electronics

MDPI

Article

Automotive Vulnerability Analysis for Deep Learning Blockchain Consensus Algorithm

Seong-Kyu Kim [1,2]

[1] Department of Information Security, Joongbu University, Goyang-si 10279, Korea; skkim@joongbu.ac.kr
[2] Department of Public Policy and Information Technology, Seoul National University of Science and Technology, Seoul 01811, Korea

Abstract: In this study, future cars are attempting self-driving around the world. However, hacking, such as ECUs in automobiles, creates problems that are directly connected to human life. Therefore, this study wrote a paper that detects anomalies in such cars by field. As a related study, the study investigated the vulnerabilities of the automobile security committee and automobile security standards and investigated the detection of abnormalities in the hacking of geo-train cars using artificial intelligence's LSTM and blockchain consensus algorithm. In addition, in automobile security, an algorithm was studied to predict normal and abnormal values using LSTM-based anomaly detection techniques on the premise that automobile communication networks are largely divided into internal and external networks. In the methodology, LSTM's pure propagation malicious code detection technique was used, and it worked with an artificial intelligence consensus algorithm to increase security. In addition, Unity ML conducted an experiment by constructing a virtual environment using the Beta version. The LSTM blockchain consensus node network was composed of 50,000 processes to compare performance. For the first time, 100 Grouped Tx, 500 Channels were tested for performance. For the first time, the malicious code detection rate of the existing system was verified. Accelerator, Multichannel, Sharding, Raiden, Plasma, and Trubit values were verified, and values of approximately 15,000 to 50,000 were obtained. In this paper, we studied to become a paper of great significance on hacking that threatens human life with the development of self-driving cars in the future.

Keywords: algorithm; information security; deep learning; LSTM; blockchain

Citation: Kim, S.-K. Automotive Vulnerability Analysis for Deep Learning Blockchain Consensus Algorithm. *Electronics* **2022**, *11*, 119. https://doi.org/10.3390/electronics11010119

Academic Editor: Krzysztof Szczypiorski

Received: 31 October 2021
Accepted: 24 December 2021
Published: 30 December 2021

1. Introduction

Smart cars are cars that incorporate ICBM (IoT, Cloud, Big Data, Mobile) technology into existing cars. Specifically, the "smart car" is a car that combines technologies such as sensors, machine learning, and systems to help drivers. Smart car technology can be divided into two categories: ADAS-Advanced Driver Assistance Systems (ADAS-Advanced Driver Assistance Systems) and self-driving cars depending on whether some of the cars are supported or not 100% [1–3]. The advanced driving support system is a system that partially assists the driving of cars. A representative example of the advanced driving support system is the "driver health check" function developed by Hyundai Motor. This function monitors the driver's heart rate and pupil condition and automatically reduces the speed of the car, and causes an emergency stop if the driver is deemed unhealthy to drive. Accidents are prevented in advance by preventing malfunctions caused by deteriorating driver health. Dynamic Light Spot NV (Night Vision) technology applied to cars that are off-work is also a representative technology of advanced driving support systems. This technology identifies moving objects on a dark night and displays them in yellow on the display so that the driver can detect objects in front of the car well. If the driver does not detect the object in front of him and is expected to collide with the object in front of him, the car will operate on its own with a sudden-stop brake function [4–6].

This prevents collisions. In addition, Tesla's Autopilot function is one of the advanced driving system technologies. Tesla's Autopilot is a function that allows cars to identify lanes on their own on highways and drive themselves along lanes and allows them to park on their own by recognizing obstacles around them as sensors. Self-driving is a technology that enables a car to drive itself perfectly without a driver [7–10]. Self-driving cars do not need any help from drivers except for having a "break" function in case of an emergency. If you enter a destination in the navigation system, it will automatically operate according to GPS (Global Positioning System) transmission and reception of the satellite (Figure 1).

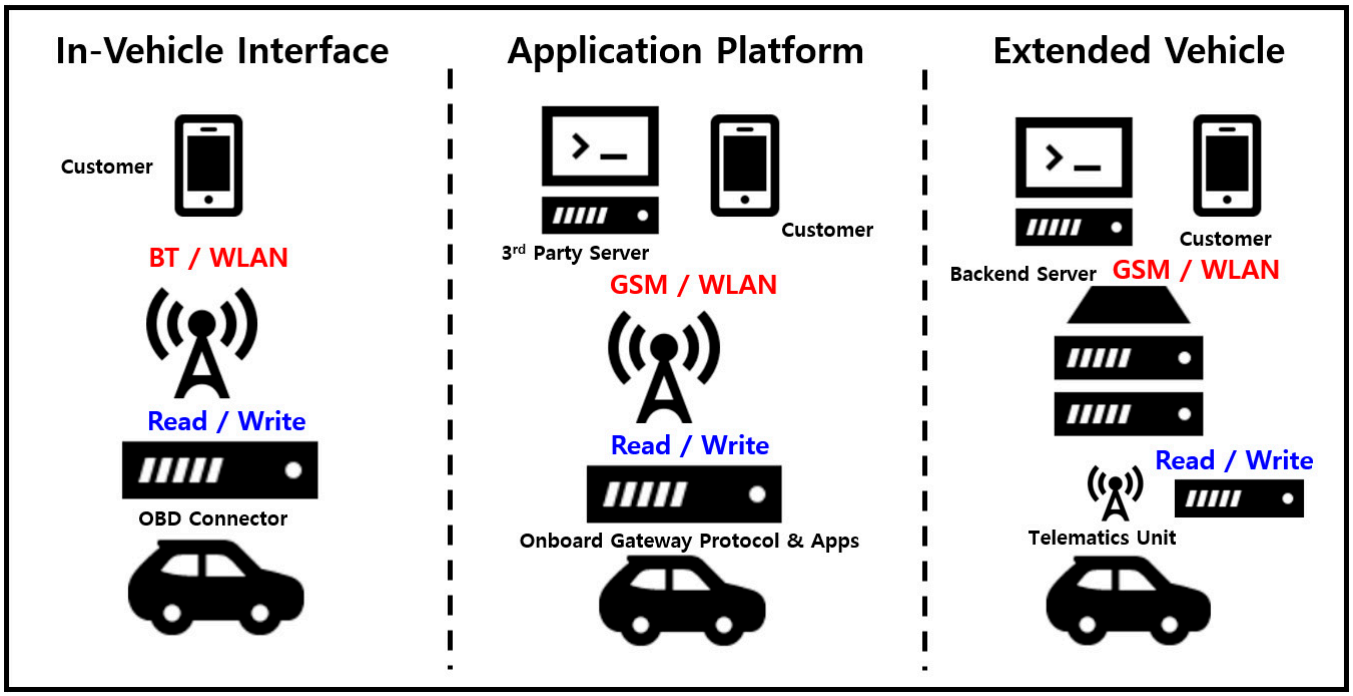

Figure 1. Interface for Supporting Automobile Cloud Services (Source: ISO TC22/SC31 WG6, 2020).

