
Downloaded from: https://e-space.mmu.ac.uk/627198/
Version: Accepted Version
Publisher: Institute of Electrical and Electronics Engineers (IEEE)
DOI: https://doi.org/10.1109/jiot.2020.3034156

Please cite the published version
Hybrid Deep Learning for Botnet Attack Detection in the Internet of Things Networks
Segun I. Popoola, Bamidele Adebesi, Mohammad Hammoudeh, Guan Gui, Haris Gacanin

Abstract—Deep Learning (DL) is an efficient method for botnet attack detection. However, the volume of network traffic data and memory space required is usually large. It is, therefore, almost impossible to implement the DL method in memory-constrained IoT devices. In this paper, we reduce the feature dimensionality of large-scale IoT network traffic data using the encoding phase of Long Short-Term Memory Autoencoder (LAE). In order to classify network traffic samples correctly, we analyse the long-term inter-related changes in the low-dimensional feature set produced by LAE using deep Bidirectional Long Short-Term Memory (BLSTM). Extensive experiments are performed with the Bot-IoT dataset to validate the effectiveness of the proposed hybrid DL method. Results show that LAE significantly reduced the memory space required for large-scale network traffic data storage by 91.89%, and it outperformed state-of-the-art feature dimensionality reduction methods by 18.92–27.03%. Despite the significant reduction in feature size, the deep BLSTM model demonstrates robustness against model under-fitting and over-fitting. It also achieves good generalisation ability in binary and multi-class classification scenarios.

Index Terms—Internet of Things, botnet detection, dimensionality reduction, Long Short-Term Memory, autoencoder.

I. INTRODUCTION

NOWADAYS, critical infrastructures such as power generation [1]–[4], communications [5]–[7], healthcare [8]–[10], manufacturing [11]–[13], transportation [14], [15], water treatment [16] and agriculture [17], [18] are interconnected in order to tap into the various benefits of the Internet of Things (IoT) [19]–[21]. The increased inter-connectivity and the use of Industrial Control Systems (ICS) renders smart critical infrastructures vulnerable to cyberattacks from terrorists or “hacktivists”. For instance, ICS and IoT devices have been proven to be easily hackable and remotely controllable to form IoT-based botnets [22], [23]. Successful exploitation of a single vulnerable IoT device can lead to leakage of sensitive information and serious security breaches in the wider IoT-enabled system [24]. This makes them an attractive target to Advanced Persistent Threats (APT) of diverse botnet attacks, especially when they are deployed in critical environments.

To secure connected IoT devices against complex botnet attacks, Machine Learning (ML) techniques have been employed to develop Network Intrusion Detection Systems (NIDS), e.g., [25]. Such NIDS can be installed at strategic points within an IoT network. Specifically, Deep Learning (DL), an advanced ML approach, offers a unique capability for automatic extraction of features from large-scale, high-speed network traffic generated by interconnected heterogeneous IoT devices [26]. Considering the resource-constraints in IoT devices, NIDS techniques used in classical computer networks are not efficient for botnet detection in IoT systems due to high computation and memory requirements [27]. In order to develop an efficient DL method for botnet detection in IoT networks, sufficiently large network traffic information is needed to guarantee efficient classification performance [28]. However, processing and analyzing high-dimensional network traffic data can lead to curse of dimensionality [29]. Also, training DL models with such high-dimensional data can cause Hughes phenomena [30]. High-dimensional data processing is complex and requires huge computational resources and storage capacity [31], [32]. IoT devices do not have sufficient memory space to store big network traffic data required for DL. Therefore, there is a need for end-to-end DL-based botnet detection method that will reduce high dimensionality of big network traffic features and also detect complex and recent botnet attacks accurately based on low-dimensional network traffic information.

Currently, Bot-IoT dataset [33] is the most relevant publicly available dataset for botnet attack detection in IoT networks because it: (a) has IoT network traffic samples; (b) captured complete network information; (c) has a diversity of complex IoT botnet attack scenarios; (d) contains accurate ground truth labels; and (e) provides massive volume of labeled data required for effective supervised DL. The original feature dimensionality1 of the Bot-IoT dataset is 43, and the memory space required to store this network traffic data is 1.085 GB. So far, feature dimensionality reduction methods that have been applied to the Bot-IoT dataset were all based on feature selection techniques. These techniques include the filter

Manuscript received April 16, 2020; revised August 27, 2020.

This work is supported in part by Cyraatek Ltd UK, the Faculty of Science & Engineering, Manchester Metropolitan University, and in part by ENERGY-IQ project, a UK-Canada Power Forward Smart Grid Demonstrator project funded by The Department for Business, Energy and Industrial Strategy (BEIS) under Grant 7454460; and the NICE (Nigerian Intelligent Clean Energy) Marketplace project funded by the Department for International Development (DFID).

S. I. Popoola and B. Adebesi are with the Department of Engineering, Faculty of Science and Engineering, Manchester Metropolitan University, Manchester M1 5GD, United Kingdom. E-mail: s.popoola@mmu.ac.uk; b.adebesi@mmu.ac.uk

M. Hammoudeh is with the Department of Computing and Mathematics, Manchester Metropolitan University, Manchester M1 5GD, United Kingdom. E-mail: m.hammoudeh@mmu.ac.uk

G. Gui is with the College of Telecommunications and Information Engineering, Nanjing University of Posts and Telecommunications (NJUPT), Nanjing, 210003 China. E-mail: guiguian@njupt.edu.cn

H. Gacanin is with the Institute for Communication Technologies and Embedded Systems, RWTH Aachen University, Aachen, Germany. E-mail: harisg@ice.rwth-aachen.de

1Feature dimensionality is the total number of network traffic features in a given dataset
method with Pearson Correlation and Entropy (PCE) [33], X-Mean clustering with Particle Swarm Optimisation (XMP) [34], and Information Gain (IFG) [35].

On the other hand, Long Short-Term Memory Autoencoder (LAE) is an effective DL method that produces a low-dimensional latent-space feature representation of a high-dimensional feature set at its hidden layer. To the best of our knowledge, this DL method has not been previously applied to reduce the dimensionality of the feature set in the Bot-IoT dataset. Also, deep Bidirectional Long Short-Term Memory (BLSTM) is a DL method that learns hierarchical feature representations and long-term inter-related changes directly from raw data using multiple hidden layers. However, this DL method has not been previously used to classify latent-space representation of network traffic features.

