DeepCoin: A Novel Deep learning and Blockchain-based Energy Exchange Framework for Smart Grids

Mohamed Amine Ferrag, Leandros Maglaras, *Senior Member, IEEE*

***Abstract*—In this paper, we propose a novel deep learning and blockchain-based energy framework for Smart Grids, entitled DeepCoin. The DeepCoin framework uses two schemes, a blockchain-based scheme and a deep learning-based scheme. The blockchain-based scheme consists of five phases; setup phase, agreement phase, creating a block phase and consensus-making phase, and view change phase. It incorporates a novel reliable peer-to-peer energy system that is based on the practical Byzantine fault tolerance algorithm and it achieves high throughput. In order to prevent smart grid attacks, the proposed framework makes the generation of blocks using short signatures and hash functions. The proposed deep learning-based scheme is an intrusion detection system (IDS), which employs recurrent neural networks (RNNs) for detecting network attacks and fraudulent transactions in the blockchain-based energy network. We study the performance of the proposed IDS on three different sources the CICIDS2017 dataset, a Power System dataset, and a Bot-IoT dataset.**

***Index Terms*—Blockchain, Smart Grid, Machine Learning, Security, IDS.**

I. Introduction

The world's electricity consumption is increased every year, reflecting the growth in the number of electric devices. According to a report published in 2018 [1] about energy consumption in the UK provides that electricity consumption in the UK increased by 33% to 15 ktoe. The renewable energy sources provide nearly 20% of U.S. electricity (e.g., hydropower, biomass, biofuels, wind geothermal, solar) in 2017 [2]. In order to modernize public electricity infrastructures, many electric power companies are interested in deploying smart grids using communication networks and renewable energies.

The smart grid consists of a set of computers, controllers, automation, and standard communication protocols, which are connected on the Internet, all of which are used in order to manage the generation and distribution of electricity to consumers through these digital technologies [3]. A community solar panels network in the smart grid could consolidate to the overall energy infrastructure of the smart city by creating additional energy storage. The main issue in the development of a smart grid is not located at the physical support but mainly in ensuring both security and privacy, which has become a major concern for the cyber security research community [4]. An adversary can launch internal and external attacks (e.g., false data injection and denial of service attacks) in order to disrupt the operation of the smart grid [5]. Examples include modification operations on the electricity data via eavesdropping attack and distributed denial of service attacks on the network communication protocols (i.e., TCP/IP, HTTP, UDP)[6].

Blockchain is an emerging technology that can leverage enterprise data using secure transactions among parties. Blockchain technology-enabled business and technological systems is a new trend rising fast in the area of engineering management [7]. Managers are expected to use Blockchain technology in the near future in order to solve problems and create new opportunities across industries (e.g., supply chain management, logistics, and product origin tracking) [8]. Recent studies have utilized blockchain technology to establish secure data sharing for the management of Smart Grid [9], [10], [11], [12]. Although blockchain is one of the promising technologies for securing the management of technology and innovation, it suffers from some vulnerabilities related to data privacy, along with a number of personal identity risks as discussed in [13]. In order to deal with these issues, novel technologies and approaches such as intrusion detection systems and machine learning techniques should be adopted.

In this paper, we propose a novel deep learning and blockchain-based energy framework, named DeepCoin, for protecting the smart grid from cyber attacks. Specifically, the main contributions of this paper are as follows.:

- We propose a new blockchain-based scheme for facilitating the exchange of excess energy among neighboring nodes. To achieve privacy-preservation, the proposed scheme employs bi-linear pairing, short signatures, and hash functions.
- We introduce how the practical Byzantine fault tolerance (PBFT) algorithm can achieve consensus inside blockchain-based energy network.
- We propose a novel Deep learning-based scheme using a recurrent neural network (RNN) for detecting network attacks and fraudulent transactions. For training an RNN, we use the truncated backpropagation through time (BPTT) algorithm. To the best of our knowledge, this is the first study that combines blockchain technology with deep learning using a truncated BPTT algorithm into one architecturally secure framework for the smart grid.

(Corresponding author: Mohamed Amine Ferrag)

M. A. Ferrag is with Department of Computer Science, Guelma University, B.P. 401, 24000, Algeria e-mail: ferrag.mohamedamine@univ-guelma.dz, phone: +213661-873-051

L. Maglaras is with School of Computer Science and Informatics, De Montfort University, Leicester, UK, and also with General Secretariat of Digital Policy, Athens, Greece, e-mail: leandrosmag@gmail.com

- We provide various experimental results using TensorFlow on three datasets, the CICIDS2017 dataset, a Power System dataset, and a Bot-IoT dataset.

The remainder of this paper is organized as follows. We review related work in section I and provide the preliminaries in Section II. Section III presents our DeepCoin framework, followed by performance evaluation in Section IV. Finally, we conclude this paper in Section V.

II. Related work

In this section, we mainly describe the relevant work in three different areas of a smart grid, blockchain, IDS, and combinations blockchain technology with deep learning.

1) Blockchain for Smart Grid: In the work by Pop et al. [9], an architecture based on the blockchain is proposed for distributed management in low/medium voltage smart grids. The blockchain distributed consensus is used for demand response verification. The Ethereum coin is used as the coin to pay for energy while Solidity is used for implementing smart contracts. In addition, energy traces of UK buildings were used to validate and test the proposed model.

For privacy preserving of energy trading, Aitzhan and Svetinovic [10] combined blockchain with multi-signatures and anonymous messaging streams for decentralized energy trading network. The study implemented a proof-of-concept, entitled PriWatt, which was inspired from the Bitcoin. The PriWatt system is efficient against double-spending attacks using hashes of blocks, which are computationally difficult. Li et al. [11] introduced a credit-based payment scheme for IIoT. The study proposed three ideas, including, 1) a unified energy blockchain, 2) a credit-based payment, and 3) an optimal pricing strategy. The unified energy blockchain consists of three entities, energy nodes, energy aggregators, and smart meters. Li et al.'s scheme is efficient against double-spending attacks.

Guan et al. [12] proposed a privacy-preserving and data aggregation scheme for power grid communications. The study divided users into different groups and each group has a private blockchain. To hide user's identity, the study applies the idea of multiple pseudonyms. Compared with the conventional authentication framework, the study shows some significant advantage in term of computational cost. To provide transparency and provenance, Gao et al. [14] presented a secured sovereign blockchain framework, named GridMonitoring, for the smart grid. The GridMonitoring framework uses smart contracts between consumers and utility companies for protecting consumer data recorded and transferred. In addition, GridMonitoring adopts three cryptographic primitives, including, consumer private key, consumer public key, and authenticator contract key. Fan and Zhang [15] proposed a data aggregation and regulation mechanism for the smart grid. Based on a consortium blockchain and the hybrid signcryption scheme, the proposed mechanism can be applied to multidimensional data acquisition and multiple receivers. The performance evaluation shows that the proposed mechanism significant advantages in terms of computation and communication overhead for multidimensional data compared to multidimensional data aggregation schemes in the multilevel network.

Gai et al. [16] introduced a privacy-preserving energy trading framework using consortium blockchain. The Gai et al.'s framework implemented B-Box on the smart contract of consortium blockchain. For detecting fixed bounds, the study uses a dynamic-style bound, which can detect data mining-based attacks. To achieving a parallel trading growth, the study introduced a parameter, called an Approximate Maximum Estimate (AME). The AME parameter can estimate value for an individual seller's trading volume. The performance evaluation in Hyperledger Fabric 1.0 shows that the proposed model can find many sellers whose energy sale were noticeably different from other sellers. In order to address the privacy protections and energy security, Gai et al. [17] proposed a model permissioned blockchain edge model, named PBEM, which it combines blockchain and edge computing technologies. Specifically, the PBEM model uses group two techniques, including, signatures and covert channel authorization, to guarantee users' validity. The experiment results show that the execution time range of the proposed PBEM model was from 1ms to 14ms, for task allocations in edge computing.

2) IDS for Smart Grid: To protect advanced metering infrastructure (AMI) from malicious attacks in smart grids, Faisal et al. [18] proposed an IDS based on the evolving classification algorithms. The study uses an open source data stream mining framework, named MOA. The approach was tested on the KDD CUP 1999 dataset and was proven to perform well in terms of memory and time consumption. The KDD dataset that was used is outdated and of very limited practical value for a modern IDS since it doesn't consider threats of a smart grid. Jokar and Leung [19] presented an intrusion detection system, entitled HANIDPS, for smart grids using ZigBee. The HANIDPS system considers three categories of three models against advanced metering infrastructure, including, 1) Illegitimate remote turn-on/off commands, when an adversary passively eavesdrops the network traffic, 2) Stealing customer information, when an adversary impersonates ID of the energy management system, and 3) Denial of service against network nodes, when an adversary tries to gain control of a specific device. The HANIDPS system combines a model-based intrusion detection method and a machine learning-based prevention technique. The HANIDPS system was tested on an IEEE 802.15.4 network (it contained six nodes, including 4 air monitors, an adversary, and a genuine node) and was proven to be efficient against IEEE 802.15.4 attacks, i.e., radio Jamming, replay-attacks, back-off manipulation, DoS against GTS requests, etc.

Zhou et al. [20] proposed a deep neural network model in order to classify cyber-attacks in the smart grid. To detect smart grid attacks, the study follows three steps, a data acquisition step, a data pre-processing step, and a deep neural network classification step. The data acquisition step collected 4559799 samples, 1064720 being positive samples and 3495079 negative samples (R2L attack, U2R attack, DOS attack, and PROBING attack). The study showed good accuracy with 96,31% compared to traditional machine learning algorithms, including, K-Nearest Neighborhood, Linear Regression, and Random Forest.

