A DEEP TRANSFER LEARNING APPROACH FOR IOT/IIOT CYBER ATTACK DETECTION USING TELEMETRY DATA

S. Poonkuzhali[*], *M. Shobana*[*], *J. Jeyalakshmi*[†]

Abstract: The rise of internet connectivity across the globe increases the count of IoT (internet of things)/IIoT (industrial internet of things) devices exponentially. The objects/devices which are connected to the internet are always prone to malicious attacks at various levels, such as physical, network, fog, and applications, which exist in the IoT architecture. Many researchers have addressed this issue and designed their own solutions based on machine and deep learning techniques. It is undeniable that deep learning outperforms machine learning (ML), but it necessitates a massive amount of datasets with appropriate labels. In this work, the deep transfer learning (TL) technique has been adapted for gated recurrent unit (GRU). Each model is trained using a dataset that belongs to one source IoT device (source domain), and this trained model is used to classify the malicious traffic in another dataset that belongs to some other IoT device (target domain). This approach is used for binary classification. These transfer learning models have been evaluated using an IoT/IIoT telemetry dataset called ToN_IoT which comprises the sensor data generated from the seven different types of IoT devices. The highest accuracy achieved by IoT garage door was upto 99.76% as a source domain by fixing IoT thermostat as target domain. These models were also evaluated using some more metrics such as precision, recall, F1-measure, training time and testing time. By implementing transfer learning based GRU model, the accuracy of the model is improved from 69.20% to 99.76%. Moreover, to prove the efficiency of the proposed model, it is compared with state of art deep learning model and its results were analyzed in a detailed manner.

Key words: *GRU, transfer learning, intrusion detection, sensor or telemetry data*

Received: October 5, 2021
Revised and accepted: August 27, 2023

DOI: 10.14311/NNW.2023.33.014

[*]S. Poonkuzhali – Corresponding author; M. Shobana; Department of Computer Science and Engineering, Rajalakshmi Engineering College, Chennai, Tamil Nadu, India, E-mail: poonkuzhali.s@rajalakshmi.edu.in, divyashobana.m@gmail.com

[†]J. Jeyalakshmi; Department of Computer Science and Engineering, Amrita School of Computing, Amrita Vishwa Vidyapeetham, Chennai, Tamil Nadu, India, Email: j_jeyalakshmi@ch.amrita.edu

©CTU FTS 2023

1. Introduction

The term "internet of things" plays a vital role in improvising the connected community in order to drag each object that belongs to different technologies under a single umbrella. This community enhances the data communication among those connected objects, or IoT devices, without any human intervention [15]. This kind of communication nature of IoT technology leads to its spreading wide across the globe and it gives rise to an increasing number of IoT devices. This popularity makes the vendors concentrate only on the operation of IoT devices rather than providing security. IoT devices are always easy targets for security attacks, rather than computers or mobile phones [16]. The design of the security framework for the IoT environment should meet the required specifications accordingly. Traditional malware detection approaches can be broadly divided into signature-based detection, honeypot-based detection, and behaviour-based approaches [8]. The signature-based approach [26, 5, 17, 3] for IoT devices is done by extracting the signature of existing malwares and it forms the database. In the future, this designed database will be used to compare the signatures of the incoming samples. This technique is not capable of detecting unknown malware since it lacks the signature of the new malware. Secondly, the honeypot based approach involves creating a virtual machine which is purposely exposed to all kinds of incoming malicious attacks from the nearer connected network [22]. Finally, the behaviour-based approach aims to detect malware samples by tracing the behaviour of the attacked system. This approach is achieved by implementing machine learning (ML) and deep learning techniques. Some of the behavior-based approaches to IoT security are achieved by using machine learning techniques such as SVM (support vector machine), decision tree, naive Bayes, KNN (k-nearest neighbour), etc., and they face many hurdles to achieve better performance [20, 11]. These drawbacks were overcome by deep learning techniques in terms of performance in predicting and detecting malicious behaviour. In some cases, the biggest challenge in deploying deep learning techniques requires labelling large amounts of data for training [9]. In the field of IoT, the availability of real-time existing datasets along with labelling of IoT security attacks is very low for deep learning computation. Apart from these issues, some of the IoT real-time datasets also suffer from class imbalance problems due to non-availability of IoT network traffic as well as IoT malware samples. In order to overcome these kinds of challenges, many of the intrusion detection or malware detection systems proposed for the IoT environment make use of deep transfer learning (TL) techniques in a consistent manner. But those DTL (deep transfer learning)-based security frameworks suggested for IoT platforms require high training time and are computed only on network traffic attributes alone. The following is a summary of my overall contribution to this research project:

1. The transfer learning-based GRU model has been designed to perform knowledge transfer among different IoT devices using two GRU architectures.

2. The intrusion detection system is designed at the sensory layer using a TL-based GRU model by training the model using one device as the source domain and the knowledge transferred to another model by testing the GRU model using some other IoT device dataset as the target domain.

3. The proposed model has been compared with the traditional deep learning techniques such as CNN (convolutional neural network), RNN (recurrent neural network) and DNN (deep neural network) using various performance metrics such as accuracy, precision, recall and F1-measure.

2. Related work

The relevant research work that contributes to transfer learning techniques used for IoT security, as well as the work that deals with intrusion detection techniques using a transfer learning approach, were discussed and analyzed in detail in this section.

2.1 Transfer learning for IoT security

Vu et al. [22] developed MultiMaximum Mean Discrepancy AE (autoencoder), a deep transfer learning-based intrusion detection system for IoT devices that employs two autoencoders and MultiMaximum Mean Discrepancy (MMD). This model utilizes the first autoencoder for the training process along with a class label, and the second autoencoder is trained to classify the target dataset without labelling. This model takes more time for training purposes. This approach is somewhat not appropriate for IoT devices. Taheri et al. [20] depicted transfer learning along with a convolutional neural network for classifying images generated from network traffic attributes. Those attributes were taken from incoming and outgoing traffic generated by the IoT device, which includes both benign and malicious traffic.