In this paper, we propose a hybrid DL framework, called LAE-BLSTM, for efficient botnet detection in IoT networks using LAE and deep BLSTM algorithms. The main contributions of this paper are as follows:

1) A single hidden-layered LAE is proposed for feature dimensionality reduction. This method reduces the dimensionality of large-scale IoT network traffic data and produces a low-dimensional latent-space feature representation at the hidden layer without losing useful intrinsic network information;

2) A deep BLSTM is proposed for network traffic classification. This method analyses the long-term inter-related changes in low-dimensional feature set produced by LAE to distinguish botnet attack traffic from benign traffic in IoT networks;

3) Extensive experiments are performed with the Bot-IoT dataset to validate the effectiveness of LAE-BLSTM in binary and multi-class classification scenarios;

4) The performance of state-of-the-art optimisation algorithms was investigated and compared to ensure efficient feature dimensionality reduction and botnet attack detection in IoT networks.

The remaining parts of the paper are organized as follows: In Section II, we review related state-of-the-art methods for feature dimensionality reduction and network traffic classification. In Section III, we describe the hybrid DL framework (LAE-BLSTM) proposed for botnet detection in IoT networks. Extensive experiments are performed in Section IV to validate the effectiveness of LAE-BLSTM. Experimentation results are presented and discussed in Section V. Finally, we conclude the paper in Section VI.

II. RELATED WORK

Although several datasets are available for network intrusion detection, they have various challenges, including lack of reliable labels, low attack diversity, redundancy of network traffic, and missing ground truth. For instance, KDD Cup99 and NSL-KDD datasets are popularly used, but they are outdated, and they do not reflect current normal and attack scenarios [36], [37]. The application of the DEFCON-8 dataset is limited because of the low number of benign traffic samples [36]. The attack scenarios in the UNIBS dataset are limited to DoS; the network traffic data are presented in packets with no extracted features, and the labels were not provided [38]. CAIDA datasets have no ground truth information about the attack samples [38]. Network traffic samples in the LBNL dataset were not labeled, and the features were not extracted from the packet files [38]. Attack samples in the UNSW-NB15 dataset were generated in a synthetic environment [39]. ISCX and CICIDS2017 datasets were generated based on the concept of profiling, and this can be due to their innate complexity. Also, the ground truth of these datasets is not available to enhance the labeling process. Not much information is given about the botnet scenarios that were used in most datasets [36], [38]–[40]. Also, IoT network traffic data was not included in related datasets [36], [39], [41]–[44].

Bot-IoT dataset [33] is the most relevant dataset that is publicly available for network-based botnet attack detection in IoT networks. To realise this dataset, an IoT network testbed was set up to generate benign and malicious network traffic using heterogeneous communication protocols which include User Datagram Protocol (UDP), Transmission Control Protocol (TCP), Address Resolution Protocol (ARP), Internet Control Message Protocol (ICMP), Internet Protocol version-6 ICMP (IPv6-ICMP), Internet Group Management Protocol (IGMP), and Reverse Address Resolution Protocol (RARP). The testbed setup comprised a variety of IoT devices, including a weather station, smart fridge, motion-activated lights, remotely-activated garage door, and smart thermostat. Also, millions of IoT botnet attack traffic samples were included in Bot-IoT. These attack traffic samples can be categorized into four IoT botnet scenarios, namely: DDoS, DoS, reconnaissance and information theft.

Feature dimensionality reduction is mostly achieved by applying either linear or non-linear transformation technique to high-dimensional feature set. Principal Component Analysis (PCA) [45] is one of the common linear transformation methods while kernel methods [46], spectral methods [47] and DL methods [48] employ non-linear transformation techniques. Autoencoder is an unsupervised DL method that produces latent-space representation of input data at the hidden layer. Different autoencoder architectures have been proposed to reduce the feature dimensionality in most popular network intrusion datasets. These methods were implemented and evaluated with the network traffic data in publicly available datasets which include KDD-Cup99 [49]–[56], NSL-KDD [49], [57]–[60], UNSW-NB15 [50], [59], [61] and CICIDS2017 [62]. Table I shows the autoencoder-based feature dimensionality reduction techniques in the literature. The original feature dimensionality, feature reduction method, new feature dimensionality, classifier and classification scenarios were provided. Long Short-Term Memory (LSTM) is a variant of Recurrent Neural Network (RNN) and it has the capacity to learn long-term dependencies in network traffic features [63]–[65]. However, none of the proposed autoencoder-based methods in Table I was implemented nor validated with Bot-IoT dataset.

Different feature selection methods have been proposed to reduce the dimensionality of network traffic features in Bot-IoT dataset. Table II presents an overview of state-
TABLE I
DIMENSIONALITY REDUCTION OF NETWORK TRAFFIC FEATURES USING AUTOENCODER

