Deep Learning and Smart Contract-Assisted Secure Data Sharing for IoT-Based Intelligent Agriculture

Randhir Kumar, Indian Institute of Technology Hyderabad, Hyderabad, 502285, India
Prabhat Kumar, LUT University, 53850, Lappeenranta, Finland
Ahamed Aljuhani, University of Tabuk, Tabuk, 47512, Saudi Arabia
A. K. M. Najmul Islam, LUT University, 53850, Lappeenranta, Finland
Alireza Jolfaei, Flinders University, Adelaide, SA, 5042, Australia
Sahil Garg, Resilient Machine Learning Institute (ReMI), Montreal, QC, H3C 1K3, Canada

The recent development of Internet of Things (IoT) and unmanned aerial vehicles (UAVs) has revolutionized traditional agriculture with intelligence and automation. In a typical intelligent agriculture (IA) ecosystem, massive and real-time data are generated, analyzed, and sent to the cloud server (CS) for the purpose of addressing complex agricultural issues, such as yield prediction, water feed calculation, and so on. This helps farmer and associated stakeholders to take correct decision that improves the yield and quality of agricultural product. However, the distributed nature of IA entities and the usage of insecure wireless communication open various challenges related to data sharing, monitoring, storage, and further makes the entire IA ecosystem vulnerable to various potential attacks. In this article, we exploit deep learning and smart contract to propose a new IoT-enabled IA framework for enabling secure data sharing among its various entities. Specifically, first we develop new authentication and key management scheme to ensure secure data transmission in IoT-enabled IA. The encrypted transactions are then used by the CS to analyze and further detect intrusions by a novel deep learning architecture. In CS, the smart contract (SC)-based consensus mechanism is executed on legitimate transactions that verifies and adds the formed blocks into blockchain by a peer-to-peer CSs network. In comparison to existing competing security solutions, a rigorous comparative research demonstrates that the proposed approach provides greater security and more utility characteristics.

According to a UN study, the world’s population will reach 9.8 billion people by 2050. This rise in population demands nearly 70% increase in current food production rate. Agriculture is the world’s most important industry, contributing significantly to social stability and economic progress. The transition from traditional agriculture (also known as Agriculture 1.0) to intelligent agriculture (IA) (also known as Agriculture 4.0) is the only alternative to meet the growing demand efficiently. IA is a new approach that uses the current information and communication technologies in conjunction with conventional farming practices to improve the quality and quantity of agricultural...
products. The IA helps in intelligent decision making and provides various personalized services through different key technologies including, Internet of Things (IoT), unmanned aerial vehicles (UAVs), cloud computing, and artificial intelligence.

In a typical IA ecosystem, several data acquisition technologies, such as IoT devices and actuators are deployed to collect both field and crop growth information. In addition, UAVs are used to gather data from IoT devices and, in certain cases, they may collect data directly from particular flying zones (FZ). The data acquired are forwarded to cloud servers (CSs) for the purpose of addressing complex agricultural issues, such as yield prediction, water feed calculation, and so on, assisting farmers and other stakeholders in making smart decisions that increase agricultural production and quality. However, the distributed nature of IA entities (including IoT devices, UAVs, and CSs), and the usage of insecure wireless communication open various challenges related to data sharing, monitoring, storage, and further makes the entire IA ecosystem vulnerable to various potential attacks including impersonation, replay, man-in-the-middle, data poisoning, brute-force, physical smart devices, and UAVs capture attacks.

In the literature, several key management mechanisms, blockchain- and SC-based authentication strategies for enhancing security of IoT-enabled IA have been put forth. For instance, works presented in Bera et al. Vangala et al. Rahman et al., and Vangala et al. were mainly based on user authentication/authorization and session key management. However, all abovementioned solution used blockchain as a distributed storage mechanism to store entire agricultural transactions. Unfortunately, blockchain becomes inefficient when complete transaction are offloaded to the distributed ledger but works better with data hashes. Furthermore, we believe that all of the abovementioned authorization and authentication techniques are insufficient for addressing security issues in IoT-enabled IA networks since they only ensure that data transmission is secure but do not guarantee or check the type of data (attack or normal) before it is added to blockchain.

Motivated from the aforementioned challenges, we exploit deep learning and SC to propose a new IoT-enabled IA framework for enabling secure data sharing among its various entities. Specifically, first we develop new authentication and key management scheme to ensure secure data transmission in IoT-enabled IA. The encrypted transactions are then used by the CS to analyze and further detect intrusions by a novel deep learning architecture. The latter is a novel architecture that is designed using a contractive sparse autoencoder (CSAE), gated recurrent unit (GRU) networks, multilayer perceptrons (MLPs), and softmax classifier for attack detection. In CS, the SC-based proof of authority (PoA-Aura) consensus mechanism is executed on legitimate transactions that verifies and adds the formed blocks into interplanetary file system (IPFS) by a peer-to-peer (P2P) CSs network. The returned cryptographic hash if further stored on blockchain.

**SYSTEM MODELS**

We introduce a network model in this part, followed by a threat model, both of which were used in the design of the proposed framework.

