and most IIoT applications generate data over time. Taheri et al. [18] present a malware detection application using a federated learning architecture for the IIoT. It applies to a generative adversarial network (GAN) for edge devices, and the server controls the collaboration and performs aggregation. This is also an offline learning approach, which is not applicable to continuously generated data. The authors in [12] introduced an LSTM based incremental approach that dynamically updates according to the new sequential data. The collected data is divided into subdatasets, and weak learning is applied to the newly generated data. A strong learner is formed by combining the old and new weak learners. This method consumes extra resources for combining the old and new weak learners. S. Disabato and M. Roveri [15] present an incremental model based on k-nearest neighbor and transfer learning. It is an on-device learning solution for IoT and embedded system applications. This approach is based on classical k-nearest neighbor, which is not applicable for time series applications.

Li et al. [30], present Adaptive Threshold Hierarchical Incremental Learning (ATHIL) for tackling the issue of deploying complex models on resource restricted devices. It applies the coefficient of dispersion for sparsening network weights in order to minimize the complexity of the model. This approach sparsens the weight matrix with a certain percentage, which affects the performance in critical applications. The authors in [31], present dynamic support network (DSN) incremental learning that offers a regularization approach in the feature space to address the issues of overfitting and forgetting commonly observed in Few-shot class incremental learning. In order to combine the representation ability of new classes, DSN temporarily expands network nodes. Although DSN merges old and new classes systematically to improve predictive performance, it is a complex model to apply in resource constraint devices. Chen et al. [31], propose transductive support vector-based incremental learning (ILTSVM) to predict labels for specific unlabeled instances in order to manage large volumes of data. The authors didn't mention whether the proposed method could be applied to a real-time stream of data, despite demonstrating improvements in terms of performance and complexity.

Although the aforementioned studies promised to improve intelligent data analytics in various aspects, the issue of resource optimization in line with consistence performance is still critical for tiny devices in the IIoT and needs additional investigation. Hence, this paper provides resource efficient approach by maintaining the performance.

A. INDUSTRIAL INTERNET OF THINGS ARCHITECTURE

IIoT is a combination of several sensors, actuators, and other controlling embedded devices [2], [18]. These smart devices should be connected to the internet for collaboration, exchanging information, and sharing resources. These emerging technologies allow physical objects to understand the environment (thinking, hearing, seeing, and performing other tasks) by communicating with each other [28]. IIoT supports industries by improving supply chain optimization, plant safety, quality control, predictive maintenance, employee and environmental safety, automated and remote equipment, monitoring, etc. Data visualization enabled by IIoT can be used to track the movement of products and control the physical environment for better human decision-making [19].

There is no consistent IoT architectural framework for industrial applications [19], [28]. A typical general communication architecture of the IIoT is depicted in Figure 1. It has three layers communicating with each other through different protocols [19]. The bottom one is the device layer which includes different kind of sensors and actuators for collecting information from the environment. The next is network layer which contains different communication technologies such as WiFi, ZigBee, RFID and Bluetooth for managing secure transmission of data collected by sensors and actuators. The application layer is on the top which includes specific intelligent data processing tasks. Services requested by customers are handled by the application layer. Nowadays, the majority of IIoT devices are powered by the well-known open- source Android operating system, which is suitable for artificially intelligent developers [18]. The application layer performs data processing, cloud computing, and decision-making. The raw data sent from the device should be preprocessed and analyzed for characterization and better decision-making. Specific intelligent machine learning algorithms are also executed in this layer after technical preprocessing. The predicted output from machine learning algorithms will make machine-based and human decision-making. The network layer allows the secure transmission of raw data from the device layer to the application. This could be achieved by one of the application specific protocols among several alternatives [28]. Some examples of IIoT communication protocols are RFID, ZigBee, Bluetooth, IEEE 802.15.4, Z-wave, 5G, and WiFi. Targeted information from the environment like temperature, pressure, location, and vibration could be gathered by the edge devices. Specific information gathered for different purposes is transferred to the application layer for further processing and decision-making.