3) Combining blockchain with machine learning: Liu et al. [21] proposed a data collection sharing framework for IIoT applications. They proposed the combination of Ethereum blockchain and deep reinforcement learning (DRL). For storing and sharing data, they used Ethereum nodes, which were classified into two categories: 1) Mining nodes and 2) Non-mining nodes. The DRL algorithm used three basic components, states, actions and rewards. A state is a description of the environment for IIoT applications. An action is the moving direction and distance of mobile terminals. A reward is based on the collection amount achieved, geographical fairness and energy consumption. According to the study, the DRL algorithm can increase the geographical fairness ratio by 34.5% compared to a random solution.

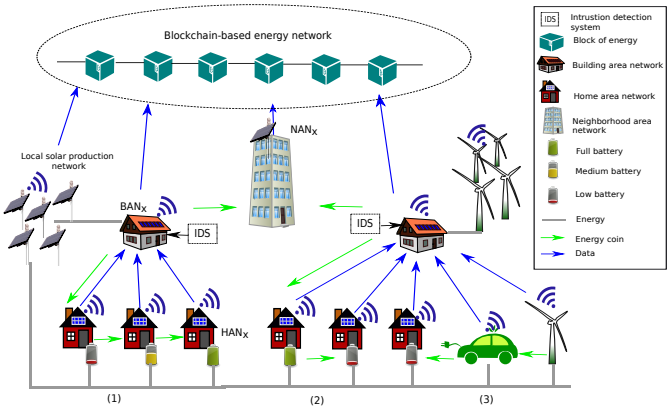

Fig. 1: Conceptual architecture of a smart grid. 1) HANs sell the surplus production to the electricity grid. 2) HANs have solar panels installed on their roofs which are connected to the local solar production network. 3) Electric vehicles act as energy storage devices.

III. Preliminaries

A. System model

As presented in Fig. 1, the smart grid architecture consists of three types of network architecture, including, home area network (HAN), building area network (BAN), and neighborhood area network (NAN) [3]. In order to facilitate communication among devices in each home, the HAN network uses two types of digital networks, namely, wireless Local Area Network (WLAN) and Wide Area Network (WAN), which operate at a frequency of 2.4 GHz under 802.11 wireless standards. The BAN network and the NAN network connect various departmental networks within a single building. In each network type, there are smart meters that measure a building's energy production and the quantities consumed at different times. Both BAN and NAN networks contain mining GPUs with a lot of graphics memory in order to manage high hash rates as well as low power draw (e.g., Nvidia GeForce GTX 1070, AMD Radeon RX580, Nvidia GeForce GTX 1060...etc). All entities of the smart grid use solar panels or wind generators, which consolidates the energy infrastructure by creating additional energy storage. This smart grid architecture provides a peer-to-peer (P2P) energy trading mechanism, which each consumer can use in order to sell or buy energy from neighboring nodes.

B. Short signatures

The short signatures scheme [22] is based on the following algorithms: *Setup* (1^λ): This algorithm generates a bilinear environment $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e, \psi)$ based on the q-Strong Diffie-Hellman (q-SDH) problem. $\mathbb{G}_1$ and $\mathbb{G}_2$ are two groups of prime order p with g_1 is a generator of $\mathbb{G}_1$ and g_2 is a generator of $\mathbb{G}_2$, and ψ is an isomorphism where $g_1 = \psi(g_2)$. The system settings is defined as follows:

$$Sign\_param = (\lambda,\ p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e, \psi) \tag{1}$$

Key generation: Using the *Sign_param*, This algorithm chooses randomly x and y in Z_p^* and computes $u = g_2^x \in \mathbb{G}_2$ and $v = g_2^y \in \mathbb{G}_2$. The secret key is $sk = (x, y)$ and the corresponding public key is $pk = (Sign\_param, u, v, z)$ which $z = e(g_1, g_2) \in \mathbb{G}_T$.

Sign phase: Using the secret key sk, the corresponding public key pk, and the bloc B. This algorithm chooses randomly r in Z_p^*, which $x + B + yr \neq 0 \bmod p$ and computes the signature as follows:

$$S = g_1^{\frac{1}{x+B+yt}} \tag{2}$$

Sending phase: Given the signature $Sig = (S, r)$ of the bloc B, the algorithm sends Sig.

Reception phase: The receiver checks the following formula

$$e\left(S, u, g_2^B \cdot v^r\right) =? \ z \tag{3}$$

C. Complexity assumptions

The q-Strong Diffie-Hellman Problem (q-SDH) is defined by Boneh and Boyen [23] as follows: given a $(q+2)$-tuple $(g_1, g_2, g_2^x, g_2^{(x^2)}, \ldots, g_2^{(x^q)})$ as input where as above $g_1 = \psi(g_2)$, output a pair $(c, g_1^{1/(x+c)})$ where $c \in Z_p^*$, $g_1 \in \mathbb{G}_1$, and $g_2 \in \mathbb{G}_2$. An algorithm A has advantage ϵ in solving q-SDH in $(\mathbb{G}_1, \mathbb{G}_2)$ if the probability

$$\Pr\left[A\left(g_1, g_2, g_2^x, g_2^{(x^2)}, \ldots, g_2^{(x^q)}\right) = (c, g_1^{1/(x+c)})\right] \geq \epsilon \tag{4}$$

IV. DeepCoin Framework

This section overviews the architecture of the proposed DeepCoin framework. It is composed of a blockchain-based scheme and a Deep learning-based scheme. DeepCoin framework consists of four network entities: *Energy buyer*, *Energy vendor*, *Blockchain*, and *IDS*, which are described below.

- *Energy buyer*: We assume three types of energy buyers, including, EB_{HAN}, EB_{BAN}, EB_{NAN}, which are energy network entities located in the HAN network, the BAN network, and the NAN network, respectively. These entities plan to trade with energy vendors. An energy buyer entity proves that he has enough energy money, named CoinEnergy, that satisfy the energy vendor's minimum asset requirement.
- *Energy vendor*: We assume four types of energy vendors, including, EV_{HAN}, EV_{BAN}, EV_{NAN}, EV_{COM} which are energy network entities located in the HAN network,

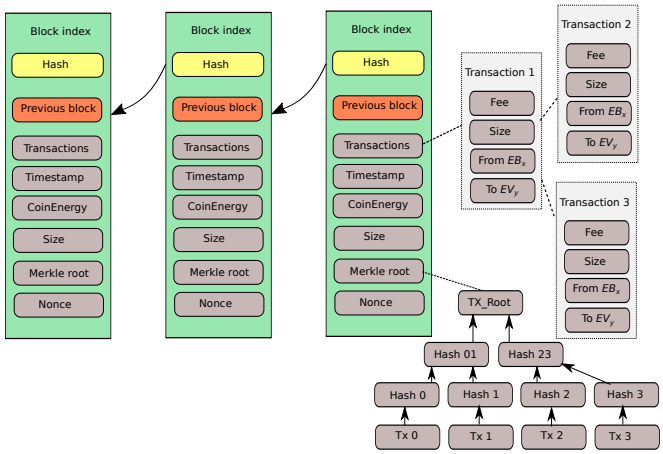

Fig. 2: The data structures of Blocks used by the DeepCoin framework.

BAN network, NAN network, and energy company, respectively. These entities prove that they have enough energy to sell to energy buyers entities (i.e., EB_{HAN}, EB_{BAN}, EB_{NAN}). Note that the energy company is an entity that produces energy from renewable sources (e.g., solar energy, wind energy, and biomass).

- *Blockchain*: An entity which is a distributed digital ledger containing all energy transactions in the smart grid. This distributed ledger is replicated and stored in different nodes of the smart grid, including, HAN_x, BAN_x, NAN_x, and COM_x. The data structures of Blocks used by the DeepCoin framework are presented in Fig. 2. Specifically, the structure of each block includes nine fields. (1) Block index field is a number that is unique for each block. (2) Hash field is the hash value of the block content. (3) Previous block field is the hash value of the previous block. (4) Transactions field containing the details of the purchases made between energy buyers and energy vendors. (5) Timestamp field is date and time of the creation of the block. (6) CoinEnergy field is the money of energy. (7) Size field is the size of the block. (8) Merkle root field is the descendant of all the hashed pairs in the tree. (9) Nonce field is the number that blockchain miners are solving for. To add a block to the blockchain, we assume that the BAN_x, NAN_x, and COM_x nodes mining new blocks, which is hard work, have to be properly rewarded. Note that the data structures of Blocks used by DeepCoin framework are inspired from Bitcoin's structure.
- *Intrusion Detection System (IDS)*: An entity installed at the BAN and NAN nodes, which is responsible for verifying that the frames running on the energy transaction network comply with a set of rules. Specifically, this entity detects the network attacks (e.g., brute force attack, botnet, DoS attack, DDoS attack, web attack, infiltration attack, ...etc) as well as fraudulent transactions to prevent future illegal actions.

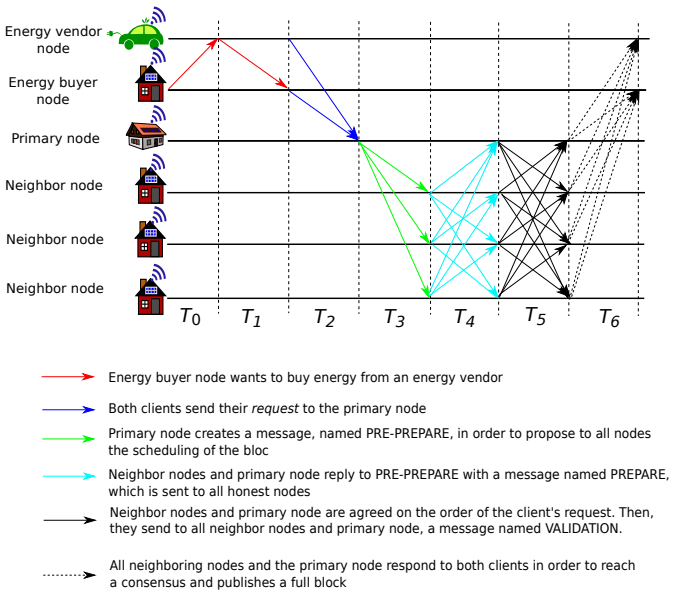

Fig. 3: The consensus process for blockchain-based energy exchange framework.