IT (Information Technology) companies Google and Apple, and automakers BMW and Audi, are also developing commercialized autonomous vehicles. However, according to the survey, 60% of drivers prefer to drive themselves rather than leave them to autonomous driving. Until now, people do not prefer it due to the stability of the autonomous driving system [11]. Furthermore, I think that defending the hacking of these cars will make a very important contribution. Automobiles are being used as the most important means of movement for humans to move. However, it is true that there are many traffic accidents. In addition, a representative example of cybersecurity threats to automobiles is the Jeep Cherokee hacking incident. This is a case in which American white hackers Charlie Miller and Chris Volosec succeeded in remote control by hacking into a digital system mounted on Fiat Chrysler's automobile model Jeep Cherokee and stealing control of the vehicle. The incident forced Chrysler to recall 1.4 million of the models. In addition, Hiroshima University in Japan succeeded in operating the vehicle wirelessly with its own application after hacking the car.

In addition, the automobile security guidelines explain details and examples of about 30 security threats, including threats to backend servers, threats to automobile data and codes, and potential vulnerabilities.

However, as IoT (Internet of Things) develops, smart car trends are expected to lean toward self-driving cars. Since the problem of preference for autonomous vehicles is a safety problem, the market will be activated once this part is resolved. Therefore, in this paper, abnormality detection technology has been studied for a long time, and, recently, machine learning and deep learning technologies have developed, showing excellent

results in abnormality detection based on time-series data. Therefore, in this study, artificial intelligence-based technologies such as machine learning and deep learning were used to apply them to various fields such as behavior pattern analysis, industrial control systems, and automobile security [12–15]. Abnormal detection techniques are defined and can be applied differently according to each field, so there are several limitations. In this study, various artificial intelligence-based techniques were used and applied directly to various domains to increase usability in the field of abnormality detection. In this study, the concept of anomaly detection was summarized, and artificial intelligence-based anomaly detection techniques were directly applied to various fields. The methodology for anomaly detection techniques and their strengths and weaknesses were compared. A study was also conducted applying deep learning and machine learning techniques and rule-based abnormality detection techniques. In this study, research was conducted on behavioral abnormality detection in the field of image recognition, abnormality detection in the industrial control system, and an intrusion detection system (IDS) using automobile data. In addition, accurate security and verification were performed by applying the blockchain consensus algorithm [16–20].

To verify the information security of automobiles, we propose an algorithm that verifies using LSTM's pure propagating malicious code detection model design, vanishing gradient profile model, and commission calculation process.

(1) Section 1 is platooned using an introduction.
(2) Section 2 introduces automobile security, automobile security standards, and artificial intelligence algorithms as related studies.
(3) Section 3 introduced LSTM's Code Detection Model Design through Model Design.
(4) Section 4 explains the love of experimental results.
(5) Section 5 explains the conclusions and future studies.

In addition, this study conducted a study on the security of these cars and examined the mood of ISO/SAE 21434, an automobile security standard. In addition, a study was conducted on the same storage space of LSTM, a key hacking prediction algorithm, in a gated state or a gated memory. Further, in order to increase security, security was increased by using a blockchain consensus algorithm, and, among the various experimental methods, a parallel consensus algorithm was used.

2. Related Research

2.1. Automobile Security

It introduces security threats to cars and security technologies currently being applied or studied. Various security threats to automobiles and various security solutions are being researched and developed, but the biggest security problem in automobiles is that unauthorized data is injected into the vehicle's internal network, and the availability of automobiles is violated through attacks such as DoS (Denial-of-Service). For this purpose, there are various attack methods and threats exist. Security threats to self-driving cars were classified and organized in terms of the platform, network, and management/diagnosis [21].

In addition, security technologies currently being developed or partially applied were classified. As the functions of automobiles related to electric platforms are advanced, high-performance ECUs (Electronic Control Unit) are being installed, and the application of the AUTOSAR standard platform is gradually expanding. AUTOSAR has included security standards since version 4.0 and has provided HSM-based security functions. VECTOR and ETAS Electrobit have AUTOSAR solutions. Currently, ECU-related security technologies include secure boot, secure flash, and secure access control, which are used not only as ECU protection but also as element technologies for safe remote updates such as SOTA/FOTA (Software-over-the-air/Firmware-over-the-air) [22–25].

As the connectivity of automobiles expands, research on the protection of head unit platforms is also being actively conducted recently. Some technologies such as application sandboxes and secure booting are being applied, and in particular, research such as platform

virtualization, Adaptive Autosar, Secure SocketCAN, and firewall is actively expanding the Linux security module (LSM) and controlling the kernel-level Socket API [26–28].

Technologies such as firewalls and IDS have been developed and partially applied to vehicle internal network security. Currently, it is centered on known threat detection technologies, and rule and whitelist-based detection technologies are the mainstream. In foreign countries, Harman, Agus, and SMT have developed firewall technologies for vehicles in Penta Security and Pescaro in Korea. Recently, research on artificial intelligence-based anomalies and abnormal behavior analysis technologies has been actively conducted to detect unknown threats. In the case of vehicle-to-vehicle V2X security, technology has been developed based on IEEE 1609.2 and CAMP VSC3 standards and is currently in the stage of establishing and demonstrating testbeds domestically and internationally [29]. IEEE 1609.2 is a communication security standard between the vehicle and the vehicle and between the vehicle and the road base station and includes information related to message authentication, integrity, confidentiality, and automobile certificates. Vehicle-to-vehicle authentication is applied with a vehicle PKI (Public key Infrastructure) technology called SCMS (Security Credential Management System) defined in IEEE 1609.2 and CAMP VSC3 Figure 2.

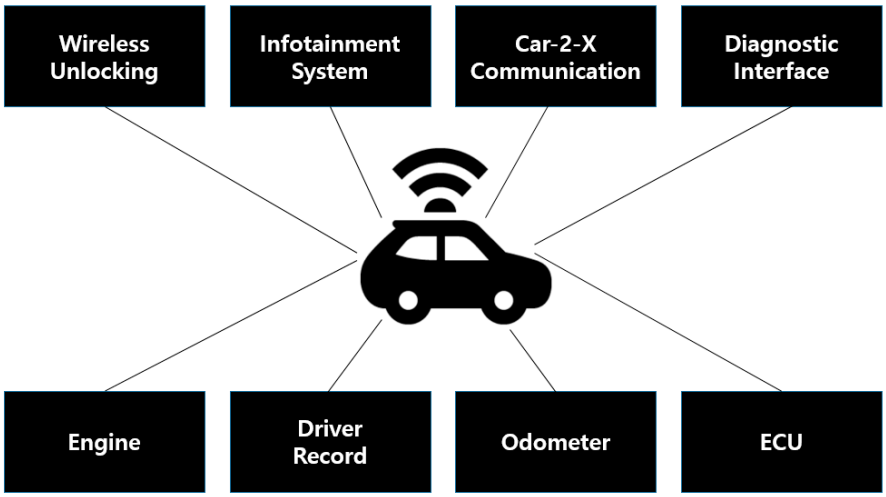

Figure 2. Vulnerability of Automobile Security (Source: Artificial News 2021).

2.2. Automobile Security Standards

Automobiles are evolving in line with these technology trends to meet the needs of users seeking smarter things [30–32].