2.2 Transfer learning for intrusion detection system

Singla et al. [18] presented a network-based intrusion detection framework using a deep learning model along with a transfer learning technique. The author utilised one deep neural network to train the source dataset and another deep neural network to detect new attack types given in the target dataset which are not in the source dataset. Wu et al. [24] presented a transfer learning-based convolutional neural network to provide a network-based intrusion detection system. Here two concatenated convolutional neural networks are deployed to detect new variant types of attack by the second neural network model. Tariq et al. [21] suggested a security framework for deep transfer learning using an LSTM (long short-term memory) model called CANTransfer for in-vehicle communication with a minimal amount of dataset. This model is trained using known attacks in order to predict unknown attacks. Taghiyarrenani et al. [19] implemented an intrusion detection framework based on transfer learning by training the attack from one network and detecting attacks generated from another kind of network using SVM (support vector machine) and a baseline method such as DAMA. Wen et al. [23] proposed an intrusion detection system based on time series forecasting and transfer learning. The author has used convolutional neural networks (CNN) to predict an unknown anomaly with minimal training samples. Li et al. [12] proposed an anomaly detection system using active transfer learning techniques, namely ACTrAdaBoost

and maximum mean discrepancy knowledge. Xu et al. [25] implemented an intrusion detection framework using the grayscale images generated from network traffic. To detect the malicious grayscale image, transfer learning along with deep learning techniques (CNN) is used. Li et al. [13] suggested an intrusion detection system for the IoV (internet of vehicles) network on the IoV cloud platform. The author has utilized deep learning along with transfer learning for secure in-vehicle communication.

2.3 Intrusion detection system for IoT without transfer learning technique

Tim et al. [2] has explained about the features exist in the novel dataset called ToN_IoT. The telemetry dataset has been analyzed and evaluated using various machine learning techniques. The performance of the ToN_IoT dataset has been compared with the other existing dataset proposed for IoT intrusion detection system. Abdallah et al. [6] proposed an intrusion detection framework for the internet of vehicles platform. Author has utilized chi square technique for feature reduction process followed by sampling process has been carried out using synthetic minority oversampling technique (SMOTE). Finally XGBoost has been used for classification process and it is compared with various ML algorithms to prove its efficiency. Amir andalib et al. [1] proposed a security solution for IoT environment using autoencoder. Author has been designed novel autoencoder using various latent space to produce better result with minimum number of features.

Tab. I demonstrates about the some of the demerits in the existing solution. On the whole, the finding of the demerits can be summarized as techniques based on transfer learning concentrates only on detecting non labelled attacks rather than the device's type. Hence those techniques were failed to produce common models for all types of IoT devices. Then some of there used machine learning technique which is not suitable for real time application based on high dimensional dataset.

3. Methods and materials

In this section, the required dataset and techniques used to design the intrusion detection system for the IoT environment using telemetry dataset have been described as given as below.

3.1 Dataset description

The ToN_IoT dataset contains telemetry datasets generated from various IoT and IIoT sensors embedded in various types of IoT devices such as a IoT fridge, IoT garage door, IoT GPS tracker, IoT Modbus unit, IoT motion light sensor, IoT thermostat, and IoT weather sensor. The telemetry dataset for each device has different input attributes based on the IoT device. But the number of input parameters for all the IoT devices was equal. The parameter are common for all the given IoT dataset. In this transfer learning approach, only binary classification has been implemented.

Author & year	Technique used	Limitation
Vu et al. [22] & 2020	Two autoencoder along with multi-maximum mean discrepancy	This model takes more time for training purposes, this approach is somewhat not appropriate for IoT devices
Taheri et al. [20] & 2018	Transfer learning along with convolutional neural network for classifying images generated from network traffic attributes	This model acquires high memory to process and store the image dataset instead of network traffic
Xu et al. [25] & 2019	To detect the malicious grayscale image, transfer learning along with deep learning technique (CNN) is used	This model acquires high memory to process and store the image dataset instead of network traffic
Singla et al. [18] & 2019	Two DNN (deep neural network) used along with TL	Knowledge transfer was carried out only between types of attacks but fails to prove for device based transfer
Wu et al. [24] & 2019	Two CNN (convolutional neural network) used along with TL	Knowledge transfer was carried out only between types of attacks but fails to prove for device based transfer
Tariq et al. [21] & 2020	Using LSTM model called CANTransfer for in-vehicle communication with minimal amount of dataset	Knowledge transfer was carried out only between types of attacks but fails to prove for device based transfer
Taghiyarrenani et al. [19] & 2018	Intrusion detection framework based on transfer learning by training the attack from one network and to detect attack generated from another kind of network using SVM and a baseline method such as DAMA	Utilization of machine learning technique will produce lesser performance than the machine learning technique for larger database
Li et al. [12] & 2019	Active transfer learning techniques namely ACTrAdaBoost and maximum mean discrepancy knowledge	Utilization of machine learning technique will produce lesser performance than the machine learning technique for larger database
Wen et al. [23] & 2019	Time series forecasting using transfer learning. By implementing convolutional neural network (CNN) the unknown anomaly is predicted	Implementing U-net architecture is time consuming process and this approach yields very low accuracy
Li et al. [13] & 2020	Author has utilized deep learning along with transfer learning for secure in-vehicle communication.	The model updation has been done via cloud which consume more time

Tab. I *Analysis of existing solution.*

The types of attacks considered in the research work are DoS (denial of service), DDoS (distributed denial of service) and backdoor attacks. The normal category is a non-malicious case of classification. Each device has various parameters based on its functionality. For example IoT fridge has features like date, time, temperature, condition as compared to threshold is high or low, label etc., for IoT garage door the fields are date, time, door_state, door_signal on a phone where the signal is true or false, label etc., IoT GPS tracker is a device that has features like date, time, latitude, longitude and label. IoT motion light has features like date, time, motion_status, light_status and label. IoT thermostat has features like date, time, current_temperature, thermostat_status showing either on or off and label. Basically these features are used for classification. But the label field in all devices is most influential that shows an indication that a record is normal or attack based on value as 0 or 1. These are the features used for classification of different types of attack.