A. Blockchain-based scheme

In order to secure this energy environment, we propose a blockchain based-energy scheme, where the nodes of the smart grid network can exploit their excess energy and sell it to other neighboring nodes (e.g., HAN) or neighbors on the other side of the street (e.g., BAN, NAN). This scheme aims to strengthen the local energy network and reduce the disruption caused by dangerous situations such as natural disasters. In addition, this scheme allows creating a secure and stable energy environment at the grid edge.

Generally, there are two types of network consensus, namely, 1) public network consensus and 2) private network consensus [13]. To achieve consensus in a blockchain network, there are five approaches, including, the practical Byzantine fault tolerance (PBFT) algorithm, proof of work (proposed by Nakamoto in the bitcoin network), proof of stake (used by Ethereum), Proof of Authority (PoA), and RAFT. The proposed DeepCoin framework uses the PBFT algorithm [24] for achieving consensus in the smart grid. The overall message complexity of PBFT's normal operation is $(O(N^2))$ [25].

Figure 3 shows the consensus process for blockchain-based energy exchange framework. We assume that the smart grid is composed of a set of four types of nodes $\mathcal{N} = \{HAN_x,\ BAN_x,\ NAN_x\ COM_x\}$, where $x = \{1 \dots n\}$, connected by a reliable peer-to-peer energy network. Therefore, we propose that the total ledger is maintained by BAN_x, COM_x, and NAN_x nodes while HAN_x nodes do not participate in the consensus making. Our concrete DeepCoin framework is outlined below:

- ***Setup phase***: Based on the q-Strong Diffie-Hellman (q-SDH) problem, let $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ be bilinear groups where $|\mathbb{G}_1| = |\mathbb{G}_2| = p$ for some prime p. Two different hash functions are given: $H_1 : \Theta \times \{0,1\}^* \rightarrow \{0,1\}$ and : $H_2 : \{0,1\} \times \{0,1\}^* \rightarrow \Omega$, which we assume that the proposed scheme signs blocks in some finite set Ω and

Algorithm 1 Consensus-making phase

Input: P_x, EV, EB, σ_i, PK_i, $Bloc_i$
Initialization: Set the time intervals of the block generation as t
Output: Success or Failure

P_x checks the validity of the block using the following formula: $e\left(\sigma_i, u \cdot g_2^{Bloc_i},\ v^{r_i}\right) = ?z$. If the equality is true then Validity_Bloc $= valid$ otherwise the result is Validity_Bloc $= invalid$;
P_x checks if the EV node has enough energy to sell. If the equality is true then Validity_Energy $= valid$ otherwise the result is Validity_Energy $= invalid$;
P_x checks if the EB node has enough CoinEnergy to buy. If the equality is true then Validity_Coin $= valid$ otherwise the result is Validity_Coin $= invalid$;
if Validity_Bloc $= invalid$ **then**
    P_x sends a penalty to the EB node;
    return Failure;
else
    if Validity_Energy $= invalid$ **then**
        P_x sends a penalty to the EV node;
        return Failure;
    else
        if Validity_Coin $= invalid$ **then**
            P_x sends a penalty to the EB node;
            return Failure;
        else
            Run the algorithm 2;
            return Success;
        end if
    end if
end if

Algorithm 2 Scheduling a bloc

Input: t, $\mathcal{N} = \{HAN_x,\ BAN_x,\ NAN_x,\ COM_x\}$, where $x = \{1 \dots n\}$
Initialization: Set the time of the view as $VIEW$
Output: Reached or Not reached

Set $VIEW = 0$;
P_x creates a message <PRE-PREPARE,$Bloc_i, \sigma_i, Sec$>, with a unique sequence number Sec;
After the time t, the P_x node sends <PRE-PREPARE,$Bloc_i, \sigma_i, Sec$> to $\mathcal{N}$;
After receiving the proposal, each node $\in \mathcal{N}$ replies with a message <PREPARE, $Bloc_i$, Sec> to all $\mathcal{N}$ ▷ the P_x and EB node do not participate in sending replies phase;
Each node $\in \mathcal{N}$, upon receiving at least $2f$ <PRE-PREPARE,$Bloc_i, \sigma_i, Sec$> and <PREPARE, $Bloc_i$, Sec>, sends a message <COMMIT, $Bloc_i$, Sec> to all $\mathcal{N}$ ▷ the EB node does not participate in this step;
Each node $\in \mathcal{N}$, upon receiving at least $2f+1$ <COMMIT, $Bloc_i$, Sec>, reaches a consensus and publishes a full block and **return** Reached;
if the EB node does not receive a response after $2^{VIEW} \cdot t$ **then**
    run the algorithm 3 and **return** Not reached;
end if

Algorithm 3 View change phase

Input: t,$VIEW$, $\mathcal{N} = \{HAN_x,\ BAN_x,\ NAN_x,\ COM_x\}$, where $x = \{1 \dots n\}$
Initialization: Set the counter of the view as $Counter = 1$

$VIEW$=$VIEW + Counter$;
The EB node creates a message <CHANGE-VIEW,$Bloc_i, \sigma_i, Sec$,$VIEW$>, with a unique sequence number Sec;
The EB node sends <CHANGE-VIEW,$Bloc_i, \sigma_i, Sec$,$VIEW$> to all $\mathcal{N}$;
Choose a new P_x node;
Run Algorithm 2;
if the EB node does not receive a response after $2^{VIEW} \cdot t$ **then**
    $Counter = Counter + 1$;
    Back to Step 1;
end if

the secret keys are in some set Θ. The public key is $pk = (g_1,\ g_2,\ u,\ v,\ z)$. The corresponding secret key is $sk(x,\ y)$, and $z = e(g_1, g_2) \in \mathbb{G}_T$. The address set is $S_i(pk_i, sk_i)$ [26].

- ***Agreement phase***: When an energy buyer node EB wants to buy energy from an energy vendor EV, they negotiate the price, quantity, and period of validity. All this information is contained in a file and embedded in the blockchain block, as presented in Fig. 2.
- ***Creating a block***: Given a secret key $x_i, y_i \in Z_p^*$ and a bloc $Bloc_i \in \{0,1\}^*$, an energy buyer node EB_i picks a random number $r_i \in Z_p^*$ and compute $\sigma_i = H_1\left(g_1^{\frac{1}{(x_i + Bloc_i + y_i + r_i)}}\right) \in \{0,1\}$. Then, the EB_i compute $b_i = H_2(\sigma_i, Bloc_i) \in \Omega$ and set the time intervals of block generation as T. The signature of bloc $Bloc_i$ is $SigBloc(\sigma_i, b_i, r_i)$. At the end, the EB_i broadcasts transaction data attached with $SigBloc$ to the entire network.
- ***Consensus-making phase*** : This algorithm is an active replication protocol and uses a special replica, named the primary node (leader) P, that proposes a query execution order. Generally, this algorithm requires $N = 3f + 1$ replicas to tolerate f simultaneous Byzantine faults. In this phase, we propose that only the BAN_x, NAN_x, and COM_x can be one of the primary nodes P_x in each transaction. The choice of P_x is based on the distance of the transmission range. Once the primary node P_x receives a transaction from the EB node, P_x run the algorithm 1. After verifying the block, the primary node P_x creates a message, named PRE-PREPARE, in order to propose to all nodes the scheduling of the bloc. The honest nodes reply to PRE-PREPARE with a message, named PREPARE, which is sent to all honest nodes. When each node $\in \mathcal{N}$, upon receiving at least $2f$ PRE-PREPARE and PREPARE, he sends a message, named COMMIT, to all honest nodes. Then, when each node $\in$

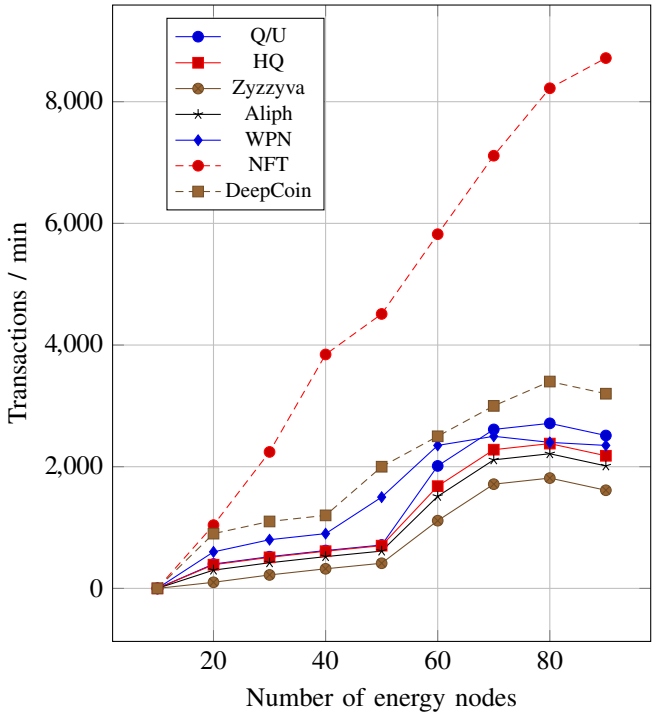

Fig. 4: Throughput of DeepCoin for different number of energy nodes. Energy block size: 1 MB. Timeout= 100ms. Number of BAN nodes=10. Number of NAN nodes=5.

$\mathcal{N}$, upon receiving at least $2f+1$ COMMIT, he reaches a consensus and publishes a full block.

- ***View change phase*** : If the EB node does not receive a response after a predefined period of time, it re-transmits its request to all honest nodes by running the algorithm 3. During the view change phase, from $VIEW$ to $VIEW+1$, the primary node of $VIEW$ leaves its role of leader, and a new primary node is elected as leader. The honest nodes in this phase store the messages that they receive and send to a log file. This allows to retransmit them if needed (e.g., if a message is lost on the blockchain network). Since the honest nodes can not store an infinite amount of message, we propose that they truncate these messages periodically.

