The Use of Blockchain to Support Distributed AI Implementation in IoT Systems

Subhi Alrubei, *University of Sheffield,* Edward Ball, *University of Sheffield,* and Jonathan Rigelsford, *University of Sheffield*

***Abstract*—This paper presents a distributed and decentralized architecture for the implementation of Distributed Artificial Intelligence (DAI) using hardware platforms provided by the Internet of Things (IoT). A trained DAI system has been implemented over the IoT, where each IoT device acts as one or more of the neurons within the DAI layers. This is accomplished through the utilization of decentralized, self-managed blockchain technologies that allow trusted interactions and information to be exchanged between distributed neurons. The platform was built and customized to be used within the IoT system, and it is capable of handling DAI-related tasks. A new consensus mechanism based on Proof of Authority (PoA) and Proof of Work (PoW) has been designed and implemented, along with bespoke block and transaction formats. The proposed architecture was analyzed, implemented, and tested using a dedicated testbed with low-cost IoT devices. A quantitative measurement and performance evaluation of the system based on a real-world IoT application was conducted. The implemented DAI is found to have an accuracy of 92%-98%, with an energy cost of 0.12 joules (J) when utilizing a Raspberry Pi to run one neuron. The measured hash per joule (h/J) when using a Raspberry Pi for mining is 13.8Kh/J compared to 54Kh/J using an ESP32. The results showed that it is feasible to implement a DAI system utilizing the IoT hardware platform while maintaining the system's accuracy. The integration of the blockchain has added an element of security and trust to the data and the interaction between system components.**

***Index Terms*—Distributed Artificial Intelligence (DAI), Blockchain, IoT, Consensus Mechanisms, Performance Evaluation.**

I. Introduction

THE Internet of Things (IoT) is a major source of big data, which is generated from the huge number of smart devices connected to the internet. This data provides users with the ability to generate valuable information and knowledge. One promising technology in this context is artificial intelligence (AI) that can be utilized within the IoT to provide an intelligent means of processing data to produce valuable insights and predictions, and to enhance the process of decision-making automation. Depending on the application and its requirements, the processing of this data by an AI system may be in the cloud layer, in an edge layer, and/or in the sensing layer (the smart devices). However, such intelligent implementation of AI into the IoT realm faces challenges, especially the implementation into the edge and sensing layers. In particular, these devices often lack adequate computational resources [1].

A distributed system may be a collection of autonomous nodes communicating with each other over a communication channel. It has the ability to run software in parallel among these nodes closer to where computing is needed [2]. This attractive ability of the distributed approach helps process data in near real-time and reduces the communication overhead needed to transfer data from end devices to a central entity, such as cloud computing. To realize the benefits of true parallelism and distribution offered by AI in a fully distributed computing system, a scalable hardware platform is required. The distributed nature of the IoT, where thousands of smart devices are available and can communicate with each other, offers such a platform [3].

Distributed computing requires each node in the system to carry out a parallel computation every round [4]. The number of rounds needed to complete the task and the number of messages exchanged between the nodes will result in a complex and undesirable situation. To avoid this complexity and reduce latency by providing the AI system with historical data to facilitate future decisions, the IoT system requires implementation in an architecture that combines both decentralization and distribution. Blockchain technology is an ideal solution that enables distributed computing and achieves data storage in a large number of devices over a wide area network.

The integration of blockchain into IoT can provide reliable control of the IoT network's ability to distribute computation over a large number of devices, improve overall security by enhancing data integrity, ensure accountability, and provide a way to implement better access control [5]. It also allows the AI system to use trusted data for analyses and forecasts while utilizing the available IoT hardware to coordinate the execution of tasks in parallel, using a fully distributed approach.

The contributions of this paper are summarized as follows:

- A novel, and secure blockchain architecture for supporting DAI on low-power and low-cost IoT devices.
- Practical implementation of DAI using scalable and distributed IoT hardware platform.
- Prediction and measurements of DAI using blockchain in IoT devices with a performance analysis that includes, accuracy, energy consumption, and overall system latency utilizing data from trusted and robust platforms.
- A blockchain protocol that includes a new consensus mechanism, and transaction and block formats that help

The authors are with the Department of Electronic and Electrical Engineering, The University of Sheffield, Sheffield,UK, e-mail: {salrubei1, e.a.ball, j.m.rigelsford}@sheffield.ac.uk.

nodes handle DAI-based transactions and prediction requests.

The rest of the paper is organized as follows. Section II presents the related work, followed by the proposed architecture design in Section III. Section IV presents the system analysis; this is followed by the system implementation and deployment in Section V. Details of the results are in section VI, and finally, the paper is concluded in Section VII.

II. Related Work

Blockchain platforms are usually built around one of two main approaches: *On-Chain and Off-Chain*. On-chain is a transparent approach where transactions are executed and stored on the public blockchain. It is implemented by most of the well-known blockchain platforms, such as Bitcoin [6] and Ethereum [7]. This means all transactions and communications between the nodes are executed on the blockchain, and each transaction is stored on the chain. While this approach can result in increased latency, it provides a trusted method that makes AI predictions traceable and easy to understand, allowing users and organizations to determine how and why any decisions were made.

Off-chain, as described by [8], is intended to move some of the computational efforts from the main chain to an off-chain platform. The transactions are executed by the nodes off-chain, and only the final outcome is committed to the main chain. There are different implementations of the off-chain methods, e.g., Bitcoin's Lightning network [9] and Plasma of Ethereum [10]. While this approach reduces latency, it does not provide the complete trusted and transparent process intended by blockchain nor does it allow for full traceability. It may not provide validations for all transactions, which could compromise the system's security.

In terms of combining blockchain, AI, and IoT, [11] proposed a platform named NeuRoNt based on the Ethereum blockchain and an edge layer hosted a smart contract. The platform consists of multiple agents powered by smart contracts that can solve complex problems. Ethereum and smart contract-based mobile edge sharing systems were proposed by [12]. AI was used for data processing, and blockchain and cloud platforms were used to facilitate the sharing of services in IoT-enabled smart cities. ModelChain, proposed by [13], aims to maintain the privacy of health records while allowing multiple institutions to train the medical health prediction framework using blockchain and machine learning.

In [14] the BlockDeepNet framework was proposed, which combined the implementation of deep learning, blockchain, and smart contracts for data analyses in IoT. Blockchain was used to securely exchange local and global updates of the deep learning model. The work by [15] proposed the DeepCoin framework for smart grids based on blockchain and deep learning. The deep learning used is an intrusion detection systems (IDS) scheme for detecting fraudulent transactions and attacks in the blockchain-based network. Another framework proposed by [16] is based on deep learning, SDN, and blockchain for enabling high-performance and cost-effective computing resources for smart city applications. Nevertheless, both [15] and [16] frameworks suffer from centralization issues.

The authors of [17] introduced a distributed AI system enabled by multiple layers of fog networking for smart shopping advertisements. They offloaded some of the AI analyses and data processing to fog layers while utilizing cloud platforms to perform the main analyses and choose advertisements based on age and gender. The works in [18]–[20] provided a framework and a simulation study to deploy a distributed Hopfield neural network through the use of a Wireless Sensor Network (WSN) as a hardware platform. While this work provided a robust architecture for utilizing an IoT system for the implementation of distributed AI, validation in the form of practical deployment and a real-world use case is needed. The authors of [21] have proposed distributed deep neural networks (DDNNs) architecture, which consists of the cloud, the edge, and IoT end devices. Another work by [22] proposed a distributed machine

TABLE I
Comparison Between this Proposed Architecture and other Related Works

Paper	Main Contribution	Technologies Utilized	Distributed and Decentralized Implementation	Prototype and System Deployment	Measurement and Performance Analyses	Distributed Neurons Implementation on IoT devices
[12]	Blockchain and smart contract based framework for sharing economy in smart cities	Fog, IoT, AI, Blockchain and cloud	No, blockchain for distributed data sharing but relies on central AI engine	Yes, using smartphone, private blockchain and Amazon AWS	Yes, system's latency	No
[13]	Blockchain-based framework to improve the robustness and security of distributed healthcare predictive modeling	Blockchain and Machine learning	No	No	No	No
[14]	Blockchain based Deep Learning collaborative algorithms for IoT.	Blockchain, Smart Contract, Edge, IoT, and AI	Partially, IoT devices rely on central edge server	Yes, using edge, clustering, access point, Ethereum blockchain, and smart contracts	Yes, Accuracy, Latency, and memory and CPU Usage	No
[15]	Framework for smart grids based on deep learning (as IDS) and blockchain	Blockchain and Recurrent neural Networks (RNNs)	No, uses central RNN	Yes, using private blockchain	Yes , IDS detection rate and accuracy	No
[16]	Framework for enabling high performance and cost-effective computing resources for smart city applications.	Blockchain, Deep Learning, and IoT	No, central cloud for AI	Partial implementation, Corda and CordaDApp to simulate blockchain nodes setups	Yes, Latency and Scalability	No
[17]	AI-based smart shopping advertisements.	AI, Fog, and cloud computing	Partially, still relies on central cloud for AI implementation	Yes, using OM2M IoT platform and central cloud	Yes, latency and Data transfer	No
[18]–[20]	Framework to deploy a distributed Hopfield neural network using WSN	AI and WSN	Distributed AI	No, simulation only	Simulated results only	Yes
[21]	Distributed deep neural networks (DDNNs) architecture which consists of the cloud, the edge, and IoT enddevices.	AI, Cloud, Edge, and IoT	Partially, needs aggregator and cloud for some processing	Yes, using six IoT devices and a cloud	Yes , measured accuracy	No
[22]	Machine learning (ML) architecture called Parallel Channel Artificial Neural Networks (PCANN) for image recognition on IoT devices	AI and IoT	Partially, IoT devices rely on a controller	No, simulation only	Yes, classification accuracy	No
This Paper	A novel and secure blockchain architecture for supporting fully distributed AI on low-power and low-cost IoT devices	Blockchain, AI, Edge, and IoT-end devices	Decentralized and fully distributed AI based on blockchain	Yes, using customized-built blockchain that includes IoT-centric consensus mechanism and DAI over IoT devices (23 Raspberry Pis and six ESP32)	Yes, latency, accuracy, devices hash power, and energy consumption	Yes

learning (ML) architecture called Parallel-Channel Artificial Neural Networks (PCANN) for image recognition tasks on IoT devices. It works by dividing the ML model into small models distributed on these devices, with one designated as a controller to control the process.

