Proactive Attack Detection at the Edge through an Ensemble Deep Learning Model

Panagiotis Fountas
Department of Informatics and Telecommunications
University of Thessaly
Lamia, Greece
email: pfountas@uth.gr

Maria Papathanasaki
Department of Informatics and Telecommunications
University of Thessaly
Lamia, Greece
email: mpapathanasaki@uth.gr

Kostas Kolomvatsos
Department of Informatics and Telecommunications
University of Thessaly
Lamia, Greece
email: kostasks@uth.gr

Nikos Tziritas
Department of Informatics and Telecommunications
University of Thessaly
Lamia, Greece
email: nitzirit@uth.gr

Abstract*—The new form of the Web involves numerous devices present in two infrastructures, i.e., the Internet of Things (IoT) and the Edge Computing (EC) infrastructure. IoT devices are adopted to record ambient data and host lightweight processing to provide support for applications offered to end users. EC is placed between the IoT and Cloud and can be the host of more advanced processing activities. It has gained popularity due to the increased computational resources compared to the IoT and the decreased latency in the provision of responses compared to the Cloud. A high number of nodes may be present at the EC that should secure the Quality of Service (QoS) of the desired applications. Apparently, EC nodes become central points where the collected data are collected and processed. Data processing (especially when data are sensitive) imposes various security issues that should be mitigated in order to maintain high QoS levels and the uninterrupted functioning of EC nodes. In this paper, motivated by the need of the increased security, we propose an ensemble scheme for the detection of attacks in the EC. Our distributed scheme relies on the adoption of deep learning to proactively detect potential malfunctions. Our model is embedded in EC nodes and is continuously applied upon the streams of data transferred by IoT devices to the EC. We present the details of our approach and evaluate it through a variety of simulation scenarios. Our intention is to reveal the strengths and weaknesses of the provided model when adopted in a very dynamic environment like the EC.

Keywords—Attack Detection, Internet of Things, Edge Computing, Cloud Computing, Deep Learning.

I. INTRODUCTION

Nowadays, Web is widely used in industrial systems, in the evolution of the Internet of Things (IoT) applications, the development of computer networks, and so on and so forth. The IoT is a vast network of billions of different interconnected devices that produce incessantly data [1]. These data are generated or recorded by IoT devices which are constantly increasing due to Industry 4.0, environmental monitoring, smart mobility, etc. The growth of these devices, in both private and public environments, has made them an integral part of our daily lives, thus, attracting a high number of attackers [20]. The uncontrolled deployment of these devices in the environment inevitably leads to attacks, compromising the leakage of sensitive data, unauthorized access to websites, or even Denial of Service (DoS). We need to keep in mind that many IoT devices have low-end specifications that often make them more prone to attacks and may compromise the entire network. Another security risk faced by IoT devices is Advanced Persistent Threats (APTs) [21]. Groups of well-trained and well-funded individuals discover security vulnerabilities in IoT systems and attack in a highly organized manner. The security of IoT devices is crucial targeting to the provision of confidentiality, integrity, availability, but also privacy for users. For the above reasons, security in IoT devices has been increasing in recent years. Modern tools have significantly improved the pre-existing security gaps; however, the need remains and undoubtedly intensifies. The heterogeneity of IoT devices, in both their protocols and at the physical level, makes the provision of security models particularly challenging. Equally demanding is the fact that there are innumerable nodes, but also the absence of modern security methods in them [24].

Due to the computational limitations of IoT devices, often, 'heavy' tasks such as classification, which is particularly important in detecting attacks, are transferred to edge devices to distribute the workload close to data sources [2]. All these data are used to export analytics [3] and support efficient decision making. Frequently, data are characterized by missing values [4], outliers, and other factors that affect the data quality and make it unsuitable for decision making. Data quality is a multidimensional concept. Frequently mentioned dimensions are accuracy, completeness, consistency, and timeliness. We should consider that any of these aspects can be adversely affected by any abnormal behavior, such as an outlier, a missing value, etc. Such behavior can be caused by unusual behavior of an IoT device that has either been attacked or has technical failure [25].