**Network Model**

The network model of the proposed framework is illustrated in Figure 1. In this model, we have mainly eight entities, trusted authority (TA), IoT device (IoTD), UAV, intrusion detection system (IDS), CS, IPFS, SC, and blockchain network (BN). In IoT-enabled IA, the TA is responsible to register IoT devices, and other entities like IDS, CS, IPFS, SC, and blockchain network (BN) are responsible to process data. The network model comprises of trusted authority (TA), IoT device (IoTD), UAV, intrusion detection system (IDS), cloud server (CS), interplanetary file system (IPFS), SC, and blockchain network (BN). The trusted authority (TA) is responsible for registering IoTDs, and the CS is responsible for processing data.
their deployment. Initially, the authentication and key management phase includes mutual authentication and key agreement between two IoT$_D$ blocks, between IoT$_D$ and its associated UAV, and between UAV and CS using the established session keys. This phase ensures secure communication among the participating entities. Once the communications starts, the IoT$_D$ placed in each FZ has the capability to extract crop readings from its zone. Each FZ is associated with a UAV that collects the readings from IoT$_D$. These data or transactions include the status of standing crops, quantity of chemicals used as pesticides at various locations, and so on. These transactions are forwarded to CS, where the proposed IDS checks and marks the transaction as normal and abnormal based on the behavior. The valid transactions are then used by each CS for mining using SC. Specifically, each CS mines and stores the valid transactions into IPFS, keeping the returned transaction hash into the global BN.

The IPFS hashes of the verified transactions are packed into the current block by each miner, who also creates the merkle root and block hash while calculating the subsequent block. If $CS_A$ calculates a block hash that satisfies the difficulty, it will be broadcast to miners $CS_B$, $CS_C$, and $CS_B$, and so on. After receiving the block, miners $CS_B$, $CS_C$, and $CS_B$ must check the transactions and block hash. The majority of transactions received by miners $CS_B$, $CS_C$, and $CS_B$ throughout the mining process are similar to those received by miner $CS_A$. They only alter a small number of transactions in their transaction pool because of network transmission delays. As a consequence, the vast majority of IPFS hashes for transactions in the new block match those in miner $CS_A$, then the transaction has already been confirmed by these miners and does not require downloading from IPFS. The IPFS network must be accessed using the proper IPFS hashes in order to receive the data for the remaining transactions. The authenticity of the block and the transactions would then be verified. The BN could then be updated with the new block.

**Threat Model**

The “Dolev-Yao” threat model, often known as the DY model, is the first one we employ in this article. This theory states that an adversary designated as $A$ has the ability to not only intercept, alter, or delete communication messages between any two participants, but also to introduce harmful messages into the channel. (TA) stands for trusted authority, which is meant to be a totally trustworthy organization. IoT$_D$ and UAV are regarded as untrusted entities, although CS are regarded as semitrusted. The Canetti and Krawczyk adversary model (also known as CK-adversary), which is another threat model, is also used. An attacker $A$ in this scenario has the ability to hijack the session key/state on a live session between two network users, and steal confidential credentials.

**PROPOSED FRAMEWORK**

### Deep Learning Module

In this section, a deep learning model is proposed that is used to detect intrusion in the IoT-enabled IA ecosystem. When dealing with large amounts of data in IoT-enabled IA, deep learning models surpass conventional statistical or machine learning techniques. This phenomenon has been discovered and validated in a number of research articles and publications.

As a consequence, when compared to other statistical or machine learning methodologies, deep learning is a better option. This article introduces a novel deep learning architecture for developing a better IDS for IoT-enabled IA. In this approach, we have combined CS AE, GRU network, MLPs, and softmax classifier. Each of them is explained in the following.

**Contractive Sparse AutoEncoder Layer**

The autoencoder (AE) is a technique for unsupervised learning that consists of two components: the encoder and decoder. As seen here, the encoder uses a deterministic affine transformation matrix with non-linearity to transform the input $D_T$ into a hidden representation $Y_T$.

$$Y_T = F(L_1 D_T + b_1)$$

where $L_{strok}^1$ is the weight between the input $D_T$, and the hidden representation $Y_T$ and $b_1$ denotes the bias. The $Y_T$ variable is used by the decoder to recreate the output $\hat{D}_T$

$$\hat{D}_T = F'(L_2 Y_T + b_2)$$

where the weight of the hidden representations $Y_T$, and $\hat{D}_T$ is denoted by $L_2$ and bias is represented by $b_2$. $\hat{D}_T$ is the name given to the reconstruction of $D_T$. The purpose of AE is to minimize the reconstruction error for a given training set, which is performed by decreasing the following cost function while learning the AE parameters $L_1, L_2, b_1, b_2$: 

$$C = \frac{1}{2} \sum \frac{1}{2} \left\| X - \hat{X} \right\|^2$$
The training sample and its reconstruction output are represented by $D_T$ and $\tilde{D}_T$, respectively. $N$ is the total number of training samples, and $L(D_T, \tilde{D}_T)$ represents the loss function. Using square error or cross entropy, this can be decreased. $\lambda$ stands for regularization term, which aids model generalization. From the original dataset, the sparse autoencoder (SAE) attempts to learn sparse yet inherent features. The SAE loss function is stated as, and it is produced by adding a sparsity penalty term to the AE loss function:

$$
\mathcal{L}_{SAE}(L, \hat{L}, \hat{b}, \hat{b}) = \mathcal{L}_{AE}(L, \hat{L}) + \frac{\lambda}{2} \sum_{i=1}^{M} \frac{1}{N} \sum_{j=1}^{N} \left( \| D_T \|_2^2 + \eta \sum_{j=1}^{M} \text{KL}(\theta \| \hat{\theta}_j) \right).
$$