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Ref.</th>
<th>Input features</th>
<th>Reduction method</th>
<th>Output features</th>
<th>Classifier</th>
<th>Classification scenarios</th>
</tr>
</thead>
<tbody>
<tr>
<td>KDD-Cup99</td>
<td>[49]</td>
<td>41</td>
<td>Stacked deep autoencoder</td>
<td>28</td>
<td>RF</td>
<td>Binary, multi-class</td>
</tr>
<tr>
<td></td>
<td>[50]</td>
<td>41</td>
<td>Stacked deep autoencoder</td>
<td>10</td>
<td>Softmax</td>
<td>Binary, multi-class</td>
</tr>
<tr>
<td></td>
<td>[51]</td>
<td>122</td>
<td>Autoencoder</td>
<td>100</td>
<td>CNN, softmax</td>
<td>Multi-class</td>
</tr>
<tr>
<td></td>
<td>[52]</td>
<td>41</td>
<td>Autoencoder</td>
<td>13</td>
<td>k-means</td>
<td>Binary</td>
</tr>
<tr>
<td></td>
<td>[53]</td>
<td>41</td>
<td>Stacked autoencoder</td>
<td>5</td>
<td>Decision tree</td>
<td>Binary</td>
</tr>
<tr>
<td></td>
<td>[54]</td>
<td>41</td>
<td>Variational autoencoder</td>
<td>20</td>
<td>k-means</td>
<td>Multi-class</td>
</tr>
<tr>
<td></td>
<td>[55]</td>
<td>41</td>
<td>Autoencoder</td>
<td>18</td>
<td>k-means</td>
<td>Multi-class</td>
</tr>
<tr>
<td></td>
<td>[56]</td>
<td>39</td>
<td>Autoencoder</td>
<td>3</td>
<td>k-means</td>
<td>Binary</td>
</tr>
<tr>
<td>NSL-KDD</td>
<td>[49]</td>
<td>41</td>
<td>Stacked deep autoencoder</td>
<td>28</td>
<td>RF</td>
<td>Binary, multi-class</td>
</tr>
<tr>
<td></td>
<td>[57]</td>
<td>122</td>
<td>Stacked sparse autoencoder</td>
<td>2</td>
<td>SVM</td>
<td>Binary, multi-class</td>
</tr>
<tr>
<td></td>
<td>[58]</td>
<td>115</td>
<td>Stacked sparse autoencoder</td>
<td>10</td>
<td>Logistic</td>
<td>Binary, multi-class</td>
</tr>
<tr>
<td></td>
<td>[59]</td>
<td>41</td>
<td>Deep autoencoder</td>
<td>3</td>
<td>Deep FFNN</td>
<td>Binary, multi-class</td>
</tr>
<tr>
<td></td>
<td>[60]</td>
<td>52</td>
<td>Autoencoder</td>
<td>2</td>
<td>-</td>
<td>Multi-class</td>
</tr>
<tr>
<td>UNSW-B15</td>
<td>[50]</td>
<td>42</td>
<td>Stacked deep autoencoder</td>
<td>10</td>
<td>Softmax</td>
<td>Binary, multi-class</td>
</tr>
<tr>
<td></td>
<td>[61]</td>
<td>207</td>
<td>Semi-supervised autoencoder</td>
<td>2</td>
<td>Decision tree</td>
<td>Binary, multi-class</td>
</tr>
<tr>
<td></td>
<td>[59]</td>
<td>41</td>
<td>Deep autoencoder</td>
<td>3</td>
<td>Deep FFNN</td>
<td>Binary, multi-class</td>
</tr>
<tr>
<td>CICIDS2017</td>
<td>[62]</td>
<td>81</td>
<td>Stacked sparse autoencoder</td>
<td>64</td>
<td>RF</td>
<td>Binary, multi-class</td>
</tr>
<tr>
<td>Bot-IoT</td>
<td>This paper</td>
<td>37</td>
<td>LAE</td>
<td>6</td>
<td>Deep BLSTM</td>
<td>Binary, multi-class</td>
</tr>
</tbody>
</table>

TABLE II
DIMENSIONALITY REDUCTION OF NETWORK TRAFFIC FEATURES IN Bot-IoT DATASET

<table>
<thead>
<tr>
<th>Ref.</th>
<th>Method</th>
<th>Feature size</th>
<th>Classifier</th>
<th>Classification scenarios</th>
</tr>
</thead>
<tbody>
<tr>
<td>[33]</td>
<td>PCE</td>
<td>10</td>
<td>SVM, RNN, LSTM</td>
<td>Binary</td>
</tr>
<tr>
<td>[34]</td>
<td>XMP</td>
<td>10</td>
<td>SVM, DNN, C4.5 decision tree</td>
<td>Binary</td>
</tr>
<tr>
<td>[35]</td>
<td>IFG</td>
<td>13</td>
<td>C5 decision tree, One-class SVM</td>
<td>Multi-class</td>
</tr>
<tr>
<td></td>
<td>This paper</td>
<td>6</td>
<td>Deep BLSTM</td>
<td>Binary, Multi-class</td>
</tr>
</tbody>
</table>

of-the-art feature dimensionality reduction methods that are related to this paper. Koroniotis et al. [33] employed PCE method for feature selection. The authors reported that 10 optimal network traffic features were selected. These include seq, stddev, N_IN_Conn_P_SrcIP, min, state_number, mean, N_IN_Conn_P_DstIP, drate, srcrate and max. Support Vector Machine (SVM), Recurrent Neural Network (RNN) and Long Short Term Memory (LSTM) models were trained with the optimal features to perform binary classification in general and specific attack detection scenarios. Asadi et al. [34] identified optimal feature clusters and eliminated outliers using XMP technique. Most relevant network traffic features were selected based on the XMP method. The best binary classification performance was recorded when SVM, Dense Neural Network (DNN) and C4.5 decision tree classifiers were trained with 10 network traffic features. These network traffic features include Proto, state_number, durs, mean, dpkts, drate, TnPBPDstIP, TnP_PSrcIP, TnP_P_Dport and N_IN_Conn_P_DstIP. Khraisat et al. [35] selected 13 network traffic features using information gain (entropy) method. These features include dport, seq, dur, fflags_number, fflags, sport, N_IN_Conn_P_DstIP, srcrate, AR_P_Protocol_Sport, daddr, TnPBPDstIP, rate and AR_P_Protocol_SrcIP. An ensemble classifier, which comprised of C5 decision tree and One-Class SVM (OCSVM) models, was trained with the selected network traffic features to perform multi-label classification.

To ensure a fair comparison, feature dimensionality reduction methods that do not include benign network traffic traces and all the four botnet attack scenarios in the Bot-IoT dataset were not included in this paper. For instance, Soe et al. [66] did not consider the DoS attack scenario. Also, the performance of the method in detecting benign network traffic was not reported. In a similar work [67], the authors did not evaluate the performance of the proposed method. In another work [68], Guerra-Manzanares et al. did not evaluate the performance of the proposed method with the network traffic data in the Bot-IoT dataset.

In summary, the state-of-the-art methods in the related work focused on the selection of specific features from available network traffic information available in the Bot-IoT dataset. However, this approach may likely affect the efficiency of botnet attack detection in IoT networks because the classifiers will not have access to some relevant network information during training, validation, and testing. Consequently, the feature selection approach may lead to low botnet attack detection accuracy and a high false alarm rate in IoT networks. On the other hand, LAE reduces the dimensionality of big IoT network traffic data and produces a low-dimensional latent space feature representation at the hidden layer without losing useful intrinsic network information.

III. PROPOSED METHOD FOR BOTNET ATTACK DETECTION IN IoT NETWORKS

In this section, we propose a hybrid DL framework, named LAE-BLSTM, for efficient botnet attack detection in IoT networks. The description of LAE and deep BLSTM methods is presented in Algorithms 1 and 2, respectively. In this paper, boldface uppercase alphabets and boldface lower case alphabets represent matrices and column vectors respectively.
A. LSTM Autoencoder

Autoencoder is an unsupervised DL method which analyses the dynamic relationships between the features of high-dimensional data and produces a low-dimensional latent-space representation. Unlike the conventional Autoencoder, LAE employs LSTM units to model the long-term inter-related changes in network traffic features. In this subsection, LAE method is developed to reduce feature dimensionality of big IoT network traffic data and obtain a low-dimensional latent-space feature representation with minimum reconstruction error.