In Figure 4, we compare the throughput of DeepCoin framework with the state of the art of Byzantine fault tolerance protocols in term of the different number of energy nodes. Specifically, we have implemented six approaches, including, Q/U, HQ, Zyzzyva, Aliph, WPN, and NFT. The Q/U protocol was presented in 2005 by Abd-El-Malek et al. in [27] where it involves a subset of replicas, called quorum, to order to perform queries. The HQ protocol is proposed by Cowling et al. in [28] which is based on a BFT protocol that requires $3f+1$ replicas. The Zyzzyva protocol [29] presented in 2009 by Kotla et al., which is a BFT protocol and based on speculation (i.e., in the case without fault, the replicas do not need to agree on the order of execution of the queries). The Aliph protocol [30] combines three BFT protocols, namely, Quorum, Chain, Backup. The WPN protocol uses the same consensus-making phase of DeepCoin but the primary node is selected randomly. The NFT protocol does not use a fault tolerance strategy for transaction execution in the consensus protocol. From Figure 4, we can observe that the DeepCoin framework can validate more transactions per minute than five approaches, including, Q/U, HQ, Zyzzyva, Aliph, and WPN. The reason is that the DeepCoin framework chooses BAN and NAN nodes with low failure probabilities as primary nodes and the HAN nodes cannot be selected as primary nodes.

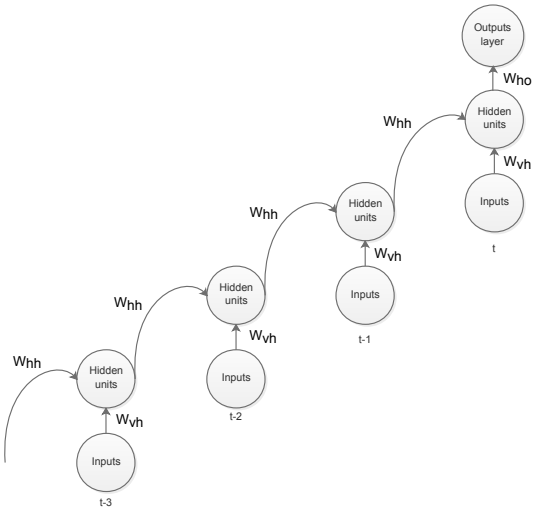

Fig. 5: RNN using the backpropagation through time algorithm.

B. Deep learning-based scheme

In order to detect network attacks and fraudulent transactions, we propose an intrusion detection system (IDS) based on a deep learning approach. This approach is inspired by recurrent neural networks. Note that the proposed IDS is executed only by the BAN and NAN nodes in the blockchain-based energy network.

There are many recurrent neural network (RNN) algorithms proposed in the literature, such as, Real-Time Recurrent Learning (RTRL), Long Short-Term Memory (LSTM), Echo-State Networks (ESN), and Truncated Backpropagation Through Time (TBTT) [31]. The RNN can extended deeper based on three points defined by Pascanu et al. in [32], including, input-to-hidden function, hidden-to-hidden transition, and hidden-to-output function.

The standard RNN is a neural network that simulates a discrete-time dynamical system, which is formalized as follows: Given a sequence of input vectors $x(t)$, a sequence of output vectors is $z(t)$, which $t \in \left[1, t_f\right]$ and internal state vectors are as follow: $h_0(t) = x(t), \forall\ t \in \left[1, t_f\right]$ and $h_j(0) = x(t), \forall\ j \in [1, N]$. By applying the affine transformation a_j to the output vector of the previous layer and adding the linear transformation $V_j \in \mathrm{R}^{n_j \times n_j}$, the parameters of an RNN can be estimated by the following cost functions:

$$A_j(t) = W_j \times h_{j-1}(t) + V_j \times h_j(t-1) + b_j \tag{5}$$

$$h_j(t) = \delta_j(A_j(t)) \tag{6}$$

Algorithm 4 RNN using truncated BPTT algorithm

Input: $x(t)$
Initialization: $h_0(t) = x(t), \forall\ t \in [1, t_f]$
Output: $\frac{\partial C}{\partial \theta}$
for $j = 1$ `to` N **do**
  for $t = 1$ `to` t_f **do**
    $A_j(t) = W_{vh} x(t) + W_{hh} h_{t-1} + b_j$;
    $h_j(t) = e(A_j(t))$ ▷ $e(\cdot)$ is the hidden nonlinearities;
    $o_j(t) = W_{ho} h_j(t) + b_o$;
    $z(t) = \delta(o_j(t))$ ▷ $\delta(\cdot)$ is the output nonlinearities;
  end for
  Compute the loss of the RNN: $L(z, y) = \sum_{t=1}^{t_f} L(z_t; y_t)$ ▷ L is loss function when predict y as z
end for
for N `to` $j = 1$ **do**
  for t_f `to` $t = 1$ **do**
    $\frac{\partial C}{\partial o_j}(t) = \delta'(o_j(t)) . \frac{\partial C}{\partial L(z(t);y(t))} / \frac{\partial C}{\partial z(t)}$;
    $\frac{\partial C}{\partial b_o} = \frac{\partial C}{\partial b_o} + o_j(t)$;
    $\frac{\partial C}{\partial W_{ho}} = \frac{\partial C}{\partial W_{ho}} + o_j(t)\, h_j^{\mathrm{T}}(t)$;
    $\frac{\partial C}{\partial h_j}(t) = \frac{\partial C}{\partial h_j}(t) + W_{ho}^{\mathrm{T}} \frac{\partial C}{\partial o_j}(t)$;
    $\frac{\partial C}{\partial z}(t) = e'(z(t)) \cdot \frac{\partial C}{\partial h_j}(t)$;
    $\frac{\partial C}{\partial W_{vh}} = \frac{\partial C}{\partial W_{vh}} + \frac{\partial C}{\partial z}(t) x(t)^{\mathrm{T}}$;
    $\frac{\partial C}{\partial b_h} = \frac{\partial C}{\partial b_h} + \frac{\partial C}{\partial z}(t)$;
    $\frac{\partial C}{\partial W_{hh}} = \frac{\partial C}{\partial W_{hh}} + \frac{\partial C}{\partial z}(t) h_j^{\mathrm{T}}(t-1)$;
    $\frac{\partial C}{\partial h_j}(t-1) = W_{hh}^{\mathrm{T}} \frac{\partial C}{\partial z}(t)$
  end for
end for
$\frac{\partial C}{\partial \theta} = \left(\frac{\partial C}{\partial W_{hv}}, \frac{\partial C}{\partial W_{hh}}, \frac{\partial C}{\partial W_{ho}}, \frac{\partial C}{\partial b_h}, \frac{\partial C}{\partial b_o}, \frac{\partial C}{\partial h_o} \right)$;
return $\frac{\partial C}{\partial \theta}$;

Where $h_j(t)$, $A_j(t) \in \mathrm{R}^{n_j}$, and $z(t)$=$h_j(t)$.

The details of the proposed IDS methodology are illustrated in Fig. 6. Specifically, the proposed method consists of four stages: 1) datasets stage, 2) pre-processing stage, 3) training stage and 4) testing stage. The detailed design of each stage is further discussed in the following sub-sections.

1) Datasets stage: We use three different sources for the experiments, the CICIDS2017 dataset [33], a Power System dataset [34] and a Bot-IoT dataset [35].

The CICIDS2017 dataset is developed by Sharafaldin et al. [33] at Canadian Institute for Cybersecurity (CIC). Compared to the previous datasets (e.g., DARPA98, KDD99, and NSL-KDD), the CICIDS2017 dataset contains benign and seven common attack network flows, including, brute force attack, heartbleed attack, botnet, DoS attack, DDoS attack, web attack, infiltration attack.

The Power System dataset [34] is developed by Mississippi State University and Oak Ridge National Laboratory. Compared to the CICIDS2017 dataset, the Power System dataset contains 37 scenarios, which are divided into 8 natural events, 1 no events, and 28 attack events. Three categories of attacks in power system are considered, including, 1) data injection, 2) remote tripping command injection, and 3) relay setting change. This dataset has been used for power system cyber-attack classification (e.g., common path mining [36], sequential pattern mining approach [37])

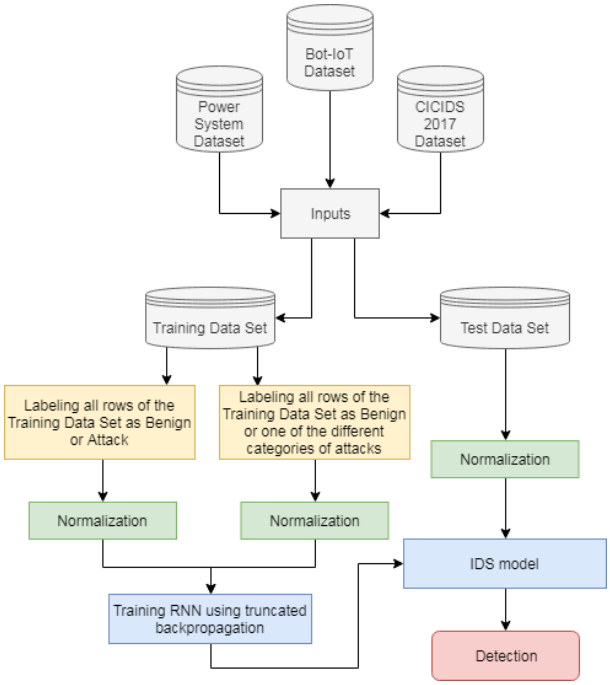

Fig. 6: Flowchart of the proposed IDS methodology.

The CICDS2017 and the Power System dataset lack the inclusion of IoT-generated traffic. In order to evaluate the performance of DeepCoin framework in IoT-generated traffic (e.g., the Internet of Energy (IoE)), we use another dataset, named Bot-IoT dataset. The Bot-IoT dataset is developed by Koroniotis et al. [35] at the University of New South Wales Canberra. The attacks in the Bot-IoT dataset are categorized into three attack types: information gathering, DoS, and information theft.