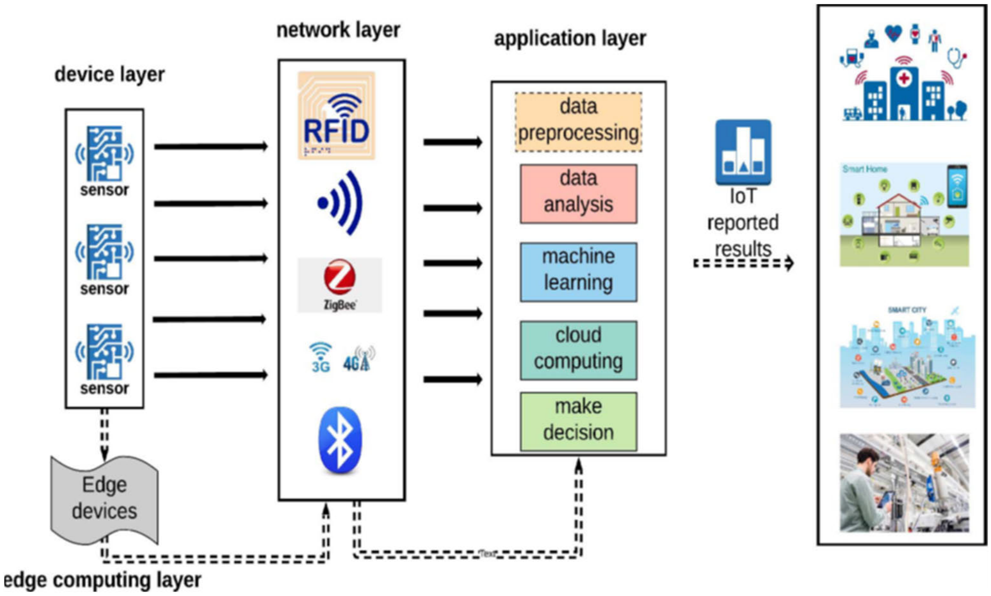

FIGURE 1. A typical Industrial IoT architecture [19].

B. SEQUENTIAL LEARNING

Sequence learning is designed to model sequential problems with sequential input or output using an ordered dataset. Sequential problems include sequence generation, sequence prediction, sequence recognition, and sequence classification [22]. Sequence classification takes a sequence of inputs to distinguish the class label. Sequence prediction helps in predicting the next value from the existing sequence. Sequence generation produces another sequence based on the given input values that has similar behavior to the previous input sequence [29]. Sequence recognition decides if the sequence is legitimate based on a certain scenario. The ordered data can be the sensor's time-series data, audio clips, video clips, text streams, etc. [9], [22]. It can be applied to various sensor time series predictions, sequence value predictions, name entity recognition, financial time series predictions, text processing, etc. [9].

Several machine learning algorithms have been proposed for solving sequence problems. Some of the most widely mentioned ones are ARMA and RNN, with their variants [23]. ARMA is a classic method for modeling single-time series data, a combination of AR (Auto-Regressive) and Moving Average (MA). Autoregressive Integrated Moving Average Model (ARIMA) and SARIMA (Seasonal Autoregressive Integrated Moving Average Model) are also the two most widely used improved variants of ARMA. RNN is variant of neural network for handling time dependent problems. An LSTM is a category of RNN that memorizes long-term dependency problems.

RNN can memorize the past information to improve the performance of sequential problems [23]. In traditional artificial neural networks (ANN), the output solely resides on the current input. However, the output on the current point depends on the past information and the current input. It is an aggregate of connected ANN model in which the current model can acquire previous knowledge from the past model. The information acquired in the previous stages is stored in the hidden layers of the RNN network. Figure 2, depicts the architecture of RNN which takes sequences of inputs (x_i) at the bottom, undergoes step by step process (A) and reveals outputs at the top (h_i).

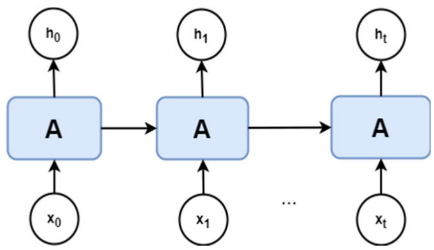

FIGURE 2. Recurrent neural networks structure.

However, vanilla RNN has two main challenges. The method used for updating the weights is gradient descent algorithm, which may be affected by vanishing and exploding gradient [24]. Vanishing gradient occurs when the gradient decreases and gets to zero over time which leads the weights to be constant. Initially, the LSTM structure decides whether to throw or keep the information, which is accomplished with a forget gate (sigmoid function) in Eq. (1) [8], [24]. In exploding gradient, the gradient increases through time which leads to larger weights. A special type of RNN has been proposed with an internal memory to remember long term dependency. LSTM network is able to remove or add information to the cell state which is adjusted by gated

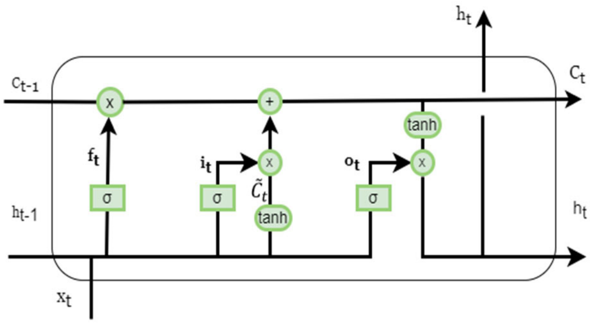

FIGURE 3. Basic LSTM structure.

structures.

$$
\begin{aligned}
i_t &= \sigma\left(W_i . [h_{t-1}, x_t] + b_i\right) \\
f_t &= \sigma\left(W_f . [h_{t-1}, x_t] + b_f\right) \\
\tilde{C}_t &= tanh\left(W_c . [h_{t-1}, x_t] + b_c\right) \\
C_t &= f_t * C_{t-1} + i_t * \tilde{C}_t \\
o_t &= \sigma\left(W_o . [h_{t-1}, x_t] + b_o\right) E \\
h_t &= o_t * tanh(C_t)
\end{aligned} \tag{1}
$$

where, h_t =hidden state, o_t = output gate, C_t = cell state, i_t = input gate, f_t =forget gate.