The model becomes less sensitive to modest changes in the input values as a result of this process. It simply instructs the neurons to ignore little data changes and respond only to larger, more meaningful ones. This “penalty” is only applied during the training of the model; therefore, it has no bearing when the network is employed. As a result, the cost function of the CSAE can be written as follows:

$$
\mathcal{L}_{CSAE}(L, \hat{L}, \hat{b}, \hat{b}) = \mathcal{L}_{CSAE}(L, \hat{L}) + \frac{\lambda}{2} \sum_{i=1}^{M} \frac{1}{N} \sum_{j=1}^{N} \left( \| J_T(D_T) \|_F^2 + \eta \sum_{j=1}^{M} \text{KL}(\theta \| \hat{\theta}_j) \right).
$$

where $\| J_T(D_T) \|_F^2$ represents the square of the Jacobian matrix’s Frobenius norm. For attack detection, the acquired features are fed into the following module, which comines GRU+MLP+softmax classifiers.

### GRU Network Layer

The GRU network receives the low-dimensional feature vector from CSAE layer. The GRU can be single-layered or multilayered (stacked), depending on the hyperparameter optimization. The $D_T$ is the input for a given time step $T$, and the computations are

$$
\begin{align*}
R_{T} &= \sigma(D_T \mathbf{L}_R + \mathbf{H}_{T-1} \mathbf{L}_R + \mathbf{b}_R) \mathbf{Z}_T \\
U_{T} &= \tanh(D_T \mathbf{L}_U + \mathbf{H}_{T-1} \mathbf{L}_U + \mathbf{b}_U) \mathbf{C}_T \\
Z_T &= R_T \odot \mathbf{H}_{T-1} + (1 - Z_T) \odot \mathbf{C}_T.
\end{align*}
$$

The previous time-hidden state’s step is $\mathbf{H}_{T-1}$, the reset gate is $R_T$, the update gate is $Z_T$, the weight parameters are $\mathbf{L}_R$ and $\mathbf{L}_U$, and the biases are $\mathbf{b}_R$ and $\mathbf{b}_U, \mathbf{C}_T$ are the hidden candidate state, whereas $\mathbf{H}_T$ is the new state. ReLU function is denoted by the letter $\odot$, which stands for Hadamard product. The output of a multilayered GRU network is the hidden state $\mathbf{H}_T$ of the preceding layer, and there is no dropout between the layers.

### MLPs Layer

The dense layer of MLP uses the output vector of the GRU layer $\mathbf{H}_T$ to represent the output activation of its node in the following way:

$$
\begin{align*}
D_1(a) &= f(L_1^T \mathbf{H}_T + b_1) \\
D_2(M) &= f(L_2^T D_1(a) + b_2).
\end{align*}
$$

The weight matrix $L_1$ connects the output of the GRU layer with the first hidden layer, and the bias vector $b_1$.
is connected with that layer. Where $L_2$ is the weight matrix linking the first and last hidden layers, and $b_2$ is the bias associated with the weight matrices.

**Softmax Classifier Layer**

Finally, the softmax classifier is integrated with the proposed deep learning architecture to determine the likelihood that the projected type belongs to each category. We use (11) to compute it, where $M$ is the previous layer’s output and softmax’s input, $C$ is its dimension, and $k$ is the probability of the projected type belonging to a certain class. Equation (12) is used to compute the loss function

$$f(M_{l}) = e^{M_{l}}\sum_{j=1}^{C} e^{M_{j}}, \quad k = 1, \ldots, C$$

$$\text{LOSS}(y, M) = -\sum_{k=1}^{C} y_k \log(f(M_k)))$$

**SC Module**

In this section, we have discussed the steps used by the proposed SC-based authentication and key management module.

**Initialization Phase**

This phase explores, how TA chooses the parameters to register the entities of framework. The detailed process is discussed in the following. First, nonsingular elliptic curve is selected by the TA, i.e., $E(\beta, \gamma) S^2 = T^3 + aT + \gamma$ (mod $W_n$), where $W_n$ is a large prime value and $\beta, \gamma \in V^* = \{1,2,3, \ldots, W_n\}$ are the two points, i.e, infinity point and zero point $2O$. Further, the TA chooses a base point $BP \in E(\beta, \gamma)$ of order $I$ as bigger as $W_n$. Furthermore, TA chooses a cryptographic hash function (HF) using SHA-512. In addition, TA chooses an identity $TD_{TA}$, and picks a private key $TP_{PR} \in V^*$ and evaluates a public key $TA_{PR} = TP_{PR} * BP$. Finally, the TA preserves a private key ($TP_{PR}$) secret and disseminates public parameters($E(\beta, \gamma), BP, (HF)(\cdot), TA_{PR}$).