The work by [23] proposed a Federated Learning (FL) system for helping manufacturers develop smart home systems. It uses consumer's data for training an ML model to assesses home appliance manufacturers. Blockchain is used to ensure accountability within the system, especially when a model performs an update operation. The authors of [24] proposed BAFFLE, a blockchain-based FL environment that leverages smart contracts for coordinating the model aggregation.

Although each of these proposed frameworks and architectures provides different advantages, none has yet exploited the potential provided by blockchain technology for supporting and facilitating the implementation of AI in a decentralized and fully distributed approach through IoT systems. Table I shows a comparison between the architecture proposed in this paper and related research.

III. Proposed Architecture Design

Artificial neural networks are a branch of artificial intelligence, and a multilayer perceptron (MLP) is a type of neural network. An MLP is usually made up of at least one input layer, one hidden layer, and an output layer [25]. The number of neurons in the input layer is equal to the features of the dataset, while the number in the hidden and output layers can vary depending on multiple factors, such as training and the type of implementation and problem at hand (e.g., regression or classification). Layers are fully connected, and neurons communicate their values to each other using synaptic connections represented by weights [25].

MLP is based on supervised learning techniques, and one of the learning algorithms that is used to train MLPs is the back-propagation algorithm. In this algorithm, the first step is to initialize network weights to random values, then present the first input values from a training dataset to the network. This data is propagated through the network, and each node produces an output that is a function of the sum of the input values to the node and its weights, modified by an activation function, such as the sigmoid function $(S(x) = 1/1 + e^{-x})$. This is done by each neuron in the network until a final output is produced, then an error is calculated that compares the actual output to the target output. This error propagates back through the network and weights are adjusted to minimize the overall error. These steps are repeated until the overall error is satisfactorily small.

A. *Design Overview*

Distributed Artificial Intelligence (DAI) is an approach to exploit the resources provided by large-scale distributed computing. The aim of this design is to develop a blockchain platform that can support the implementation of a DAI that utilizes the capabilities of IoT systems. The general workflow of the proposed architecture is presented in Fig. 1. In this design, with the integration of a blockchain that is trustworthy,

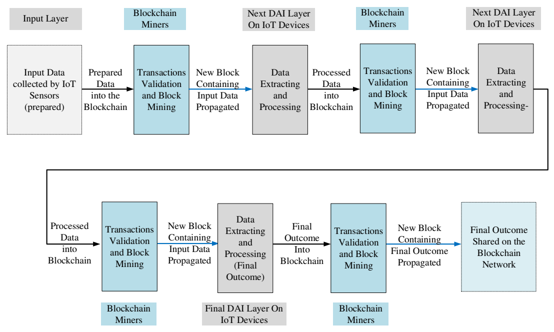

Input Layer
Blockchain Miners
Next DAI Layer On IoT Devices
Blockchain Miners
Next DAI Layer On IoT Devices
Input Data collected by IoT Sensors (prepared)
Prepared Data into the Blockchain
Transactions Validation and Block Mining
New Block Containing Input Data Propagated
Data Extracting and Processing
Processed Data into Blockchain
Transactions Validation and Block Mining
New Block Containing Input Data Propagated
Data Extracting and Processing-
Processed Data into Blockchain
Transactions Validation and Block Mining
New Block Containing Input Data Propagated
Data Extracting and Processing (Final Outcome)
Final Outcome Into Blockchain
Transactions Validation and Block Mining
New Block Containing Final Outcome Propagated
Final Outcome Shared on the Blockchain Network
Blockchain Miners
Final DAI Layer On IoT Devices
Blockchain Miners

Fig. 1. Proposed Architecture - General Workflow.

self-managed, and self-regulated, the DAI engine will have a platform that provides a secure way to handle and protect data, yielding a better decision-making process. Regardless of the type of DAI implemented, the data flow will be the same. Devices and nodes will also be able to interact and communicate with each other in a secure way that will ensure efficient processing and flow of data through the system's different layers.

While the architecture has the ability to support any form of DAI implementation, for this paper the implemented DAI is in the form of Distributed Multilayer Perceptrons (DMLP). The main idea is to build an architecture that supports DAI implementation in general, and DMLP was selected as an example because it will allow deployment up to the neuron level, allowing for more focus on the evaluation of the blockchain aspects of this architecture. For DMLP, the exploitation of the resources provided by large-scale distributed IoT systems can be achieved by hosting one or more neurons on an IoT device. Each device would then act independently and utilize the blockchain platform to ensure the integrity of the processed data and its transfer to other devices. Processed data will then flow from one layer to another until the desired outcome is achieved. A trained DMLP will be implemented and tested over this blockchain platform.

In this design, the on-chain approach has been chosen where all transactions are executed, validated, and committed on the main chain. This method allows the blockchain platform to record all the AI transactions and variables that are used by the trained AI engine to make decisions. As the AI engine becomes smarter as a result of continuous training and is able to process large amounts of data, it becomes more difficult for scientists to understand how the AI systems came to specific conclusions and decisions. However, through the implementation of AI on blockchain platforms, they will have immutable records of all the data and variables used by AI for its decision-making processes. This will provide data scientists with the ability to easily audit and trace the entire process.

B. *System Components*

The system proposed in this designed as a decentralized and fully distributed architecture, which provides added value to the computational ability of an AI system through the utilization of a scalable IoT-hardware platform. This architecture

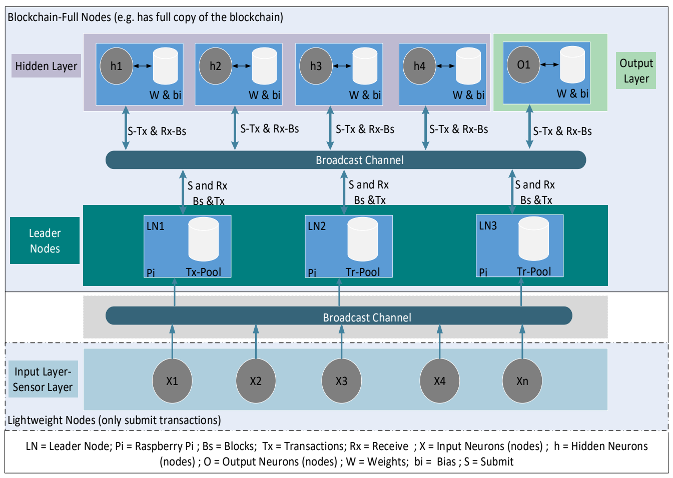

Blockchain-Full Nodes (e.g. has full copy of the blockchain)
Hidden Layer
h1
h2
h3
h4
O1
W & bi
Output Layer
S-Tx & Rx-Bs
Broadcast Channel
S and Rx Bs &Tx
Leader Nodes
LN1
LN2
LN3
Pi
Tr-Pool
Broadcast Channel
Input Layer-Sensor Layer
X1
X2
X3
X4
Xn
Lightweight Nodes (only submit transactions)
LN = Leader Node; Pi = Raspberry Pi ; Bs = Blocks; Tx = Transactions; Rx = Receive ; X = Input Neurons (nodes) ; h = Hidden Neurons (nodes) ; O = Output Neurons (nodes) ; W = Weights; bi = Bias ; S = Submit

Fig. 2. Proposed Architecture - System Components.

takes advantage of the IoT sensing capabilities, blockchain immutability and trustworthiness, and the capabilities of DAI intelligence. This results in the design of a computationally intelligent, scalable, distributed, and decentralized DMLP architecture based on the blockchain (see Fig.2). The architecture consists of five main components.