We assume that the smart grid is divided into two completely separated networks, namely, victim-smart grid and attack-smart grid. The victim-smart grid contains the necessary equipment for building a smart grid communication, including, smart meters, routers, firewalls, switches...etc. To launch a brute force attack, an adversary uses a Python script, named Patator, which contains 30 modules (e.g., SMTP raw force, Raw force to HTTP, SSH raw force...etc). The DoS attack can be launched in the smart grid by four tools, namely, Hulk, GoldenEye, Slowloris, and Slowhttptest, in order to target the blockchain web hosting. A web attack can be launched using a PHP/MySQL web application, entitled Damn Vulnerable Web Application (DVWA), in order to hack a neighbor's smart meter. The infiltration attack can be launched by the Metasploit tool to develop and execute exploits against the victim-smart grid. Specifically, the attack-smart grid uses three modules, namely, 1) *Exploit module*, which is used to exploit a vulnerability on the victim-smart grid, 2) *Payload module*, which is used for opening a port on the victim-smart grid connected to a shell or opening a virtual network computing session, and 3) *Auxiliary modules*, which are used for various tasks (e.g., execute the Nmap and portscan). To change power flows on the lines in the victim-smart grid, an adversary's botnet

disrupt the power grid's normal operation using different tools for Botnet attacks, such as Ares. In addition, an adversary can launch a DDoS attack and PortScan in the AMI network environment using the LOIC tool (e.g., sending the UDP, TCP, or HTTP requests to the victim-smart grid), in order to deplete the resources, deteriorate the performance of packet delivery, and drop an amount of the legitimate packets of the victim-smart grid. For detail explanation of other attacks in Smart Grid, we refer the readers to [3].

2) Pre-processing stage: The CSV version of CICIDS 2017 contains 2,830,743 rows devised on 8 files, containing 79 features for every traffic record. The CSV version of Power System dataset contains 78404 rows devised on 15 files with 128 features for every traffic record. The CSV version of Bot-IoT dataset contains 73,360,900 rows with 32 features for every traffic record. For each dataset, we concatenate the files in one same table that contains all benign and attacks rows. Then, we create a training and test subset for each data set.

3) Normalization stage: In order to help the DeepCoin framework to converge and achieve its objectives we performed scaling of the data into a specific range [0,1] using the Min-Max transformation :

$$\overline{x_i(j)} = \frac{x_i(j) - Min(x(j))}{Max(x(j)) - Min(x(j))} \tag{7}$$

Where Max denotes the maximum value and Min denotes minimum value from the original set for each value x_i of the feature j.

4) Training stage: For training an RNN, we use the truncated BPTT algorithm. Refer to Fig. 5, BPTT defined by Werbos [38] and Learning representations by back-propagating errors defined by Rumelhart et al. [39], the RNN using truncated BPTT algorithm is described in an Algorithmic way in 4.

5) Testing stage: In order to test the proposed RNN-IDS model, we process each row of the test data set by the training RNN using truncated backpropagation. Then, we classify the result rows as Benign or a specific type of attack, as presented in Fig 6.

V. Performance evaluation and analysis

A. Evaluation settings

Table I summarizes the statistics of attacks in Training and Test datasets, including, CICIDS2017 dataset [33], Bot-IoT dataset [35], and Power System dataset [34]. The experiment is performed on Google Colaboratory[1] under python 3 using TensorFlow library and three types of hardware accelerators, including, Central Processing Unit (CPU), Graphics Processing Unit (GPU), and Tensor Processing Unit (TPU). We used four packages, NumPy, Pandas, Scikit-learn and Keras. The NumPy is used for manipulating multidimensional arrays as well as mathematical functions operating. The Pandas library is used for manipulating and analyzing data. The Scikit-learn library and Keras library are used for deep neural network algorithms and machine learning. Finally, we compare the

[1]https://colab.research.google.com

TABLE I: Statistics of attacks in Training and Test datasets

Dataset	Attack		Flow Count	Training	Test
CICIDS dataset	BENIGN	BENIGN	2273097	30000	30000
	DoS	DDoS	128027	3700	4300
		Heartbleed	11	5	5
		DoS slowloris	5796	2350	2650
		DoS GoldenEye	10293	2300	1700
		DoS Hulk	231073	5500	6500
		DoS Slowhttptest	5499	2161	1159
	Web Attack	Web Attack Sql Injection	21	15	5
		Web Attack Brute Force	1507	920	480
		Web Attack XSS	652	480	160
	Infiltration	Infiltration	36	20	10
	PortScan	PortScan	158930	3800	4200
	Brute-Force	FTP-Patator	7938	910	1090
		SSH-Patator	5897	910	1090
	Bot	Bot	1966	930	630
Bot-IoT dataset	BENIGN	BENIGN	9543	4000	4000
	Information gathering	Service scanning	1463364	36700	36468
		OS Fingerprinting	358275	9002	8911
	DoS	DDoS TCP	19547603	498602	478778
		DDoS UDP	18965106	484127	464128
		DDoS HTTP	19771	594	394
		DoS TCP	12315997	317899	297900
		DoS UDP	20659491	526487	506487
		DoS HTTP	29706	942	543
	Information theft	Keylogging	1469	106	98
		Data theft	118	102	96
Power System dataset	Natural Events	Natural Events	1221	500	500
	Attack	Attack	3711	2010	890
	No event	No event	294	208	92

TABLE II: Accuracy of the proposed IDS using the CICDS2017 dataset with different hardware accelerators and hidden nodes.

	Accuracy	Training time (s)			Test time (s)		
		CPU	GPU	TPU	CPU	GPU	TPU
HN = 10	96.776%	110.4	15.2	13.1	2.15	1.21	1.20
HN = 20	97.116%	313.1	30.3	27.6	7.33	3.82	3.22
HN = 30	98.349%	824.2	55.7	51.6	15.45	6.57	6.52
HN = 40	98.444%	921.3	78.1	62.2	33.22	16.22	16.13
HN = 50	98.941%	1002.4	95.2	73.3	41.61	31.27	29.31
HN = 60	99.811%	1772.9	155.9	102.2	123.98	72.35	71.39

HN: Hidden nodes. Number of epochs = 5. The batch size=100.

TABLE III: Accuracy of the proposed IDS using the Bot-IoT dataset with different hardware accelerators and hidden nodes.

	Accuracy	Training time (s)			Test time (s)		
		CPU	GPU	TPU	CPU	GPU	TPU
HN = 10	97.177%	300.1	60.3	55.2	14.32	8.24	7.88
HN = 20	97.336%	500.4	79.1	71.6	21.87	12.77	11.83
HN = 30	97.809%	991.2	92.7	85.9	34.24	18.23	17.82
HN = 40	98.114%	1010.2	103.2	92.5	67.23	21.13	20.12
HN = 50	98.881%	1200.8	140.4	133.8	70.65	27.14	26.18
HN = 60	99.912%	2012.9	201.7	191.6	90.11	44.23	43.12

HN: Hidden nodes. Number of epochs = 5. The batch size=100.

TABLE IV: Accuracy of the proposed IDS using the Power System dataset with different hardware accelerators and hidden nodes.

	Accuracy	Training time (s)			Test time (s)		
		CPU	GPU	TPU	CPU	GPU	TPU
HN = 10	96.127%	20.2	2.3	2.1	21.31	1.13	1.01
HN = 20	96.341%	33.6	4.1	4.3	18.02	2.12	2.03
HN = 30	96.451%	54.2	6.7	6.3	15.24	2.22	2.11
HN = 40	96.684%	87.2	13.4	13.1	22.14	2.88	2.56
HN = 50	96.742%	94.6	16.5	16.2	25.77	4.14	4.03
HN = 60	96.822%	132.2	20.7	21.4	30.16	7.21	7.22

HN: Hidden nodes. Number of epochs = 5. The batch size=100.

TABLE V: Performance comparison with other models in terms of detection rate under multi-class classification.

	Proposed IDS	SVM	RF	NB
DDoS	99.899%	98.988%	98.719%	65.656%
Heartbleed	100%	100%	100%	85.000%
DoS slowloris	98.111%	91.258%	92.647%	81.556%
DoS GoldenEye	77.122%	69.252%	65.462%	63.255%
DoS Hulk	97.793%	96.275%	94.043%	72.661%
DoS Slowhttptes	94.104%	88.229%	80.041%	71.172%
Sql Injection	80.000%	70.000%	100%	100%
Brute Force	82.376%	81.487%	81.517%	59.512%
XSS	91.715%	87.612%	38.687%	90.875%
Infiltration	100%	84.296%	82.129%	82.429%
PortScan	98.995%	97.884%	99.763%	98.388%
FTP-Patator	99.747%	97.249%	98.616%	98.344%
SSH-Patator	99.923%	96.643%	97.128%	98.291%
Bot	97.585%	69.699%	98.567%	30.987%
Service scanning	87.912%	72.823%	69.823%	65.212%
OS Fingerprinting	92.218%	70.139%	82.198%	68.675%
DDoS TCP	100%	89.564%	88.281%	78.669%
DDoS UDP	100%	98.138%	55.258%	78.500%
DDoS HTTP	100%	62.239%	82.257%	50.775%
DoS TCP	100%	71.255%	81.771%	65.555%
DoS UDP	100%	100%	82.988%	100%
DoS HTTP	100%	70.139%	82.198%	68.675%
Keylogging	77.909%	65.123%	70.119%	65.618%
Data theft	99.749%	89.668%	86.551%	66.546%
Power Attack	98.876%	82.366%	81.556%	82.277%

SVM: Support Vector Machine. RF: Random Forest. NB: Naive Bayes.