3.2 Transfer learning

Transfer learning is a kind of approach that is used to transfer the knowledge extracted from one task to another new task with the same or different feature space. Transfer learning can be used to provide huge benefits in the case of non-availability of a required dataset, so this approach can be efficiently used for deep learning rather than applying to machine learning techniques [7]. Because, unlike deep learning techniques, machine learning techniques are capable of achieving a high accuracy rate on smaller datasets. In the case of IoT devices, the computational workload should be minimal, so that knowledge retrieved from one task performed for one IoT device can be transferred to another different IoT device. The main terminologies used for transfer learning are source domain and target domain. The source domain is defined as the base model which performs well for a certain task when compared to other existing models. Meanwhile, the target domain can be defined as the new dataset which is given to the pretrained model to produce the new model without a training process [14].

3.3 Gated recurrent unit

Recurrent neural networks act as the basic unit of gated recurrent units to keep track of information involved in the uneven length of input sequence with small variants of hidden memory cells. Like the long short-term memory unit [10], the gated recurrent unit (GRU) also comprises gating units which control the flow of information inside the unit rather than including additional memory cells. GRU has the ability to store knowledge about the previous input of a sequence, and that knowledge will be used for the prediction of the target. The working principle of the GRU relies on two basic gated units, which consist of one reset gate and an update gate. The functionality of the reset gate can be defined as the adaptive nature of the gate to receive new inputs, whereas the update gate can be used to keep track of old information. Moreover, these gates [4] were controlled in an efficient way to work over the hidden nodes to read or write the time related sequences for prediction or some other tasks and it is given in Fig. 2.

3.3.1 Hyperparameter tuning

The architecture of gated recurrent unit has five GRU layers with an input and output dense layer. The number of neurons used in the input layer is 9 which equal to the number of input features. The output layer has only one neuron with ReLU activation function since this model performs binary classification. The number of neurons used in the consecutive three GRU layers are 256 and the number of neurons in the fourth GRU layer is 64 with ReLU activation function. The optimization function used for this transfer learning model is RMSprop with binary cross entropy as loss function. The model is trained for 10 iteration or epochs with batch size of 1000. After 10th epoch, the performance of the model becomes stable without any improvement so the value of epoch is fixed to 10.

4. System overview

In this section, the functionality of each component present in the proposed architecture has been explored in a detailed manner and it can be visualized in Fig. 1 as given below.

4.1 Source and target domain

The main terminologies used in the concept of transfer learning are source domain, target domain, source task and target task. Here, the source domain is referred to as the pretrained model, which performs well for source tasks using an IoT device dataset. In this work, both the source and target tasks are the classification of IoT telemetry data into normal and malicious classes. In the target domain, the remaining data of five IoT devices is considered individually.

4.1.1 Pseudocode of the IoT/IIoT cyber attack classification

In Algorithm 1, the steps involved in the detection of the attack at the physical layer in the IIoT environment has been elucidated. The IIoT attack classification

Algorithm 1 Pseudocode for IoT/IIoT cyber attack classification.

Input: Source dataset with labels (S_{dataset}) and target dataset without labels (T_{dataset})
Output: Label of target dataset (T_{label})
function IIoT attack classification
begin function
$S_{\text{dataset}}new \leftarrow Preprocess(S_{\text{dataset}})$
$T_{\text{dataset}}new \leftarrow Preprocess(T_{\text{dataset}})$
Train the $GRU_{\text{basemodel}}$ with $S_{\text{dataset}}new$
Save the weights
Transfer the weights and layers to $GRU_{\text{targetmodel}}$
Train the $GRU_{\text{targetmodel}}$ with $T_{\text{dataset}}new$
$T_{\text{label}} \leftarrow Predict(GRU_{\text{targetmodel}})$
end function

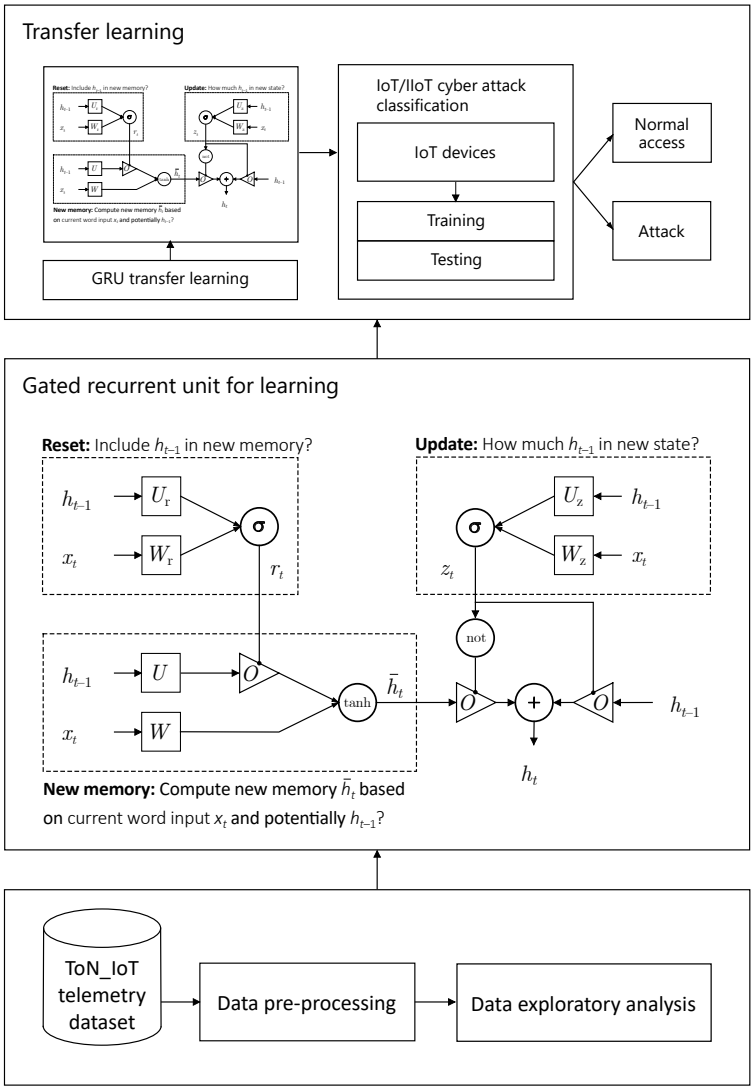

Fig. 1 *System overview.*

process takes the ToN_IoT as source dataset. It is collection of seven different IoT devices and the type of attack they are subjected to. The dataset namely consists of devices IoT fridge, IoT garage door, IoT GPS tracker, IoT Modbus unit, IoT motion light, IoT thermostat and IoT weather sensor activity. The dataset for each device is taken for preprocessing. The missing fields are removed, normalization and standardization are performed over the dataset. The dataset thus prepared can provide a better outcome when model is fitted over it. Each dataset is trained with GRU model and weights are initialized for the same. The trained model is used over the test data to classify the type of attack. For example for an IoT fridge