A sequence of network traffic features is represented by a three-dimensional matrix, \( X \in \mathbb{R}^{n \times 1 \times k} \), in Equation 1:

\[
X = \{x_{d,j}\}_{j=1}^{k} \}_{d=1}^{n},
\]

where \( \{x_d\}_{d=1}^{n} = x_1, x_2, \ldots, x_n \), \( k \in \mathbb{Z}^{+} \) is the number of network traffic features, and \( n \in \mathbb{Z}^{+} \) is the total instances of network traffic available in the dataset. An instance of network traffic is represented by Equation 2:

\[
\{x_j\}_{j=1}^{k} = x_1, x_2, \ldots, x_k,
\]

where \( x_j \in \mathbb{R}^{u} \) is a network traffic feature vector.

The encoder part of a single hidden layered LSTM autoencoder [69] was used to compress the network traffic feature matrix given by Equation 2 with the aim of reducing its dimensionality without losing the information contained in the original data. LSTM has the capability to learn long time-dependencies with its feedback connections and a recurrent memory unit that is controlled by three gates, namely, input gate, a forget gate and an output gate [70]. LAE accepts high-dimensional network traffic feature set, \( X \), and produces a low-dimensional latent-space representation, \( \tilde{X} \), at the hidden layer. The input gate vector (\( i_j \)), forget gate vector (\( f_j \)), memory cell state vector (\( c_j \)), output gate vector (\( o_j \)) and hidden state vector (\( h_j \)) were formed based on the LAE algorithm presented in Algorithm 1. The dimension of the column vectors is expressed by \( i_j, f_j, c_j, o_j, h_j \in \mathbb{R}^{u} \), where \( u \) is the number of LSTM hidden units that represent the desired network traffic feature dimensionality.

Weight matrices and bias vectors were obtained by training the LAE using the Back Propagation Through Time (BPTT) algorithm [71]. The weight matrices for the connections between the input and the recurrent gates are given as \( W_{ix}, W_{fx}, W_{ex}, W_{ox} \in \mathbb{R}^{u \times n} \), and these are the weight matrices of input-to-input gate connection, input-to-forget gate connection, input-to-cell connection and input-to-output gate connection respectively. Similarly, weight matrices for connections between the recurrent gates and the hidden state are given as \( W_{ih}, W_{fh}, W_{eh}, W_{oh} \in \mathbb{R}^{u \times n} \), and these are the weight matrices of input-to-hidden state connection,
forget gate-to-hidden state connection, cell state-to-hidden state connection and output gate-to-hidden state connection respectively. On the other hand, \( b_1, b_f, b_c, b_o \in \mathbb{R}^n \) are the bias vectors of input gate, forget gate, cell state and output gate respectively. Recurrent activation function is a sigmoid function and it is represented by \( \sigma_r \). Hidden layer activation function is represented by \( \sigma_h \). Diagonal matrices, \( D_i \) and \( D_f \), are formed with input gate vector and forget gate vectors as represented by Equations 3 and 4 respectively:

\[
D_i = \text{diag}(i) = \begin{bmatrix} i_1 & \cdots & i_h \end{bmatrix}, \quad (3)
\]

\[
D_f = \text{diag}(f) = \begin{bmatrix} f_1 & \cdots & f_h \end{bmatrix}. \quad (4)
\]

For \( x_j \), the encoded output is given in Equation 5.

\[
\hat{s}_j = k_\phi(x_j, c_{j-1}), \quad (5)
\]

where \( c_{j-1} \) is the previous cell state vector and \( k_\phi \) is the encoding function. Finally, the low-dimensional network traffic feature matrix is represented by Equation 6:

\[
\bar{X} = \left\{ \{\hat{s}_{d,j}\}^n_{j=1} \right\}^n_{d=1}. \quad (6)
\]

### B. Bidirectional LSTM

Conventional LSTM is unidirectional and it can only capture the dependence of the current state based on previous context [72]. Meanwhile, BLSTM has full access to both past and future sequential information using two LSTM hidden layers to scan input data sequence in positive and negative time directions respectively [73]. Therefore, a deep BLSTM method is developed to efficiently detect IoT botnet attack traffic by analysing the long-term inter-related changes in low-dimensional features produced by LAE.

Reduced network traffic feature set, \( \bar{X} \), and its corresponding target vector, \( \bar{y} \), were fed into a deep BLSTM model to produce input gate vectors (\( \tilde{I}_j, \tilde{F}_j \)), forget gate vectors (\( \tilde{F}_j, \tilde{F}_j \)), memory cell state vectors (\( \tilde{C}_j, \tilde{C}_j \)), output gate vectors (\( \tilde{O}_j, \tilde{O}_j \)) and hidden state vectors (\( \tilde{H}_j, \tilde{H}_j \)) based on the BLSTM algorithm provided in Algorithm 2. These column vectors are represented as (\( \tilde{I}_j, \tilde{F}_j, \tilde{F}_j, \tilde{C}_j, \tilde{C}_j, \tilde{O}_j, \tilde{O}_j, \tilde{H}_j, \tilde{H}_j \)). Weight matrices (\( \tilde{W}_{i}, \tilde{W}_{i} \)) and bias vectors (\( \tilde{b}_{i}, \tilde{b}_{i} \)) were obtained by training the BLSTM using the BPTT algorithm. Recurrent activation function is a sigmoid function and it is represented by \( \sigma_r \); hidden layer activation function is represented by \( \sigma_h \); while output layer activation function is a softmax function and it is represented by \( \sigma_y \). The parameters of the forward LSTM hidden layer are computed from the past to the present input data sequence while those of the backward LSTM hidden layer are calculated starting from the future to the present input data sequence. The LSTM hidden layers in the positive and negative time directions are jointly connected to the output layer. The difference between the predicted output of LAE-BLSTM, \( \hat{y} \), and the target output, \( y \), is minimized in Equation 7.

\[
\theta(y, \hat{y}) = \left[ \theta(y_d, \hat{y}_d) \right]_{d=1}^{n}, \quad (7)
\]

where \( \theta \) is either a binary cross-entropy loss function for binary classification or a categorical cross-entropy loss function for multi-class classification scenarios.