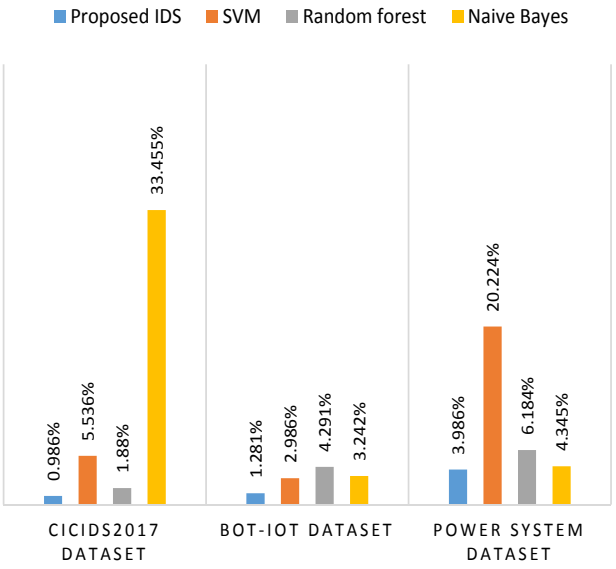

Fig. 7: Performance comparison with other models in terms of false alarm rate.

performance of the IDS model with three machine learning methods, including, Support Vector Machine, Random Forest, and Naive Bayes.

In order to evaluate the performance of our energy blockchain network, we have created a private blockchain using MultiChain[2], which is an open source blockchain platform. We consider that the energy blockchain network contains three types of nodes, $HAN = \{30, 40, 60\}$, $BAN = \{5, 10, 15\}$ and $NAN = \{5, 10, 15\}$. In addition, we assume that an attacker A uses Sybil attack to launch two types of attacks, a selfish

[2]https://www.multichain.com/

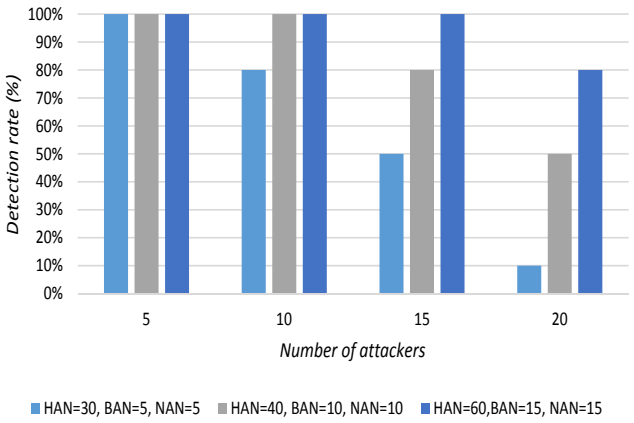

Fig. 8: The detection rate of attacks in energy blockchain network varies with the number of attackers and $HAN = \{30, 40, 60\}$, $BAN = \{5, 10, 15\}$, and $NAN = \{5, 10, 15\}$. These attackers launch two types of attacks against the energy blockchain network, a selfish mining attack and a double spending attack.

mining attack and a double spending attack. Both honest nodes and attackers have 30% of the hashing power in the beginning.

B. Evaluation metrics

In order to measure the performance of the proposed IDS model, we use the most important performance indicators, accuracy, detection rate (DR), and false alarm rate (FAR), which are defined below:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{8}$$

$$Detection\ rate = \frac{TP}{TP+FN} \tag{9}$$

$$False\ alarm\ rate = \frac{FP}{FP+TN} \tag{10}$$

The accuracy measures the proportion of the total number of correct classifications. The detection rate measures the proportion of detection of an attack. The false alarm measures the proportion of benign events incorrectly classified as attacks [56]. However, these measures are based on four metrics, True Positive (TP), False Negative (FN), True Negative (TN) and False Positive (FP). TP is equivalent to attack data that are correctly classified as an attack. FN is equivalent to attack data that are incorrectly classified as normal. TN represents normal data that are correctly classified as normal. The FP means normal data that are incorrectly classified as an attack.

C. Evaluation results

Accuracy of the proposed IDS using the CICIDS2017 dataset with different hardware accelerators (i.e., CPU, GPU, TPU) and hidden nodes $HN = \{10, 20, 30, 40, 60\}$ is shown in Table II. Due to the high-performance of GPU and TPU with hundreds of cores, both training time and test time are fast

TABLE VI: Performance comparison with other Intrusion detection systems that are based on deep learning

Framework	Year	Data set	Classification method	Blockchain	Capacity for feature diversity	Machine learning library	CPU	GPU	TPU	Accuracy
Gao et al. [40]	2014	KDD Cup 1999	DBN	No	Low	MATLAB 7.0	Yes	No	No	N/A
Alom et al. [41]	2015	NSL-KDD	DBN	No	Low	N/A	N/A	N/A	N/A	97.50%
Tang et al. [42]	2016	NSL-KDD	DNN	No	Low	N/A	N/A	N/A	N/A	75.75%
Niyaz et al. [43]	2016	NSL-KDD	STL	No	Low	N/A	N/A	N/A	N/A	79.10%
Kim et al. [44]	2016	KDD Cup 1999	RNN using LSTM	No	Low	N/A	Yes	Yes	No	96.93%
Yin et al. [45]	2017	NSL-KDD	RNN using FPBP	No	Low	Theano	Yes	No	No	81.29%
Shone et al. [46]	2018	KDD Cup 1999	NDAE	No	Low	TensorFlow	Yes	Yes	No	97.85%
		NSL-KDD			Low					89.22%
Tang et al. [47]	2018	NSL-KDD	RNN using GRU	No	Low	Keras	Yes	No	No	89%
Diro et al. [48]	2018	NSL-KDD	Deep model	No	Low	N/A	N/A	N/A	N/A	N/A
Jiang et al. [49]	2018	NSL-KDD	RNN using LSTM	No	Low	TensorFlow	Yes	Yes	No	97%
Zhou et al. [20]	2018	N/A	DNN	No	Low	TensorFlow	N/A	N/A	N/A	96.31%
Yang et al. [50]	2019	NSL-KDD	MDPCA with DBN	No	Low	TensorFlow	Yes	No	No	82.08%
		UNSW-NB15			Medium					90.21%
Zhang et al. [51]	2019	KDD Cup 1999	DGNN	No	Low	N/A	Yes	Yes	No	93.93%
Basumallik et al. [52]	2019	IEEE-30 bus	CNN	No	Medium	Keras	Yes	No	No	98.67%
		IEEE-118 bus			Medium					94.53%
DeepCoin	/	CICIDS2017	RNN using BPTT	Yes	High	TensorFlow	Yes	Yes	Yes	98.23%
		Bot-IoT			High					98.20%
		Power System			Medium					96.52%

Abbreviation & Terms. N/A: means no available results, NSL-KDD: Data set contains four attacks (DoS,U2R,R2L,Probe), STL: Self-taught Learning, RNN: Recurrent Neural Network, FPBP: Forward propagation and back propagation, LSTM: Long short term memory, BPTT: Truncated back propagation through time, GRU: Gated recurrent unit, DBN: Deep belief neural, KDD Cup 1999: Data set contains four attacks (DoS,U2R,R2L,Probe), CPU: Central processing unit, GPU: Graphics processing unit, TPU: Tensor processing unit, NDAE: an auto-encoder featuring non-symmetrical multiple hidden layers, DNN: Deep neural network, MDPCA: Modified density peak clustering algorithm. UNSW-NB15: Data set contains nine attacks (Generic, Exploits, Fuzzers, DoS, Reconnaissance, Analysis, Backdoor, Shellcode, and Worms), DGNN: Deep generative neural network, CNN : Convolutional neural network, IEEE-30 bus and IEEE-118 bus systems contain six attacks (Bus/Branch faults, Line Trip, Load Changes, Generation Changes, Shunt Disconnection, and False Data).

TABLE VII: Comparison between DeepCoin and other related frameworks in the non-blockchain systems

Metric	[53]	[19]	[54]	[20]	[55]	DeepCoin
Ledger distribution	N	N	N	N	N	Y
Fault tolerance	N	N	N	N	N	Y
Participation in consensus	N	N	N	N	N	Y
Smart contracts	N	N	N	N	N	Y
Intrusion detection system	N	Y	N	Y	N	Y
Adaptability	P	P	P	P	P	Y
Security	Y	P	Y	P	Y	Y
Privacy	Y	N	Y	N	Y	Y
Trust	P	N	P	N	P	Y
Reduced maintenance cost	N	N	N	N	N	Y
Internet of Energy (IoE) e-business model	N	N	N	N	N	Y

Abbreviation & Terms. Y: Yes; P: Partial; N: No.

TABLE VIII: Comparison between DeepCoin and other related frameworks in the blockchain systems

Metric	[11]	[14]	[15]	[16]	[17]	DeepCoin
Ledger distribution	Y	Y	Y	Y	Y	Y
Fault tolerance	P	P	Y	P	P	Y
Participation in consensus	Y	Y	Y	Y	Y	Y
Smart contracts	Y	Y	Y	Y	Y	Y
Intrusion detection system	N	N	N	N	N	Y
Adaptability	Y	Y	P	Y	Y	Y
Security	Y	Y	Y	Y	Y	Y
Privacy	Y	Y	Y	Y	Y	Y
Trust	Y	Y	Y	Y	Y	Y
Reduced maintenance cost	P	P	Y	Y	Y	Y
Internet of Energy (IoE) e-business model	Y	Y	Y	Y	Y	Y

Abbreviation & Terms. Y: Yes; P: Partial; N: No.

compared to the performance with a CPU. The accuracy of the proposed IDS increases when the number of hidden nodes increases; the better accuracy is 99.811% which is achieved with 60% hidden nodes.

Accuracy of the proposed IDS using the Bot-IoT dataset with different hardware accelerators (i.e., CPU, GPU, TPU) and hidden nodes $HN = \{10, 20, 30, 40, 60\}$ is depicted in Table III. For the case of $HN = 10$, the results show that for both the number of epochs = 5 and the batch $size = 100$, the accuracy is 97.177%.