Initially, the LSTM structure decides whether to throw or keep the information and it is accomplished with a forget gate (sigmoid function) in Eq. (1) [8], [24]. It takes xt and ht-1 as input and outputs a decimal between zero and one. Zero indicates throwing completely, and one indicates keeping completely. The current information that should be added to the cell is decided by the input gate and candidate cell. An input gate is a sigmoid function that computes the values to update the cell state. The candidate cell in Eq. (1). computes some values, that could be added to the cell state. A combination of the input gate and candidate cell will be used to update the cell state. The basic structure of LSTM network is presented in Figure 3, which takes sequence of inputs and predicts an output after a successful computation of complex mathematical equations [1]. The circle and arrows represent point wise operation and the flow of the vector respectively, and activation functions are exclusively defined.

The new cell state is calculated from the input gate, the previous cell state, the forget gate, and the candidate cell. For throwing information, the forget gate is multiplied by the previous cell state. The input gate is also multiplied by the candidate cell to decide how far to overhaul the cell state. The final output gate in Eq. (1) is computed from the previous input and hidden state using a sigmoid function. As a final step, the current hidden state is obtained from the output gate and the current candidate gate, which are shown in the last line of Eq. (1).

III. METHODOLOGY

Recent advancements in IIoT edge device hardware capacity (CPU, Memory, and Battery) enables the development of on-device learning. However, this hardware is not large enough to process machine learning algorithms. Hence, analyzing a large amount of data that has been collected continuously for a long time is a challenging task for resource deficient devices. This study presents a typical on-device incremental learning in which edge devices collect their own data and analyze it by themselves. Edge devices can analyze data to predict a certain task over time. The machine learning model deployed on these devices can be updated to enhance existing knowledge with newly generated data. This paper puts forward a modularized incremental approach that learns over time with the newly generated data and extracts knowledge from the previously processed data. Machine learning algorithms should be executed modularly by taking input data within a certain range of time. Hence, the previously processed data would not be further analyzed since the model had already extracted knowledge. In this manner, only essential information can be transferred from a previously executed model using a dataset collected over a certain period of time. This reduces resource consumption by maintaining performance as it only transfers essential information.

The RALSTMM modifies the LSTM network through an incremental approach. The data set collected from the environment can be analyzed incrementally, as illustrated in Figure 4 and Algorithm 1. The dataset is grouped into sub-datasets with a certain time interval. A dataset analyzed at a certain time will not be analyzed again. It transfers essential information from the previous sub-dataset to the current sub-dataset during the analysis. During the first sub-dataset, the entire process is identical to the original LSTM. After processing the first sub-dataset, the last values of cell state, hidden state and the predicted value for computing the cost function are transferred to the second sub-dataset. Similarly, the second sub-dataset transfers the last values of cell state, hidden state, and predicted value (y_{pred}), which continue until the time step t_n.

During the analysis of the first dataset, the equations of LSTM remained the same since there was no modification on the initial random input parameters. From the second sub-dataset, the first sequence takes the previous dataset's last hidden state and cell state values. Hence, all the sub-datasets except the initial sub-dataset can learn from their antecedent by taking the last hidden and cell state parameters instead of initializing randomly, as presented in Eq. (3).

The whole dataset at time t_n is the aggregate of all the sub-datasets.

$$D_{n\_total} = D_1 \cup D_2 \cup \ldots \cup D_n \tag{2}$$

The performance of the RALSTMM can be computed with a cost function. The cost function calculates the error between the predicted and actual quantities [27]. Mean Square Error (MSE) is one of the most widely used cost functions and is computed by the sum of the squared differences between the actual and predicted quantities. The MSE error of each sub-dataset could be computed from the predicted and actual values of its own dataset (Eq. 4). However, the MSE for more

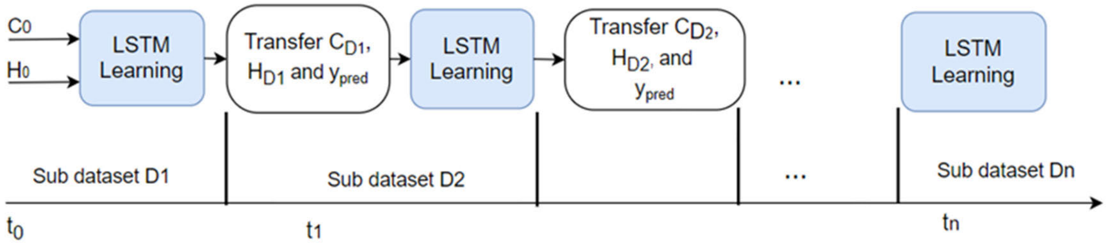

FIGURE 4. Structure of the RALSTMM.