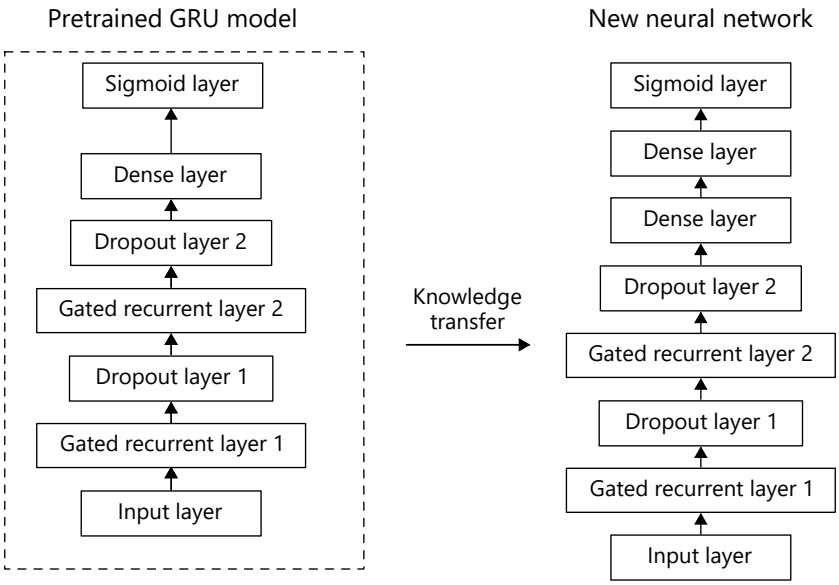

Fig. 2 *TL based GRU model.*

the types of attack are normal, DoS, DDoS and backdoor attacks. Based on the attributes like data, time, temperature, condition and label the type of attack is predicted.

4.1.2 Pseudocode of gated recurrent unit

Gated recurrent units are designed to have persistent memory unlike other neural networks which can help to generate the next hidden state h_t from h_{t-1} which is the previous state and the x_t input. The following are steps involved in the process and it is given Algorithm. 2.

4.2 TL based GRU classifier model

This module consists of two GRU models in which the first GRU model is pretrained using any one of the IoT datasets. The layers and weights of the high-performance pretrained GRU model are transferred to another GRU classifier with an additional sigmoid layer. This second is used to detect attacks on the remaining five IoT datasets with the help of the transferred knowledge. This scenario prevents keeping track of different datasets for different IoT devices. Since this TL-based model is capable of detecting attacks using the training knowledge of a single IoT device. In Tab. II, the sample distribution for all the given IoT devices has been listed.

5. Result and discussion

The experimental setup used to implement the proposed model is the Windows 10 operating system on an i3 processor at 2.30 GHz with 4 GB of RAM. Here, the deep

Algorithm 2 Pseudocode for GRU model.

Input: x_t and h_{t-1}
Output: h_t
1. Generate new memory h_t from input x_t and previous state h_{t-1} using softmax consolidation $h_t = (1–z_t O\bar{h}_t + z_t O h_{t-1})$.
2. Check the relevance of the past state and diminish it using the reset gate in case it is relevant. Else proceed to the new memory generation process in step 1.
3. Update the signal z_t if decided from the previous state that it should be carried forward to the next iteration state h_t.
If $z_t == 1$ **then**
    h_{t-1} is forwarded to the next state.
else
    h_t is forwarded to the next state.
end if
$z_t = \sigma(W_z x_t + U_z h_{t-1})$
$r_t = \sigma(W_r x_t + U_r h_{t-1})$
4. Generate the new memory $\bar{h}_t$ with the help of the update gate.
$\bar{h}_t = \tanh(r_t O U h_{t-1} + W x_t)$

Device name	Training size	Testing size
IoT fridge	469660	117416
IoT garage door	414012	177434
IoT GPS tracker	416980	178707
IoT motion light	316584	135679
IoT thermostat	309560	132669

Tab. II *Sample distribution for TL model.*

learning model GRU was developed using Python version 3.7 software installed on the aforementioned system. The performance metrics of the target domain are measured using various performance metrics such as accuracy, precision, recall, F1-measure, training time, and prediction time. The equation used to calculate the performance metrics has been given in Tab. III.

Metrics	Equation
Accuracy	$\frac{\text{TP} + \text{TN}}{\text{TP} + \text{FN} + \text{FP} + \text{TN}} \cdot 100\%$
Precision	$\frac{\text{TP}}{\text{TP} + \text{FP}}$
Recall	$\frac{\text{TP}}{\text{TP} + \text{FN}}$
F1-measure	$\frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$

Tab. III *Equation for performance metrics.*

The parameters used in the above-mentioned equations can be described and they are given below.

- TP (true positive) – This term refers to normal sensor data that has been correctly predicted as normal.
- FP (false positive) – It denotes attack sensor data which is wrongly detected as normal.
- TN (true negative) – It denotes the attack sensor data which is correctly predicted as an attack.
- FN (false negative) – It denotes the normal sensor data which is incorrectly predicted as an attack.