### IV. FEATURE DIMENSIONALITY REDUCTION AND NETWORK TRAFFIC CLASSIFICATION

In this section, we implement the hybrid DL framework proposed in Section III (i.e. LAE-BLSTM) and perform extensive experiments with Bot-IoT [33] to validate its effectiveness for botnet attack detection in IoT networks. The overall architecture of LAE-BLSTM is shown in Fig. 1.

All the experiments performed in this paper leveraged Numpy, Pandas, Scikit-learn and Keras libraries in Python programming language. Python codes were written and implemented within Spyder Integrated Development Environment (IDE) running on Ubuntu 16.04 LTS workstation with the following specifications: Random Access Memory (32 GB), Processor (Intel Core i7-9700K CPU @ 3.60GHz × 8), Graphics (GeForce RTX 2080 Ti/PXCl/e/SSE2) and 64-bit Operating System (OS).

#### A. Data pre-processing

Large network traffic data in Bot-IoT dataset was pre-processed to transform the features and the ground truth labels into appropriate formats for ease of computation. In this study, data pre-processing involves the following: (a) elimination of redundant network information; (a) random division of complete network traffic data into training, validation and testing sets; (b) selection of network traffic features and ground truth labels; (c) normalization or scaling of network traffic features; and (d) integer encoding of ground truth labels.

Redundant network traffic information (\( pkSeqID, sadd, daddr, proto, state \) and flgs) were removed from Bot-IoT. Therefore, only 37 out of the 43 network traffic features were selected to form high-dimensional network traffic feature set. The description of each of the 43 features are provided in [33]. The high-dimensional feature set was randomly divided into training set (70%), validation set (15%) and testing set (15%) with reference to most recent works covering diverse areas of application, e.g., [74]–[79]. Table III presents the distribution of data samples in training, validation and testing sets under binary and multi-class classification scenarios. Elements of the high-dimensional feature set were normalized to a range of \([0,1]\) using min-max transformation method given by Eq. (8):

\[
x_{\text{norm}} = \frac{x - x_{\text{min}}}{x_{\text{max}} - x_{\text{min}}}, \quad (8)
\]

where \( x \) is a network traffic feature vector; while \( x_{\text{min}} \) and \( x_{\text{max}} \) are the minimum and maximum values of \( x \) respectively. This method retains the original distribution of network traffic features. The pre-processed high-dimensional network traffic feature set was named RAW-F.
Fig. 1. LAE-BLSTM architecture for botnet attack detection in IoT networks

<table>
<thead>
<tr>
<th>Classification scenario</th>
<th>Target</th>
<th>Training</th>
<th>Validation</th>
<th>Testing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary</td>
<td>Attack</td>
<td>2,567,548</td>
<td>550,303</td>
<td>550,194</td>
</tr>
<tr>
<td></td>
<td>Normal</td>
<td>325</td>
<td>67</td>
<td>85</td>
</tr>
<tr>
<td>Multi-class</td>
<td>DDoS</td>
<td>1,348,654</td>
<td>288,809</td>
<td>289,161</td>
</tr>
<tr>
<td></td>
<td>DoS</td>
<td>1,155,031</td>
<td>247,680</td>
<td>247,549</td>
</tr>
<tr>
<td></td>
<td>Normal</td>
<td>325</td>
<td>67</td>
<td>85</td>
</tr>
<tr>
<td></td>
<td>Reconnaissance</td>
<td>63,806</td>
<td>13,476</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Theft</td>
<td>57</td>
<td>14</td>
<td>8</td>
</tr>
</tbody>
</table>

On the other hand, binary and multi-class ground truth labels were converted into integer format for ease of computation. Specifically, binary ground truth labels i.e. attack and normal were represented by 0 and 1 respectively. Similarly, multi-class ground truth labels i.e. DDoS, DoS, normal, reconnaissance and theft were represented by 0, 1, 2, 3 and 4 respectively.

B. LAE for feature dimensionality reduction

LAE model was developed to reduce the feature dimensionality and the data size of RAW-F such that the limited available memory space in IoT devices will be sufficient for network traffic data storage. The internal architecture of LAE is shown in Fig. 1, and its working principles were discussed earlier in Section IIIA. This DL method employed the following hyperparameters: a single LSTM layer, 37 input neurons, six neurons at the LSTM layer, a Rectified Linear Unit (ReLU) activation function, Mean Square Error (MSE) as a loss function, a learning rate of 0.001, 10 epochs, and a batch size of 64. LAE models were trained with state-of-the-art optimisation algorithms that are available in Keras ML library\(^2\). These optimisation algorithms include Adaptive moment estimation (Adam) [80], Stochastic Gradient Descent with Nesterov momentum (SGD) [81], RMSprop [82], Adadelta [83], Adagrad [84], Adamax [80], Adam with Nesterov momentum (Nadam) [85], and Follow The Regularised Leader (FTRL) [86]. To ensure efficient feature dimensionality reduction, we investigated and compared the performance of LAE model when each of these optimisation algorithms was used. The performance evaluation was based on reconstruction loss, data size of low-dimensional features, and the rate of feature dimensionality reduction. Finally, the performance of the optimal LAE model was compared with that of three state-of-the-art feature dimensionality reduction methods i.e. PCE [33], XMP [34], and IFG [35]. The low-dimensional features produced by LAE, PCE [33], XMP [34], and IFG [35] methods are referred to as PCE-F, XMP-F, and IFG-F, respectively.

\(^2\)https://keras.io/api/optimizers
C. Deep BLSTM for network traffic classification

Given the low-dimensional feature set, LAE-F, binary and multi-class deep BLSTM models were developed to correctly detect benign network traffic and botnet attack traffic in IoT networks. The network architecture of deep BLSTM model is shown in Fig. 1, and its principles of operation were previously explained in Section III B. In this paper, a deep BLSTM model comprised of a single LSTM layer with six input neurons, four dense hidden layers, and an output layer. The LSTM layer and the four dense hidden layers have 100 output neurons each. The output layer used a single neuron and five neurons for binary and multi-class classification, respectively. ReLU activation function was employed in all the layers of deep BLSTM model, except the output layer. A sigmoid activation function and a softmax activation function were employed at the output layer for binary and multi-class classification, respectively. The hyperparameters that were used to train deep BLSTM model include: a learning rate of 0.0001, 20 epochs, a batch size of 64, as well as a binary cross-entropy loss function and a categorical cross-entropy loss function for binary and multi-class classification, respectively. Deep BLSTM models were trained with Adam, SGD, RMSprop, Adadelta, Adagrad, Adamax, Nadam, and FTRL optimisation algorithms.