As a result of this evolution, according to data from HIS (International Hybrid System) Markit, a market research firm released in 2018, sales of self-driving cars are expected to increase from 510,000 in 2021 to 33.7 million in 2040. The evolution of automobiles with technological innovation can provide many services for convenience and safety to relevant stakeholders, including users. However, there may be threats arising from this, one of which is the threat from cyberattacks. Following Miller and Valasek's announcement in 2015, studies on cyberattacks on automobiles have been conducted by several researchers, and vulnerabilities to automobiles' cybersecurity have been reported. Accordingly, NHTSA in the United States published guidelines for dealing with cybersecurity issues for automobiles in Best Practices for Cybersecurity of Automobiles 1. Meanwhile, in Europe, not only did the EVITA project aim at designing, verifying, and prototyping architectures for cybersecurity of vehicle internal communication networks, but ENISA also made recommendations to be carried out in the industry to ensure cybersecurity in smart cars in 'Smart Car' 2. Both documents 1 and 2 value processes that define the entire life-cycle (i.e., from initial conceptual design to development, production, operation, and scrapping) and engineering processes that can be managed and supported, which are recommended to

be integrated with existing engineering processes. In response to these demands, the US SAE standardized cybersecurity guidelines formed a WG11: Cybersecurity under the SC32 of ISO TC22. Further, ISO and SAE jointly began developing standard documents. WG11 developed related content with four part groups (PGs) under the workbench and integrated them to distribute the first CD edition in September 2018. In addition, the CD edition was revised twice, reflecting the opinions received from experts from various countries. The DIS (Draft International Standard) edition will then be distributed in February 2020 after reviewing by experts who participated in standard development in ISO and SAE, and document work in ISO Figure 3.

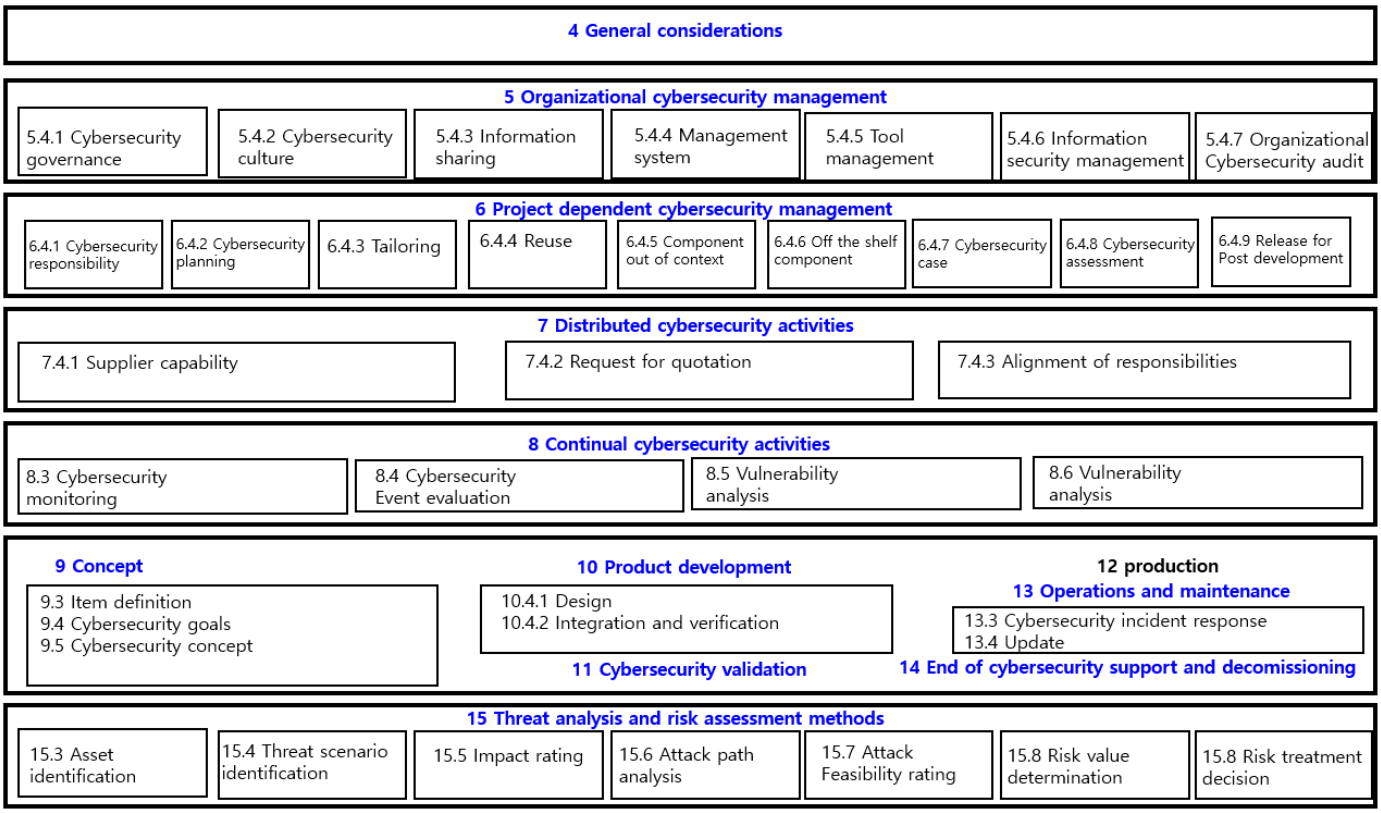

Figure 3. ISO/SAE 21434 Overall Composition (Source: ISO 21434 WG6, 2020).

The main parts modified from the first CD plate in the DIS plate are as follows. First of all, the overall structure and wording were modified to meet ISO's RISK Management standard [33–37]. In addition, the contents of cybersecurity management were divided into overall cybersecurity management and project-dependent cybersecurity management. In addition, cybersecurity monitoring and incident reporting, which have been dealt with in production and operation since development, were organized into separate sections as continuous cybersecurity activities, including vulnerability analysis and management. In addition, activities on distributed development, including manufacturers, suppliers, and third parties, were added. In addition, documents are numbered so that the requirements (RQ), recommendations (RC), allowances (PM), and work output (WP) specified in the documents can be easily distinguished. It was then described to facilitate the evaluation of the correlation between each requirement (RQ), recommendation (RC), input (Input), and work output. The overall composition of the document is shown in Figure 1. After the distribution of the DIS edition, experts from each country will collect opinions on the DIS edition until early May 2020 and review the collected opinions, and reflections on standard documents will proceed until October 2020. Next, it was decided whether to proceed to FDIS. Since the DIS edition is the final revision of the technical content, it is very important to collect opinions from experts from each country on the DIS edition scheduled to run

from February 12 to May 6. Therefore, it would be desirable not only for the automobile industry but also for for cybersecurity experts in the IT field to participate in the review of the DIS to express their opinions. In addition, as the next task of the WG11 workshop, discussions are expected to begin to add content on TAF (Target Attack Feasibility) and Cyber-Security Assurance Level (CAL) in the 21434 revision and standards that stipulate evaluation standards and related requirements for auto cybersecurity.