Algorithm 1 RALSTMM incremental learning

Input: Data collected by edge devices in a certain period
Output: Predicted results
if input data is D1:
  Execute original LSTM
  Save the last C_{D1}, H_{D1} and y_{pred}
  Report the predicted attacks
else
  Execute LSTM with initial values of cell state C_{Dn-1}, hidden state H_{Dn-1} and y_{pred}
  Save the last C_{Dn}, H_{Dn} and y_{pred}
  Report the predicted attacks
end if

than one sub-dataset is the aggregated MSE of each sub-dataset (Eq. 5)

$$\begin{aligned} f_t &= \sigma\left(W_f.[h_{Dn-1}, x_t] + b_f\right) \\ i_t &= \sigma\left(W_i.[h_{Dn-1}, x_t] + b_i\right) \\ \tilde{C}_t &= tanh\left(W_c.[h_{Dn-1}, x_t] + b_c\right) \\ C_t &= f_t * C_{Dn-1} + i_t * \tilde{C}_t \\ o_t &= \sigma\left(W_o.[h_{Dn-1}, x_t] + b_o\right) \\ h_t &= o_t * tanh(C_t) \end{aligned} \tag{3}$$

where, h_{Dn-1} and C_{Dn-1} is the previous transferred hidden and cell states respectively

$$MSE = \frac{1}{N_{Dn}} \sum\nolimits_{i=0}^{N_{Dn}} (y_i - \bar{y}_i)^2 \tag{4}$$

$$\begin{aligned} MSE &= \frac{1}{N_{D1}} \sum\nolimits_{i=0}^{N_{D1}} (y_i - \bar{y}_i)^2 + \frac{1}{N_{D2}} \sum\nolimits_{i=D1+1}^{N_{D2}} (y_i - \bar{y}_i)^2 \\ &\quad + \ldots + \frac{1}{N_{Dn}} \sum\nolimits_{i=Dn-1+1}^{N_{Dn}} (y_i - \bar{y}_i)^2 = \sum\nolimits_{i=0}^{N} (y_i - \bar{y}_i)^2 \end{aligned} \tag{5}$$

where, $\bar{y}_i$ is the predicted value, y_i is the actual value, N_{Dn} is the size of the dataset for sub-dataset n, and N is the size of the dataset at time t_n.

IV. RESULTS AND DISCUSSION

The RALSTMM has been tested with three different IoT application specific datasets in order to characterize domain-specific applicability and robustness based on [1], [9], and [26]. Specifically, a testbed dataset undertaken on fridge, thermostat, and weather IoT devices which is found at the University of New South Wales website has been used [25], [26]. The testbed was collected from different normal and attacked activities of the IIoT network. The architecture of the network has three layers, namely cloud, fog, and edge layers, to represent the current real-world scenarios of Industry 4.0 networks. The experiment was conducted with Python and the hardware specification of an Intel (R) Core (TM) i5-8350U CPU (1.70 GHz) and 8 GB of main memory.

The dataset needs to be sliced into sub-datasets for demonstration as incremental learning needs continuous group of datasets and demonstrating real-world streaming problems as in [9] and [12]. Hence, the dataset has been grouped into four sub-datasets based on the number of samples and may vary for different samples and scenarios. Hence, each of the three datasets used for this experiment was partitioned into four parts, i.e., D1, D2, D3 and D4, chronologically. The experiment was performed for both the existing and the proposed approaches with sub-datasets. Initially, the LSTM network executes the first dataset (D1) and records the result. The second experiment was executed by merging the first and second sub-datasets. Similarly, the next experiments were performed by merging the previous sub-datasets and recording the results for each step. The RALSTMM network executes each sub-dataset independently in chronological order. Results of each sub-dataset were reported, and important updated parameters of each sub-dataset were transferred to the next experiment. From the second sub-dataset, updated cell state and hidden state values from the previous experiment were taken instead of initializing randomly.

The most widely used performance evaluation metrics for such scenarios are Mean Squared Error (MSE), accuracy, recall, precision and F1score. Processing time, memory and CPU depletion are used to measure the resource consumption of such scenarios [5], [12], [15], [32]. We have further evaluated the model's parsimony through the utilization of information criteria. The Bayesian Information Criteria (BIC) and Akaike Information Criteria (AIC) are the most widely used information criteria for selecting the best model in time series analysis [36], [37], [38]. Hence, we have used Mean Squared Error (MSE), accuracy, recall, precision, processing time, BIC and AIC for evaluating the RALSTMM