- *Worker Nodes* (WN): These are low-cost and low-power devices, such as the Raspberry Pi and ESP32 microcontrollers. When a node joins the network for the first time, it will join as a worker node. All worker nodes are able to participate in the mining process by carrying out mining tasks on the network. By submitting correct answers to any task and behaving honestly on the network, these nodes can build up their honesty. Once they achieve the required honesty level and have enough resources, they can be promoted to the Leader Node category. Leader nodes will also perform mining tasks as required to maintain their honesty and ensure sufficient computation power is available in the network.
- *Leader Nodes* (LN): These are low-cost devices, such as the Raspberry Pi, that form the heart of the blockchain network by acting as miners and validators. They store a full copy of the blockchain locally that is then synced with the latest block in the network. They have a high enough honesty level to act as coordinators of the mining process, validate the work of workers, sign and propagate blocks, and validate each other's newly propagated blocks. Leader nodes are the only nodes with an honesty level that allows them to run a blockchain network and handle all the tasks related to the DMLP engine in the network, which adds further trust to the handling of the data and prediction operations.
- *Input Layer / Sensing Layer*: This is where many small, low-power sensors can be used for monitoring and data collection. These sensors can be directly connected to other devices from the WNs or LNs. They can sense and collect data, which will then be locally prepared and submitted to the network for processing. These sensors are part of the blockchain network in the form of lightweight nodes that can submit transactions through the nodes they are connected to. They are the data feeder to the DMLP engine and the first layer in it.
- *Hidden Layer*: These are low-cost devices, that will act individually as one or more neurons in the hidden layer. In this proposed architecture, there can be more than one hidden layer, but for the proof of concept implementation, only one hidden layer was deployed where each device acts as one neuron.
- *Output layer*: This is the final layer of the DMLP, and it also consists of low-cost devices. Each device can act as one or more neurons of the output layer. This design allows for the implementation of multiple output layers, where each one implements different activation functions and produces its own final predictions. This enhances the forecasting ability and allows for better performance.

Another important part of this architecture is the propagation of transactions and blocks. Broadcast is the most commonly used network operation in blockchain networks, where nodes issue transactions and blocks by broadcasting them in the network. In this architecture, we assume that all transactions and blocks that are related to the DMLP are broadcast to secure channels that are only accessed by the honest and trusted leader nodes. However, depending on the IoT application under consideration, there might be a need for the integration of suitable encryption algorithms such as AES-128 [26] into the architecture to encrypt the DMLP-related data within transactions (i.e., transaction's payload).

C. Blockchain Platform

Designing a blockchain platform that is reliable, secure, and has acceptable latency is an essential part of this design. All transactions are executed on-chain, ensuring full traceability, and the validation of all transactions before processing enhances the system's security. A public blockchain platform, including its consensus mechanism, transactions, and block formats, was designed.

1) Consensus Mechanism: The main objective is to design a consensus mechanism that is suitable for non-financial IoT applications based on the concept of resources in exchange for services. We have considered different algorithms such as Proof of Stack (PoS), Proof of Stake (PoW) [6], and Byzantine Fault Tolerance (BFT) and its variation Proof of Authority (PoA) [27]. PoS have lately gained a lot of hype within the financial blockchain platforms. In PoS the trust is bounded on digital/currency assets, where the miners stake their coins to the network to be able to mine and validate blocks. However, it is vulnerable to Nothing-at-Stake attack [28] and Coin Age Accumulation attack [29], and constitutes a consensus disadvantage to nodes that do not have a high stake in the currency. This would result in rich nodes becoming richer.

PoA implemented by [30] is a lightweight consensus algorithm, offering two key advantages, it does not require much energy to mine a block, and it has lower latency as it does not require confirmation rounds as shown by [31] making it suitable for IoT implementations. However, one disadvantage is that it is intended for permission and private blockchain platforms, which is against the decentralized approach.

Proof of Work (PoW) is one of the most secure consensus algorithms for a public blockchain, yet it suffers from a long transaction confirmation time, approximately six blocks on Bitcoin. Many authors including [32]–[34] shown through their works it is possible to integrate PoW into IoT-blockchain applications.

In this paper, the security advantages provided by PoW have been realized while its long confirmation time was mitigated by combining it with PoA in a single consensus mechanism called Honesty-based Distributed Proof of Authority via Scalable Work (HDPoA). PoA depends on a number of trusted nodes called authorities, that are supposed to be honest (at least 51% of them) to mine and validate blocks. In classical PoA, leader or authorities nodes are assigned and authorized by the owner of the network. In the proposed HDPoA leaders nodes have to perform works and build their honesty level to earn the privilege of mining and validating blocks. The work can be in any form such as by carrying a small work of the PoW. The detailed implementation of HDPoA is provided in Section V.

2) Block and Transaction Formats: Another key aspect of this blockchain platform was the format of blocks and transactions. First, block headers were designed to allow for the inclusion of both the worker node's public key and the leader's signature. It was also adapted to distinguish between blocks that carry DMLP related transactions and those that do not. This will help the neurons in each layer deal with the blocks accordingly. The transaction format is an essential part and was designed to allow for different transaction types; the different fields of both block headers and transactions will be discussed in detail in section V.

D. Data Flow in the System

The data flow in the system is illustrated by Fig. 3. The data is propagated through the network from one layer to another as follows:

1) First, the sensor nodes in the sensing layer collect data and pass it to the first neuron in the input layers. Neurons then process this data locally and create and submit their input transactions.
2) Transactions arrive at the miners' transaction pool. The first leader node in the round-robin (e.g., LN1) then collects all transactions, validates them, and adds them into a new block.
3) Next, LN1 creates mining tasks and sends them to the assigned WNs.
4) Once a worker node accepts the work, it will conduct the mining until either the node finds a solution, it receives an abort message from LN1, or it completes the iteration through all of the nonces assigned to it. If it finds a solution, it will submit its finding to all of the leader nodes.
5) When LN1 receives the WN solution (i.e., the nonce) to the task, it will validate the work (see sub-subsection V-A-1). If valid, it will send an abort message to all WNs. It will then sign and propagate the new block to the network.
6) When neurons in the Hidden Layer receive the new block, they will open the block, extract the relevant input values, perform the required calculation, and apply the activation functions. Their hidden values are then included in a new transaction and propagated through the broadcast channel to the miner nodes.
7) The next leader node in the round-robin (e.g., LN2) validate transactions and adds them into a new block. It will then create mining tasks and send them to the assigned WNs.
8) Once a worker node accepts the work, it will start the mining process until it either finds a solution, it receives an abort message from LN2, or it completes the iteration. If it finds the target nonce, it will submit its finding to all of the leader nodes.
9) Once LN2 receives the WN solution, it will validate the work. If valid, it will send an abort message to all WNs. It will then sign and propagate the new block to the network.
10) After neurons in the Output Layer receive the new block, they will open the block, extract the relevant hidden values, perform the required calculation, and apply the activation functions. They then include their output value, which is the prediction of the DMLP expert engine, in a new transaction and propagates it through to the LNs.
11) The next leader node in the round-robin (e.g., LN3) validate transactions and adds them into a new block. It will then create mining tasks and send them to the assigned WNs.
12) Once a worker node accepts the work, it will start the mining process until it either finds the correct nonce, it receives an abort message from LN3, or it completes the iteration. If it finds the target nonce, it will submit its finding to all of the leader nodes.
13) Once LN3 receives the WN nonce, it will validate the work. If valid, it will send an abort message to all WNs. It will then create and sign the block, and propagate it to the network.

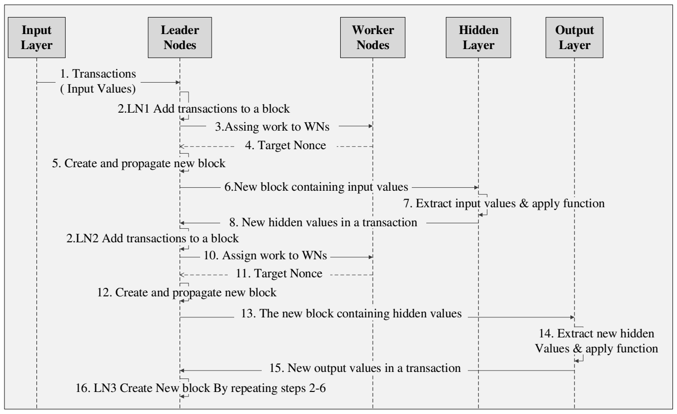

Input Layer
Leader Nodes
Worker Nodes
Hidden Layer
Output Layer
1. Transactions (Input Values)
2.LN1 Add transactions to a block
3.Assing work to WNs
4. Target Nonce
5. Create and propagate new block
6.New block containing input values
7. Extract input values & apply function
8. New hidden values in a transaction
2.LN2 Add transactions to a block
10. Assign work to WNs
11. Target Nonce
12. Create and propagate new block
13. The new block containing hidden values
14. Extract new hidden Values & apply function
15. New output values in a transaction
16. LN3 Create New block By repeating steps 2-6

Fig. 3. The Data Flow Between the System Different Components.

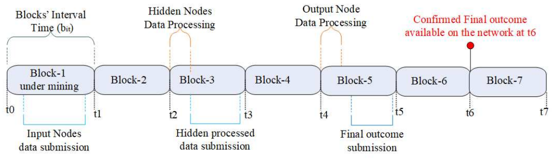
14) Finally, once the new block arrives and is validated in the network, the final outcome of the DMLP expert engine is now available to all interested nodes.