2.3. LSTM (Long Short-Term Memory)

The Current Neural Network (RNN) is a type of artificial neural network that features a cyclical structure of connections between units. This structure allows us to store states inside neural networks to model time-varying dynamic characteristics, so unlike forward neural networks, sequence-type inputs can be processed using internal memory. Therefore, the circulating artificial neural network may be applied to data processing that has time-varying characteristics, such as handwriting recognition or speech recognition [38–40].

The name cyclic neural network is given as it processes dynamic data without restrictions on the length of the input signal and represents both a finite impulse structure and an infinite impulse structure. The finite impulse cyclic neural network is a directional acyclic graph, so if properly solved and reconstructed, it can be expressed as a forward neural network, but it is impossible to represent an infinite impulse cyclic neural network as a directional graph [41].

Circular neural networks may have additional storage space. This storage space has a graph shape, so it can function as a time delay or have a feedback loop. Such storage space is referred to as a gate state or a gate memory and is a typical example in which the LSTM and the gate circulation unit (GRU) are equally applied.

LSTM has also been developed in the field of speech recognition and speech synthesis for large words and is currently applied to Google Android. In 2015, Google's voice recognition ability improved by 49% through CTC-based LSTM [42].

In addition, records in the fields of machine translation, language modeling, and multilingual language processing are being updated one after another with excellent abilities. Applied with convolutional neural networks, it has also made significant progress in the field of automatic image captioning. In order to reduce the hardware burden caused by the enormous amount of computation required to rotate the LSTM, research to accelerate the LSTM using a hardware accelerator is also ongoing.

LSTM (short- and long-term memory) is a deep learning system designed to solve the slope loss problem (English). The LSTM has an additional gate called a forget gate. Through this gate, it is possible to prevent a problem where the slope value rapidly disappears or increases during reverse propagation. This made it almost impossible for conventional RNNs to learn from distant past events. However, LSTM learned from events millions of units ago, allowing them to process not only high-frequency signals but also low-frequency signals, and soon developed their performance dramatically. As a result, many neural networks with similar structures to LSTM have been announced.

After accumulating LSTM, this neural network has been learned as a connectionist temporal classification (CTC) and is widely used in practical research fields. In particular, CTC is bringing good results in alignment and recognition. In addition, it was found that context-dependent language learning, which was impossible with the existing hidden Markov model (HMM), was possible.

In addition, short-term memory (LSTM) refers to a neural network structure designed to enable short- and long-term memory by compensating for the shortcomings of existing RNNs that do not remember information far from the output. It is mainly used for time-series processing or natural language processing.

Furthermore, the default RNN is a structure in which, when input x is input, output y comes out over several cycles. It consists of a chain-shaped structure to continue the information obtained in the previous step. The tanh layer is used as a module. RNN uses the back propagation through time (BPTT) backpropagation method (Figure 4).

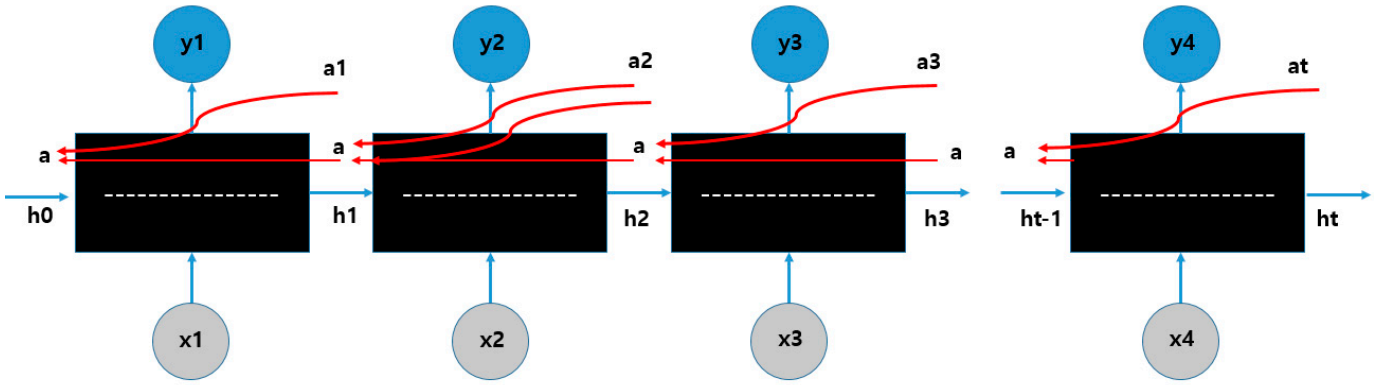

Figure 4. BPTT (Back Propagation Through Time).

This method reverses from beginning to end for each time step, and if the time step is large, it becomes a very deep network, resulting in gradient extinction/explosion problems. There is also the problem that long-term patterns cannot be learned. To solve this problem, an LSTM cell that can have a long-term memory has been proposed. LSTM also has the same chain structure as RNN, but the repeating module has a structure in which four layers exchange information with each other, not one tanh layer. In LSTM cells, states are largely divided into two vectors. Ht can be considered short-term, and ct can be considered long-term [43].

2.4. Blockchain Consensus Algorithm

The consensus algorithm is a key element of all blockchain networks and plays a role in maintaining the integrity and security of decentralized systems [44–48]. Proof of work (PoW), the first consensus algorithm, was designed and applied as a way to overcome the Byzantine disorder [49]. The consensus algorithm can be defined as a mechanism for achieving consensus on a blockchain network. Since the public (decentralized) blockchain consists of a decentralized system and does not rely on central authorities, decentralized nodes need to agree on the validity of transactions [50–52]. This is where the consensus algorithm begins. The consensus algorithm verifies that protocol rules are observed and ensures that all transactions proceed in a reliable manner so that the blockchain is used only once. Algorithms and protocols are often used interchangeably, but they are not exactly the same. In short, protocols are the basic rules of the blockchain, and algorithms can be defined as mechanisms that follow these rules [53–55].

Blockchain technology can be applied to various businesses and can be suitable for other uses. However, under any circumstances, the blockchain network will be built on top of a protocol that determines how the system works, so all elements of the system and network participants will have to follow the basic protocol rules. If the protocol stipulates what the rules are, the algorithm complies with these rules and instructs the system to go through the procedures to derive the desired results [56,57]. For example, the blockchain consensus algorithm determines the validity of transactions and blocks. It should also be considered that these protocols define how nodes interact, how data are transmitted, and the necessary conditions for successful block verification [58]. On the other hand, the consensus algorithm verifies signatures, approves transactions, verifies node information in automobiles, and actually validates blocks, all of which depend on network consensus. As mentioned earlier, consensus algorithms are important to maintain the integrity and security of cryptocurrency networks. The consensus algorithm allows decentralized nodes to agree on which version of the blockchain is the real version [59]. The high cost of mining makes it very difficult for minors to invest their resources to interfere with the network (Figure 5).