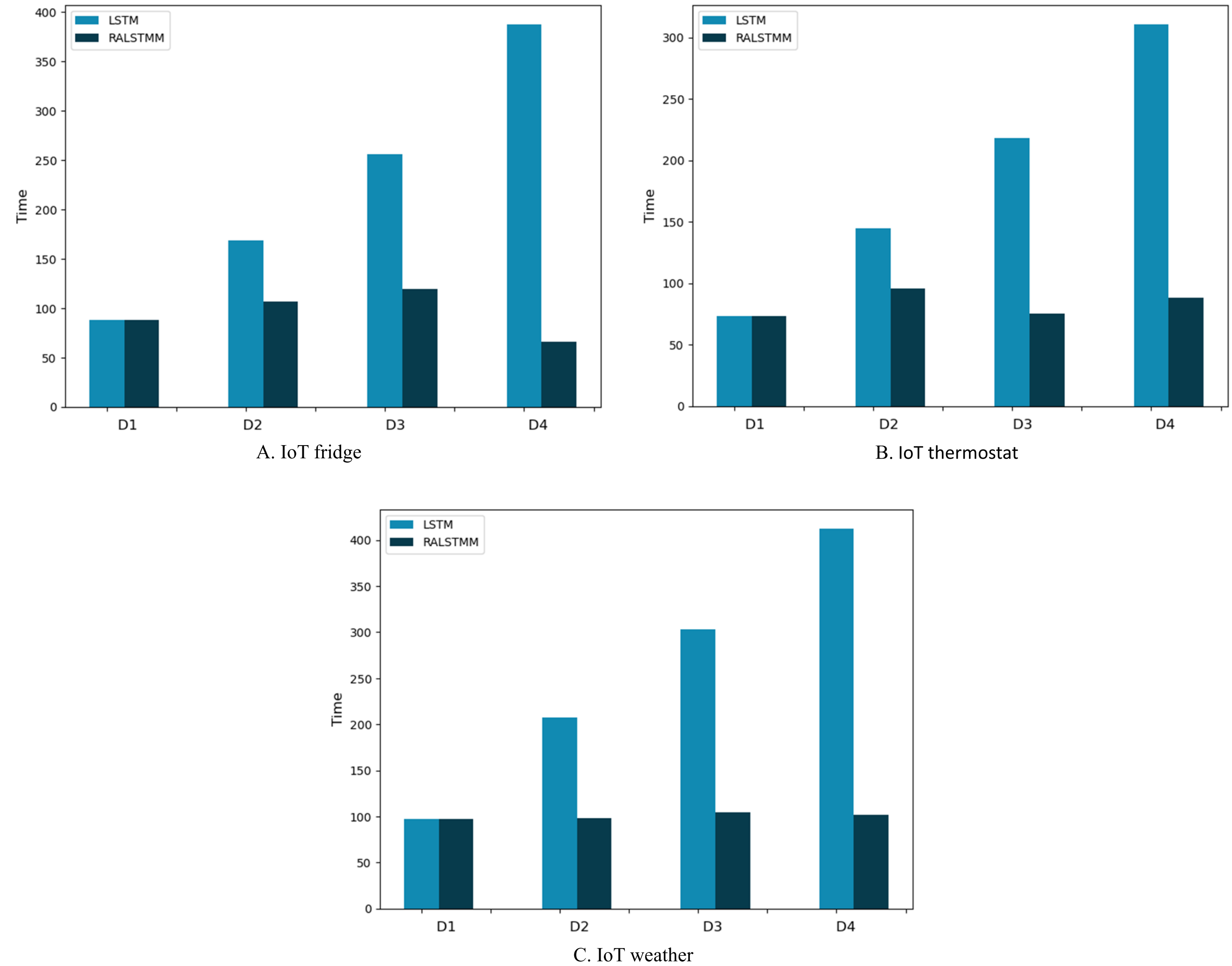

FIGURE 5. Comparison of the RALSTMM and existing LSTM with their processing time.

TABLE 1. Comparison of mean square error for the RALSTMM and existing LSTM model.

	IoT Fridge MSE		IoT Thermo-stat MSE		IoT Weather MSE	
Sub-datasets	LSTM	RALSTMM	LSTM	RALSTMM	LSTM	RALSTMM
D1	0.08	0.08	0.04	0.04	0.05	0.04
D2	0.10	0.03	0.13	0.03	0.04	0.04
D3	0.03	0.01	0.06	0.02	0.04	0.03
D4	0.02	0.02	0.01	0.01	0.03	0.02

with different IoT datasets (fridge, thermostat, and weather).

Tables 1 and 2 indicate the comparison of the RALSTMM and existing work with MSE, accuracy, recall, and precision. In the first group of sub-datasets, the result for the RALSTMM and existing LSTM models is the same due to an identical initial parameter. In the second sub-dataset, the RALSTMM outperforms the existing model as it effectively transfers important parameters. The quantity of MSE decreases from the first to the last sub-dataset, which proves better prediction performance as the size of the dataset increases. Similarly, the processing time of both the proposed and existing LSTM networks was measured during the experiment. As clearly shown in Table 3, RALSTM has lower values of BIC and AIC compared to the existing LSTM. This proves that RALSTM is better, as a lower value of BIC and AIC indicates the effectiveness of the model.

As shown in Figure 5, the time elapsed for training the RALSTMM is smaller than the time elapsed for training the existing model from the second sub-dataset. This significant improvement is due to the fact that the RALSTMM trains only the specific sub-dataset by receiving important transferred parameters from the predecessor sub-dataset (i.e., if sub-dataset D3 is currently running, it accepts important parameters from sub-dataset D2). However, the existing model processes the aggregate of the previous sub-datasets.

TABLE 2. Comparison of accuracy, recall and precision for the RALSTMM and existing LSTM model.