IV. System Analyses

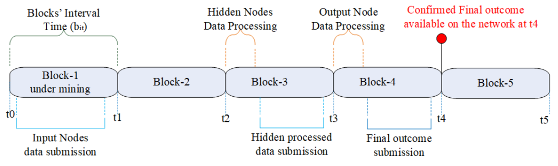
The system under consideration is based on blockchain technology where multiple nodes are connected in a peer-to-peer network via wireless links. Two types of nodes are on the blockchain network: leader nodes (LN) and worker nodes (WN). Only trusted LN can be utilized as a neuron in any of the DMLP layers, ensuring added security. The difficulty level D of the blockchain network is at its lowest (i.e $D = 1$) when the target hash value $h_v = 2^{232}$. The data traffic generated on the network is from two main processes; both are broadcast transmissions: propagation of transactions and propagation of blocks to all nodes. Different types of transactions can be generated on the network; where DMLP related transactions always have the highest priorities when adding transactions into a new block. For systems analysis and implementation two different cases regarding the processing of DMLP related transactions were considered:

- Case I: In this case, the LN that is assigned to perform the next block mining process; once it receives a new block it will immediately start the mining process without waiting (see Fig. 4a).
- Case II: In this case, the LN that is assigned to perform the next block mining process; once it receives a new block will not immediately start the mining process of the next block. A waiting time, ΔT, is introduced, which can be calculated by: $\Delta T = t_{pd} + t_p + \alpha$. Where t_{pd} is the transactions propagation time, t_p is the time needed by the neurons in the network to process DMLP related transactions, and α is the time needed to validate transactions by the LNs. This is to allow neurons to perform necessary processing and submit their finding in new transactions, with the aim of reducing the overall system latency (see Fig. 4b).

All the system's parameters are defined and explained in Table II.


Blocks' Interval Time (b_{it})
Hidden Nodes Data Processing
Output Node Data Processing
Confirmed Final outcome available on the network at t6
Block-1 under mining
Block-2
Block-3
Block-4
Block-5
Block-6
Block-7
t0
t1
t2
t3
t4
t5
t6
t7
Input Nodes data submission
Hidden processed data submission
Final outcome submission

(a)


Blocks' Interval Time (b_{it})
Hidden Nodes Data Processing
Output Node Data Processing
Confirmed Final outcome available on the network at t4
Block-1 under mining
Block-2
Block-3
Block-4
Block-5
t0
t1
t2
t3
t4
t5
Input Nodes data submission
Hidden processed data submission
Final outcome submission

(b)

Fig. 4. Block Mining Process. (a) Case I: Mining immediately Without Waiting. (b) Case II: Mining After Waiting for Time equal to ΔT

A. Overall Confirmation Time (OCT)

The probability of the final DMLP outcome arrival in the network is based on the Poisson process with arrival rate, λ.

$$P(T \leq t) = 1 - e^{-\lambda t} \tag{1}$$

We let λ represent the rate at which blocks are added to the network and, since there is no need for extra time for confirmations, we use trusted leader nodes for validating the blocks $\lambda = \frac{1}{T_m}$ block/sec where $T_m = \frac{D \times 2^{24}}{hp(i) \times I}$. The time t depends on the number of blocks (n) for which we need to wait before the final outcome arrives in the network, the block propagation delay b_{pd}, the block validation time b_{vt}, and most importantly the total number of DMLP layers N_L. Two different cases were considered in the design as stated above, for both cases, OCT has been analyzed as follows:

1) Case I: As shown in Fig. 4a, in this case, LN starts the mining process immediately, and the probability of the DMLP final outcome confirmation can be calculated by:

$$P(n) = \begin{cases} 1 - e^{-\left[(\frac{1}{\frac{D \times 2^{24}}{hp(i) \times I}}) \times \left(\frac{n}{N_L + 1} \times (\frac{D \times 2^{24}}{hp(i) \times I} + b_{vt} + b_{pd}) \right) \right]} & \text{if } n \geq (2N_L) \\ 0 \quad \text{if } n < (2N_L) \end{cases} \tag{2}$$

This represents the probability of confirming the outcome of the DMLP including the confirmation of the input values and all hidden values. The total time for this confirmation process (OCT) can be calculated by:

$$OCT = (2 \times N_L) \times \left[\frac{\ln(1 - P(n))}{\frac{-1}{\frac{D \times 2^{24}}{hp(i) \times I}}} \right] \tag{3}$$

2) Case II: As shown in Fig. 4b, in this case, we force LN to wait for DMLP related transactions to be processed, by introducing the waiting time, ΔT. This means the probability of the AI final outcome confirmation can be calculated by:

TABLE II
Description of The System's Parameters

Parameter	Description
D	Mining Difficulty
b_{it}	Interval time between two consecutive blocks
T_m	Time to mine a block (i.e time workers need to find the target hash)
b_{vt}	Block validation time by LNs.
Overall Confirmation Time (OCT)	The time from the initiation of the prediction process (i.e inputs neurons submit their reading) until the final outcome from the output neurons is confirmed on the blockchain network.
$hp(i)$	The hash per seconds produced by node i
I	Total number of worker nodes needed to mine a block
$P(n)$	Probability that AI output confirmed after n blocks
t_{pd}	Transactions propagation delay
b_{pd}	Block propagation delay
t_p	The time needed by each neuron to process DMLP-related transactions and produce its own new value.
α	The time each LN needs to process and validate every transaction before adding it to a new block
N_L	Total Number of Layer in the DMLP network

$$P(n) = \begin{cases} 1 - e^{-\left[(\frac{1}{\frac{D \times 2^{24}}{hp(i) \times I}}) \times \left(\frac{n}{N_L} \times (\frac{D \times 2^{24}}{hp(i) \times I} + b_{vt} + b_{pd} + \Delta T) \right) \right]} & \text{if } n \geq (N_L + 1) \\ 0 \quad \text{if } n < (N_L + 1) \end{cases} \tag{4}$$

This represents the probability of confirming the outcome of the DMLP including the input and hidden values. The total time for this confirmation process (OCT) can be calculated by:

$$OCT = \left[(N_L + 1) \times \left(\frac{\ln(1 - P(n))}{\frac{-1}{\frac{D \times 2^{24}}{hp(i) \times I}}} \right) \right] + [\Delta T \times (N_L - 1)] \tag{5}$$

B. Energy Cost

For a system where P_s is the power consumption during the sleep mode, P_t is the power consumption during data transmission, P_r is the power consumption during data reception and preparation, and P_{AI} is the power consumption during the processing of DMLP transactions. Then the total power consumption of one node in the DMLP prediction process is:

$$P_n = P_s + P_t + P_r + P_{AI} \tag{6}$$

These powers depend on the time for each event assuming that: T_r is the total time for receiving the block and extracting AI data. T_t is the time required to prepare and transmit the transaction T_{AI} the processing time of the DMLP data taking a node to produce its own final calculation. Then the total energy of a node E_n can be calculated as:

$$E_n = (P_s \times T_s) + (P_r \times T_r) + (P_t \times T_t) + (P_{AI} \times T_{AI}) \tag{7}$$

Therefore the total energy required for the DMLP process is:

$$E_{AI} = \sum_{n=1}^{n=N} E_n \tag{8}$$

The amount of energy consumed depends on the node's state, the node will be in one of five different states. *Idle State (i)* where the node is on and not connected to the wireless channel, and energy consumed in this state is defined by E_i, this is the reference state. *Connection State (cx)* where the node is on and connected to the available wireless channel (i.e. Wi-Fi connectivity), and energy consumed in this state is defined by E_{cx}. *Blockchain State (bc)*, where the node is connected to the blockchain network but is not performing any actions apart from submitting transactions and receiving and adding blocks to its internal storage, and energy consumed in this state is defined by E_{bc}. *AI State (AI)*, this is where the node, in addition to the actions performed in the blockchain state, is acting as one neuron of any of the DMLP layers, and energy consumed in this state is defined by E_{AI}. *Mining State (m)* where the node, in addition to the actions performed in the blockchain state, acts as a worker node and carries out some of the mining calculation; the energy consumed in this state is defined by E_m.

Based on these states, the difference in energy consumption between two states δE can be calculated by:

$$\delta E = E_{state1} - E_{state2} \tag{9}$$

The i state represents the reference state, this will allow for the calculation of the energy consumed by a node when in any state in comparison to the reference state, for example, the energy consumed by a node when in the mining state is:

$$\delta E_m = E_m - E_i \tag{10}$$

V. System Implementation and Deployment

For practical trial purposes, we have developed a proof of concept system. It consists of our own public blockchain platform and DMLP engine. The following subsections describe the implemented system.

A. Blockchain Implementation

A customized blockchain platform that implements the proposed consensus mechanism HDPoA discussed above was created. Currently, the network consists of 23 Raspberry Pi devices used for both blockchain and DMLP implementations and six ESP32 microcontrollers used only for blockchain implementation as WNs. Below is a description of the implementation of the essential parts of the blockchain.