Table III shows that the distribution of network traffic samples in Bot-IoT dataset is highly imbalanced across the classes in both binary and multi-class classification scenarios. It has been experimentally proven that class imbalance affects the value and meaning of traditional classification performance metrics as they become bias in favour of the majority class [87]. Therefore, highly imbalanced data classification cannot be evaluated using traditional performance metrics. In a case where both classification successes and errors are to be considered, Matthews Correlation Coefficient (MCC) is the best null-biased performance metric for highly imbalanced data classification [87]. The value of MCC is calculated using Eq. (9):

\[
MCC = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP + FP) \times (TP + FN) \times (TN + FP) \times (TN + FN))}},
\]

where TP is the number of IoT botnet attack samples that are correctly classified as IoT botnet attack traffic; FP is the number of normal samples that are misclassified as IoT botnet attack traffic; TN is the number of normal samples that are correctly classified as normal traffic; and FN is the number of IoT botnet attack samples that are misclassified as normal traffic.

For efficient network traffic classification, we investigated and compared the performance of deep BLSTM model when each of these optimisation algorithms was used. The performance of these models was evaluated with the training, validation and testing sets. We analysed the training loss, validation loss, and MCC values to determine the most efficient optimisation algorithm for deep BLSTM model. In binary classification scenario, the performance of the optimal deep BLSTM model was compared with that of four state-of-the-art classifiers i.e. SVM [33], RNN [33], LSTM [33], and VCDL [88]. In multi-class classification scenario, the performance of the optimal deep BLSTM model was compared with that of three state-of-the-art classifiers i.e. FNN [89], SNN [89], and C5-0CSVM [35].

V. RESULTS AND DISCUSSION

In this section, we present and analyse the results of the experiments performed in Section IV to validate the effectiveness of LAE-BLSTM model for botnet attack detection in IoT networks. The performance of LAE-BLSTM model is compared with that of the state-of-the-art ML and DL models in binary and multi-class classification scenarios. In this context, LAE-BLSTM model is considered to be efficient if it meets all the following conditions:

1) Relatively low memory space requirement for big network traffic data storage. This is evaluated based on reconstruction loss, data size of low-dimensional features, and the rate of feature dimensionality reduction. Low reconstruction loss, low data size, and high reduction rate imply that relatively low memory space is required for big network traffic data storage.

2) Robustness against model under-fitting and over-fitting. Model under-fitting is evaluated based on the training loss and the MCC value obtained when deep BLSTM model was evaluated with the training set. Model over-fitting is evaluated based on the validation loss and MCC value obtained when deep BLSTM model was evaluated with the validation set. Low training loss and high MCC value on training set imply that deep BLSTM model is robust against model under-fitting. On the other hand, low validation loss and high MCC value on validation set imply that deep BLSTM model is robust against model over-fitting.

3) Good generalization ability. This is evaluated based on the MCC values obtained when deep BLSTM model was evaluated with the testing set. High MCC values on testing set implies that deep BLSTM model performs well on previously unseen network traffic data.

A. Results of feature dimensionality reduction

To determine the most suitable optimiser for efficient feature dimensionality reduction, we analyse the reconstruction losses generated when Adam, SGD, RMSprop, Adadelta, Adagrad, Adamax, Nadam, and FTRL optimisation algorithms were used to train LAE model. Fig. 2 shows that the use of Adam optimiser produced the lowest reconstruction loss (0.0041 - 0.0023) throughout the 10-epoch training period. Table IV shows that the use of Adam optimiser outperformed the use of the other seven optimisers by 39.55 – 89.33%.

Data sizes of the low-dimensional feature sets (PCE-F, XMP-F, IFG-F, and LAE-F) were compared with that of the original feature set (RAW-F) to evaluate the impact of feature dimensionality reduction methods. Table V shows that LAE and state-of-the-art methods significantly reduced the data size of RAW-F. However, LAE achieved the lowest data size (88.04 MB) with a reduction rate of 91.89%. This method outperformed XMP [34], IFG [35], and PCE [33] methods by 18.92%, 18.92%, and 27.03%, respectively. The
implementation of the DL method in memory-constraint IoT devices becomes more practicable for efficient botnet detection when the data size of the network traffic feature set is as small as possible.

B. Results of network traffic classification in binary scenario

To determine the most suitable optimiser for efficient network traffic classification in binary scenario, we analyse the training losses, validation losses, and MCC values obtained when deep BLSTM model was trained with Adam, SGD, RMSprop, Adadelta, Adagrad, Adamax, Nadam, and FTRL optimisation algorithms.

Fig. 3 shows the training losses of deep BLSTM model in binary classification scenario when each of the eight optimisation algorithms was used. In all cases, the training losses reduced as the number of epochs increased from 1 to 20. However, deep BLSTM model had the lowest training losses when Nadam optimiser was employed. The values of training losses produced by Adam optimiser were very close to those of Nadam optimiser. At the end of the 20-epoch training, Nadam optimiser achieved a training loss of $8.19 \times 10^{-5}$ while that of Adam optimiser was $8.39 \times 10^{-5}$. On the other hand, SGD, RMSprop, Adadelta, Adagrad, Adamax and FTRL optimisers produced relatively high training losses of 0.0011, 0.0006, 0.0013, 0.0013, 0.0003, and 0.0014, respectively.