Sub-dataset	LSTM	RALSTMM	LSTM	RALSTMM	LSTM	RALSTMM
	IoT Fridge Accuracy		IoT Thermo-stat Accuracy		IoT Weather Accuracy	
D1	0.63	0.63	0.56	0.56	0.67	0.67
D2	0.66	0.68	0.59	0.64	0.70	0.69
D3	0.75	0.71	0.79	0.80	0.84	0.82
D4	0.83	0.80	0.81	0.82	0.86	0.81
	IoT Fridge Recall		IoT Thermo-stat Recall		IoT Weather Recall	
D1	0.73	0.73	0.85	0.85	0.75	0.75
D2	0.77	0.79	0.88	0.89	0.88	0.95
D3	0.76	0.80	0.89	0.91	0.94	0.96
D4	0.84	0.82	0.91	0.89	0.88	0.79
	IoT Fridge Precision		IoT Thermo-stat Precision		IoT Weather Precision	
D1	0.71	0.71	0.62	0.62	0.83	0.83
D2	0.68	0.83	0.65	0.68	0.77	0.71
D3	0.69	0.71	0.67	0.69	0.54	0.87
D4	0.75	0.78	0.72	0.70	0.77	0.79

TABLE 3. Comparison of AIC and BIC for the RALSTMM and existing LSTM model.

Sub-data set	LSTM	RALSTMM	LSTM	RALSTMM	LSTM	RALSTMM
	IoT Fridge BIC		IoT Thermo-stat BIC		IoT Weather BIC	
D1	14232	14232	12605	12605	13109	13109
D2	16120	14052	16214	11808	17052	13901
D3	19920	13980	20810	13582	19204	11082
D4	21021	13549	19901	10255	20954	12548
	IoT Fridge AIC		IoT Thermo-stat AIC		IoT Weather AIC	
D1	7104	7104	8102	8102	7668	7668
D2	7698	6850	8421	7808	8102	7669
D3	7669	6404	8925	7541	9170	7750
D4	8015	6835	8890	7651	9050	7180

V. CONCLUSION

Current industrial control systems have several bottlenecks, such as security, scalability, and resource limitations for data processing. Analyzing the whole dataset, which has been collected continuously for a long time, is a challenging task for resource deficient devices. This paper proposes a typical on-device incremental learning approach (RALSTMM) for edge devices of industrial control system to minimize resource utilization and privacy. It can be deployed on edge nodes and updated to enhance existing knowledge with newly generated data. The RALSTMM transfers important parameters from the previous training. The experimental results with three IoT testbed datasets show that the proposed RALSTMM saves processing time and maintains the performance. Tiny resource constraint industrial devices need a resource optimized algorithm for intelligent data processing and RALSTMM model could be applicable in such industrial applications.

Next direction of the study could be conducting tests using a real-time stream of data generated from resource restricted edge devices in real-world scenarios. The RALSTMM could be extended to other neural network architectures such as convolutional neural network, generative adversarial network and reinforcement learning for addressing wide range of application.

REFERENCES

[1] J. Liu, J. Bai, H. Li, and B. Sun, "Improved LSTM-based abnormal stream data detection and correction system for Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 18, no. 2, pp. 1282–1290, Feb. 2022, doi: 10.1109/TII.2021.3079504.

[2] M. S. S. Garmaroodi, F. Farivar, M. S. Haghighi, M. A. Shoorehdeli, and A. Jolfaei, "Detection of anomalies in industrial IoT systems by data mining: Study of CHRIST Osmotron water purification system," *IEEE Internet Things J.*, vol. 8, no. 13, pp. 10280–10287, Jul. 2021, doi: 10.1109/JIOT.2020.3034311.

[3] M. Javaid, A. Haleem, R. P. Singh, S. Rab, and R. Suman, "Significance of sensors for Industry 4.0: Roles, capabilities, and applications," *Sensors Int.*, vol. 2, Jan. 2021, Art. no. 100110, doi: 10.1016/j.sintl.2021.100110.

[4] G. Aceto, V. Persico, and A. Pescapé, "A survey on information and communication technologies for Industry 4.0: State-of-the-art, taxonomies, perspectives, and challenges," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 4, pp. 3467–3501, 4th Quart., 2019, doi: 10.1109/COMST.2019.2938259.

[5] T. T. Huong, T. P. Bac, D. M. Long, T. D. Luong, N. M. Dan, B. D. Thang, and K. P. Tran, "Detecting cyberattacks using anomaly detection in industrial control systems: A federated learning approach," *Comput. Ind.*, vol. 132, Nov. 2021, Art. no. 103509, doi: 10.1016/j.compind.2021.103509.

[6] B. Qolomany, I. Mohammed, A. Al-Fuqaha, M. Guizani, and J. Qadir, "Trust-based cloud machine learning model selection for industrial IoT and smart city services," *IEEE Internet Things J.*, vol. 8, no. 4, pp. 2943–2958, Feb. 2021, doi: 10.1109/JIOT.2020.3022323.