Except from the data quality a significant task in EC, is to ensure the integrity of the data from various attacks and malwares. The growing need for ever-greater remote connectivity has sparked attacks on computer systems. Therefore, the detection of such attacks became a critical and integral part in the creation and use of computer systems in order to ensure their integrity and the security of the data used [5, 6]. Attacks such as brute-force, buffer overload, phishing, trojan horses etc. can cause significant problems in the computer systems that are attacked. Over the years a great deal of effort has been made to ensure security in the Edge. The constant increase of well-organized attacks leads to the creation of imaginative but quite complex solutions that remained in the spotlight for a short time. However, there are models that adopt artificial intelligence techniques to perform

978-1-6654-6667-7/21/$31.00 ©2021 IEEE
DOI 10.1109/IUCC-CIT-DSCI-SmartCNS55181.2021.00018

unsupervised learning on sensor data for anomaly detection. More specifically, autonomous security nodes (Intrusion Detection Systems - IDS) are used, which are placed close to the IoT device we are interested in and have the ability to perform either in real time or with minimal delay, checks for possible intrusions, identification and potentially deal with them [26]. Edge security issues have also been addressed in scenarios involving real-time data streams collected by weak IoT devices and sent to EC nodes for further processing to protect sensitive data. In this case, the use of standard encryption mechanisms, such as AES, may not be applicable at the device level. A lighter encryption model is therefore adopted in which a driver device obscures the identity of individual devices, while at the same time allowing easier management of encryption keys and encrypted data [27].

In this paper, we depart from the state-of-the-art solutions and propose an ensemble model for the detection of attacks. Our approach combines two deep learning models, i.e., an autoencoder [28] and a Long Short Term Memory LSTM model [29] in order to autonomously detect attacks upon streams of data collected by IoT devices. Before applying the attack detection scheme as dictated by the LSTM model, we pre-process the incoming data and perform a dimensionality reduction through the use of the autoencoder. This step is necessary because in order to rely on the most significant features of data and reveal hidden aspects of the attacks. In addition, dimensionality reduction will assist in speeding up the detection process as less features will be fed into the LSTM model minimizing the decision-making time. Both models are 'connected' in a sequential order, i.e., the outcome of the autoencoder is fed into the aforementioned LSTM model that decides for the presence of an attack. Among various types of deep learning schemes, we strategically select the use of the LSTM model for its capability to learn long-term dependencies. Compared to past efforts in the domain, the novelty of our approach is found in the model's ability to perform dimensionality reduction in data with nonlinear correlation and also to identify long term dependencies on dimensionality reduced data. The following list presents the salient contributions of our paper:

- We provide an attack detection model for Edge Computing (EC), combining two deep learning models;
- We enhance the behavior of EC nodes with a mechanism that conducts dimensionality reduction, even in nonlinear correlated data, using an autoencoder;
- We evaluate the proposed model through its comparison to decision tree classifiers, a Naive Bayes classifier and a probabilistic model proposed in [32], using two datasets.

The paper is organized as follows. In section II, we present the prior work in the specific domain Section III reports on the preliminary information and the high level architecture of our model. The proposed model is thoroughly described in Section IV. Section V discusses the experimental setup as well as the evaluation of our model by giving numerical performance outcomes. Finally, in Section VI, we conclude our paper and present future extensions that will expand the research outcomes.

II. RELATED WORK

The fast increase of usage of the Edge and Cloud computing creates new security challenges in which the research community has to propose mechanisms against the threats and vulnerabilities [19]. The detection of attacks has always been a challenge for the research community which studies algorithms that efficiently protect various systems from attacks. The existence of an abnormal behavior in a computer system may indicate the infection of the system, e.g., by a malware. Many research activities focus on the detection of such anomaly behaviors. In [7], the authors propose an outlier detection method which takes under consideration a vector of data to confirm a candidate outlier using a sliding window approach. The detection of outliers may be considered as a means for the detection of attacks in the sense that outliers depict abnormalities in the usual, normal data. A high number of research efforts deal with the detection of outliers. The majority of them adopt statistical methods to detect the deviation of data from the 'common' distribution as detected in historical measurements [30, 31, 32, 33]. In [8] is proposed a system level Device-Edge split IDS for IoT devices which first analyze the behavior of devices and then detect anomalous behaviors which may constitute an intrusion on the devices. The advantage of this system is the ability to detect efficiently the anomalous behaviors with the minimum possible overhead for the IoT device. Moreover, in [11], the authors develop an intelligent Multi-Task Learning framework, which is called Cyber-Typhon, combining On-Line Sequential Extreme Learning Machines (OS-ELM) and Restricted Boltzmann Machines (RBMs) to detect Advanced Persistent Threats (APT) attacks through the anomaly detection. Cyber-Typhon detects correlated features with the network traffic to identify if the traffic of the network is normal using the OS-ELM. In case OS-ELM identify the traffic as threat then puss it forward to responsible RBM to identify the type of the threat. A survey on EC security is presented in [17] where the challenges are described while pointing out the problems of the EC in five categories, i.e., access control, key management, privacy protection, attack migration and anomaly detection. Furthermore, the authors analyze the existing solutions which have been proposed by the research community and discuss about the future directions on EC security. The authors of [9] present a lightweight attack detector in the IoT infrastructure which is based on a learning recurrent Random Neural Network. The low computational cost of the detector proposed in [9] makes it suitable for the detection of certain types of Botnet attacks in IoT system and for deployment by edge devices. In [10], a distributed detection scheme is proposed that uses ELM classifiers. It should be mentioned that the proposed scheme adopts High Performance Computing (HPC) cluster resources, improving the performance of the model. A Blockchain Security Architecture is presented in [12] to secure the network communications between traded Industrial IoT devices based on deep learning smart contracts. These contracts execute a mutual traffic control agreement using a trained deep autoencoder neural network to detect anomalies. The research effort presented in [13], proposes a web attack detection system which relies on analysis of URLs and can be applied at any EC node to detect web attacks. The web attack detection system consists of three phases i.e., data preparation, discrimination and action. It relies on two deep learning models i.e., Convolutional Neural Networks (CNN) and models in Natural Language Processing (NLP). In addition, the paper presented in [18] proposes a Channel State