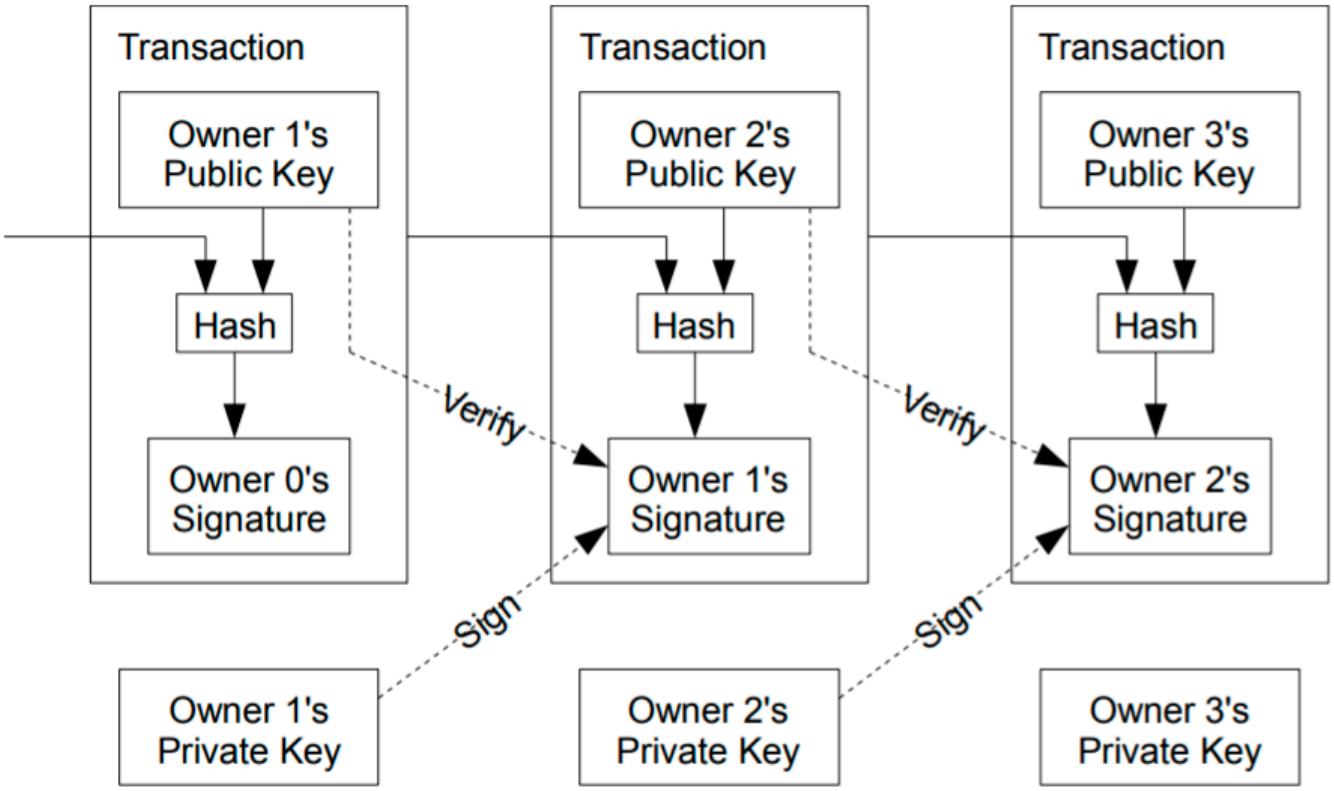

Figure 5. The Algorithm Structure of Bitcoin Consensus Algorithm (Source: Bitcoin Peer to peer system 2008).

3. Model Design

3.1. Issue Raising

In automobile security, an electronic control unit (hereinafter referred to as an electronic control system) as a communication system between automobiles is an engine control device that controls one or more of the automobile electrical systems or subsystems. The technological development of this system has led to the advent of automobile communication networks, which began with the controller area network (CAN), first developed by Bosch in 1988. The measurement control communication network contributed to making the existing automobile components connected in a wired manner, which ultimately reduces the weight of the vehicle and fuel efficiency significantly, thereby greatly improving vehicle performance.

Automobile communication networks are largely divided into in-vehicle networks and vehicular ad hoc networks. The internal network consists of a wired and wireless communication network between detectors or electronic devices, a measuring controller communication network (CAN) controlled by connecting the vehicle body and chassis parts, a vehicle network system (MOST) for multimedia access such as the audio, amplifier, and compact disk player. In addition, the external network is a wireless communication system that smoothly supports the external communication network of automobiles and is generally used to focus on supporting telematics services. First, the vehicle's internal network is a part of computer hardware for sensors or individuals in a vehicle and serves to connect all device files connected to the computer to devices and electronic components and to send control information data to the vehicle's electronic control system. In addition, in order to ensure the normal operation of the smart transportation system, integrity verification of the main setting values and execution codes must be accompanied. Integrity verification should be performed (required), user request, or periodically (selected) when starting the smart transportation system, and appropriate response procedures should be performed in case of integrity verification failure (error). In addition, vehicle applications

must prevent and detect abnormal updates and application changes through electronic signatures. In this study, research was conducted as a way to overcome new malicious codes in the future using artificial intelligence systems and blockchain consensus algorithms.

3.2. Research Methodology

This methodology was designed using artificial intelligence detection techniques to prevent new cyber-attacks such as CAN (Controller Area Network) command attacks through hacking and vulnerability in automobiles and smart car information theft attacks using "Sniffing" that steal network packets. Abnormal detection techniques are defined by dividing them into "normal" and "abnormal" values. In addition, LSTM's RNN method was introduced to take this methodology. Hackers who mostly hack cars have normal patterns, such as Jeongtam and misjudgment. In addition, as connectivity increases through these various wired and wireless interfaces, the attack vector has expanded.

Increased possibility of illegal infiltration of internal networks, limited exposure to attack detection such as random access using poor security design, spoofing and malicious data injection, increased malfunction of embedded devices through electronic control units, sensors, actuators, and memory attacks. This methodology initially designed a model. First, LSTM's Code Detection Model Design model is adopted. In addition, the Commission Calculation Process was designed to automatically detect false detection and false detection of hacking. To verify this information, the vanishing gradient problem model is used. Finally, a model was designed to prove the effectiveness of the final cyber threat using artificial intelligence LSTM gate verification, and in the existing LSTM.

By implementing a blockchain artificial intelligence consensus algorithm, we created a model to create consensus nodes between data. Through the artificial intelligence consensus algorithm, a model for finding consensus between nodes is completed, and, eventually, a model for the safety of automobiles made on possible nodes to a view change protocol and a deep neural network is presented in the methodology.

First, model design was used in the methodology. The second in the LSTM's Code Detection Model Design and Commission Calculation Process model is verified by the vanishing gradient program model and LSTM gate verification. Third, the blockchain is verified by the intelligence consensus algorithm and artificial. Finally, a methodology is presented using a view change protocol and deep neural network (Figure 6).