[7] M. Zolanvari, M. A. Teixeira, L. Gupta, K. M. Khan, and R. Jain, "Machine learning-based network vulnerability analysis of industrial Internet of Things," *IEEE Internet Things J.*, vol. 6, no. 4, pp. 6822–6834, Aug. 2019, doi: 10.1109/JIOT.2019.2912022.

[8] Y. Liu, N. Kumar, Z. Xiong, W. Y. B. Lim, J. Kang, and D. Niyato, "Communication-efficient federated learning for anomaly detection in industrial Internet of Things," in *Proc. GLOBECOM IEEE Global Commun. Conf.*, Dec. 2020, pp. 1–6, doi: 10.1109/GLOBECOM42002.2020.9348249.

[9] P. Narkhede, R. Walambe, S. Poddar, and K. Kotecha, "Incremental learning of LSTM framework for sensor fusion in attitude estimation," *PeerJ Comput. Sci.*, vol. 7, p. e662, Aug. 2021, doi: 10.7717/peerj-cs.662.

[10] I. T. Christou, N. Kefalakis, A. Zalonis, and J. Soldatos, "Predictive and explainable machine learning for industrial Internet of Things applications," in *Proc. 16th Int. Conf. Distrib. Comput. Sensor Syst. (DCOSS)*, May 2020, pp. 213–218, doi: 10.1109/DCOSS49796.2020.00043.

[11] Z. Xiao, H. Fang, and X. Wang, "Anomalous IoT sensor data detection: An efficient approach enabled by nonlinear frequency-domain graph analysis," *IEEE Internet Things J.*, vol. 8, no. 5, pp. 3812–3821, Mar. 2021, doi: 10.1109/JIOT.2020.3024763.

[12] H. Wang, M. Li, and X. Yue, "IncLSTM: Incremental ensemble LSTM model towards time series data," *Comput. Electr. Eng.*, vol. 92, Jun. 2021, Art. no. 107156, doi: 10.1016/j.compeleceng.2021.107156.

[13] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

[14] R. R. Ade and P. R. Deshmukh, "Methods for incremental learning : A survey," *Int. J. Data Mining Knowl. Manag. Process*, vol. 3, no. 4, pp. 119–125, Jul. 2013, doi: 10.5121/ijdkp.2013.3408.

[15] S. Disabato and M. Roveri, "Incremental on-device tiny machine learning," in *Proc. 2nd Int. Workshop Challenges Artif. Intell. Mach. Learn. Internet Things*, Nov. 2020, pp. 7–13, doi: 10.1145/3417313.3429378.

[16] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 50–60, May 2020, doi: 10.1109/MSP.2020.2975749.

[17] Y. Liu, S. Garg, J. Nie, Y. Zhang, Z. Xiong, J. Kang, and M. S. Hossain, "Deep anomaly detection for time-series data in industrial IoT: A communication-efficient on-device federated learning approach," *IEEE Internet Things J.*, vol. 8, no. 8, pp. 6348–6358, Apr. 2021, doi: 10.1109/JIOT.2020.3011726.

[18] R. Taheri, M. Shojafar, M. Alazab, and R. Tafazolli, "Fed-IIoT: A robust federated malware detection architecture in industrial IoT," *IEEE Trans. Ind. Informat.*, vol. 17, no. 12, pp. 8442–8452, Dec. 2021, doi: 10.1109/TII.2020.3043458.

[19] Y. Liu, T. Dillon, W. Yu, W. Rahayu, and F. Mostafa, "Missing value imputation for industrial IoT sensor data with large gaps," *IEEE Internet Things J.*, vol. 7, no. 8, pp. 6855–6867, Aug. 2020, doi: 10.1109/JIOT.2020.2970467.

[20] A. K. Takele and B. Villányi, "Anomaly detection using hybrid learning for industrial IoT," in *Proc. IEEE 2nd Conf. Inf. Technol. Data Sci. (CITDS)*, May 2022, pp. 262–266, doi: 10.1109/CITDS54976.2022.9914338.

[21] J. Li, H. Izakian, W. Pedrycz, and I. Jamal, "Clustering-based anomaly detection in multivariate time series data," *Appl. Soft Comput.*, vol. 100, Mar. 2021, Art. no. 106919, doi: 10.1016/j.asoc.2020.106919.

[22] J. Brownlee, *Long Short-Term Memory Networks With Python: Develop Sequence Prediction Models With Deep Learning* (Machine Learning Mastery), 2017. [Online]. Available: https://books.google.hu/books?id=m7SoDwAAQBAJ

[23] S. Siami-Namini, N. Tavakoli, and A. S. Namin, "A comparison of ARIMA and LSTM in forecasting time series," in *Proc. 17th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2018, pp. 1394–1401, doi: 10.1109/ICMLA.2018.00227.

[24] K. Smagulova and A. P. James, "A survey on LSTM memristive neural network architectures and applications," *Eur. Phys. J. Special Topics*, vol. 228, pp. 2313–2324, Oct. 2019, doi: 10.1140/epjst/e2019-900046-x.