1) Consensuses Mechanism: Leader nodes, as stated above, coordinate and manage the consensus process. Leader node selection is based on a round-robin process. Before each block mining process, one leader node is elected as a primary miner and another as a secondary miner. This is to ensure that a block is propagated to the network in each round. All other nodes apart from the elected primary and secondary miners will act as worker nodes. The consensus process begins with the leader nodes and takes place over four phases before the release of a new block to the network as follows:

- First, *LN1* will start the process by executing the initiation phase, which includes adjusting the difficulty level, validating transactions and adding them to a new block, preparing the work for the WNs, assigning WNs, and then sending the work to the assigned WNs.
- Multiple WNs will then conduct the hashing work. The worker node that finds the target nonce that satisfies the required difficulty level will submit the nonce along with its public key to all LNs currently available; this will allow other LNs to validate the current LN proposed block.
- Next, *LN1* will conduct the validation of the worker-reported results. This is done by executing a single hash of the new block's header, combined with the received nonce. If the resulting hash satisfies the target difficulty, then it is a valid solution. Otherwise, it is rejected and *LN1* should wait for other WNs. If valid, it will send an abort message to all WNs, then create and sign the block, and propagate it to the network.
- Finally, other leader nodes will validate the propagated block. This is achieved by ensuring the *LN1* is one of

the trusted LNs, by validating the block's hash and all transactions in the block.

Honesty Level: First, the node joins the network with an honesty level of zero and becomes a WN. As the node behaves honestly, obeys system rules, is available to perform any required work, and ensures it only submits correct answers to any work it carries out, its honesty value increases, giving the node the chance to be promoted to the leader node category. We define that node *n* has honesty level value of H_n and the target honesty level or threshold of the network is H_T. Node *n* can be leader only if $H_n > H_T$.

All work has a value of honesty W_v, and any correct answer to any work carries a positive honesty value of H_P, and a wrong answer carries a negative value of H_N. The honesty level for each node can be calculated by:

$$H_n = H_P + H_N \tag{11}$$

Once a node becomes a leader node, it can coordinate the mining process, validate the worker results, and sign and propagate new blocks when it is its turn to lead. For more details and the main concept of the algorithm please see our previous work in [35].

Scalable Work: the introduction of this concept provides three important advantages to the design. First, it allows the nodes to increase their honesty level, whereby if all nodes behave according to the network rules they all can be promoted to the Leader Nodes category; realizing the full potential of the decentralization concept. Secondly, it allows for the integration of mining tasks to incorporate the security advantages provided by PoW facilitating a public blockchain platform. Finally, it is used to increase the security of the network in the case of miss behaving WNs. By majority vote, leader nodes can penalize these nodes by increasing the work required from them.

The workload can be defined as W_L, the node hash power factor can be defined as HPF, and the total work is defined as W_T. The node's honesty factor HF can be calculated by:

$$HF(i) = \begin{cases} \frac{H_T - H_n}{HPF} & \text{if } H_T > H_n \\ \frac{1}{HPF} & \text{if } H_T \le H_n \end{cases} \tag{12}$$

The assigned W_L to any node i can be calculated by:

$$W_L(i) = W_T \times \frac{HF(i)}{\sum_{k=1}^{k=N} HF(k)} \tag{13}$$

As the node's honesty level increases, the work assigned to that node decreases, helping the node save energy. Conversely, if the node's honesty level decreases, more mining work will be assigned to it.

2) Block and Transactions Format: Table III shows the different fields of both transaction and block headers. Different fields have been introduced within the block header and the transaction to allow neurons within the DMLP engine to distinguish between different values and easily extract relevant values to process them.

Different types of transactions have been identified. These include DMLP-related transactions such as input-layer transactions, hidden-layer transactions, output-layer transactions, notifications transactions, and parameter update transactions (administrative transactions). Non-DMLP related transactions include data transfer between nodes, payment transactions, and reward transactions. The transaction type field is designed to allow the assigning of relevant types to each transaction. If the transaction is from an input neuron, then the type will have a value of one. Similarly, if the transaction is from a neuron in the first hidden layer then the value will be 2.0 etc. In this implementation, only one hidden layer was utilized, however, the transaction was designed to allow for a distinction between neurons if the system has more than one hidden layer.

The block header was also designed with the DMLP in mind. The DMLP_ Flag field will be used by the LN in case the block carries any DMLP-related transactions, alerting other LNs and neurons once they receive the block.

B. DMLP Implementation

A multilayer perceptron AI system was developed as a proof of concept and to test the validity of the architecture. It consists of three layers: input, hidden, and output. The process of developing the DMLP system spans over three phases. The first phase encompasses the implementation and testing on a standalone PC running i5-8250U CPU @ 1.60GHz. This is then followed by the implementation and testing using a Raspberry Pi 3 Model B+ (1.4GHz 64-bit quad-core processor) as a standalone system. The trained DMLP was finally implemented as a distributed system on the blockchain network, where Raspberry Pi devices are used to act as individual neurons.

TABLE III
FORMAT OF THE BLOCK HEADER AND THE TRANSACTION

Field	Description	Size
Block Header		
Block Height	Number of blocks in order	4 bytes
Previous Hash	The hash of the previous/ parent block	32 bytes
Merkle Root	Merkle root of all transactions	32 bytes
Block Time	Time of the creation of the block	4 bytes
Difficulty	The difficulty level.	4-byte
Nonce	The target nonce that produced the desired difficulty level	4-byte
DMLP_Flag	Indicate the presence of DMLP data. Optional, always 1 if used	2 byte
Hash	The hash of this block produced by the LN	32 bytes
Public Key	The public key of the WN	33 bytes
Signature	LN should sign the block for validation	64 bytes
Transaction		
Transaction-ID	Unique transaction identifier	128 bits
Transaction Type	0, default. 1, DMLP input values. 2.0, DMLP values from the first hidden layer. 2.1, DMLP values from the second hidden layer. 3, DMLP values form the output layer. etc.	4 bytes
Reading Value	Sensors /Input reading, hidden, output values	1-4 bytes
Recipient	The address of the receiver of the transactions, if it is intended for DMLP then the layer's name is used	2-33 bytes
Node Type	Distinguish between different neurons in different layers	1-4 bytes
Node Name	To help distinguish the flow of DMLP values from one layer to another	1-4 bytes
Timestamp	Time of the creation of the transaction	4 bytes
Signature	Sender should sign the transactions for validation	64 bytes

C. Dataset Tested

For practical trial and testing purposes, we have used three different datasets; the Iris flower dataset [36], air quality dataset [37] and the Bot-IoT dataset from [38]. First, the DMLP system was utilized to forecast the type of flower from the iris dataset; the system consists of four neurons in the input layer, five neurons in the hidden layer, and three neurons in the output layer; each neuron was deployed on a Raspberry Pi. The dataset contained 150 instances, with 4 features and one output. Then the system was configured to test future occurrences of air pollution that could affect air quality. It consists of 12 neurons in the input layer, 15 neurons in the hidden layer, and three neurons in the output layer. This dataset contains a total of 9,358 instances with 12 features, it has been prepared and normalized for the purpose of this implementation. A Raspberry Pi was used to emulate all input nodes, 15 Raspberry Pis used for the hidden neurons, and one Pi for the output neurons.

Finally, the system was configured to classify the type of attacks either Denial of Service (DoS), Distributed Denial of Service (DDoS), or Reconnaissance based on the Bot-IoT dataset. The configured system consists of 10 neurons in the input layer, 12 neurons in the hidden layer, and three neurons in the output layer. The dataset we used contained 10,000 instances with 10 features and three outputs, and this data has been prepared and normalized for the purpose of this implementation, the complete description of this dataset can be found in [38]. Ten Raspberry Pis used as neurons for the input layer, 12 used for the neurons in the hidden layer, and one used for the neurons in the output layer.

VI. Results

The system was tested while the nodes were deployed over two cities in the UK, Sheffield and Edinburgh, separated by a distance of about 310km. The following sections include the results and their evaluation.

A. Overall Confirmation Time

In both cases (see Fig. 4), the system was tested using a different number of workers, three leader nodes, and different difficulty levels. As shown by Fig. 5, the system was tested for two difficulties (D=1 and D=2). Both the predicted (using 3 and 5) and the measured OCT indicated that the overall system end-to-end confirmation time can be reduced as the network grows in size without compromising its security by reducing the mining difficulties. In a network of 24 WNs with a difficulty of one, the OCT for Case I was 3.4 minutes and 2.16 minutes for Case II. For the same network with a difficulty of two, the measurements were 6.8 and 4.1 minutes, respectively. The T_m of the block has the most effect on the OCT, nevertheless, the block and transaction propagation delays are important parameters. The results shown in Table IV indicate that since the test was conducted with reliable Wi-Fi connectivity, the average of b_{pd} and t_{pd} have little effect on the OCT. However, as shown by our previous study in [31], the b_{pd} is expected to have more influence on the OCT if the size of the block is increased or different communication methods, such as 3G and LoRaWAN, are used.

TABLE IV
Latency Measurements of Most important System Parameters

Parameter	Average	Maximum	Minimum
b_{pd}	295 ms	682 ms	227 ms
t_{pd}	66 ms	99 ms	41 ms
P_t	200 ms	220 ms	190 ms

The network is small with regards to the number of workers; nevertheless, within IoT, with the presence of thousands of devices, the difficulty can be increased according to the number of available nodes in the network while maintaining an acceptable OCT according to the application. To better understand the effect on the OCT in the presence of hundreds of devices, OCT was calculated for different difficulty levels (1-16) using 3 and 5 and a different number of WNs, ranging from 100 to 1,000. A network that has more than 500 WNs can achieve OCT in less than two minutes for Case I and less than a minute for Case II. It is clear from Fig. 6 that as the number of WNs increases, the OCT would be significantly lowered.