Information (CSI) management frame authentication system for spoofing attack detection, which is called PHYAlert and is appropriate for protecting Wi-Fi-based edge networks. Similarly, the authors in [22] propose FlowGuard, an edge centric IoT defense scheme relying on two deep learning models i.e., LSTM model and CNN, for the detection, identification, classification, and mitigation of IoT DDoS attacks. In [23], an intrusion detection system which is called APAE is presented. APAE relies on an asymmetric parallel Autoencoder which adopts standard and dilated convolutional filters. Its architecture makes it capable of detecting in real time various categories of attacks in IoT networks.

III. PRELIMINARIES & HIGH LEVEL ARCHITECTURE

We consider a network of n IoT devices depicted by the following set $I = \{I_1, I_2, I_w \dots, I_n\}$. IoT devices collect data from their environment using sensors and interact among them using the network. In our scenario, we adopt a set of m EC nodes $EN = \{EN_1, EN_2, EN_i, \dots, EN_m\}$ which gather the data from the IoT devices to perform processing activities. Without loss of generality, every EN_i 'supervises' a group G_j of IoT devices such that $G = \{G_1, G_2, G_j, \dots, G_k\}$ and $G_1 \cap G_2 \cap G_j \cap G_k = \emptyset$ (see Fig. 1). More specifically, EC nodes receive the data in the form of multivariate vectors i.e., $X_t^w = \{x_1^w[t], x_2^w[t], x_z^w[t], \dots, x_d^w[t]\}$ where indexes w, t represent the w^{th} IoT device and the time instance when the data vector is reported. Also, every dimension $x_z^w[t]$ is characterized by indexes t and z, which express the time instance and the dimension of X_t^w from the w^{th} IoT device respectively. Every time an EC node receives a data vector passes it through the proposed ensemble scheme to detect if the IoT device which reports the data vector has been infected by a malware.

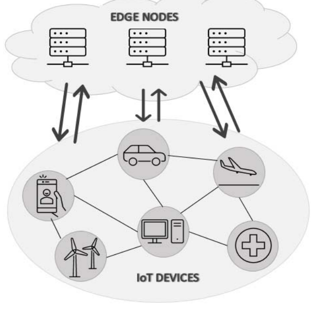

Fig. 1. The network of our scenario

Our proposed ensemble scheme consists of a sequential connection of two deep learnings models i.e., autoencoders and a LSTM model network. More specifically, we use the former model to perform a dimensionality reduction over the data received by streams. The autoencoder conducts the dimensionality reduction based on its ability to compress data with non-linear correlations which differs with other relevant techniques like the Principal Component Analysis (PCA) [34]. After the dimensionality reduction step, the outcome of the autoencoder, i.e., $X'^w_t = \{x'^w_1[t], \dots, x'^w_r[t]\}, r < d$ is fed as input to the LSTM model. On the other hand, the LSTM model is used for binary classification of reduced data vectors into the proper class. The reason that we decide to adopt an LSTM model as it is capable of learning long-term dependencies upon data. We consider the set of classes $C = \{A, \overline{A}\}$, i.e., A depicts an attack and $\overline{A}$ depicts a normal data vector. The LSTM model output expresses the decision of the proposed scheme for the data vector X_t^w using a representation into lower dimensions.

IV. ATTACK DETECTION BASED ON ENSEMBLE SCHEME

A. Dimensionality Reduction for Attacks Detection

Autoencoders belong to the category of unsupervised learning algorithms, i.e., no training data should be taken into consideration. One of the most significant applications of autoencoders is dimensionality reduction where the autoencoder finds the representation of data in less dimensions than the original ones, focusing only on the most significant features. They are also adopted to eliminate noise from images by learning the noise during the training process, in order to remove the noise from new images [35]. Another application of autoencoders is anomaly detection where the model is fed with data and learns to reconstruct them with a low error. When the autoencoder is fed with anomaly data, the error is high enough that it exceeds a predefined threshold, thus, it can easily detect anomalies. The 'architecture' of an autoencoder consists of: (i) an encoder; (ii) a bottleneck, and; (iii) a decoder.