**Registration Phase**

This phase describes a registration process of each entities and shares the communication parameters.

a) $TD_{TD}$ Registration: The $TA$ registers an IoT nodes $TD_{TD}$, where $TD_{TD}$ is connected with that layer. $TA$ chooses an unique identity $ID_{IoT}$ for registration of IoT devices. Further, $TA$ evaluates a pseudo-identity $PID_{IoT} = H(ID_{IoT}) || C_{IoT_{PR}} || TD_{TD}$, where $TD_{TD}$ is a registration time of $TD_{TD}$ and generates a certificate $CRT_{IoT} = TA_{PR} + H(PID_{IoT} || C_{IoT_{PR}} || TA_{PR}) * C_{IoT_{PR}}$ mod ($V_n$).

b) $TD_{TD}$ Registration: The $TA$ registers an IoT nodes $TD_{TD}$, where $TD_{TD}$ is connected with that layer. $TA$ chooses an unique identity $ID_{IoT}$ for registration of IoT devices. Further, $TA$ evaluates a pseudo-identity $PID_{IoT} = H(ID_{IoT}) || C_{IoT_{PR}} || TD_{TD}$, where $TD_{TD}$ is a registration time of $TD_{TD}$ and generates a certificate $CRT_{IoT} = TA_{PR} + H(PID_{IoT} || C_{IoT_{PR}} || TA_{PR}) * C_{IoT_{PR}}$ mod ($V_n$).

**Key Agreement and Authentication Phase**

We have discussed various steps used in key agreement and authentication. i) IoT nodes to UAV Authentication

Step 1: $TD_{TD}$ chooses an unique random number $N_1 \in \mathbb{Z}_p$ with valid timestamp $TSTP$, and evaluates $L_1 = h(PID_{IoT} || CR_{IoT} || PR_{IoT} || TD_{TD})$. Furthermore, $TD_{TD}$ encrypts the $L_1$ as $L_1 = E_{PR_{IoT}}(L_1)$. Moreover, $TD_{TD}$ evaluates the $L_1 = h(L_1 || CR_{IoT} || TD_{TD} || PID_{IoT} || PR_{IoT} || TSTP)$ and makes a request message $M_1 = (PID_{IoT} || CR_{IoT} || TD_{TD} || TSTP || L_1)$ and transmits to UAV through open channel.
Step-2: After receiving successful message $M_1$, timestamp gets validated $TSP_T^2$, using UAV $| TSP_T^2 - TSP_T^1 | < \Delta T$. After successful verification of timestamp, UAV checks certificates using $CRT_{T_D} = B = PB_{TA} + h(PB_{UA})$ if it matches successful then UAV receives $PSID_{T_D}$, respect to $PPR_{T_D}$ from the database and evaluates $L_1 = h(L_2 || PSID_{T_D} || PPR_{T_D} || CRT_{T_D})$ to verify whether $L_2 = L_3$. If it matches successful then UAV uses decryption $L_2$ as $L_3 = D_{PR_{UA}}(L_2)$.

Step-3: Further, UAV picks a unique random number $UAVV_1 \in Z_p$ and valid timestamp gets recorded $TSP_T^2$ and generates a temporary identity $PPR_{T_D}$ and evaluates $UAV_1 = h(PSP_{T_D} || PSID_{UA} || UAVV_1 || TSP_T^2)$ and uses encryption $UAV_1$ as $UAV_2 = E_{PR_{UA}}(UAV_1)$. Next, UAV executes a session key $SES_{UA} = h(PPR_{T_D} || L_1 || UAV_1 || TSP_T^2 || TSP_T^3)$, and creates reply message $M_2 = \{PPR_{T_D}, UAV_2, t_{CRUAV}, PSID_{UA}, TSP_T^3\}$ and transmit to $T_D$ through open channel.

Step-4: After successful receive of reply message ($M_2$) timestamp gets validated $TSP_T^3$ by $T_D$. Whether $| TSP_T^3 - TSP_T^2 | < \Delta T$ is denoting correct timestamp if it matches successful, then $T_D$ checks certificate by $CRT_{UA} = B = PB_{TA} + h(PB_{UA})$. Further, $T_D$ uses decryption, the $UAV_2$ computes $UAV_1 = D_{PR_{UA}}(UAV_2)$. Next, $T_D$ evaluates $UAV_3 = h(PPR_{T_D} || L_1 || UAV_1 || TSP_T^3 || TSP_T^2)$ and verifies if $UAV_3 = UAV_1$ then $T_D$ evaluates $PPR_{T_D} = PPR_{T_D} \oplus h(PSID_{UA} || PPR_{T_D} || TSP_T^2)$. Further, $UAV_3$ evaluates a session key $SES_{UA} = h(PPR_{T_D} || L_1 || LAV_1 || TSP_T^3)$ and disseminates to $UAV$. Further, $T_D$ chooses a valid timestamp $TSP_T^2$ and verify session key $SES_{T_D}$ by $SES_{T_D} = h(SES_{T_D} || TSP_T^2)$ and makes changes to $PPR_{T_D}$ and $PPR_{T_D}^{new}$ in the database. Furthermore, $T_D$ generates acknowledges message $M_3 = \{SES_{T_D(T_D', TSP_T^2)}, TSP_T^2\}$ and transmit to $UA$ through open channel.

Step-5: After successfully receiving the acknowledgment message $M_3$, timestamp gets validated $TSP_T^2$, by $UA$ $| TSP_T^2 - TSP_T^1 | < \Delta T$ is denoting correct timestamp. Next $UA$ checks $SES_{T_D} = h(SES_{UA} || TSP_T^3)$, after successful match, the $UA$ makes establishment of the session key $SES_{T_D} = h(SES_{UA} || TSP_T^3)$ by $T_D$. Finally, $UA$ makes changes with $PPR_{T_D}$ and $PPR_{T_D}^{new}$ in database.

ii) UAV to CS Authentication

Step-1: $UAV_1$ chooses a unique random number $\{t, t \in Z_p \}$ and valid timestamp $TSP_T^1$ and evaluates $L_1 = h(TSC_{UA} || PPR_{UA} || \{t, t \in Z_p \})$. Further, $UAV_1$ makes encryption $L_1$ as $L_2 = E_{PR_{UA}}(L_1)$. Furthermore, $UAV_1$ evaluates $L_1 = h(C_{UA} || TSP_T^1 || TSP_T^2)$ and creates message request for access $M_1 = \{PPR_{UA}, TSP_T^1, L_1\}$ and transmit to CS through open channel.