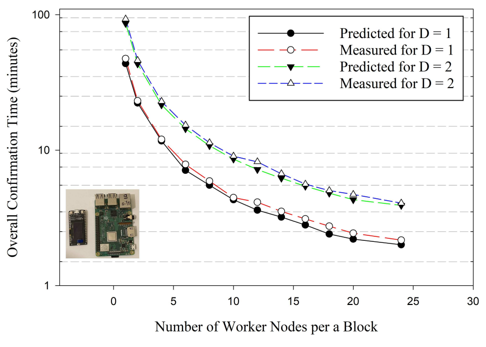
Another test was conducted to measure OCT using a network consisting of 20 workers for four different difficulties ($D = 1$, $D = 2$, $D = 4$, $D = 8$, and $D = 16$). As shown by Fig. 7, when the difficulty increases, the OCT also increases; however, if more devices were available to use, the time could be lowered while maintaining an acceptable level of D.

B. DMLP Accuracy

For each dataset, the system was first trained and tested on a standalone PC and on a Raspberry Pi. Each dataset was divided into 70% for training, 10% for testing on the PC, 10% for testing on the Raspberry Pi, and 10% for testing on the blockchain network, with the aim of comparing the resulting accuracies. This is to ensure the trained DMLP accuracy results are consistent with the results from both implementations on the standalone PC and Raspberry Pi. The results showed that the accuracy resulting from testing over the blockchain is in line with both tests on the individual Raspberry Pi and the PC. All tests produced the same accuracy, 98% for the iris dataset, 96% for the air quality dataset, and 92% for the Bot-IoT dataset. This means it is practically possible to implement DAI engines on IoT devices based on blockchain technology with the accuracy unaffected.

C. Computational Power and Energy Consumption

Energy consumption is a vital aspect of this system; the system was designed with the intention of allowing low-power devices to be part of the blockchain and benefit from the DMLP services offered. This is done in exchange for a small amount of power, where these devices participate in the mining process and ensure blockchain network security. The hash power of devices used in the implementation was measured



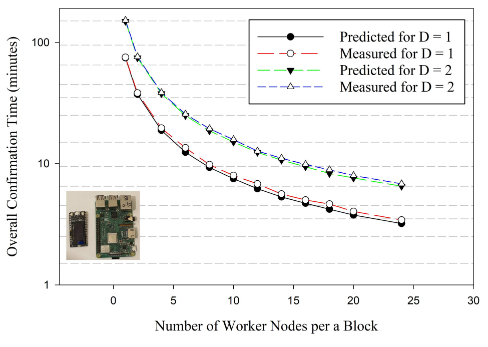
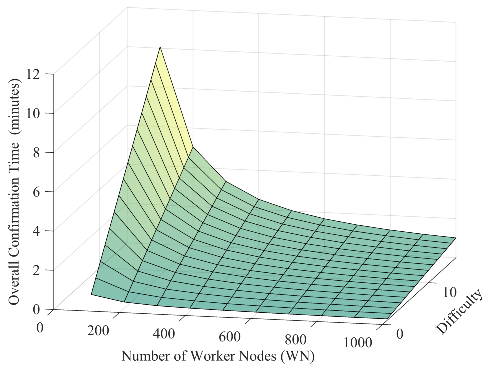


Fig. 5. Overall Confirmation Time of the DMLP outcome. (a) Case I: Mining immediately. (b) Case II: Mining After Waiting for Time equal to ΔT



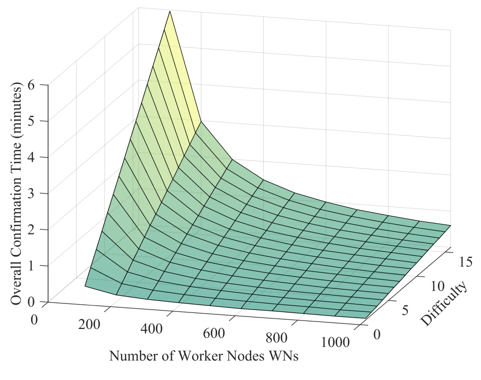


Fig. 6. Predicted OCT for different Difficulties and WNs. (a) Case I: Mining immediately. (b) Case II: Mining After Waiting for Time equal to ΔT

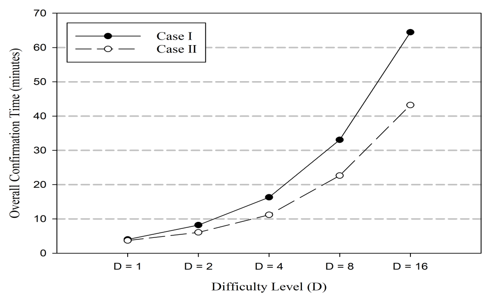


Fig. 7. Overall Confirmation Time Using 20-WNs when Mining in Different Difficulties.

based on the number of hashes the device can perform per second (h/s). The Raspberry Pi average hash power was 35 Kh/s while the ESP32 averaged 17.4 Kh/s.

The energy consumption was measured for each of the different system states when the system is implemented on different devices: a Raspberry Pi 3 Model B+ and an ESP32. Figure. 8a shows the average energy consumption of both devices for different states. Most of the energy consumed is due to running the device's operating system (idle state). Using 9, it can be seen that the Raspberry Pi consumes 0.12 joules per second (J/s) when the system only performs DMLP-related tasks. However, when the system is in the mining state, it consumes 0.53 J/s, compared to only 0.15 J/s when mining using an ESP32 (see Fig. 8b).

Based on the above measurements of the hash power and the energy consumption, the hash per joule was calculated and is shown in Fig. 8c. When using a Raspberry Pi, a total of up to 13.8 Kh can be performed at the cost of one joule, and when using an ESP32, up to 54 Kh can be achieved per joule. This demonstrates that the ESP32 microcontroller is more efficient in terms of power when it is used for blockchain mining in comparison with the Raspberry Pi, which provides faster operations but with a slight increase in energy cost.

D. Evaluation

The DMLP performance on the blockchain network was not affected, and the results showed that it achieved the same accuracy as both of the standalone implementations. Measurements and prediction showed that a network with thousands of available workers can achieve a reasonable difficulty to secure the network while managing the mining work between the workers to save energy and significantly reduce the OCT. The network currently consists of 23 Raspberry Pis and six ESP32, nevertheless, all of the measured results, especially those related to OCT, were almost identical to the predictions that were based on the system analyses in section IV (see Fig. 5 and Fig. 6). This indicates that the proposed architecture is valid and the blockchain platform can support the distributed implementation of AI using IoT hardware capabilities.

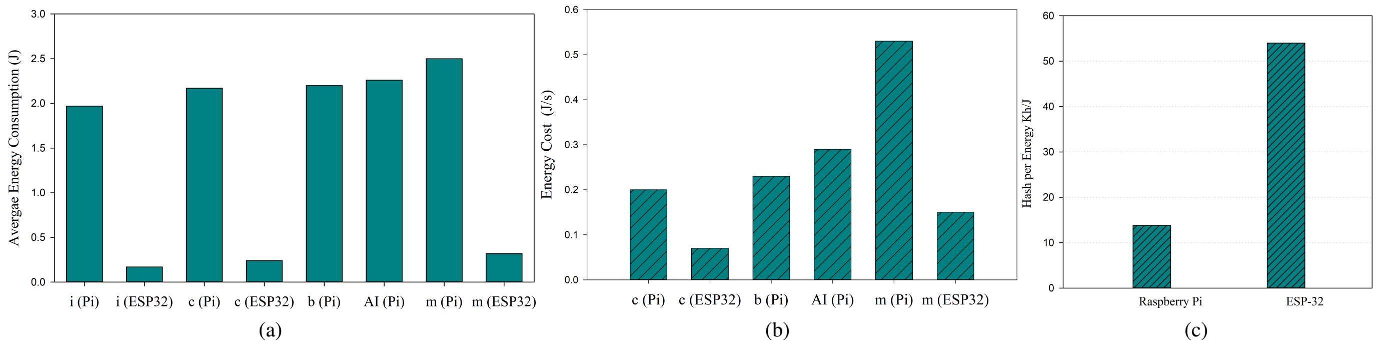


Fig. 8. Energy and Hash Power Measurements For Raspberry Pi and ESP32 when the system is in idle (i), connection (cx), blockchain (bc), AI, and mining (m) states. (a) Average energy Consumption. (b) Energy Cost δE During Different States, (c) Hash Power per Energy (During Mining State)

In terms of computational power, the results showed that low-cost devices such as Raspberry Pi and ESP32 have the hashing power required to participate in the mining process of the HDPoA consensus mechanism. This is because the implemented consensus mechanism was designed based on sharing the computational power among IoT devices, and it can efficiently utilize the power from these devices. This is reflected in the power consumption results above.