Fig. 4 shows the validation losses of deep BLSTM model in binary classification scenario when each of the eight optimisation algorithms was used. In all cases, the validation losses reduced as the number of epochs increased from 1 to 20. However, deep BLSTM model had the lowest validation losses when Nadam optimiser was employed. The values of validation losses produced by Adam optimiser were very close to those of Nadam optimiser. At the end of the 20-epoch validation, Nadam optimiser achieved a validation loss of $8.75 \times 10^{-5}$ while that of Adam optimiser was 0.0001. On the other hand, SGD, RMSprop, Adadelta, Adagrad, Adamax and

<table>
<thead>
<tr>
<th>Optimiser</th>
<th>Reconstruction loss</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adam</td>
<td>0.0023</td>
</tr>
<tr>
<td>SGD</td>
<td>0.0106</td>
</tr>
<tr>
<td>RMSprop</td>
<td>0.0038</td>
</tr>
<tr>
<td>Adadelta</td>
<td>0.0199</td>
</tr>
<tr>
<td>Adagrad</td>
<td>0.0046</td>
</tr>
<tr>
<td>Adamax</td>
<td>0.0062</td>
</tr>
<tr>
<td>Nadam</td>
<td>0.0038</td>
</tr>
<tr>
<td>FTRL</td>
<td>0.0217</td>
</tr>
</tbody>
</table>
FTRL optimisers produced relatively high validation losses of 0.0011, 0.0008, 0.0013, 0.0013, 0.0003, and 0.0014, respectively.

Table VI presents the MCC values of deep BLSTM model in binary classification scenario when each of the eight optimisation algorithms was used. For SGD, Adadelta, Adagrad, and FTRL optimisers, deep BLSTM model correctly classified all the samples in the attack class as botnet attack traffic but it misclassified all the samples in the normal class as botnet attack traffic. In these cases, the values of TN and FN were zero. Based on Eq. (9), the MCC values were undefined when deep BLSTM model was evaluated with network traffic data in the training, validation, and testing sets. This shows that SGD, Adadelta, Adagrad, and FTRL optimisers could not handle the class imbalance problem in Bot-IoT dataset. The classifier was biased in favour of the attack (majority) class because the number of samples in this class is far greater than those in the normal class (see Table III). On the other hand, Nadam optimiser produced the highest MCC values when deep BLSTM model was evaluated with network traffic samples in the training, validation, and testing sets. Nadam optimiser achieved an overall MCC of 93.17%, and it outperformed Adam, RMSprop, and Adamax optimisers by 1.91%, 33.69% and 12.03%, respectively. Fig. 5 shows the confusion matrix of deep BLSTM model when Nadam optimiser was used for binary classification. The detection rate of normal traffic was 92.90% while that of botnet attack traffic was 100%.

Finally, the performance of LAE-BLSTM model was compared with that of state-of-the-art models in binary classification scenario. Table VII shows that LAE-BLSTM model outperformed PCE-SVM [33], PCE-RNN [33], PCE-LSTM [33], and PCE-VCDL [88] models by 90.03%, 87.19%, 86.14%, and 72.75%, respectively.

C. Results of network traffic classification in multi-class scenario

To determine the most suitable optimiser for network traffic classification in multi-class scenario, we analyse the training losses, validation losses, and MCC values obtained when deep BLSTM model was trained with Adam, SGD, RMSprop, Adadelta, Adagrad, Adamax, Nadam, and FTRL optimisation algorithms.

![Confusion matrix of deep BLSTM model in binary classification scenario](image)

![Training losses of deep BLSTM model in multi-class classification scenario](image)
Fig. 7 shows the validation losses of deep BLSTM model in multi-class classification scenario when each of the eight optimisation algorithms was used. In all cases, the validation losses reduced as the number of epochs increased from 1 to 20. However, deep BLSTM model had the lowest validation losses when Adam optimiser was employed. The values of validation losses produced by Nadam optimiser were very close to those of Adam optimiser. At the end of the 20-epoch validation, Adam optimiser achieved a validation loss of 98.28% while that of Adam optimiser was 0.0326. On the other hand, SGD, RMSprop, Adadelta, Adagrad, Adamax, and FTRL optimisers produced relatively high validation losses of 0.6503, 0.6040, 0.6988, 0.6310, 0.0656, and 0.7920, respectively.

### TABLE VIII

<table>
<thead>
<tr>
<th>Optimiser</th>
<th>DDoS</th>
<th>DoS</th>
<th>Norm</th>
<th>Recon</th>
<th>Theft</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adam</td>
<td>98.33</td>
<td>98.31</td>
<td><strong>93.34</strong></td>
<td>99.96</td>
<td>93.66</td>
<td>96.72</td>
</tr>
<tr>
<td>SGD</td>
<td>57.83</td>
<td>57.18</td>
<td>null</td>
<td>99.62</td>
<td>null</td>
<td>42.93</td>
</tr>
<tr>
<td>RMSprop</td>
<td>92.78</td>
<td>92.71</td>
<td>null</td>
<td>99.47</td>
<td>null</td>
<td>56.99</td>
</tr>
<tr>
<td>Adadelta</td>
<td>27.86</td>
<td>24.79</td>
<td>null</td>
<td>97.05</td>
<td>null</td>
<td>24.74</td>
</tr>
<tr>
<td>Adagrad</td>
<td>36.07</td>
<td>35.71</td>
<td>null</td>
<td>98.73</td>
<td>null</td>
<td>34.10</td>
</tr>
<tr>
<td>Adamax</td>
<td>94.98</td>
<td>94.94</td>
<td>73.05</td>
<td>99.82</td>
<td>null</td>
<td>72.56</td>
</tr>
<tr>
<td>Nadam</td>
<td><strong>98.97</strong></td>
<td><strong>98.96</strong></td>
<td>92.88</td>
<td><strong>99.96</strong></td>
<td><strong>93.68</strong></td>
<td><strong>96.89</strong></td>
</tr>
<tr>
<td>FTRL</td>
<td>19.24</td>
<td>15.00</td>
<td>null</td>
<td>98.10</td>
<td>null</td>
<td>26.47</td>
</tr>
</tbody>
</table>