Algorithm 1. Proof-of-Authority (Aura Algorithm) for Block Verification and Addition

1: State: $CS \in T_D$ Set of miners,
2: $C_i = (A_i, F_i)$. $A$, local blockchain of node $F_i$ is a DAG of block $A_i$ and pointer $F_i$
3: $b$ denotes Block records
4: $F_i$: parent, preceding node of $b$
5: $m$, miners who mines and sign block $b$
6: $s$, new block added to the network
7: $duration$, each step takes time to validate and added
8: function propose
9: while True do
10: step #: duration, #clock time
11: if $k \in CS_i \land step \ mod(CS_i) = k$ then
12: $b$:parent $\leftarrow b(c_i, l) \rightarrow last block
13: $b$:CS $\leftarrow F_i$
14: $b$:step $\leftarrow step
15: $C_i = (A_i \cup b, F_i \cup b.parent)
16: disseminate ($C_i$)
17: sleep(duration)
18: end if
19: end while
20: end Function
21: function Score($A_i, F_i$)
22: return UNIT256-MAX * height($A_i, F_i$) - step-num($A_i, F_i$)
23: end Function
24: function Deliver($A_i, F_i$)
25: if Score($A_i, F_i$) $> Score(A_i, F_i)$ then
26: Score($A_i, F_i$) $\leftarrow Score(A_i, F_i)$
27: end if
28: end Function
29: function isDecide($b$
30: $V = \{b: i \in A_i \land b.step > b.step\} \rightarrow |V| * 2 > |CS_i|$
31: end Function

Step-2: After successfully receiving the message $M_1$, timestamp gets validated $TSP_T^1$ by CS $| TSP_T^1 - TSP_T^1 | < \Delta T$, if timestamp validated successfully, then CS checks certificate by $CRT_{UA} = B = PB_{TA} + h(PB_{UA} || PB_{UA})$ if it matches successful, then CS receives $PSID_{UA}$ with respect to $PPR_{UA}$ from the database and evaluate $L_2 = h(L_2 || PSID_{UA} || PPR_{UA} || CRT_{UA}^1)$.
to verify whether \( L_1 = L_3 \). If it matches successfully, then CS uses decryption \( L_3 = D_{PRCS}(L_3) \).

**Step-3:** Further, CS picks a unique random number \( CS_{UAV} \in \mathbb{Z}_n \) and valid timestamp \( TSTP_2 \) and generates temporary identity \( PPR_{new}^{UAV} \) and evaluates \( CS_{UAV} = h(PSID_{UAV} || TSTP_2 || CS_{UAV} || TSTP_2) \) and uses encryption \( CS_{UAV} \) as \( CS_{UAV} = E_{PRCS}(CS_{UAV}) \). Furthermore, CS generates a session key \( SES_{UAV} = h(PPR_{new}^{UAV} || L_1 || CS_{UAV} || TSTP_2, || TSTP_2) \), \( PPR_{new}^{UAV} = PPR_{new}^{UAV} \oplus h(PSID_{CS} || PPR_{new}^{UAV} || TSTP_2, || TSTP_2) \), and \( CS_{UAV} = h(PPR_{new}^{UAV} || L_1 || CS_{UAV} || TSTP_2) \) and generates replay message \( M_2 = \{PPR_{new}^{UAV}, CS_{UAV}, CS_{UAV}, TSTP_2, PSID_{CS}, TSTP_2\} \) and transmit to \( UAV \) through open channel.

**Step-4:** After successfully receiving the reply message \( M_2 \) from CS, timestamp gets validated \( TSTP_2 \) by \( UAV \), i.e., \( |TSTP_2 - TSTP_2| < \Delta T \) is denoting valid timestamp or invalid timestamp. If matches successfully, then \( UAV \) checks for certificate using \( CRC \). \( PSID_{CS} || TSTP_2 \) and verify \( CS_{UAV} = CS_{UAV} \) then \( UAV \) uses decryption \( CS_{UAV} \) to \( CS_{UAV} = D_{PRCS}(CS_{UAV}) \). \( CS_{UAV} \) evaluates \( CS_{UAV} = h(PPR_{new}^{UAV} || CS_{UAV} || CRC || PSID_{CS} || TSTP_2) \) and verify \( CS_{UAV} = CS_{UAV} \) then \( UAV \) uses encryption \( PPR_{new}^{UAV} \). Further, \( UAV \) chooses valid timestamp \( TSTP_3 \) and generates a session key \( SES_{UAV} = h(PPR_{new}^{UAV} || L_1 || CS_{UAV} || TSTP_2) \) and dissemi\(nates to CS. Furthermore, UAV chooses valid timestamp \( TSTP_3 \) and generates session key \( SES_{UAV} = h(PPR_{new}^{UAV} || L_1 || CS_{UAV} || TSTP_2) \) and dissemi\(nates to CS. Finally, UAV generates an acknowledgment message \( M_3 = \{SES_{UAV}, TSTP_3\} \) and transmit to CS through open channel.