In this paper the energy cost during different system states including during executing DAI-related tasks and the mining process have been measured, however, it was difficult to compare these measurements with any of the related works because all of them have not provided such measurements. An added contribution of our paper in comparison to other works in its class. In our HDPoA, when running a mining task for 82.9 sec on a Raspberry Pi, the total energy cost was 43.9 J and for the ESP32 it was 12.4 J, this compares to 54.9 J energy cost measured by [39] when executing PoW on a Raspberry Pi. This indicates that the proposed consensus mechanism of HDPoA is more efficient than PoW in terms of energy consumption and that nodes can participate in the mining process while also being used for data analysis and sharing without substantial power cost. As the number of participating nodes increases, the effect on their individual power will decrease. In a network with a few thousand IoT devices, a node might spend a day without executing tasks.

In terms of security and user trust, this architecture provides a distributed public blockchain platform that ensures the security of data through the proposed HDPoA that is suitable for implementation within the IoT realm. Users can see that this is a trustworthy platform because it relies on an immutable, transparent, and secure blockchain. *Malicious or misbehaving leader nodes* that try to sign and propagate an invalid block can be dealt with. First, the network implements a mechanism that only accepts a block from any leader node every N blocks, where N is equal to the total number of active leaders at the time. Second, each new block in the network will be validated by every LN in the network, both the hash of the block and all the transactions in it.

The consensus mechanism can adjust the mining difficulty according to the available mining power on the network and the required mining rate (interval between blocks). As the number of nodes in the network increases, mining power will also increase as more WNs participate in the mining process. Accordingly, the difficulty will be increased. For example, with a total of 20 WNs, the network was able to increase the mining difficulty up to 16 and mine blocks; the only downside was the increase in the mining time (see Fig. 7). When coupled with the added security of the presence of LNs running the mining process, this prevents dishonest nodes from breaching the consensus rules of the network or harming it. In terms of robustness and redundancy, this proposed architecture does not rely on a third-party central entity to process and share data, which eliminates a single point of failure by leveraging distributed architecture.

Table V provides a detailed performance comparison between our paper and other related works in four performance metrics; communication overhead, risk of a single point of fail-

TABLE V
PERFORMANCE COMPARISON WITH RELATED WORKS

Paper	Communication Overhead	Risk of Single Point of Failure?	Data Security (Integrity and Availability)	Accuracy
14	Local learning model processing (+) Blockchain used to distribute parameters (+) PS = 2	Yes, need cloud offloading PS = 1	Blockchain is utilized to ensure the protection of data starting from IoT devices (++) PS = 2	75% PS = 1
[16]	Utilized blockchain for data handling from the edge layer only (+) No local processing PS = 1	Yes, relies on cloud and data center PS = 1	Blockchain is utilized to ensure the protection of data starting from the edge (+) PS = 1	Not measured PS = 0
[17]	Uses two levels of fog processing in a hierarchical structure (++) PS = 2	Yes, relies on central cloud for some of the AI Implementation PS = 2	No mechanism in place PS = 0	Not measured PS = 0
[18-20]	Nodes need to connect to each other from layer to another to exchange data PS = 0	Yes, clusters' heads could be a problem PS = 2	No mechanism in place PS = 0	Not measured PS = 0
[21]	Some local processing but there is a need to exchange data with aggregator and prone to bottleneck (+) PS = 1	Yes, needs an aggregator and cloud PS = 0	No mechanism in place PS = 0	Up to 97% PS = 2
[22]	Much local processing but devices need to exchange data with a controller (++) PS = 2	Yes, devices rely on a controller PS = 0	No mechanism in place PS = 0	Vary between 90%-96.7 PS = 2
This Paper	Some local processing (+) Blockchain utilized to handle data starting from sensing layer (+) Devices only need to propagate transactions once to leader nodes (+) PS = 3	Fully decentralized (+) Any IoT devices can be utilized as a neuron (+) It also allows for the processing of the same data by multiple DAI models (+) PS = 3	Blockchain is utilized to ensure the protection of data integrity and ensures its availability starting from the sensing layer (++) PS = 2	Vary between 92%-99% PS = 2
PS = Performance Score. For each category, the score is assigned between 0 to 3 where 0 is the worst in class and 3 best in class.				

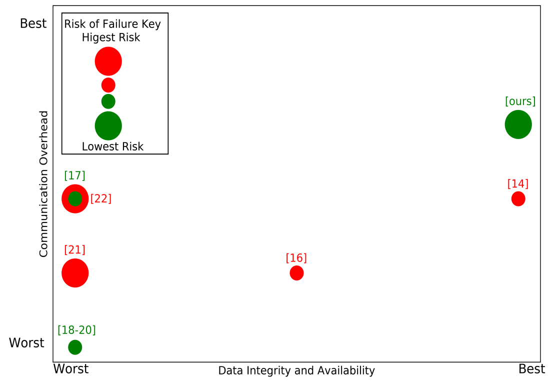


Fig. 9. Illustration of Comparative Performance Showing Relative Best in Class/Worst in Class for Different Metrics as Listed in Table V. Numbers in [ ] are the Paper's References.

ure, data integrity and availability, and DAI engine measured accuracy. Figure 9 shows the score for each architecture in both data integrity and availability and communication overhead metrics along with the risk associated with each one of them. Our proposed architecture is the best for communication overhead and has the lowest risk of a single point of failure. It also ranked among the best architectures in both data integrity and availability and measured accuracy metrics. Finally, this architecture provides a trustworthy, self-managed, and self-regulated, public platform that can be utilized to integrate DAI into IoT hardware.

VII. Conclusion

A distributed, decentralized, and secure blockchain-based architecture for supporting DAI on low-power and low-cost IoT devices was proposed. A practical implementation of DMLP using a distributed IoT hardware platform was accomplished using a real-world example application. The results showed that, in terms of prediction accuracy, it is possible to implement a DAI system over an IoT platform based on blockchain technology. It also showed that this architecture provides a secure, scalable, and distributed approach that utilizes IoT devices as a platform for AI implementation with a minimal impact on their computational resources.

Currently, the proposed architecture is implemented based on an on-chain approach; future work will include exploring the viability of utilizing an off-chain approach, which may be ideal for applications with near real-time response requirements. Additional future research will include the integration of an encryption algorithm into the system and the deployment of another type of AI, such as a Convolutional Neural Network (CNN). The evaluation of the system performance over different communication links, such as 4G and LoRaWAN, will also be investigated.

References

[1] A. F. R. Neto, F. C. Delicato, T. V. Batista, and P.F. Pires . "Distributed Machine Learning for IoT Applications in the Fog," In Fog Computing (eds A. Zomaya, A. Abbas and S. Khan). 2020, doi:10.1002/9781119551713.ch12.

[2] M. V. Steen and A.S Tanenbaum, "A brief introduction to distributed systems," Computing 98, 967–1009 (2016). https://doi.org/10.1007/s00607-016-0508-7.

[3] G. Serpen, J. Li, L. Liu and Z. Gao, "WSN-ANN: Parallel and distributed neurocomputing with wireless sensor networks," The 2013 International Joint Conference on Neural Networks (IJCNN), Dallas, TX, 2013, pp. 1-8, doi: 10.1109/IJCNN.2013.6706764.

[4] N. Lynch, "Distributed Algorithms," San Francisco, CA, USA:Morgan Kaufmann, 1996.

[5] 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," in IEEE Communications Surveys and Tutorials, vol. 21, no. 2, pp. 1508-1532, Secondquarter 2019, doi: 10.1109/COMST.2019.2894727.

[6] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," Manubot, Tech. Rep., 2008.

[7] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," Yellow Paper, vol. 151, pp. 1–32, Apr. 2014.

[8] Q. Zhou, H. Huang, Z. Zheng and J. Bian, "Solutions to Scalability of Blockchain: A Survey," in IEEE Access, vol. 8, pp. 16440-16455, 2020, doi: 10.1109/ACCESS.2020.2967218.

[9] J. Poon and T. Dryja. (2016)."The Bitcoin Lightning Network: Scalable Off-Chain Instant Payments," [Online]. Available: https://lightning.network/lightning-network-paper.pdf.Accessed on: Aug. 6, 2020.

[10] J. Poon and V. Buterin, "Plasma: Scalable autonomous smart contracts," White Paper, 2017, pp. 1–47.

[11] W. Rouwer, and M. Borda, "NeuRoN:Decentralized Artificial Intelligence, Distributing Deep Learning to the Edge of the Network," (2017), [Online]. Available: https://s3-us-west-1.amazonaws.com/ai.doc.static/pdf/whitepaper.pdf. Accessed on: Jun. 13, 2020.

[12] M. A. Rahman, M. M. Rashid, M. S. Hossain, E. Hassanain, M. F. Alhamid and M. Guizani, "Blockchain and IoT-Based Cognitive Edge Framework for Sharing Economy Services in a Smart City," in IEEE Access, vol. 7, pp. 18611-18621, 2019, doi: 10.1109/ACCESS.2019.2896065.

[13] T.T. Kuo and L. Ohno-Machado, 2018. "ModelChain: Decentralized privacy-preserving healthcare predictive modeling framework on private blockchain networks". arXiv preprint arXiv:1802.01746.

[14] S. Rathore, Y. Pan, and J. H. Park, "Blockdeepnet: A blockchain-based secure deep learning for iot network," Sustainability, vol. 11, no. 14, p.3974, 2019.