In this dataset consists of telemetry attributes for seven different IoT devices. They are IoT fridge, IoT garage door, IoT GPS Tracker, IoT Modbus, IoT Motion Light, IoT Thermostat and IoT Weather sensor. But the features of IoT Modbus are date, time, FC1_Read_Input_Register, FC2_Read_Discrete_Value, FC3_Read _Holding_Register, FC4_Read_Coil, label and type. And the features of IoT Weather sensor are date, time, temperature, pressure, humidity, label and type. The features of these datasets are different when compared to all others in ToN_IoT dataset. Hence the device data alone is used. In this experiment setup, dataset of five IoT devices has been considered to perform transfer learning. The performance of the proposed model has been evaluated in ten different direction by varying the dataset for source domain and target domain. For example, when IoT fridge dataset act as source then the remaining four dataset (IoT thermostat, IoT garage door, IoT motion light, IoT GPS tracker) act as target. These models has been evaluated using accuracy, precision, recall and F1-measure individually. Those obtained values for the ten models has been given in the form of heat map for each metric separately.

In Tab. IV, the row wise IoT devices indicate the source domain whereas the column wise IoT devices indicate the target domain. The source domain dataset is used to train the GRU base model or pretrained model. The weights of this pretrained model has been transfered to the target model thereby the target dataset is fed to the model without labels. This same scenario has been followed for the Tabs. V, VI, VII, VIII and IX. Tabs. V, VI, VII, VIII and IX explained precision, recall, F1-measure, predicting time, and response time in the same way that accuracy was explained in Tab. IV.

Accuracy [%]	IoT fridge	IoT garage door	IoT gps tracker	IoT motion light	IoT thermostat
IoT fridge	0	86.040	86.230	85.750	87.030
IoT garage door	85.180	0	86.260	86.260	87.040
IoT GPS tracker	85.180	86	0	88.450	89.540
IoT motion light	85.180	93.340	92.330	0	93.450
IoT thermostat	95.450	96.890	99.760	97.760	0

Tab. IV *Matrix plot for accuracy.*

Precision [–]	IoT fridge	IoT garage door	IoT gps tracker	IoT motion light	IoT thermostat
IoT fridge	0	0.823	0.744	0.736	0.757
IoT garage door	0.725	0	0.744	0.744	0.757
IoT GPS tracker	0.725	0.739	0	0.877	0.888
IoT motion light	0.835	0.912	0.902	0	0.924
IoT thermostat	0.925	0.912	0.978	0.966	0

Tab. V *Matrix plot for precision.*

Recall [–]	IoT fridge	IoT garage door	IoT gps tracker	IoT motion light	IoT thermostat
IoT fridge	0	0.861	0.862	0.857	0.870
IoT garage door	0.851	0	0.862	0.862	0.870
IoT GPS tracker	0.851	0.860	0	0.843	0.843
IoT motion light	0.912	0.903	0.923	0	0.934
IoT thermostat	0.923	0.965	0.976	0.987	0

Tab. VI *Matrix plot for recall.*

F1-measure [–]	IoT fridge	IoT garage door	IoT gps tracker	IoT motion light	IoT thermostat
IoT fridge	0	0.793	0.799	0.792	0.810
IoT garage door	0.783	0	0.799	0.799	0.810
IoT GPS tracker	0.783	0.795	0	0.875	0.856
IoT motion light	0.909	0.899	0.873	0	0.912
IoT thermostat	0.922	0.876	0.888	0.978	0

Tab. VII *Matrix plot for F1-measure.*

Training time [s]	IoT fridge	IoT garage door	IoT gps tracker	IoT motion light	IoT thermostat
IoT fridge	0	21.223	19.118	19.223	19.201
IoT garage door	22.345	0	23.456	24.344	22.355
IoT GPS tracker	28.876	26.767	0	26.787	27.877
IoT motion light	29.776	29.988	30.765	0	31.323
IoT thermostat	25.877	26.745	27.569	27.489	0

Tab. VIII *Matrix plot for training time.*

Prediction time [s]	IoT fridge	IoT garage door	IoT gps tracker	IoT motion light	IoT thermostat
IoT fridge	0	10.555	9.455	9.987	10.112
IoT garage door	11.656	0	11.455	11.223	11.019
IoT GPS tracker	11.009	12.221	0	11.876	11.112
IoT motion light	9.445	9.767	9.665	0	9.569
IoT thermostat	9.996	9.876	9.132	9.231	0

Tab. IX *Matrix plot for prediction time.*

In Tab. IV, the row wise IoT devices indicate the source domain whereas the column wise IoT devices indicate the target domain. The source domain dataset is used to train the GRU base model or pretrained model. The weights of this pretrained model has been transfered to the target model thereby the target dataset is fed to the model without labels. This same scenario has been followed for the Tabs. V, VI, VII, VIII and IX. Tabs. V, VI, VII, VIII and IX explained precision, recall, F1-measure, predicting time, and response time in the same way that accuracy was explained in Tab. IV.

In Tab. IV, the highest accuracy value achieved by each one of the IoT devices as a source domain for its corresponding target domain is elucidated below:

- IoT fridge as a source attains 95.45% of accuracy for IoT thermostat as target domain which is the maximum as compared to other devices. Other target domains show lesser accuracy when trained with IoT fridge and similar accuracy. IoT fridge taken as source and trained over the data is not tested with the IoT fridge , hence the 0 value. Similarly same device is not considered for source and destination.

- IoT garage door as a source attains 96.89% of accuracy for IoT thermostat as target domain which is the highest among other devices. Other target test data over devices like IoT fridge gives 86.04% accuracy, IoT GPS tracker gives 86% accuracy and IoT motion light gives accuracy of 93.34% as accuracy. IoT garage door taken as source and trained over the data is not tested with the IoT garage door, hence the 0 value.

- IoT GPS tracker as a source attains 99.76% of accuracy for IoT thermostat as target domain. Other target devices taken for test data like IoT fridge gives 86.23% as accuracy, IoT garage door gives 86.26% as accuracy and IoT motion light gives 92.33% as accuracy. IoT GPS tracker taken as source and trained over the data is not tested with the IoT GPS tracker, hence the 0 value.

- IoT motion light as a source attains 97.76% of accuracy for IoT thermostat as target domain Other target domains considered as target or test data for devices like IoT fridge give 85.75% as accuracy, IoT garage door gives 86.26% as accuracy and IoT GPS tracker gives 88.45% as accuracy. IoT motion light

taken as source and trained over the data is not tested with the IoT motion light, hence the 0 value.