An autoencoder works as follows [14,15]. An input vector $(X_t^w)^T \in \mathbb{R}^{d\times 1}$ goes through the hidden layer where compression / dimensionality reduction is performed according to (1):

$$X'^w_t = \alpha(W_1 (X_t^w)^T + b_1) \qquad (1)$$

The outputs of the hidden layer are depicted by $X'^w_t \in \mathbb{R}^{r\times 1}$, with $r < d$. In our approach, we reduce the dimensions of data into a two-dimensional representation. Additional parameters (depicted by the above presented equation) are applied in the management of data as they transferred through the layers of the network. These parameters are:

- $W_1 \in \mathbb{R}^{r\times d}$ refers to the weight for the first layer, i.e., the encoder;
- $b_1 \in \mathbb{R}^{r\times 1}$ expresses the bias vector for the first layer. The bias allows the activation equation to be shifted to the right or left;
- α represents the activation equation which is usually a non-linear equation like Rectified Linear Unit (ReLU) function.

The second layer, i.e., the decoder, tries to reconstruct the input from the reduced data with the highest possible accuracy. This process is performed by applying (2).

$$\hat{X}_t^w = \alpha(W_2 X'^w_t + b_2) \qquad (2)$$

In (2), the following parameters are adopted to generate the initial inputs with the lowest possible error:

- $W_2 \in \mathbb{R}^{d\times r}$ refers to the weight for the second layer, i.e., decoder;
- $b_2 \in \mathbb{R}^{d\times 1}$ expresses the bias vector for the second layer (decoder).

For the calculation of W_1 , W_2 and B_1 , B_2 , the autoenconder tries to minimize a loss equation (usually, the Mean Square Error - MSE), that measures the difference between the input and the reconstructed vector resulting from the dimensionality reduced data. Equation (3) depicts the minimization step of the detected error.

$$minimize\ \{L\} = minimize\{f\left((X_t^w)^T, \hat{X}_t^w\right)\} \quad (3)$$

Using backpropagation and an optimizer, such as the stochastic gradient descent, each data sample $(X_t^w)^T$ goes through the autoencoder to calculate X'^w_t and $\hat{X}_t^w$.

B. The Proposed LSTM model

An LSTM model consists of three parts known as gates and a cell state or memory cell. These gates are: (i) the input gate; (ii) the forget gate and; (iii) the output gate (see Fig. 2 [36]). Assuming we have h hidden units, the batch size equal to v and the number of inputs equal to u, we notate the input, the hidden state of current and previous timestamps as $Y_t \in \mathbb{R}^{v\times u}$ and $H_t, H_{t-1} \in \mathbb{R}^{v\times h}$, respectively. In our scenario, the input of LSTM model is the output of autoencoder, i.e., $Y_t = (X'^w_t)^T$. The value of each part of the LSTM network can be calculated by the following equations [16]:

$$I_t = \sigma(Y_t W_{xi} + H_{t-1} W_{hi} + b_i) \quad (4)$$

$$F_t = \sigma\left(Y_t W_{xf} + H_{t-1} W_{hf} + b_f\right) \quad (5)$$

$$O_t = \sigma(Y_t W_{xo} + H_{t-1} W_{ho} + b_o) \quad (6)$$

where σ is the sigmoid equation $W_{xi}, W_{xf}, W_{xo} \in \mathbb{R}^{u\times h}$, and $W_{hi}, W_{hf}, W_{ho} \in \mathbb{R}^{h\times h}$ are weight parameters and $b_i, b_f, b_o \in \mathbb{R}^{1\times h}$ are biases. A significant component in the operation of LSTMs is the memory cell which is transferred through the repeating modules to solve the vanishing gradient problem [37]. The following equation holds true:

$$C_t = F_t \odot C_{t-1} + \mathrm{I}_t \odot \tilde{C}_t \quad (7)$$

In (7), the input gate decides the percentage of contribution of the new data via the candidate memory cell $\tilde{C}_t$ while the forget gate adjusts the contribution of the old Memory Cell C_{t-1}. The candidate memory cell $\tilde{C}_t$ is calculated as follows:

$$\tilde{C}_t = tanh(\mathrm{X}_t W_{xc} + H_{t-1} W_{hc} + b_c) \quad (8)$$

where $W_{xc} \in \mathbb{R}^{u\times h}$ and $W_{hc} \in \mathbb{R}^{h\times h}$ are weight parameters and $b_c \in \mathbb{R}^{1\times h}$ is a bias vector. The last but not least part in the LSTM module is the hidden state H_t which expressed as the next equation shows:

$$H_t = O_t \odot \tanh(C_t) \quad (9)$$

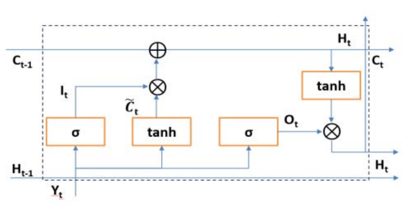

Fig. 2. The architecture of an LSTM neuron

C. Data Driven Attacks Detection

In our approach, we sequentially combine the above-described deep learning models to produce an ensemble scheme which is capable to classify data and detect potential attacks using only the most significant features of data. In particular, we use the first component of the autoencoder to perform the envisioned dimensionality reduction and identify the appropriate features, i.e., we incorporate into our decision-making module the encoder part. More specifically, the output of the encoder produces vectors with two features i.e.,$X'^w_t = \{x'^w_1[t], x'^w_2[t]\}$. Then, the reduced data are transferred as inputs to the LSTM model network to perform the desired classification. Obviously, the LSTM model is pre-trained adopting a relevant dataset defined by historical values and experts. The benefit of this approach is the improvement of time complexity that the LSTM model needs to perform the binary classification of every data vector i.e., A or $\bar{A}$. In case a data vector is detected as attack, the EC node rejects the vector and interrupts the communication with IoT device until it receives a certification that the device is no longer infected. However, this approach lies beyond the scope of the current paper.

V. EXPERIMENTAL SETUP AND EVALUATION

In our experimental evaluation, we rely on the occupancy detection dataset[a]. The dataset consists of experimental data used for binary classification (room occupancy) and contains values for the temperature, humidity, light, CO_2 and occupancy of an office room. The ground truth occupancy is obtained from time stamped pictures that were taken every minute. More specifically, the proposed model is trained over the aforementioned dataset which is already divided into three parts. Out of the total of 20,560 data samples, 8,143 (≅39%) become the training set, 2,665 (≅13%) samples is the first testing set and the remaining 9,752 (≅ 47%) is the second testing set. With this approach, we experiment with two cases: (a) in the former experimental scenario, we consider the 75% of data as the training dataset (8,143 out of 10,808 instances); (b) in the latter experimental scenario, we consider the 45% of data as the training dataset (8,143 out of 17,895 instances). The dataset is not associated with the attack detection; however, it is a representative dataset for classification problems that matches to our approach. We assume the presence of a person in the room as an attack, i.e., occupancy = 1 and his/her absence as a normal condition, i.e., occupancy = 0 . We consider three baseline models, a Decision Tree Classifier (DTC), Naive Bayes Classifier (NBC) and a statistical model proposed in [32] comparing their performance with our model, i.e., the Attack Detection based on Reduced Data (ADRD) scheme. For the evaluation, we rely on Precision, Recall, Accuracy, True Negative Rate (TNR) and F_1 score, as they are realized by the calculation of True Positives (TP), True Negatives (TN), False Positives (FP) and False Negatives (FN). We define as TP the number of predictions correctly identified in the appropriate class (attack or occupancy), TN the number of predictions correctly identified as absence of an attack (or occupancy), FP the number of instances which are wrongly characterized as positive events and FN the number of instances which are wrongly characterized as negative events. The discussed performance metrics are calculated as follows:

$$Precision = TP/(TP + FP) \quad (10)$$

$$Recall = TP/(TP + FN) \quad (11)$$

$$Accuracy = (TP + TN)/(TP + TN + FP + FN) \quad (12)$$

$$TNR = TN/(TN + FP) \quad (13)$$

[a] https://archive.ics.uci.edu/ml/datasets/Occupancy+Detection+

$$F_1\ score = (2 \cdot TP)/(2 \cdot TP + FP + FN) \quad (14)$$

The best performance of the ADRD is achieved when these metrics reach the maximum value, i.e., the unity. In that case, ADRD manages to eliminate false positives and false negatives which means that attacks are efficiently detected without missing any event.