Accuracy of the proposed IDS using the Power System dataset with different hardware accelerators (i.e., CPU, GPU, TPU) and hidden nodes $HN = \{10, 20, 30, 40, 60\}$ is shown in Table IV. For the case of $HN = 60$, the results show that for both the number of epochs = 5 and the batch $size = 100$, the accuracy is 96.822%. In addition, the results show that the performance does not improve with the number of hidden nodes. This is because the Power System dataset does not contain many attacks compared to both the CICIDS2017 dataset and the Bot-IoT dataset. It is concluded that the proposed IDS is efficient and shows higher performance with datasets that contain many types of attacks.

Table V demonstrates the performance comparison of the proposed IDS against other classifiers, including, SVM, RF, and NB, in terms of detection rate under multi-class classification. According to the results, the detection rate for each attack obtained by our proposed IDS is higher compared to the detection rate obtained by SVM, RF, and NB. In addition, the proposed IDS reaches a 100% detection rate for eight attacks, including, Heartbleed, Infiltration, DDoS TCP, DDoS UDP, DDoS HTTP, DoS TCP, DoS UDP, and DoS HTTP.

Figure 7 shows the performance comparison of proposed IDS with other models, including, SVM, RF, and NB, in terms of false alarm rate under three datasets. Mean false alarm rate of the proposed IDS is 0.986% in CICIDS2017 dataset, 1.281% in Bot-IoT dataset, and 3.986% in Power System dataset, which are better than those obtained using SVM, RF, and NB.

Figure 8 shows the detection rate of attacks in energy blockchain network varies with the number of attackers at $HAN = \{30, 40, 60\}$, $BAN = \{5, 10, 15\}$, and $NAN = \{5, 10, 15\}$. The DeepCoin framework gets a higher detection rate of attacks (i.e., selfish mining attack and double spending

attack) when there are 5 attackers, $HAN = \{30\}$, $BAN = \{5\}$, and $NAN = \{5\}$. The detection rate decreases when the number of attackers is greater than 50% of the total computing power of the entire blockchain (i.e., total number of BAN and NAN nodes).

D. Comparison between DeepCoin and other related frameworks

Table VI shows the performance of DeepCoin framework compared with the Deep learning based intrusion detection systems, that have been previously proposed. In the machine learning, there is a parameter, named hyperparameter, which value is set before the learning process begins. The deep learning-based IDS systems use two types of hyperparameters, namely, 1) hyperparameters related to the network structure and 2) hyperparameters related to training algorithm. The hyperparameters related to the network structure include the number of hidden layers and the weight initialization schemes (i.e., Sigmoid, Softmax). The hyperparameters related to training algorithm include learning rate, number of epochs, and batch size. Gao et al. [40] and Alom et al. [41] proposed IDS systems based on Deep belief networks, which the pre-training is modeled using the restricted Boltzmann machine. Niyaz et al. Niyaz et al. [43] used self-taught learning (STL) as a deep learning approach. The STL technique consists of two stages for the classification, including, 1) feature representation is learned from a large collection of unlabeled data and 2) apply the first stage to labeled data for the classification task. The recurrent neural network algorithms are used by IDS models, such as long short-term memory (LSTM) [44], [49], forward propagation and back propagation (FPBP) [45], and gated recurrent unit (GRU) [47]. Therefore, many existing IDS systems utilize three datasets, including, KDD Cup 1999 dataset, NSL-KDD dataset, and UNSW-NB15 dataset, which are outdated and of very limited practical value for a modern IDS. The DeepCoin framework is evaluated in moderns datasets, such as the CICIDS2017 dataset and the Bot-IoT dataset. In addition, there is only the DeepCoin framework that combines blockchain technology with the deep learning approach using a truncated BPTT algorithm for intrusion detection.

The comparisons between DeepCoin and other related frameworks in both non-blockchain systems and blockchain systems are presented in Table VII and Table VIII, respectively. The first metric considered is ledger distribution and this refers to the ability of replication and saving an identical copy of the ledger by each energy node in the smart grid. The next metric considered is fault tolerance and this refers to the ability to identify failures through distributed consensus protocols. The proposed DeepCoin framework uses the practical Byzantine fault tolerance algorithm for achieving consensus in the smart grid. Intrusion detection was also considered which refers to the ability to detect network attacks and fraudulent transactions. DeepCoin provides this property and makes it infeasible to change or modify energy data in the smart grid based on a deep learning approach. Security, privacy, and trust, which are the most important challenges faced by the smart grid, was also considered. The DeepCoin framework coupled with short signatures and hash functions ensures these properties. The blockchain technology enables trust between transacting energy nodes (i.e., removing the need for energy nodes to trust centralized entities to handle their energy data). In addition, DeepCoin provides an Internet of Energy (IoE) e-business model, by incentivizing users to make energy available for others to use on demand, in exchange for cryptocurrency.

E. Privacy analysis

In this subsection, we analyze the security properties of our proposed DeepCoin framework by focusing on privacy preservation.

Let $\mathcal{A}$ be an adversary attacking the proposed DeepCoin framework under strong existential unforgeability game. The result in [22] has shown that the short signatures scheme is existentially unforgeable under an adaptive chosen message attack and is data private, based on the q-SDH assumption holds in $(\mathbb{G}_1, \mathbb{G}_2)$. In the proposed DeepCoin framework, on one hand, the transaction data $SigBloc(\sigma_i, b_i, r_i)$ is protected by a hash function and a validated certificate. On other hand, by using σ_i in $SigBloc$, both energy buyer node EB_i and energy vendor node EV_j can easily check whether two signatures on the same packet are generated by the same signer or not. In addition, the blockchain combined with the short signatures scheme and hash function ensures that an adversary cannot pose as the energy buyer node or energy vendor node. Hence, the proposed DeepCoin framework satisfies privacy-preservation.

VI. Conclusion

In this paper, we have proposed a novel deep learning and blockchain-based energy framework, called DeepCoin, for Smart Grids. Based on short signatures and hash functions, in DeepCoin, users can exploit the excess energy and sell it to other neighboring users while preserving privacy. With the use of the practical Byzantine fault tolerance algorithm, DeepCoin can achieve consensus inside the blockchain-based energy network. DeepCoin includes a novel Deep learning-based scheme using a recurrent neural network algorithm. Through performance evaluations using three datasets we demonstrated the efficiency of the proposed DeepCoin framework.

For future work, we plan to study the performance of the DeepCoin framework with the integration of edge computing [57], [58] in the smart grid. We will consider an edge computing enabled blockchain network in the smart grid, where energy nodes can access and utilize computing services from an edge computing service provider. This integration may help the energy nodes achieve optimal energy management policy.

References

[1] "Energy consumption in the uk," https://www.gov.uk/government/statistics/energy-consumption-in-the-uk, accessed: 2018-01-24.

[2] "Us energy information administration, us department of energy," https://www.eia.gov/energyexplained/index.php?page=electricity_in_the_united_states, accessed: 2018-01-24.

[3] M. A. Ferrag, L. A. Maglaras, H. Janicke, J. Jiang, and L. Shu, "A systematic review of data protection and privacy preservation schemes for smart grid communications," *Sustainable Cities and Society*, vol. 38, pp. 806–835, 2018.

[4] S. Tan, D. De, W.-Z. Song, J. Yang, and S. K. Das, "Survey of security advances in smart grid: A data driven approach," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 1, pp. 397–422, 2017.

[5] P. Srikantha and D. Kundur, "A der attack-mitigation differential game for smart grid security analysis." *IEEE Trans. Smart Grid*, vol. 7, no. 3, pp. 1476–1485, 2016.

[6] M. A. Ferrag, "Epec: an efficient privacy-preserving energy consumption scheme for smart grid communications," *Telecommunication Systems*, vol. 66, no. 4, pp. 671–688, 2017.

[7] R. Yang, F. R. Yu, P. Si, Z. Yang, and Y. Zhang, "Integrated blockchain and edge computing systems: A survey, some research issues and challenges," *IEEE Communications Surveys & Tutorials*, 2019.

[8] "The essential eight technologies board byte: blockchain," https://www.pwc.com.au/pdf/essential-8-emerging-technologies-blockchain.pdf. last accessed 30 Apr. 2019.

[9] C. Pop, T. Cioara, M. Antal, I. Anghel, I. Salomie, and M. Bertoncini, "Blockchain based decentralized management of demand response programs in smart energy grids," *Sensors*, vol. 18, no. 1, p. 162, 2018.

[10] N. Z. Aitzhan and D. Svetinovic, "Security and privacy in decentralized energy trading through multi-signatures, blockchain and anonymous messaging streams," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 5, pp. 840–852, 2018.

[11] Z. Li, J. Kang, R. Yu, D. Ye, Q. Deng, and Y. Zhang, "Consortium blockchain for secure energy trading in industrial internet of things," *IEEE transactions on industrial informatics*, vol. 14, no. 8, pp. 3690–3700, 2018.

[12] Z. Guan, G. Si, X. Zhang, L. Wu, N. Guizani, X. Du, and Y. Ma, "Privacy-preserving and efficient aggregation based on blockchain for power grid communications in smart communities," *IEEE Communications Magazine*, vol. 56, no. 7, pp. 82–88, 2018.

[13] M. A. Ferrag, M. Derdour, M. Mukherjee, A. Derhab, L. Maglaras, and H. Janicke, "Blockchain technologies for the internet of things: Research issues and challenges," *IEEE Internet of Things Journal*, 2018.

[14] J. Gao, K. O. Asamoah, E. B. Sifah, A. Smahi, Q. Xia, H. Xia, X. Zhang, and G. Dong, "Gridmonitoring: Secured sovereign blockchain based monitoring on smart grid," *IEEE Access*, vol. 6, pp. 9917–9925, 2018.

[15] M. Fan and X. Zhang, "Consortium blockchain based data aggregation and regulation mechanism for smart grid," *IEEE Access*, vol. 7, pp. 35 929–35 940, 2019.

[16] K. Gai, Y. Wu, L. Zhu, M. Qiu, and M. Shen, "Privacy-preserving energy trading using consortium blockchain in smart grid," *IEEE Transactions on Industrial Informatics*, 2019.