– IoT thermostat as a source attains 93.45% of accuracy for IoT motion light as target domain. For rest of the target devices like IoT fridge , the accuracy is 87.03%. For IoT garage door, the accuracy is 87.04%. For IoT GPS tracker, the accuracy is 89.54%. For IoT motion light, the accuracy is 93.45%. IoT thermostat taken as source and trained over the data is not tested with the IoT Thermostat, hence the 0 value.

Precision is considered a better measure of performance for repeated experiments. In Tab. V, the maximum precision value achieved by each one of the IoT devices as a source domain for its corresponding target domain is elucidated below:

– IoT fridge as a source achieves precision of 0.923 for IoT thermostat as target domain. The computational model is trained with IoT fridge data and it applied over the IoT thermostat test set. The IoT garage door taken as test data provides 0.725 as precision and so does IoT GPS tracker. IoT motion light gives a precision of 0.923. The attack classification is similar for all IoT devices, Hence the application of training on a device and testing on another.
– IoT garage door as a source achieves precision of 0.912 for IoT thermostat & IoT motion light as target domain. In this instance IoT garage door is the data that is used for training and IoT motion light also has the same precision as IoT Thermostat. The IoT GPS tracker as test data gives 0.739 and IoT fridge gives 0.823 as precision. The source device dataset is not taken as test set hence the 0 for IoT garage door.
– IoT GPS tracker as a source achieves precision of 0.978 for IoT thermostat as target domain. The IoT motion light dataset used as test data for model trained over IoT GPS tracker gives a precision of 0.902 and IoT garage door gives a precision of 0.744. The IoT fridge gives a precision of 0.744.
– IoT motion light as a source achieves precision of 0.966 for IoT thermostat as target domain. The model trained over IoT motion light applied over IoT GPS tracker as test data delivers 0.877 as precision. The other devices like IoT garage door gives 0.744 as precision and IoT fridge gives 0.736 as precision.
– IoT thermostat as a source achieves precision of 0.924 for IoT motion light as target domain. Here the data of IoT thermostat used for training data and IoT GPS tracker data as test data gives 0.888 as precision. The IoT garage door device used to access the test data gives 0.757 as precision and so does IoT fridge .

Recall is a performance metric that shows ability to detect the positive samples properly. Hence it gives a different dimension of understanding. In Tab. VI, the highest recall value achieved by each one of the IoT devices as a source domain for its corresponding target domain is elucidated below:

– IoT fridge as a source achieves recall value of 0.923 for IoT thermostat as target domain. The GRU computational model trained over IoT fridge and

using IoT motion light as test data, gives 0.912 as recall. The IoT GPS tracker test data gives a recall of 0.851 and so does IoT GPS tracker and IoT garage door gives recall of 0.851. The IoT fridge test data is not applied because source is the IoT fridge itself.

- IoT garage door as a source achieves recall value of 0.965 for IoT thermostat as target domain. The IoT motion light device data taken for test set gives a recall of 0.903 and IoT GPS tracker gives a recall value of 0.86. The IoT fridge data taken for target test data gives recall value of 0.861.
- IoT GPS tracker as a source achieves recall value of 0.976 for IoT thermostat as target domain. IoT motion light gives a recall of 0.923 and IoT garage door gives a recall value of 0.862 when used as a test data. The IoT fridge gives a recall of 0.862 as test data. The same computational model is used over all the device training an test data.
- IoT motion light as a source achieves recall value of 0.987 for IoT thermostat as target domain. The IoT GPS tracker taken as test data gives recall of 0.843 and IoT garage door test data gives a recall of 0.862. The IoT fridge dataset gives a recall of 0.857. The IoT motion light source dataset taken as training data and target test data is chosen as another device. Among all IoT thermostat gives a recall of 0.987.
- IoT thermostat as a source achieves recall value of 0.934 for IoT motion light as target domain. IoT GPS tracker test data gives a recall of 0.834 and IoT garage door test data gives a recall of 0.87. And so does the IoT fridge dataset.

F1 is a performance evaluation measure which is considered an aggregate of precision and recall. In Tab. VII, the highest F1-measure value achieved by each one of the IoT devices as a source domain for its corresponding target domain is given below:

- IoT fridge as a source achieves F1-measure value of 0.922 for IoT thermostat as a target domain. IoT motion light as target dataset gives an F1-measure of 0.909 and IoT GPS tracker test data gives a F1-measure of 0.783 and so does the IoT garage door dataset.
- IoT garage door as a source achieves F1-measure value of 0.899 for IoT motion light as a target domain. The IoT thermostat test data gives a F1-measure of 0.899. IoT GPS tracker test data gives a F1-measure of 0.795. The IoT fridge test dataset gives an F1 of 0.793. The target domain is changed because the types of attacks are same across all devices. The source domain and its training data is held the same.
- IoT GPS tracker as a source achieves F1-measure value of 0.888 for IoT thermostat as a target domain. Other device test data serving as target domain like IoT motion light gives F1-measure of 0.873 and IoT garage door gives a F1-measure of 0.799. IoT fridge also gives the same value as F1-measure. The IoT GPS tracker as source domain is analyzed with different target domains

and the F1-measure performance measure shows IoT thermostat gives better performance over test data.

– IoT motion light as a source achieves F1-measure value of 0.978 for IoT thermostat as a target domain. The IoT GPS tracker taken as device test data gives an F1-measure of 0.875. The IoT garage door test data gives F1-measure of 0.799 and IoT fridge gives a F1-measure of 0.792.

– IoT thermostat as a source achieves F1-measure value of 0.912 for IoT motion light as a target domain. The IoT GPS tracker test data gives a F1-measure of 0.856. IoT garage door gives a F1-measure of 0.81 and IoT fridge gives a F1-measure of 0.81 too.

Training time is also considered a vital performance metric for evaluating computational models. If all models for target domain test data give a high value of accuracy then training time can serve as a distinguishing factor. The time also is largely affected by size of the dataset. In Tab. VIII, the least training time achieved by each one of the IoT devices as a source domain for its corresponding target domain is elucidated below:

– IoT fridge as a source domain 22.345 s while training IoT garage as a target Model. Over the dataset of IoT thermostat the time taken is 25.877 s, over IoT motion light the time taken is 29.776 and over IoT GPS tracker the time taken is 28.876 s.