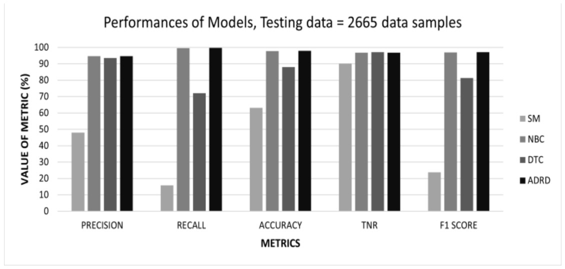

Fig. 3. Performances of Models for the 1st testing dataset

Fig. 3 presents the performance of the aforementioned models for the first testing dataset which contains 2,665 data samples. As we can see, the ADRD has better performance for all metrics compared to the remaining models except from the TNR, which is higher in DTC by 0.3544%. The metric which shows the effectiveness of our model against the others, is F_1 score which takes under consideration both Precision and Recall metrics. Tis depicts the ability of the proposed model to minimize FP and FN events.

In Fig. 4, we observe the performance evaluation in a larger dataset than in the previous experimental scenario with 9,752 data samples. Recall shows the ADRD's ability to detect attacks correctly eliminating FN events. NB has twice the FN of ADRD, which makes it unreliable in this attack detection mechanism, as it ranks more attacks as non-attacks as opposed to ADRD. We have to notice that this is critical as any false negative event jeopardizes the functioning of EC nodes. In that sense, FN events are more important than FP because any undetected attack will harm the stability of EC nodes and the network. More specifically, we observe that Recall in ADRD is 99.65% and in NBC is 99.31%, so their difference is 0.34%. This number may be relatively low, however, if we consider the number of the incoming data to EC nodes, we can easily identify the negative effects and the potentials for resulting undetected attacks. For instance, in 1,000 events, 340 FN events can be translated into 340 attacks not detected by the NBC model. TNR is 2.1 points higher in NBC than ADRD. The same stands for the Precision, which shows the model's 'anxiety' to detect attacks classifying normal events as abnormal scenarios. In general, the NBC shows a bad binning of continuous variables being not suitable for imbalanced data that it is not the case with the ADRD.

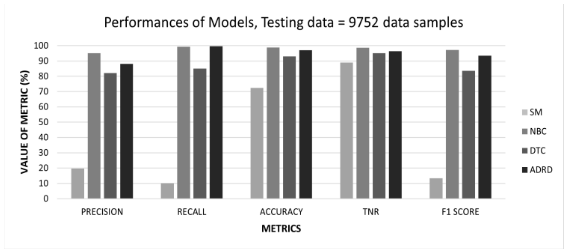

Fig. 4. Performances of Models for the 2nd testing dataset

Our evaluation reveals that the proposed model manages to perform the best possible results when the amount of the training data dominates our datasets (first experimental scenario). Apparently, a low number of training instances negatively affects the performance of our model which is expected as the LSTM model requires enough data in order to learn the hidden statistics of the defined dataset. In any case, the ADRD achieves performance outcomes above 88% for the entire set of the adopted metrics (Precision, Recall, Accuracy, TNR, F_1 score).

In Fig. 5. we present the time required for the training process of the ADRD, the DTC and the NBC.

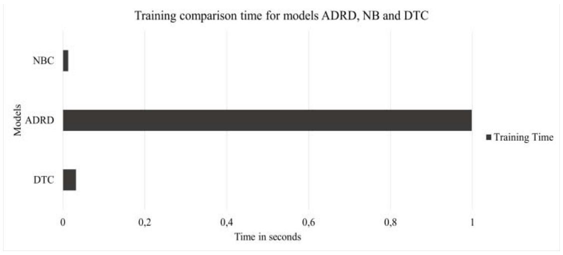

Fig. 5. Time comparison for training proccess

The time required for ADRD training is clearly affected by the number of the hidden layers of the LSTM model. Our LSTM model has 256X128X64X32X60 units placed at five layers. The time that the LSTM model requires for training is ~0.998 seconds per epoch. The NBC training time is 0.013 seconds and for the DTC it is 0.032 seconds. In summary, the DTC requires less training time than the ADRD, but the performance metrics are worse than our model. As for NBC, it is also faster in training than the ADRD, but lags behind the proposed model for the reasons already provided in the above description. The SM is not included in this evaluation process as it does not rely on a training phase.

VI. CONCLUSIONS AND FUTURE WORK

There is no doubt that attack detection is a very significant research subject that concerns the security of any ICT system. Currently, there are ongoing efforts to ensure security on the EC in a variety of ways. In the case of this paper, we proposed a model that utilizes data with non-linear correlation and after a dimensionality reduction phase adopted to detect the most significant features, data are processed by an LSTM model to decide whether they depict an attack or not. Our ensemble scheme is able to achieve very high levels of accuracy as evaluated through a high number of simulations. However, the architecture of LSTM model networks does not allow them to be efficient in non-sequential data. Undoubtedly, the proposed model responds efficiently to the detection of attacks and could be incorporated into a real system. An extension of this work can be found on the implementation of a more complex model that can respond to both sequential and non-sequential data in order to detect attacks effectively.