Furthermore, we evaluate the performance of deep BLSTM model when each of the eight optimisation algorithms was used in multi-class classification scenario. Table VIII presents the MCC values of deep BLSTM model when evaluated with network traffic samples in the training set. Adam optimiser achieved the highest MCC value in detecting normal traffic. Nadam optimiser achieved the highest MCC values in detecting DDoS, DoS, reconnaissance, and theft attacks. This optimiser outperformed Adam, SGD, RMSprop, Adadelta, Adagrad, Adamax, and FTRL optimisers by 0.08%, 53.87%, 39.81%, 72.06%, 62.70%, 24.24%, and 62.70%, respectively. Table IX presents the MCC values of deep BLSTM model when evaluated with network traffic samples in the validation set. Nadam optimiser achieved the highest MCC values in all classes. This optimiser outperformed Adam, SGD, RMSprop, Adadelta, Adagrad, Adamax, and FTRL optimisers by 0.21%, 53.82%, 39.81%, 71.97%, 23.59%, 23.59%, and 70.33%, respectively. Table X presents the MCC values of deep BLSTM model when evaluated with network traffic samples in the testing set. Nadam optimiser achieved the highest MCC values in all classes. This optimiser outperformed Adam, SGD, RMSprop, Adadelta, Adagrad, Adamax, and FTRL optimisers by 0.59%, 55.27%, 41.20%, 73.57%, 64.15%, 25.94%, and 71.91%, respectively.
Fig. 8 shows the confusion matrix of deep BLSTM model when Nadam optimiser was used for multi-class classification. The detection rates of DDoS, DoS, normal, reconnaissance, and theft were 99.5%, 99.4%, 92.90%, 100%, and 100%, respectively. Finally, the performance of LAE-BLSTM model was compared with that of state-of-the-art models in multi-class classification scenario. Table XI shows that LAE-BLSTM model outperformed PCE-FNN [89], PCE-SNN [89], and IFG-CSVM [35] models by 8.29%, 12.29%, and 10.91%, respectively.

VI. CONCLUSION

LAE-BLSTM, an hybrid DL method, was proposed for efficient botnet detection in IoT networks using LAE and deep BLSTM algorithms. The effectiveness of this method was validated by performing extensive experiments with the most relevant publicly available dataset (Bot-IoT) in binary and multi-class classification scenarios. Simulation results showed that LAE model achieved the highest data size reduction rate of 91.89%, outperforming XMP [34], IFG [35], and PCE [33] methods by 18.92%, 18.92%, and 27.03%, respectively. As the data size of big network traffic features becomes smaller, the implementation of DL method in memory-constraint IoT devices seems to be more practicable for efficient botnet detection. In addition, deep BLSTM model, which were trained to analyse the long-term inter-related changes in low-dimensional feature set produced by LAE, demonstrated robustness against model under-fitting and over-fitting as well as good generalization ability. Therefore, LAE-BLSTM has proven to be efficient for botnet attack detection in IoT networks.

REFERENCES


TABLE XI

<table>
<thead>
<tr>
<th>Model</th>
<th>DDoS</th>
<th>DoS</th>
<th>Norm</th>
<th>Recon</th>
<th>Theft</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCE-FNN [89]</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>89.00</td>
</tr>
<tr>
<td>PCE-SNN [89]</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>85.00</td>
</tr>
<tr>
<td>IFG-CSVM [35]</td>
<td>99.38</td>
<td>99.53</td>
<td>88.91</td>
<td>44.76</td>
<td>99.32</td>
<td>86.38</td>
</tr>
<tr>
<td>LAE-BLSTM</td>
<td>98.96</td>
<td>98.95</td>
<td>91.89</td>
<td>99.95</td>
<td>96.68</td>
<td>97.29</td>
</tr>
</tbody>
</table>


Segun I. Popoola received the B.Tech. degree (Hons.) in electronic and electrical engineering from the Ladoke Akintola University of Technology, Ogbomosho, Nigeria, and the M.Eng degree in information and communication engineering from the Department of Electrical and Information Engineering, Covenant University, Ota, Nigeria. He is currently pursuing the Ph.D. degree with the School of Engineering, Faculty of Science and Engineering, Manchester Metropolitan University, Manchester, U.K. He was a Lecturer with the Department of Electrical and Information Engineering, Covenant University. His research interests include wireless communications, machine/deep learning, cybersecurity, and IoT. He is a Registered Engineer with the Council for the Regulation of Engineering in Nigeria (COREN).

Bamidele Adebisi received the B.S. degree in electrical engineering from Ahmadu Bello University, Zaria, Nigeria, in 1999, the M.S. degree in advanced mobile communication engineering, and the Ph.D. degree in communication systems from Lancaster University, Lancaster, U.K., in 2003 and 2009, respectively. He was a Senior Research Associate with the School of Computing and Communication, Lancaster University, from 2005 to 2012. He joined Manchester Metropolitan University, Manchester, U.K., in 2012, where he is currently a Professor in electronic and electronic engineering. He has been involved in several commercial and government projects focusing on various aspects of wireline and wireless communications. He is particularly interested in the research and development of communication technologies for electrical energy monitoring/management, transport, water, critical infrastructures protection, home automation, the IoTs, and cyber physical systems. He has several publications and a patent in the research area of data communications over power line networks and smart grid. He is a member of the IET.

Mohammad Hammoudeh is currently the Head of the MMU IoT Laboratory and a Senior Lecturer in computer networks and security with the School of Computing, Math and Digital Technology, Manchester Metropolitan University. He has been a researcher and publisher in the field of big sensory data mining and visualization. He is a highly proficient, experienced, and professionally certified cybersecurity professional, specializing in threat analysis, and information and network security management. His research interests include highly decentralized algorithms, communication, and cross-layered solutions to Internet of Things, and wireless sensor networks.

Guan Gui received the Ph.D. degree from the University of Electronic Science and Technology of China, Chengdu, China, in 2012. From 2009 to 2014, he joined the Tohoku University as a Research Assistant as well as a Postdoctoral Research Fellow, respectively. From 2014 to 2015, he was an Assistant Professor with the Akita Prefectural University, Akita, Japan. Since 2015, he has been a Professor with Nanjing University of Posts and Telecommunications, Nanjing, China. His recent research interests include artificial intelligence, deep learning, non-orthogonal multiple access, wireless power transfer, and physical layer security. Dr. Gui has authored more than 200 IEEE Journal/Conference papers and won several best paper awards, and he is serving or served on the editorial boards of several journals. He was the recipient of the Top Editor Award of IEEE Transactions on Vehicular Technology in 2019.

Haris Gacanin received his Dipl.-Ing. degree in electrical engineering from the University of Sarajevo in 2000. In 2005 and 2008, respectively, he received M.Sc. and Ph.D. degrees from Tohoku University, Japan. He was with Tohoku University from 2008 until 2010, first as a Japan Society for Promotion of Science postdoctoral fellow and later as an assistant professor. Since 2010, he has been with Alcatel-Lucent (now Nokia), where he leads a research department at Nokia Bell Labs. His professional interest is related to the application of artificial intelligence to enable autonomous networking and design of mobile and wireless systems. He has 200+ scientific publications (journals, conferences, and patent applications) and invited/tutorial talks. He is a Senior Member of the Institute of Electronics, Information, and Communication Engineering (IEICE).