– IoT garage door as a source domain 21.223 s while training IoT fridge as a target model. The IoT thermostat target domain takes 26.745 s for training, IoT motion light takes 29.998 s, IoT GPS tracker takes 26.767 s and IoT fridge takes 21.223 s for training.

– IoT GPS tracker as a source domain 19.118 s while training IoT fridge as a target model. The target domain taken as IoT thermostat takes 27.569 s, IoT motion light takes 30.765 s and IoT garage door takes 23.456 s for training.

– IoT motion light as a source domain 19.223 s while training IoT fridge as a target model. IoT thermostat as target domain takes 27.489 s for training, IoT GPS tracker takes 26.787 s and IoT garage door takes 24.344 s for training.

– IoT thermostat as a source domain 19.201 s while training IoT fridge as a target model. IoT motion light as target domain takes 31.323 s for 31.323 s and IoT GPS tracker takes 27.877 s as target test data training time. IoT garage door takes 22.355 s.

In Tab. IX, the least prediction time achieved by each one of the IoT devices as a source domain for its corresponding target domain is elucidated below:

– IoT fridge takes 9.445 s of prediction time for the dataset of IoT motion light. IoT thermostat takes 9.996 s and IoT GPS tracker takes 11.009 s, IoT garage door takes 11.656 s for prediction time.

- IoT garage door 9.767 s of prediction time for the dataset of IoT motion light. IoT thermostat data as target domain takes 9.876 s, IoT GPS tracker takes 12.221 s and IoT fridge takes 10.555 s for predicting over the test data.
- IoT GPS tracker 9.132 s of prediction time for the dataset of IoT Thermostat. The IoT motion light device data as target domain takes 9.665 s, IoT garage door takes 11.455 s, IoT fridge takes 9.455 s for prediction time.
- IoT motion light 9.231 s of prediction time for the dataset of IoT Thermostat. IoT GPS tracker takes 11.876 s, IoT garage door takes 11.223 s and IoT fridge takes 9.987 s for prediction.
- IoT thermostat 9.569 s of prediction time for the dataset of IoT motion light. The IoT GPS tracker takes 11.112 s and IoT garage door takes 11.019 s and IoT fridge 10.112 s for prediction time.

On the basis of observation, the IoT GPS tracker as a source dataset performs well in terms of accuracy, precision, recall, and F1-measure. IoT motion light stands at the next level to the IoT thermostat in terms of performance, as a source domain or base model. The lowest performance as a source model is given by the dataset of IoT thermostat in terms of accuracy value (93.45%). As a result of the analysis of parameter time, it shows that prediction time is predominantly less than the training time. In this regard, IoT motion light seems to consume more training time and prediction time than the other IoT devices eventhough its performance is good as a source model.

5.1 Performance comparison

Tab. X illustrates about the accuracy values of four DL models namely CNN, DNN, RNN and GRU. The rows of the table denotes the dataset without labels whereas the column denotes the dataset with labels used for training these four models. On the basis of the observation of the table proposed model performs better than these three state of art DL algorithms in terms of accuracy. Tab. XI demonstrates about the performance comparison between existing solution and the proposed solution. The accuracy value for all the solutions has been elucidated in the table and it shows the better performance of the proposed model. Since the proposed solution is based transfer learning technique, it is easier to develop common model with higher performance metrics.

6. Conclusion and future work

In this work, an intrusion detection framework has been proposed for IIoT with the help of sensory data and transfer learning-based deep learning approach. Here, the gated recurrent unit has been deployed as both pretrained model and the target model. In order to deploy this model in a resource-constrained IoT device, this approach was carefully designed with a minimum computational workload.

This technique will be used in the future to deploy on fog nodes that are attached to different IIoT devices and it can also be done for even multiclass classification.

Target	Algorithm used	Source				
		IoT Fridge	IoT Garage door	IoT GPS tracker	IoT Motion light	IoT Thermostat
IoT Fridge	CNN	0	82.73	82.05	81.07	81.11
	DNN	0	75.81	77.21	76.37	76.61
	RNN	0	84.72	85.01	85.73	86.74
	GRU	0	86.04	86.23	85.75	87.03
IoT Garage door	CNN	82.71	0	83.33	83.01	82.99
	DNN	72.21	0	78.32	78.01	78.82
	RNN	86.81	0	86.66	87.91	87.92
	GRU	85.18	0	86.26	86.26	87.04
IoT GPS tracker	CNN	83.82	82.72	0	82.55	83.71
	DNN	73.34	78.99	0	78.23	77.99
	RNN	86.72	86.99	0	87.31	87.03
	GRU	85.18	86.00	0	88.45	89.54
IoT Motion light	CNN	82.92	82.02	82.23	0	81.97
	DNN	71.04	71.71	71.92	0	72.22
	RNN	86.78	86.62	86.11	0	86.78
	GRU	85.18	93.34	92.33	0	93.45
IoT Thermostat	CNN	81.72	81.22	81.34	82.09	0
	DNN	79.21	78.81	78.34	78.90	0
	RNN	87.99	87.98	87.23	87.47	0
	GRU	95.45	96.89	99.76	97.76	0

Tab. X *Performance evaluation between existing deep learning technique with GRU model.*

Author & year	Algorithm used	Accuracy [%]
Tim et al. [2] & 2021	Gradient boosting machine (GBM)	94.643
	Random forest (RF)	98.075
	Multi-layer perceptron (MLP)	97.842
Abdallah et al. [6] & 2021	Logistics regression (LR)	85.900
	Naive Bayes (NB)	69.200
	Decision tree (DT)	97.200
	Random forest (RF)	97.200
	Adaboost	90.600
	k-nearest neighbour (KNN)	98.200
	Support vector machine (SVM)	86.010
	XGBoost	98.300
Amir Andalib et al. [1] & 2020	Autoencoder (LS = 4)	98.884
	Autoencoder (LS = 3)	98.817
Proposed model	Transfer learning based GRU model	99.800

Tab. XI *Comparative analysis of proposed model with existing solution.*

Secondly, for the transfer learning based solution proposed for IIoT, its performance can be further improved by adding unsupervised domain adaptation techniques such as MMD, gradient reversal layer etc. In other direction, the designed GRU model can also be trained and evaluate using network level IDS Dataset like BoT-IoT and N-BaIoT.