REFERENCES

[1] T. Gopalakrishnan et al., "Deep Learning Enabled Data Offloading With Cyber Attack Detection Model in Mobile Edge Computing Systems," in IEEE Access, vol. 8, pp. 185938-185949, 2020, doi: 10.1109/ACCESS.2020.3030726.J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[2] Rafał Kozik, Michał Choraś, Massimo Ficco, Francesco Palmieri, A scalable distributed machine learning approach for attack detection in edge computing environments, Journal of Parallel and Distributed Computing, Volume 119, 2018, Pages 18-26, ISSN 0743-7315, https://doi.org/10.1016/j.jpdc.2018.03.006.K. Elissa, "Title of paper if known," unpublished.

[3] Fountas P., Kolomvatsos K., Anagnostopoulos C. (2021) A Deep Learning Model for Data Synopses Management in Pervasive Computing Applications. In: Arai K. (eds) Intelligent Computing. Lecture Notes in Networks and Systems, vol 284. Springer, Cham. https://doi.org/10.1007/978-3-030-80126-7_44

[4] P. Fountas and K. Kolomvatsos, "Ensemble based Data Imputation at the Edge," 2020 IEEE 32nd International Conference on Tools with Artificial Intelligence (ICTAI), 2020, pp. 961-968, doi: 10.1109/ICTAI50040.2020.00150.

[5] Panagiotis, Fountas, Kouskouras Taxiarxchis, Kranas Georgios, Leandros Maglaras, and Mohamed A. Ferrag 2021. "Intrusion Detection in Critical Infrastructures: A Literature Review" Smart Cities 4, no. 3: 1146-1157. https://doi.org/10.3390/smartcities4030061

[6] Y. Xiao, Y. Jia, C. Liu, X. Cheng, J. Yu and W. Lv, "Edge Computing Security: State of the Art and Challenges," in Proceedings of the IEEE, vol. 107, no. 8, pp. 1608-1631, Aug. 2019, doi: 10.1109/JPROC.2019.2918437.

[7] Kolomvatsos, K., Anagnostopoulos, C., 'Landmark based Outliers Detection in Pervasive Applications', ICICS, 2021.

[8] Anand Mudgerikar, Puneet Sharma, and Elisa Bertino. 2020. Edge-Based Intrusion Detection for IoT devices. ACM Trans. Manage. Inf. Syst. 11, 4, Article 18 (December 2020), 21 pages. DOI:https://doi.org/10.1145/3382159

[9] Filus K., Domańska J., Gelenbe E. (2021) Random Neural Network for Lightweight Attack Detection in the IoT. In: Calzarossa M.C., Gelenbe E., Grochla K., Lent R., Czachórski T. (eds) Modelling, Analysis, and Simulation of Computer and Telecommunication Systems. MASCOTS 2020. Lecture Notes in Computer Science, vol 12527. Springer, Cham. https://doi.org/10.1007/978-3-030-68110-4_5

[10] Rafał Kozik, Michał Choraś, Massimo Ficco, Francesco Palmieri, A scalable distributed machine learning approach for attack detection in edge computing environments, Journal of Parallel and Distributed Computing, Volume 119, 2018, Pages 18-26, ISSN 0743-7315, https://doi.org/10.1016/j.jpdc.2018.03.006.

[11] Demertzis K., Iliadis L., Kikiras P., Tziritas N. (2019) Cyber-Typhon: An Online Multi-task Anomaly Detection Framework. In: MacIntyre J., Maglogiannis I., Iliadis L., Pimenidis E. (eds) Artificial Intelligence Applications and Innovations. AIAI 2019. IFIP Advances in Information and Communication Technology, vol 559. Springer, Cham. https://doi.org/10.1007/978-3-030-19823-7_2

[12] Demertzis, K., Iliadis, L., Tziritas, N. et al. Anomaly detection via blockchained deep learning smart contracts in industry 4.0. Neural Comput & Applic 32, 17361–17378 (2020). https://doi.org/10.1007/s00521-020-05189-8

[13] Z. Tian, C. Luo, J. Qiu, X. Du and M. Guizani, "A Distributed Deep Learning System for Web Attack Detection on Edge Devices," in IEEE Transactions on Industrial Informatics, vol. 16, no. 3, pp. 1963-1971, March 2020, doi: 10.1109/TII.2019.2938778.

[14] Ferreira D, Silva S, Abelha A, Machado J. Recommendation System Using Autoencoders. Applied Sciences. 2020; 10(16):5510. https://doi.org/10.3390/app10165510

[15] Ali, S., & Li, Y. (2019). Learning multilevel auto-encoders for DDoS attack detection in smart grid network. IEEE Access, 7, 108647-108659.