[25] T. M. Booij, I. Chiscop, E. Meeuwissen, N. Moustafa, and F. T. H. D. Hartog, "ToN_IoT: The role of heterogeneity and the need for standardization of features and attack types in IoT network intrusion data sets," *IEEE Internet Things J.*, vol. 9, no. 1, pp. 485–496, Jan. 2022, doi: 10.1109/JIOT.2021.3085194.

[26] A. Alsaedi, N. Moustafa, Z. Tari, A. Mahmood, and A. Anwar, "TON_IoT telemetry dataset: A new generation dataset of IoT and IIoT for data-driven intrusion detection systems," *IEEE Access*, vol. 8, pp. 165130–165150, 2020, doi: 10.1109/ACCESS.2020.3022862.

[27] A. Kumar, A. Alsadoon, P. W. C. Prasad, S. Abdullah, T. A. Rashid, D. T. H. Pham, and T. Q. V. Nguyen, "Generative adversarial network (GAN) and enhanced root mean square error (ERMSE): Deep learning for stock price movement prediction," *Multimedia Tools Appl.*, vol. 81, no. 3, pp. 3995–4013, Jan. 2022, doi: 10.1007/s11042-021-11670-w.

[28] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A survey on enabling technologies, protocols, and applications," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 4, pp. 2347–2376, 4th Quart., 2015, doi: 10.1109/COMST.2015.2444095.

[29] X. Zhou, Y. Hu, W. Liang, J. Ma, and Q. Jin, "Variational LSTM enhanced anomaly detection for industrial big data," *IEEE Trans. Ind. Informat.*, vol. 17, no. 5, pp. 3469–3477, May 2021, doi: 10.1109/TII.2020.3022432.

[30] X. Li, S. Dong, Q. Su, M. Yu, and X. Li, "Adaptive threshold hierarchical incremental learning method," *IEEE Access*, vol. 11, pp. 12285–12293, 2023, doi: 10.1109/ACCESS.2023.3242688.

[31] B. Yang, M. Lin, Y. Zhang, B. Liu, X. Liang, R. Ji, and Q. Ye, "Dynamic support network for few-shot class incremental learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 3, pp. 2945–2951, Mar. 2023, doi: 10.1109/TPAMI.2022.3175849.

[32] H. Chen, Y. Yu, Y. Jia, and B. Gu, "Incremental learning for transductive support vector machine," *Pattern Recognit.*, vol. 133, Jan. 2023, Art. no. 108982, doi: 10.1016/j.patcog.2022.108982.

[33] L. Llopis-Ibor, C. Beltran-Royo, A. Cuesta-Infante, and J. J. Pantrigo, "Fast incremental learning by transfer learning and hierarchical sequencing," *Exp. Syst. Appl.*, vol. 212, Feb. 2023, Art. no. 118580, doi: 10.1016/j.eswa.2022.118580.

[34] Z. Fu, Z. Wang, X. Xu, D. Li, and H. Yang, "Knowledge aggregation networks for class incremental learning," *Pattern Recognit.*, vol. 137, Jan. 2023, Art. no. 109310, doi: 10.1016/j.patcog.2023.109310.

[35] L. Melgar-García, D. Gutiérrez-Avilés, C. Rubio-Escudero, and A. Troncoso, "Identifying novelties and anomalies for incremental learning in streaming time series forecasting," *Eng. Appl. Artif. Intell.*, vol. 123, Aug. 2023, Art. no. 106326, doi: 10.1016/j.engappai.2023.106326.

[36] Y. Ning, H. Kazemi, and P. Tahmasebi, "A comparative machine learning study for time series oil production forecasting: ARIMA, LSTM, and prophet," *Comput. Geosci.*, vol. 164, Jul. 2022, Art. no. 105126, doi: 10.1016/j.cageo.2022.105126.

[37] S. Chaturvedi, E. Rajasekar, S. Natarajan, and N. McCullen, "A comparative assessment of SARIMA, LSTM RNN and Fb prophet models to forecast total and peak monthly energy demand for India," *Energy Policy*, vol. 168, Sep. 2022, Art. no. 113097, doi: 10.1016/j.enpol.2022.113097.

[38] Y. Chen and K. Wang, "Prediction of satellite time series data based on long short term memory-autoregressive integrated moving average model (LSTM-ARIMA)," in *Proc. IEEE 4th Int. Conf. Signal Image Process. (ICSIP)*, Jul. 2019, pp. 308–312, doi: 10.1109/SIPROCESS.2019.8868350.

ATALLO KASSAW TAKELE received the M.Sc. degree in computer science (computer networking stream) from Jimma University, Jimma, Ethiopia, in 2018. He is currently pursuing the Ph.D. degree in computer engineering with the Budapest University of Technology and Economics, Budapest, Hungary. His research interests include machine and deep learning, the IoT, industry 4.0, security and enterprise application integration.

BALÁZS VILLÁNYI received the Ph.D. degree in computer science from the Budapest University of Technology and Economics, Budapest, Hungary. He is currently an Associate Professor and a Doctoral Advisor with the Faculty of Electrical Engineering and Informatics, Budapest University of Technology and Economics. His research interests include machine learning, schema matching algorithms, enterprise application integration, the Industrial IoT, and industry 4.0.