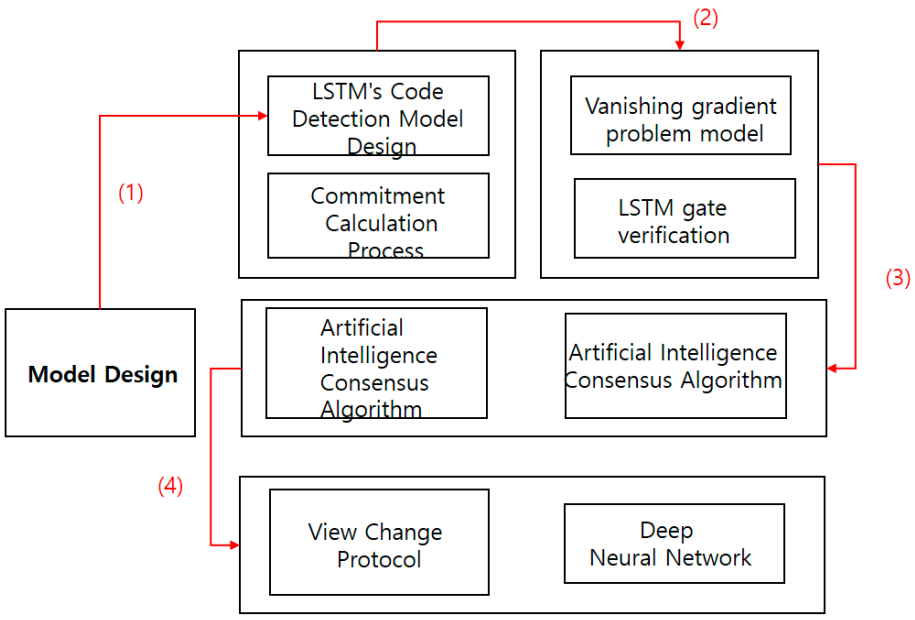

Figure 6. Research Methodology.

3.3. LSTM's Code Detection Model Design

First of all, the final output result along the forward pass is yt. The gradient (dL/dyt) of yt for the final loss appears first in the backpropagation operation of the RNN. For convenience, this was marked as dyt, and the net propagation result was marked in red compared to yt. We plan to follow this notation in the future.

Dyt is distributed in both directions on an addition graph. dWhy is obtained by multiplying the flowing gradient dyt by the local gradient ht. dht is the value obtained by multiplying the flowing gradient dyt by Why. Dhraw is obtained by multiplying the flowing gradient, dht, by the local gradient, $1 - \tanh 2$ (raw). The rest are obtained in the same way. However, it is necessary to pay attention to the picture below. RNN is a neural network in which hidden nodes have a circular structure. In other words, $ht - 1$ is reflected when making ht. In other words, $dht - 1$ in the figure below means that not only the gradient flowing from Loss at $t - 1$ but also the gradient corresponding are added and reflected at the same time. RNN is known to have a significant decrease in learning ability as the gradient gradually decreases during backpropagation when the distance between the relevant information and the point where the information is used is long. This is called a vanishing gradient program. It is LSTM designed to overcome this problem. LSTM is a structure in which the cell-state is added to the hidden state of the RNN (Figure 7).

$$\begin{aligned} \int\ t &= \sigma\left(Wxh_{fxt} + Whh_{fht} - 1 + bh_f\right) \\ it &= \sigma(Wxh_{ixt} + Whh_{iht} - 1 + bh_i) \\ ot &= \sigma(Wxh_{oxt} + Whh_{oht} - 1 + bh_o) \\ gt &= tanh\left(Wxh_{gxt} + Whh_{ght} - 1 + bh_g\right) \\ ct &= \int\ t \odot ct - 1 + it \odot gt \\ ht &= ot \odot \tan \mathrm{h}(ct) \end{aligned} \tag{1}$$

f_t has an important equation of LSTM's algorithm. In addition, i_t has an equation of negativity, and o_t has an important record. G_t records an important RNN linear algorithm for the value of tanh. C_t represents the nonlinear and linear algorithm of RNN. H_t refers to the linear algorithm of tanh's important algorithm.

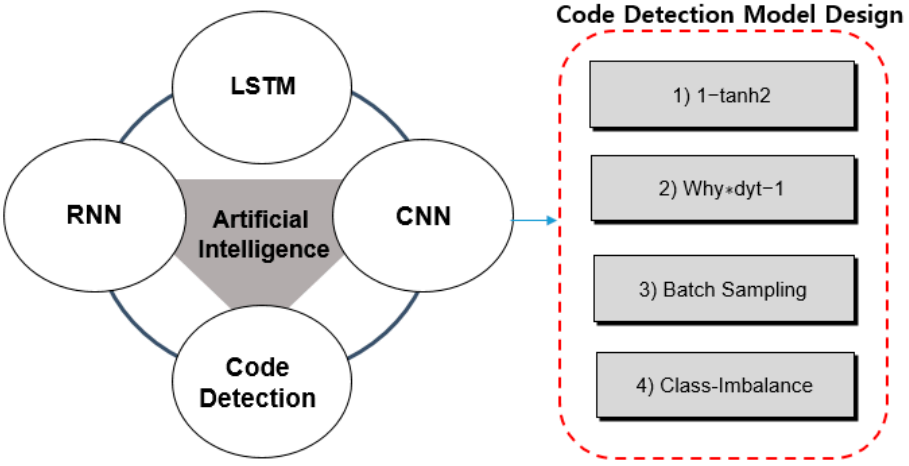

Figure 7. Code Detection Model Design.

In addition, if both the normal sample and the abnormal sample data and label exist in a given learning data set, it is called supervised anomaly detection because it is a supervised learning method. The supervised learning method has higher accuracy than other methods. Therefore, it is mainly used when requiring high accuracy, and the more abnormal samples are held, the higher the performance can be achieved. However, in general industrial sites where anomaly detection techniques are applied, the incidence of abnormal samples is significantly lower than that of normal samples. Therefore, in the case of supervised

learning, class-imbalance problems frequently occur. To solve this problem, various studies such as data augmentation, loss function redesign, and batch sampling are being conducted. Therefore, there is an advantage of high accuracy, and there is a disadvantage in that it takes a lot of time and cost to collect abnormal sample data. If both the normal sample and the abnormal sample data and label exist in a given learning data set, it is called supervised anomaly detection because it is a supervised learning method. The supervised learning method has higher accuracy than other methods. Therefore, it is mainly used when requiring high accuracy, and the more abnormal samples are held, the higher the performance can be achieved. However, in general industrial sites where anomaly detection techniques are applied, the incidence of abnormal samples is significantly lower than that of normal samples. Therefore, in the case of supervised learning, Class-Imbalance problems frequently occur. This kind of problem. In order to solve it, various studies such as Data Augmentation, Loss function redesign, and Batch Sampling are being conducted. Therefore, there is an advantage of high accuracy, and there is a disadvantage in that it takes a lot of time and cost to collect abnormal sample data.