**Step-5:** After successfully receiving the acknowledgment message \( M_3 \) timestamp gets validated \( TSTP_3 \) by CS, i.e., \( |TSTP_3 - TSTP_3| < \Delta T \) is denoting valid timestamp or invalid timestamp. Further, CS checks \( SES_{UAV} = h(SE_{CS} || TSTP_2) \). If it is valid, then CS makes the establishment of session key \( SES_{UAV} = h(SE_{CS} || TSTP_3) \) to \( UAV \). Finally, CS makes changes \( PPR_{new}^{UAV} \) and \( PPR_{new}^{UAV} \) in the database.

**Consensus Phase**

In this phase, the block verification and creation in the P2P network is discussed. IoT devices are authorized to create transactions in the network after a successful session verification, and after successful verification of transactions using consensus mechanism a block is created by the miners (CS) and added into the network. The block \( C_i \) consists of two parameters, such as local blockchain of peer, i.e., \( A_i \) and block pointer, i.e., \( F_i \). The blocks are created after voting process, when more than 50% of voting is done by the peer nodes (CS). The verification and block creations are illustrated in the Algorithm 1.

**SECURITY ANALYSIS**

This phase describes security analysis of the proposed model. It includes the formal verification to prevent various attacks. The detailed security analysis is summarized in the following.

1) **Impersonation attack:** An attacker can generate temporary identity \( ITD \), pseudoidentity \( PSID_{ITD} \), and partial private key \( PPR_{ITD} \) to perform operation as a legitimate user. Further, timestamp \( TSTP_{ITD} \) can be generated for access permissions in the framework. However, session-based approach is applied to verify the unique identity of the devices \( ITD \). If all credentials are matched then access permissions granted, else connections terminated immediately. Thus, this approach prevents from impersonation attack.

2) **Insider attack:** The attackers are privileged (can be insider) and can sniff all the credential, such as IoT device identification \( ID \), pseudoidentity \( PSID_{ID} \), and timestamp \( TSTP_{ID} \). However, access can only be permitted after session-based verification of the entities. Thus, the approach does not allow access without permissions and prevents from insider attack.

3) **MITM and replay attack:** The attacker may get the details of the IoT devices from insecure channel and communications, such as \( ITD \) and timestamp \( TSTP_{ITD} \) of registration. The attackers may send the details to the UAVs for making certain operations. However, the UAVs checks for the timestamp and verifies the session. However, it is difficult to compute all the credential at certain interval of time from id generation to session verification. Performing all the required evaluation at perfect time edge is difficult. Thus, the attacker cannot perform the MITM and replay attack.

**PERFORMANCE ANALYSIS**

The experiments were executed on a Tyronne PC with two 2.20 GHz Intel CPUs and 128 GB of RAM. The IDS was developed using the TensorFlow package Keras. The Ethereum Rinkeby network was used to create the SC module. The CSAE layer was trained for 10 epochs using two hidden layers containing (64, 32) neurons, whereas GRU used two hidden layers with (64, 32) neurons, MLP used two hidden layers with (16, 8) and last layer has softmax classifier, Adam optimizer, ReLU activation, categorical-cross entropy as loss function, and 100 batch size for 10 epochs. The intrusion performance
was evaluated using the CICIDS-2017 dataset, which contains 390,222 attack and 2,035,505 normal instances. We preprocessed both datasets using the techniques outlined in Kumar et al.’s work with 70% training and 30% testing sets. This article employs a variety of performance metrics, such as, accuracy, detection rate, precision score, F1 score, and false alarm rate. However, to calculate these values, various parameters are used, such as, true positive (a), true negative (g), false positive (b), and false negative (d) determines correct classified attack instances, correct classified normal instances, normal observations classified as attack instances, and attack observations classified as normal instances, respectively. Accuracy (AC): The percentage of all correctly identified regular and attack instances is determined by AC that is; $AC = \frac{a + g}{a + b + d + g}$. Detection rate (DR): The appropriate proportion of attacks identified is determined by DR or recall (RC) that is, $DR = \frac{a}{a + d}$. Precision (PR): PR is calculated by dividing the number of attack behaviors observed by the total number of observations classified as an attack, $PR = \frac{a}{a + b}$. F1 Score: The weighted average of PR and DR/RC is determined by the F1 score, that is, $F1 = 2 \cdot \frac{PR \cdot DR}{PR + DR}$. False alarm rate (FAR): FAR identifies cases of attack that were incorrectly identified, that is, $FAR = \frac{b}{b + g}$.

Deep Learning Module Analysis
The performance of DL approach is evaluated using a variety of assessment metrics. The proposed CSAE technique’s accuracy versus loss is depicted in Figure 2. Despite being employed to extract low-dimensional features, the CSAE method learned the dataset effectively, with a validation accuracy of 87.92% and a validation loss of 0.0546%. Table 1 shows the classwise performance of the proposed model. It is observed that the values for PR, DR, and F1 are high, and FAR is close to 0%. We have also compared the DR of the proposed model with other baseline techniques in Table 2. It is seen that the proposed model outperformed these baselines for the majority of the vectors present in the dataset. Finally, as shown in Figure 3, the overall performance of proposed model is compared with traditional approach. It is seen that the proposed model has achieved higher values and outperformed RF, DT, and NB.