[16] Yao, K., Cohn, T., Vylomova, K., Duh, K., & Dyer, C. (2015). Depth-gated LSTM. arXiv preprint arXiv:1508.03790.

[17] H. Zeyu, X. Geming, W. Zhaohang and Y. Sen, "Survey on Edge Computing Security," 2020 International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE), 2020, pp. 96-105, doi: 10.1109/ICBAIE49996.2020.00027.

[18] Jiang, Z., Zhao, K., Li, R. et al. PHYAlert: identity spoofing attack detection and prevention for a wireless edge network. J Cloud Comp 9, 5 (2020). https://doi.org/10.1186/s13677-020-0154-7

[19] ALMENDAH, Ohood M.; ALZAHRANI, Sabah M. Cloud and Edge Computing Security Challenges, Demands, Known Threats, and Vulnerabilities. Acad. J. Res. Sci. Pub, 2021.

[20] F. Meneghello, M. Calore, D. Zucchetto, M. Polese and A. Zanella, "IoT: Internet of Threats? A Survey of Practical Security Vulnerabilities in Real IoT Devices," in IEEE Internet of Things Journal, vol. 6, no. 5, pp. 8182-8201, Oct. 2019, doi: 10.1109/JIOT.2019.2935189.

[21] Cheng, X., Zhang, J., Tu, Y., & Chen, B. (2020). Cyber situation perception for Internet of Things systems based on zero‐day attack activities recognition within advanced persistent threat. Concurrency and Computation: Practice and Experience, e6001.

[22] Y. Jia, F. Zhong, A. Alrawais, B. Gong and X. Cheng, "FlowGuard: An Intelligent Edge Defense Mechanism Against IoT DDoS Attacks," in IEEE Internet of Things Journal, vol. 7, no. 10, pp. 9552-9562, Oct. 2020, doi: 10.1109/JIOT.2020.2993782.

[23] Basati, A., Faghih, M.M. APAE: an IoT intrusion detection system using asymmetric parallel auto-encoder. Neural Comput & Applic (2021). https://doi.org/10.1007/s00521-021-06011-9

[24] Hassan, W. H. (2019). Current research on Internet of Things (IoT) security: A survey. Computer networks, 148, 283-294

[25] Wand, Y., & Wang, R. Y. (1996). Anchoring data quality dimensions in ontological foundations. Communications of the ACM, 39(11), 86-95

[26] Zissis, D. (2017, June). Intelligent security on the edge of the cloud. In 2017 International Conference on Engineering, Technology and Innovation (ICE/ITMC) (pp. 1066-1070). IEEE.

[27] Jolfaei, A., & Kant, K. (2019). *Data security in multiparty edge computing environments*. Temple University Philadelphia United States.

[28] Bank, D., Koenigstein, N., & Giryes, R. (2020). Autoencoders. *arXiv preprint arXiv:2003.05991*.

[29] Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., & Schmidhuber, J. (2016). LSTM: A search space odyssey. *IEEE transactions on neural networks and learning systems*, *28*(10), 2222-2232

[30] Singh, D. A. A. G., & Leavline, E. J. (2016). Model-based outlier detection system with statistical preprocessing. Journal of Modern Applied Statistical Methods, 15(1), 39.

[31] Hubert, M., Rousseeuw, P.J. & Segaert, P. Multivariate functional outlier detection. Stat Methods Appl 24, 177–202 (2015). https://doi.org/10.1007/s10260-015-0297-8

[32] C. Manikopoulos and S. Papavassiliou, "Network intrusion and fault detection: a statistical anomaly approach," in IEEE Communications Magazine, vol. 40, no. 10, pp. 76-82, Oct. 2002, doi: 10.1109/MCOM.2002.1039860.

[33] Pittino F, Puggl M, Moldaschl T, Hirschl C. Automatic Anomaly Detection on In-Production Manufacturing Machines Using Statistical Learning Methods. Sensors. 2020; 20(8):2344. https://doi.org/10.3390/s20082344

[34] Plaut, E. (2018). From Principal Subspaces to Principal Components with Linear Autoencoders. ArXiv, abs/1804.10253

[35] Bajaj, K., Singh, D. K., & Ansari, M. A. (2020). Autoencoders based deep learner for image denoising. Procedia Computer Science, 171, 1535-1541.

[36] Kolomvatsos, K., Anagnostopoulos, C., 'A Deep Learning Model for Demand-driven, Proactive Tasks Management in Pervasive Computing', IoT, MDPI, 1(2), 240-258, 2020.

[37] Zhang, A., Lipton, Z. C., Li, M., & Smola, A. J. (2021). Dive into Deep Learning. arXiv preprint arXiv:2106. 11342, 354–357.