3.3.1. Vanishing Gradient Problem Model

In order to overcome the vulnerability of communication between automobiles, RNN is known to have a significant decline in learning ability due to a gradual decrease in gradients during backpropagation if the distance between the relevant information and the points where the information is used is long. This is called a vanishing gradient program. It was designed and created to overcome this problem. LSTM is a structure in which cell-state is added to the hidden state of the RNN. The cell state acts as a kind of conveyor belt. Thanks to this, even if the state has elapsed for quite a long time, the gradient spreads relatively well. It is also a Hadamard product operator that means multiplication by element. The forget gate ft is a gate for 'forgetting past information'. The value obtained by receiving ht − 1 and xt and taking the sigmoid becomes the value exported by the forget gate. Since the output range of the sigmoid function is between 0 and 1, if the value is 0, information on the previous state is forgotten, and if it is 1, information on the previous state is fully remembered. Further, input gate it * gt is a gate for 'remembering current information'. The value obtained by receiving ht − 1 and xt, taking sigmoid, hyperbolic tangent with the same input, and then calculating Hadamard product becomes the value exported by the input gate. Further, since it ranges from 0 to 1 and gt ranges from −1 to 1, the strength and direction are shown, respectively. The pictures listed from now on are the same as the above GIF. First, the backward pass is similar to RNN until dft, dit, dgt, and dot are obtained.

$$\begin{aligned} l(= \varnothing,\ \mu 0,\ \mu 1, \sum\ 1) &= \log \mathrm{II}(x(i), y(i)\ ; \varnothing,\ \mu 0,\ \mu 1, \sum\ 1) \\ &= \log \mathrm{II}(x(i), y(i)\ ; \varnothing,\ \mu 0,\ \mu 1, \sum\ 1)\ p.\ (y(i); \varnothing) \end{aligned} \tag{2}$$

l represents an important equation of the Vanishing Gradient program model. Log values represent three-dimensional equations and represent important points of linear equations; it means the point of the RNN algorithm.

In addition, the process of obtaining dHt can be said to be the core of the LSTM backward pass for verifying automobile malicious code. Ht is a matrix composed of it, ft, ot, and gt. This structure can create dHt by combining the gradients corresponding to each (merge). The active function of I,f,o is sigmoid, and only g is hyperbolic tangent. Find the local gradient for each active function and multiply it by the gradient flowing in. In addition, the semi-supervised learning-based abnormality detection method requires a normal sample. In order for any of the numerous data to utilize only the normal sample, a label for the normal sample is required. Assuming that most data is a normal sample, the method of learning without a label is called unsupervised learning-based anomaly detection.

The simplest method is to detect anomalies through the process of reducing and restoring dimensions using principal component analysis (PCA). Autoencoder-based anomaly

detection techniques are widely used as a representative method of deep learning based on automobile malicious code analysis.

Autoencoder proceeds with encoding, which compresses input into a latent variable, and decoding, which restores it close to the original. It can be seen as similar to PCA in that data can be extracted compressively. Methods of detecting abnormalities using Autoencoder are as follows. However, since the value of the autoencoder's latent variable depends on the overall restoration performance according to hyperparameter, the accuracy is somewhat unstable compared to the supervised learning-based abnormality detection technique. However, the biggest advantage is that it can achieve good performance by utilizing only normal data sets without a separate labelling process.

3.3.2. Commitment Calculation Process

All nodes have priorities, and the higher the amount of steak, the higher the priority. If the amount of steak is the same, the hash value of the node ID is divided by n, and if the rest is the same, the node with the small ID is assigned a high priority. In addition, among the n or fewer commission members elected for each node for verification, the node with the highest priority is designated as the block creator, and the node with the next priority on a round basis plays the role of the block creator. Since p blocks are generated in one epoch, even commission members with pth priority play the role of block generators once. The consensus network uses a separate p2p network instance for exchanging consensus messages between committee members. The committee members participate on the consensus network only during the node verification period that generates p blocks and withdraws immediately after the node verification stage is completed. Further, it is possible to minimize the impact of other messages unrelated to the agreement by separating from the main network. Although malicious nodes are likely to access the consensus network and attack it, it is presumed that side effects can be minimized if normal committee members operate the routing table, considering their eligibility. Since there may be the same problem even when only the main network is used without separating the consensus network, separating and using the consensus network does not cause additional problems. In addition, as a pre-step, the first block creator broadcasts COMMITTEE_REQUEST and seed1 messages to all nodes.

$$\begin{aligned} &DKL(q(z)||p\,(z|x)\; DKL(q(z) + \log p(x) - EZ \sim P(z)[\log p(x|z)] \\ &= Ez \sim q(z)\log\left(\tfrac{q(z)}{p(z)}\right)^{\mathrm{n}} + \log(x) - Ez \sim p(z)[\log p(x|z)] \\ &= \tfrac{1}{\mathrm{K}}\sum_{i=0}^{K}\log\left(\tfrac{q(z)}{p(z)}\right)^{\mathrm{n}} zi \sim q(z) + \log(x) - \tfrac{1}{\mathrm{K}}\sum_{i=0}^{K}[\log p(x|z)]zi \\ &= \tfrac{1}{\mathrm{K}}\sum_{i=0}^{K}[\log(zi) - \log(zi) - \log p(x|zi)]zi \sim q(z) + \log p(\mathrm{x}) \end{aligned} \tag{3}$$

DKL represents a three-dimensional equation. It refers to an equation with various tables, and the equation *Ez~p*(*z*) at the *logp*(*x*) value represents the block encryption method. Further, various values were calculated.

Start TIMERjoin with a timer. In addition, node i executes a verifiable random function (VRF) using the received seed1 value to check the eligibility of the committee candidate. When the eligibility for the committee candidate is confirmed, the JOIN message JOIN and epoch 1 are sent to the first block creator. Receives JOIN messages during TIMERjoin by prof. the first block creator who is qualified through VRF. After that, the first block generator sorts the committee candidates in priority order and selects the top n candidates as committee members, broadcasts the message ELECT and epoch 1, committee and list to verify them. The committee and list refer to a list of the selected committee members' lists and proofs.

3.3.3. LSTM Gate Verification

Among the logistic functions shown in the classification for LSTM gate verification, the sigmoid function and the logistic function are the same S-shaped functions, which convert the value of X to the value of 0–1. Meanwhile, tanh is converted into a value between -1 and -1 instead of 0 to 1, as the word hyperbolic (hybolic) is attached. Unlike sigmoid, the output of tanh may have positive and negative values, so it is possible to increase or decrease the state. Therefore, it is used for repeated connection of tanh cells and is useful for determining candidate values added therein. It is a good function to solve the slope loss problem because it can maintain a long-term value before the secondary differential becomes zero. It also converts information into a good state to use. Tanh does not flow recurrent information (memory information) from the previous cell as it is but distinguishes the point.

(1) A memory line is added to the output of all cells (ht $-$ 1 and Ct $-$ 1).

In a simple RNN, information transfer of the previous cell, which was one, is composed of two lines by adding memory in addition to the output. Roughly speaking, the recurrent side is a short-term memory, such as RNN, and the memory is long-term memory. LSTM, like its name, preserves memories in different lines while associating short and long term.

(2) The output (Recurrent) of all cells and the combination of inputs (ht $-$ 1 and Xt)

Output ht $-$ 1 (short-term memory) of the previous cell and input Xt of the current cell merge. The joined signal is copied the same information as the branch to the four lines. The result of this joining is that the input value "malicious code" is added to the short-term memory of "automobile hacking. It is also the same as the previous RNN.

(3) Forgetting gate (output of ft)

The top line is the oblivion gate. This is to select and select information for each long-term memory in the previous cell with a value ft between 0 and 1 from sigmoid function. You have to leave all 1 and throw all 0 away. When it is determined that 'pure' in long-term memory is not important at the time (t) when 'car hacking' is recognized by short-term memory ht $-$ 1 and input Xt, the output ft of 0–1 becomes a value near 0, forgetting this memory. Meanwhile, the information "malicious code" seems to be important, so ft remains at 1.