SC Module Analysis
The SC study shown in Figures 4 and 5 evaluate transaction upload time, block mining, block formation, and off-chain storage. The Figure 4(a), shows the upload

**TABLE 1.** Classwise performance analysis of proposed IDS.

| Parameters | BENIGN | DoS hulk | DDoS | PortScan | DoS GoldenEye | FTPPatator | DoS slowloris | DoS slowhtptes | SSHPatator | Bot | Web attack |
|------------|--------|----------|------|----------|---------------|------------|---------------|---------------|------------|-----|------------|
| PR         | 99.53  | 99.91    | 98.84| 84.84    | 78.68         | 98.25      | 97.58         | 96.29         | 98.99      | 95.77| 100.00     |
| DR         | 98.62  | 96.88    | 98.98| 88.56    | 98.57         | 99.65      | 97.89         | 98.24         | 96.99      | 43.28| 08.88      |
| F1         | 99.87  | 97.77    | 99.19| 88.26    | 97.17         | 96.23      | 95.88         | 98.97         | 97.88      | 60.44| 12.12      |
| FAR        | 0.050588 | 0.000338 | 0.001022 | 0.005687 | 0.000038 | 0.000018 | 0.000024 | 0.000084 | 0.000001 | 0.000015 | 0.00 |        |

**TABLE 2.** Comparison of multivector DR (%) with some commonly used baseline techniques.

| Techniques | BENIGN | DoS hulk | DDoS | PortScan | DoS GoldenEye | FTPPatator | DoS slowloris | DoS slowhtptes | SSHPatator | Bot | Web attack |
|------------|--------|----------|------|----------|---------------|------------|---------------|---------------|------------|-----|------------|
| RF         | 100.00 | 95.00    | 100.00| 97.00    | 50.00         | 72.00      | 0.00          | 55.00         | 0.00       | 0.00 | 0.00       |
| DT         | 100.00 | 90.00    | 99.00| 97.00    | 66.00         | 99.00      | 35.00         | 0.00          | 97.00      | 0.00 | 0.00       |
| NB         | 55.00  | 89.00    | 98.00| 50.00    | 99.00         | 100.00     | 60.00         | 77.00         | 97.00      | 76.00| 08.00      |
| Proposed model | 98.62 | 96.88    | 98.98| 88.56    | 98.57         | 99.65      | 97.89         | 98.24         | 96.99      | 43.28| 08.88 |
time over IPFS storage layer for different transactions. Figures 4(b) and 5(a), show block mining and block creation time with different number of nodes (NS) and transactions (Tx). It can be observed that the execution time linearly increasing as peers increasing in the network. The Figure 5(b), shows off-chain storage size in KB over IPFS for varying number of Tx. It can be observed that, storage size is increasing as the number of Tx increasing in the network.

CONCLUSION

In this article, we designed a DL and SC-assisted secure data sharing framework for IoT-based IA. Specifically, a novel DL module was designed that combined CSAE with GRU, MLPs, and softmax classifier to detect intrusion in the network. In SC module, first authentication, key management scheme was proposed. The, normal transactions received from DL-based IDS were mined by CS using SC-based PoA (aura algorithm) consensus technique. The validated transactions were added to the IPFS-based storage layer and returned cryptographic hash was stored on blockchain ledger. Experimental analysis of DL and SC module proves the effectiveness of the proposed framework. The future work includes the performance evaluation in terms of scalability and latency using different real-world datasets.

REFERENCES

1. R. Kumar, P. Kumar, R. Tripathi, G. P. Gupta, T. R. Gadekalu, and G. Srivastava, “SP2F: A secured privacy-preserving framework for smart agricultural unmanned aerial vehicles,” Comput. Netw., vol. 187, 2021, Art. no. 107819.
2. L. Cao, “Decentralized AI: Edge intelligence and smart blockchain, metaverse, web3, and DeSci,” IEEE Intell. Syst., vol. 37, no. 3, pp. 6–19, May/Jun. 2022.
3. Y. Xing et al., “A survey on smart agriculture: Development modes, technologies, and security and privacy challenges,” IEEE/CAA J. Automatica Sinica, vol. 8, no. 2, pp. 273–302, Feb. 2021.
4. P. Kumar, R. Tripathi, and G. P. Gupta, “P2IDF: A privacy-preserving based intrusion detection framework for software defined Internet of Things-fog (sdiot-fog),” in Proc. Int. Conf. Distrib. Comput. Netw., 2021, pp. 37–42, doi: 10.1145/3427477.3429989.
5. M. P. Singh, G. S. Aujla, and R. S. Bali, “Blockchain for the Internet of drones: Applications, challenges, and future directions,” IEEE Internet Things Mag., vol. 4, no. 4, pp. 47–53, Dec. 2021.
6. R. Kumar, P. Kumar, A. Kumar, A. A. Franklin, and A. Jolfaei, “Blockchain and deep learning for cyber threat-hunting in software-defined industrial IoT,” in Proc. IEEE Int. Conf. Commun. Workshops, 2022, pp. 776–781.
7. B. Bera, A. Vangala, A. K. Das, P. Lorenz, and M. K. Khan, “Private blockchain-envisioned drones-assisted authentication scheme in IoT-enabled agricultural environment,” Comput. Standards Interfaces, vol. 80, 2022, Art. no. 103567.
8. A. Vangala, A. K. Sutrata, A. K. Das, and M. Jo, “Smart contract-based blockchain-envisioned authentication scheme for smart farming,” IEEE Internet Things J., vol. 8, no. 13, pp. 10792–10806, 2021.