[15] M. A. Ferrag and L. Maglaras, "DeepCoin: A Novel Deep Learning and Blockchain-Based Energy Exchange Framework for Smart Grids," in IEEE Transactions on Engineering Management, doi: 10.1109/TEM.2019.2922936.

[16] S. Singh, Y. Jeong, J. Park, "A deep learning-based IoT-oriented infrastructure for secure smart City," Sustainable Cities and Society, Volume 60, 2020,102252,ISSN 2210-6707, doi:https://doi.org/10.1016/j.scs.2020.102252.

[17] K. L. Cai and F. J. Lin, "Distributed Artificial Intelligence Enabled by oneM2M and Fog Networking," 2018 IEEE Conference on Standards for Communications and Networking (CSCN), Paris, 2018, pp. 1-6, doi: 10.1109/CSCN.2018.8581775.

[18] G. Serpen and J. Li, "Parallel and distributed computations of maximum independent set by a Hopfield neural net embedded into a wireless sensor network," Procedia Computer Science,Volume 6,2011, pp. 390-395, doi:https://doi.org/10.1016/j.procs.2011.08.073.

[19] J. Li and G. Serpen,"nesC-TinyOS model for parallel and distributed computation of max independent set by Hopfield network on wireless sensor network," Procedia Computer Science, Volume 6, 2011, pp. 396-401,doi:https://doi.org/10.1016/j.procs.2011.08.074.

[20] J. Li and G. Serpen,"TOSSIM simulation of wireless sensor network serving as hardware platform for Hopfield neural net configured for max independent set," Procedia Computer Science, Volume 6, 2011, pp. 408-412,doi:https://doi.org/10.1016/j.procs.2011.08.076.

[21] S. Teerapittayanon, B. McDanel and H. T. Kung, "Distributed Deep Neural Networks Over the Cloud, the Edge and End Devices," 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS), Atlanta, GA, 2017, pp. 328-339, doi: 10.1109/ICDCS.2017.226.

[22] T. Bi, Q. Liu, T. Ozcelebi, D. Jarnikov and D. Sekulovski, "PCANN: Distributed ANN Architecture for Image Recognition in Resource-Constrained IoT Devices," 2019 15th International Conference on Intelligent Environments (IE), Rabat, Morocco, 2019, pp. 1-8, doi: 10.1109/IE.2019.000-3.

[23] Y. Zhao, J. Zhao, L. Jiang, R. Tan, D. Niyato, Z. Li, L. Lyu, and Y. Liu, "Privacy-Preserving Blockchain-Based Federated Learning for IoT Devices," ArXiv, 2020, odi: https://arxiv.org/abs/1906.10893v3.

[24] P. Ramanan, K. Nakayama, and R. Sharma, "Baffle: Blockchain based aggregator free federated learning," arXiv, 2020, odi: https://arxiv.org/abs/1909.07452v3.

[25] C. Arouri, E. M. Nguifo, S. Aridhi, and N. Tsopze, "Towards a constructive multilayer perceptron for regression task using non-parametric clustering. A case study of Photo-Z redshift reconstruction," ArXiv, 2014 odi:https://arxiv.org/abs/1412.5513.

[26] FIPS-197: Advanced Encryption Standard (November 2001), [Online]. Available: http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf. Accessed on: Nov. 22, 2020.

[27] S. De Angelis, L. Aniello, R. Baldoni, F. Lombardi, A. Margheri, V. Sassone, PBFT vs proof-of-authority: applying the cap theorem to permissioned blockchain,2018.

[28] Li, W.; Andreina, S.; Bohli, J.; Karame, G. Securing proof-of-stake blockchain protocols. In Data Privacy Management, Cryptocurrencies and Blockchain Technology; Springer: California, America, 2017; pp. 297–315.

[29] .P. Vasin, Blackcoins proof-of-stake protocol v2, 2014, [online] Available: https://blackcoin.co/blackcoin-pos-protocol-v2-whitepaper.pdf.Accessed on: Aug. 30, 2020.

[30] P. Szilágyi "EIP-225: Clique proof-of-authority consensus protocol," (2017), [Online]. Available:https://eips.ethereum.org/EIPS/eip-225. Accessed on: Aug. 3, 2020.

[31] S. M. Alrubei, E. A. Ball, J. M. Rigelsford and C. A. Willis, "Latency and Performance Analyses of Real-World Wireless IoT-Blockchain Application," in IEEE Sensors Journal, vol. 20, no. 13, pp. 7372-7383, 1 July1, 2020, doi: 10.1109/JSEN.2020.2979031.

[32] L. Bahri and S. Girdzijauskas. 2018. "When Trust Saves Energy: A Reference Framework for Proof of Trust (PoT) Blockchains". In Companion Proceedings of the The Web Conference 2018 (WWW '18). International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 1165–1169. DOI:https://doi.org/10.1145/3184558.3191553.

[33] J. Huang, L. Kong, G. Chen, M.-Y. Wu, X. Liu, and P. Zeng, "Towards Secure Industrial IoT: Blockchain System with Credit-Based Consensus Mechanism," IEEE Trans. Ind. Informatics, vol. 15, no. 6, pp. 1–1, 2019.

[34] G. Sagirlar, B. Carminati, E. Ferrari, J. D. Sheehan, and E. Ragnoli, "Hybrid-IoT: Hybrid Blockchain Architecture for Internet of Things - PoW Sub-blockchains," pp. 1–10, 2018.

[35] S. Alrubei, E. Ball and J. Rigelsford, "A Secure Distributed Blockchain Platform for Use in AI-Enabled IoT Applications," 2020 IEEE Cloud Summit, Harrisburg, PA, 2020, pp. 85-90, doi: 10.1109/IEEECloudSummit48914.2020.00019.

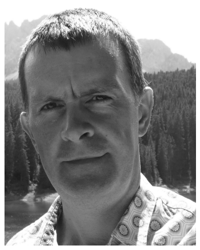
[36] Iris Dataset, UCI Machine Learning Repository, [Online]. Available: https://archive.ics.uci.edu/ml/datasets/iris. Accessed on: Jan. 15, 2021.

[37] Air quality Dataset, UCI Machine Learning Repository, [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Air+quality. Accessed on: Jul. 5, 2020.

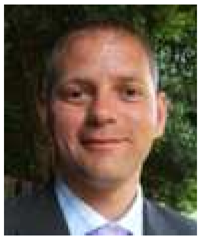
[38] N. Koroniotis, N. Moustafa, E. Sitnikova and B. Turnbull, "Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset", arXiv:1811.00701, 2018.

[39] A. Elsts, E. Mitskas and G. Oikonomou, "Distributed Ledger Technology and the Internet of Things: A Feasibility Study," Proc. 1st Wksp. Blockchain-Enabled Networked Sensor Systems, pp. 7-12, 2018.

Subhi M. Alrubei graduated in 2003 with a Bachelor of Engineering in Communication Systems Engineering from the University of Portsmouth, Portsmouth, United Kingdom. Later in 2012 Subhi received a Master of Science with Distinction in Networks and Security from the University of Kent, Canterbury, United Kingdom. He has worked for over 10 years as a communication engineer in Saudi Arabia, where he has managed and executed many communications systems projects. Since 2018 he joined the Department of Electronic and Electrical Engineering, University of Sheffield, Sheffield, UK as a PhD student. His research interest includes blockchain technology, the Internet of Things IoT, cyber security, and AI. He investigates the integration of blockchain, and AI into future IoT application for better security, privacy and performance.



Edward Ball (M 2008–present) Edward (Eddie) became a Member of IEEE in April 2008 and was born in Blackpool, United Kingdom in November 1973. Eddie graduated in 1996 with a 1st Class Master of Engineering Degree in Electronic Systems Engineering, from the University of York, York, United Kingdom. After graduating, he worked in industry for 20 years, first spending 15 years working as Engineer, Senior RF Engineer and finally Principal RF Engineer at Cambridge Consultants Ltd in Cambridge, UK. He then spent 5 years as Principal RF Engineer and Radio Systems Architect at Tunstall Healthcare Ltd in Whitley, UK. In November 2015 he joined the Department of Electronic and Electrical Engineering at the University of Sheffield, Sheffield, United Kingdom, where he now works as Reader in RF Engineering. His research interests cover all areas of radio technology, from RF system design, RF circuit design (sub-GHz to mm-wave) and the application of radio technology to real-world industrial and commercial problems. He has a particular passion for RF hardware design. Mr. Ball is a member of the IET and is a Chartered Engineer.



Jonathan M. Rigelsford (SM'13) received the MEng and PhD degrees in Electronic Engineering from the University of Hull, Hull, UK in 1997 and 2001 respectively. From 2000 to 2002, he worked as Senior Design Engineer at Jaybeam Limited. From late 2002, until 2014 he was a Senior Experimental Officer for the Communications Group within the Department of Electronic and Electrical Engineering, University of Sheffield, Sheffield, UK. He then became a Senior Research Fellow at the same institution. In 2019 Dr Rigelsford moved to Sensata Technologies as RF Engineering Lead and maintains a Visiting position in Sheffield. His research interests have included RF propagation, biomedical electromagnetics, adaptive antennas, RFID and cyber security