[17] K. Gai, Y. Wu, L. Zhu, L. Xu, and Y. Zhang, "Permissioned blockchain and edge computing empowered privacy-preserving smart grid networks," *IEEE Internet of Things Journal*, 2019.

[18] M. A. Faisal, Z. Aung, J. R. Williams, A. Sanchez *et al.*, "Data-stream-based intrusion detection system for advanced metering infrastructure in smart grid: A feasibility study." *IEEE Systems journal*, vol. 9, no. 1, pp. 31–44, 2015.

[19] P. Jokar and V. C. Leung, "Intrusion detection and prevention for zigbee-based home area networks in smart grids," *IEEE Transactions on Smart Grid*, vol. 9, no. 3, pp. 1800–1811, 2018.

[20] L. Zhou, X. Ouyang, H. Ying, L. Han, Y. Cheng, and T. Zhang, "Cyber-attack classification in smart grid via deep neural network," in *Proceedings of the 2nd International Conference on Computer Science and Application Engineering*. ACM, 2018, p. 90.

[21] C. H. Liu, Q. Lin, and S. Wen, "Blockchain-enabled data collection and sharing for industrial iot with deep reinforcement learning," *IEEE Transactions on Industrial Informatics*, pp. 1–1, 2018.

[22] D. Boneh and X. Boyen, "Short signatures without random oracles and the sdh assumption in bilinear groups," *Journal of Cryptology*, vol. 21, no. 2, pp. 149–177, 2008.

[23] ——, "Short signatures without random oracles," in *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2004, pp. 56–73.

[24] M. Castro and B. Liskov, "Practical byzantine fault tolerance," in *OSDI*, vol. 99, 1999, pp. 173–186.

[25] Y. Xiao, N. Zhang, J. Li, W. Lou, and Y. T. Hou, "Distributed consensus protocols and algorithms," *Blockchain for Distributed Systems Security*, p. 25, 2019.

[26] M. A. Ferrag and A. Ahmim, "Esspr: an efficient secure routing scheme based on searchable encryption with vehicle proxy re-encryption for vehicular peer-to-peer social network," *Telecommunication Systems*, vol. 66, no. 3, pp. 481–503, 2017.

[27] M. Abd-El-Malek, G. R. Ganger, G. R. Goodson, M. K. Reiter, and J. J. Wylie, "Fault-scalable byzantine fault-tolerant services," *ACM SIGOPS Operating Systems Review*, vol. 39, no. 5, pp. 59–74, 2005.

[28] J. Cowling, D. Myers, B. Liskov, R. Rodrigues, and L. Shrira, "Hq replication: A hybrid quorum protocol for byzantine fault tolerance," in *Proceedings of the 7th symposium on Operating systems design and implementation*. USENIX Association, 2006, pp. 177–190.

[29] R. Kotla, L. Alvisi, M. Dahlin, A. Clement, and E. Wong, "Zyzzyva: Speculative byzantine fault tolerance," *ACM Transactions on Computer Systems (TOCS)*, vol. 27, no. 4, p. 7, 2009.

[30] R. Guerraoui, N. Knežević, V. Quéma, and M. Vukolić, "The next 700 bft protocols," in *Proceedings of the 5th European conference on Computer systems*. ACM, 2010, pp. 363–376.

[31] I. Sutskever, *Training recurrent neural networks*. University of Toronto Toronto, Ontario, Canada, 2013.

[32] R. Pascanu, C. Gulcehre, K. Cho, and Y. Bengio, "How to construct deep recurrent neural networks," *arXiv preprint arXiv:1312.6026*, 2013.

[33] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization." in *ICISSP*, 2018, pp. 108–116.

[34] U. Adhikari, S. Pan, T. Morris, R. Borges, and J. Beave, "Industrial control system (ICS) cyber attack datasets," https://sites.google.com/a/uah.edu/tommy-morris-uah/ics-data-sets. last accessed 02 Jan. 2019.

[35] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, "Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset," *arXiv preprint arXiv:1811.00701*, 2018.

[36] S. Pan, T. Morris, and U. Adhikari, "Developing a hybrid intrusion detection system using data mining for power systems," *IEEE Transactions on Smart Grid*, vol. 6, no. 6, pp. 3104–3113, 2015.

[37] ——, "Classification of disturbances and cyber-attacks in power systems using heterogeneous time-synchronized data," *IEEE Transactions on Industrial Informatics*, vol. 11, no. 3, pp. 650–662, 2015.

[38] P. J. Werbos, "Backpropagation through time: what it does and how to do it," *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990.

[39] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, p. 533, 1986.

[40] N. Gao, L. Gao, Q. Gao, and H. Wang, "An intrusion detection model based on deep belief networks," in *Advanced Cloud and Big Data (CBD), 2014 Second International Conference on*. IEEE, 2014, pp. 247–252.

[41] M. Z. Alom, V. Bontupalli, and T. M. Taha, "Intrusion detection using deep belief networks," in *Aerospace and Electronics Conference (NAECON), 2015 National*. IEEE, 2015, pp. 339–344.

[42] T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho, "Deep learning approach for network intrusion detection in software defined networking," in *Wireless Networks and Mobile Communications (WINCOM), 2016 International Conference on*. IEEE, 2016, pp. 258–263.

[43] A. Javaid, Q. Niyaz, W. Sun, and M. Alam, "A deep learning approach for network intrusion detection system," in *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS)*. ICST (Institute for Computer Sciences, Social-Informatics and âĂę, 2016, pp. 21–26.

[44] J. Kim, J. Kim, H. L. T. Thu, and H. Kim, "Long short term memory recurrent neural network classifier for intrusion detection," in *Platform Technology and Service (PlatCon), 2016 International Conference on*. IEEE, 2016, pp. 1–5.

[45] C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," *IEEE Access*, vol. 5, pp. 21 954–21 961, 2017.

[46] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A deep learning approach to network intrusion detection," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 1, pp. 41–50, 2018.

[47] T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho, "Deep recurrent neural network for intrusion detection in sdn-based networks," in *2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)*. IEEE, 2018, pp. 202–206.

[48] A. A. Diro and N. Chilamkurti, "Distributed attack detection scheme using deep learning approach for internet of things," *Future Generation Computer Systems*, vol. 82, pp. 761–768, 2018.

[49] F. Jiang, Y. Fu, B. B. Gupta, F. Lou, S. Rho, F. Meng, and Z. Tian, "Deep learning based multi-channel intelligent attack detection for data security," *IEEE Transactions on Sustainable Computing*, 2018.

[50] Y. Yang, K. Zheng, C. Wu, X. Niu, and Y. Yang, "Building an effective intrusion detection system using the modified density peak clustering algorithm and deep belief networks," *Applied Sciences*, vol. 9, no. 2, p. 238, 2019.
[51] H. Zhang, X. Yu, P. Ren, C. Luo, and G. Min, "Deep adversarial learning in intrusion detection: A data augmentation enhanced framework," *arXiv preprint arXiv:1901.07949*, 2019.
[52] S. Basumallik, R. Ma, and S. Eftekharnejad, "Packet-data anomaly detection in pmu-based state estimator using convolutional neural network," *International Journal of Electrical Power & Energy Systems*, vol. 107, pp. 690–702, 2019.
[53] R. Lu, X. Liang, X. Li, X. Lin, and X. Shen, "Eppa: An efficient and privacy-preserving aggregation scheme for secure smart grid communications," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 9, pp. 1621–1631, 2012.
[54] H. Li, X. Lin, H. Yang, X. Liang, R. Lu, and X. Shen, "Eppdr: An efficient privacy-preserving demand response scheme with adaptive key evolution in smart grid," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 8, pp. 2053–2064, 2014.
[55] Y. Liu, W. Guo, C.-I. Fan, L. Chang, and C. Cheng, "A practical privacy-preserving data aggregation (3pda) scheme for smart grid," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 3, pp. 1767–1774, 2019.
[56] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 2, pp. 1153–1176, 2016.
[57] Z. Xiong, Y. Zhang, D. Niyato, P. Wang, and Z. Han, "When mobile blockchain meets edge computing," *IEEE Communications Magazine*, vol. 56, no. 8, pp. 33–39, 2018.
[58] M. Mukherjee, R. Matam, L. Shu, L. Maglaras, M. A. Ferrag, N. Choudhury, and V. Kumar, "Security and privacy in fog computing: Challenges," *IEEE Access*, vol. 5, pp. 19 293–19 304, 2017.

Mohamed Amine Ferrag received the bachelor's, master's, and Ph.D. degrees from Badji Mokhtar - Annaba University, Algeria, in 2008, 2010, and 2014, respectively, all in computer science. Since 2014, he is a senior lecturer with the Department of Computer Science, Guelma University, Algeria. His research interests include wireless network security, network coding security, and applied cryptography. He serves on the Editorial Board of several International peer-reviewed journals such as the IET Networks (IET), the International Journal of Information Security and Privacy (IGI Global), the International Journal of Internet Technology and Secured Transactions (Inderscience Publishers), and the EAI Endorsed Transactions on Security and Safety (EAI). He has served as an Organizing Committee Member (the Track Chair, the Co-Chair, the Publicity Chair, the Proceedings Editor, and the Web Chair) in numerous international conferences.

Leandros Maglaras (SM'15) received the B.Sc. degree from Aristotle University of Thessaloniki, Greece in 1998, M.Sc. in Industrial Production and Management from University of Thessaly in 2004 and M.Sc. and PhD degrees in Electrical & Computer Engineering from University of Volos, in 2008 and 2014 respectively. He is the head of the National Cyber Security Authority of Greece and a visiting Lecturer in the School of Computer Science and Informatics at the De Montfort University, U.K. He serves on the Editorial Board of several International peer-reviewed journals such as IEEE Access, Wiley Journal on Security & Communication Networks, EAI Transactions on e-Learning and EAI Transactions on Industrial Networks and Intelligent Systems. He is an author of more than 80 papers in scientific magazines and conferences and is a senior member of IEEE. His research interests include wireless sensor networks and vehicular ad hoc networks.