9. M. U. Rahman, F. Biaardi, and L. Ricci, “Blockchain smart contract for scalable data sharing in IoT: A case study of smart agriculture,” in Proc. IEEE Glob. Conf. Artif. Intell. Internet Things, 2020, pp. 1–7.

10. A. Vangala, A. K. Das, N. Kumar, and M. Alazab, “Smart secure sensing for IoT-based agriculture: Blockchain perspective,” IEEE Sensors J., vol. 21, no. 16, pp. 17591–17607, Aug. 2021.

11. P. Kumar, R. Kumar, G. P. Gupta, R. Tripathi, and G. Srivastava, “P2TIF: A blockchain and deep learning framework for privacy-preserved threat intelligence in industrial IoT,” IEEE Trans. Ind. Inform., vol. 18, no. 9, pp. 6358–6367, Sep. 2022.

12. L. Liu, M. Du, and X. Ma, “Blockchain-based fair and secure electronic double auction protocol,” IEEE Intell. Syst., vol. 35, no. 3, pp. 31–40, May/Jun. 2020.

13. H.-T. Wu and C.-W. Tsai, “An intelligent agriculture network security system based on private blockchains,” J. Commun. Netw., vol. 21, no. 5, pp. 503–508, 2019.

14. D. Dolev and A. Yao, “On the security of public key protocols,” IEEE Trans. Inf. Theory, vol. 29, no. 2, pp. 198–208, Mar. 1983.

15. R. Canetti and H. Krawczyk, “Universally composable notions of key exchange and secure channels,” in Proc. Int. Conf. Theory Appl. Cryptogr. Techn., 2002, pp. 337–351.

16. P. Kumar, R. Kumar, G. P. Gupta, and R. Tripathi, “BDEdge: Blockchain and deep-learning for secure edge-envisioned green CAVs,” IEEE Trans. Green Commun. Netw., vol. 6, no. 3, pp. 1330–1339, Sep. 2022.

17. I. Sharafaldin, “CIC-IDS2017 datasets.” Accessed: Mar. 15, 2019. [Online]. Available: http://205.174.165.80/CICDataset/CIC-IDS-2017/Dataset/

18. P. Kumaret al., “PPSF: A privacy-preserving and secure framework using blockchain-based machine-learning for IoT-driven smart cities,” IEEE Trans. Netw. Sci. Eng., vol. 8, no. 3, pp. 2326–2341, Jul.–Sep. 2021.

RANDHIR KUMAR is currently working as a postdoctoral researcher with the Department of Electrical Engineering, Indian Institute of Technology Hyderabad, Hyderabad, 502285, India. His research interest includes cryptographic techniques, information security, blockchain technology, and web mining. Kumar received his Ph.D. degree in information technology from the National Institute of Technology Raipur, Raipur, India. He is a member of the IEEE. Contact him at randhir.honeywell@ieee.org.

SAHIL GARG is currently an assistant professor of networking and cyber security with the College of Science and Engineering, Flinders University, Adelaide, SA, 5042, Australia. His research focuses on cyber-physical systems security. He is a senior member of the IEEE and a distinguished speaker of the ACM. Contact him at alireza.jolfaei@flinders.edu.au.

PRABHAT KUMAR is currently a postdoctoral researcher with the Department of Software Engineering, LUT School of Engineering Science, LUT University, 53850, Lappeenranta, Finland. His research interests include machine learning, deep learning, federated learning, Big Data analytics, cybersecurity, blockchain, cloud computing, Internet of Things, and software defined networking. Kumar received his Ph.D. degree in information technology from the National Institute of Technology Raipur, Raipur, India, under the prestigious fellowship of Ministry of Human Resource and Development funded by the Government of India in 2022. He is a member of the IEEE. Contact him at prabhat.kumar@lut.fi.

AHAMED ALJUHANI is currently an assistant professor and the chair of the Department of Computer Engineering, College of Computing and Information Technology, University of Tabuk, Tabuk, Saudi Arabia. His research interests include information security, network security and privacy, secure system design, and system development. Aljuhani received his Ph.D. degree in computer science/information security track from The Catholic University of America, Washington, DC, USA. Contact him at a.aljuhani@ut.edu.sa.

A. K. M. NAJMUL ISLAM is an associate professor of software engineering with LUT University, 53850, Lappeenranta, Finland. He is an adjunct professor of information systems with Tampere University, Tampere, Finland. Islam received his Ph.D. degree from the University of Turku, Turku, Finland. Contact him at najmul.islam@lut.fi.

ALIREZA JOLFAEI is an associate professor of networking and cyber security with the College of Science and Engineering, Flinders University, Adelaide, SA, 5042, Australia. His research focuses on cyber-physical systems security. He is a senior member of the IEEE and a distinguished speaker of the ACM. Contact him at alireza.jolfaei@flinders.edu.au.

SAHIL GARG is currently with the Resilient Machine Learning Institute, Montreal, QC, H3C 1K3, Canada. His research interests include machine learning, Big Data analytics, knowledge discovery, cloud computing, Internet of Things, software defined networking, vehicular ad-hoc networks. Garg received the Ph.D. degree from the Thapar Institute of Engineering and Technology, Patiala, India. He is a member of the IEEE. Contact him at garg.sahil1990@gmail.com.