References

[1] ANDALIB A., VAKILI V.T. A Novel Dimension Reduction Scheme for Intrusion Detection Systems in IoT Environments. *arXiv preprint arXiv:2007.05922*. 2020.

[2] BOOIJ T.M., CHISCOP I., MEEUWISSEN E., MOUSTAFA N., den HARTOG F.T. ToN_IoT: The role of heterogeneity and the need for standardization of features and attack types in IoT network intrusion data sets. *IEEE Internet of Things Journal*. 2021, 9(1), pp. 485–496.

[3] CERON J.M., STEDING-JESSEN K., HOEPERS C., GRANVILLE L.Z., MARGI C.B. Improving iot botnet investigation using an adaptive network layer. *Sensors*. 2019, 19(3), pp. 727.

[4] CHO K., VAN MERRIËNBOER B., BAHDANAU D., BENGIO Y. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*. 2014.

[5] DIETZ C., CASTRO R.L., STEINBERGER J., WILCZAK C., ANTZEK M., SPEROTTO A., PRAS A. IoT-botnet detection and isolation by access routers. In: *2018 9th International Conference on the Network of the Future (NOF)*, 2018, pp. 88–95.

[6] GAD A.R., NASHAT A.A., BARKAT T.M. Intrusion Detection System Using Machine Learning for Vehicular Ad Hoc Networks Based on ToN-IoT Dataset. *IEEE Access*. 2021, 9, pp. 142206–142217, doi: 10.1109/ACCESS.2021.3120626.

[7] GOODFELLOW I., BENGIO Y., COURVILLE A. *Deep Learning*. 2016. http://www.deeplearningbook.org.

[8] HABIBI J., MIDI D., MUDGERIKAR A., BERTINO E. Heimdall: Mitigating the internet of insecure things. *IEEE Internet of Things Journal*. 2017, 4(4), pp. 968–978.

[9] HLAVÁČ V. Neural Network for the identification of a functional dependence using data preselection. *Neural Network World*. 2021, 31(2), pp. 109.

[10] HOCHREITER S., SCHMIDHUBER J. Long short-term memory. *Neural computation*. 1997, 9(8), pp. 1735–1780.

[11] JONÁKOVÁ L, NAGY I. Power purchase strategy of retail customers utilizing advanced classification methods. *Neural Network World*. 2021, 31(2), pp. 89.

[12] LI J., WU W., XUE D. An intrusion detection method based on active transfer learning. *Intelligent Data Analysis*. 2020, 24(2), pp. 363–383.

[13] LI X., HU Z., XU M., WANG Y., MA J. Transfer learning based intrusion detection scheme for Internet of vehicles. *Information Sciences*. 2021, 547, pp. 119–135.

[14] LIANG H., FU W., YI F. A survey of recent advances in transfer learning. In: *2019 IEEE 19th international conference on communication technology (ICCT)*, 2019, pp. 1516–1523.

[15] LUONG N.C., HOANG D.T., WANG P., NIYATO D., KIM D.I., HAN Z. Data collection and wireless communication in Internet of Things (IoT) using economic analysis and pricing models: A survey. *IEEE Communications Surveys & Tutorials*. 2016, 18(4), pp. 2546–2590.

[16] MEIDAN Y., BOHADANA M., SHABTAI A., OCHOA M., TIPPENHAUER N.O., GUARNIZO J.D., ELOVICI Y. Detection of unauthorized IoT devices using machine learning techniques. *arXiv preprint arXiv:1709.04647*. 2017.

[17] NOBAKHT M., SIVARAMAN V., BORELI R. A host-based intrusion detection and mitigation framework for smart home IoT using OpenFlow. In: *2016 11th International conference on availability, reliability and security (ARES)*, 2016, pp. 147–156.

[18] SINGLA A., BERTINO E., VERMA D. Overcoming the lack of labeled data: Training intrusion detection models using transfer learning. In: *2019 IEEE International Conference on Smart Computing (SMARTCOMP)*, 2019, pp. 69–74.

[19] TAGHIYARRENANI Z., FANIAN A., MAHDAVI E., MIRZAEI A., FARSI H. Transfer learning based intrusion detection. In: *2018 8th International Conference on Computer and Knowledge Engineering (ICCKE)*, 2018, pp. 92–97.

[20] TAHERI S., SALEM M., YUAN J.-S. Leveraging image representation of network traffic data and transfer learning in botnet detection. *Big Data and Cognitive Computing*. 2018, 2(4), pp. 37.

[21] TARIQ S., LEE S., WOO S.S. CANTransfer: transfer learning based intrusion detection on a controller area network using convolutional LSTM network. In: *Proceedings of the 35th Annual ACM Symposium on Applied Computing*, 2020, pp. 1048–1055.

[22] VU L., NGUYEN Q.U., NGUYEN D.N., HOANG D.T., DUTKIEWICZ E. Deep transfer learning for IoT attack detection. *IEEE Access*. 2020, 8, pp. 107335–107344.

[23] WEN T., KEYES R. Time series anomaly detection using convolutional neural networks and transfer learning. *arXiv preprint arXiv:1905.13628*. 2019.

[24] WU P., GUO H., BUCKLAND R. A transfer learning approach for network intrusion detection. In: *2019 IEEE 4th international conference on big data analytics (ICBDA)*, 2019, pp. 281–285.

[25] XU Y., LIU Z., LI Y., ZHENG Y., HOU H., GAO M., SONG Y., XIN Y. Intrusion Detection Based on Fusing Deep Neural Networks and Transfer Learning. In: *International Forum on Digital TV and Wireless Multimedia Communications*, 2019, pp. 212–223.

[26] ZHANG C., GREEN R. Communication security in internet of thing: preventive measure and avoid DDoS attack over IoT network. In: *Proceedings of the 18th Symposium on Communications & Networking*, 2015, pp. 8–15.