(4) Input gates (Ct' and it)

After converting input data summed up by short-term memory ht $-$ 1 and input Xt for long-term preservation, it controls which signal to be stored in long-term memory by how much weight. This is handled in two steps.

- Transformation by tanh (outputting Ct'): Instead of flowing the information as it is, the amount of information can be reduced and is good to use. Further, tanh was said to be a useful function for determining candidate values added internally. For example, "I love you, unwilling" is replaced by a candidate for "I like you." In this way, it is simply converted to output Ct'.
- Screening by input gate (it): LSTM adjusts the weight by back propagation through time. Normal error backpropagation is the weight control of the input Xt, but synchronic error backpropagation is also affected by the information of the short-term memory ht $-$ 1 in the previous cell. Therefore, in order to prevent the weight from being incorrectly updated by irrelevant information coming from ht $-$ 1, the input gate is controlled to properly transmit only the required error signal. In addition, from the information "Younghee's Unwilling Cake" made of ht $-$ 1 + Xt, the input gate (sigmoid function) selects what to leave and what to let go.

(5) Output gate (outputting ot)

ht is the output of short-term memory. Through the above process, short-term memory is added to long-term memory, and only the short-term memory is output from the selected value (output Ct of long-term memory). Here, as before, it is processed in two steps.

- Transformation by tanh: Tanh's input is the long-term memory Ct − 1 in the previous cell plus the short-term memory Ct' converted Xt. Each is selected as an oblivion gate and an input gate. Ct is to print this out as it is as long-term memory, but it is easier to convert than when there is only short-term memory because it also includes short-term memory.
- Short-term memory selection: Just as the input gate protected the cell by itself, the output gate also prevented the propagation of bad information about the next cell. When updating the weight ht for activating the next cell, related information should be leaked so that it does not have a bad effect. Ot is output in the range from 0 to 1 by the output gate (Sigmoid function), and only the signal required for the short-term memory output ht is controlled to be properly transmitted (Figure 8).

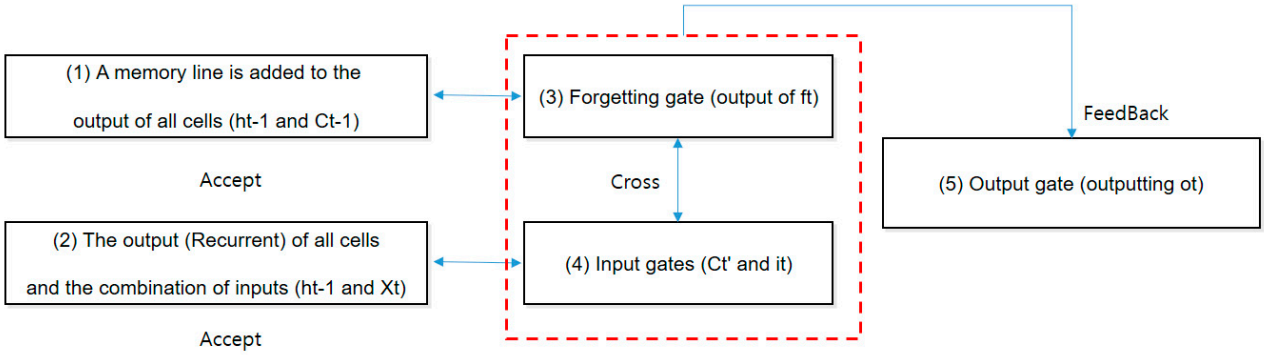

Figure 8. LSTM Gate Verification.

3.4. Artificial Intelligence Consensus Algorithm

It is a consensus algorithm that adds artificial intelligence (AI) functions to the existing method of proving the delegated stake. It is a consensus algorithm used for an artificial intelligence-based blockchain platform. The artificial intelligence consensus algorithm basically follows the existing method of proving the delegated stake, but it is characterized by the fact that artificial intelligence autonomously judges and selects a representative to give it the authority to create blocks. Through this, we are trying to solve the problems of alternatives to consensus algorithms, including Bitcoin's traditional proof of work system and Piercoin and NXT's proof of share system. In addition, by using other parameters to determine the most operational and efficient nodes, including the reliability of the nodes themselves, and allowing only these nodes to agree, malicious or corrupted nodes are removed, and the entire blockchain is protected. Each node is rated by Bellas artificial intelligence. Ratings create a reputation for the Bellas system to determine nodes to select in the future. Node selection depends on four basic elements: (1) the number of transactions (2) the staking point (3) the block creation (4) the operating time. Artificial intelligence delegation equity proof uses the order of algorithms to distinguish relationships and patterns in a series of data. Neural networks adapt themselves with new inputs and strive to achieve optimal results. Prior to commercial involvement, neural networks use genetic algorithms to ideally maintain platforms and successfully create blockchains. Blockchain transactions are demonstrated by the staking process, and the more cryptocurrencies an actor has, the more likely it is to be selected to verify the blockchain. Verifiers receive financial rewards for their active participation in the network. Neural networks, part of the artificial intelligence delegation equity certification system, are used to make accurate decisions about the platform, including compensation for node operators and time calculations required to form blocks. The time until a new block is created is determined by the transaction calculated per second. The more transactions are completed, the shorter the time to form the next block.

3.4.1. Artificial Intelligence Consensus Algorithm

It is a consensus algorithm that adds artificial intelligence (AI) functions to the existing method of proving the delegated stake. It is a consensus algorithm used for an artificial intelligence-based blockchain platform. The artificial intelligence consensus algorithm.

Among the logistic functions shown in Classification' for LSTM gate verification, the Sigmoid function and the logistic function are the same S-shaped functions, which convert the value of X to the value of 0–1. Meanwhile.

3.4.2. View Change Protocol

To replace the view change conditions and block constructors, step 1 refers to the case where the M_propose message received from Propose is invalid, or step 2 refers to the case where multiple blocks are proposed if an agreement is not reached in Vote. When the TIMERview_change expires and the block creator does not propose a block, a view change is attempted. If more than half of the committee members are Byzantine errors, view change may fail, and rollback (reversal of agreement) may also be required after consensus is reached. In this case, replacement of the entire commitment is required, and for this purpose, replacement of the previous epoch with the commitment is executed. In addition, if the view change is successful, it is designated as a block constructor from a commissioner with a p + 1 priority. If the epoch's committee member is less than 2p, and block constructor replacement occurs more than p times, the next block constructor is re-designated from the first committee member. In addition, rotation occurs.

3.5. Ant Colony

Ant colony is the basic unit by which ants organize their life cycle. Ant colonies are socially, communal, and efficiently organized, very similar to those found in other social groups, but independently developed sociality through convergent evolution. A typical colony consists of a queen ant that lays one or more eggs, numerous females and, depending on the season, winged males and females. To establish new colonies, the ants make species-specific flights during the day. The swarm of winged adults leaves the nest in search of another nest. The males die soon along with most of the females. Only a small percentage of the females survive and build new nests. Colony size is very important to ants: it can affect how they find food, how they guard their nests, how they mate, and even their appearance. Body size is often considered the most important factor in shaping the natural history of non-colonial organisms; Likewise, colony size is a key influencing how the colony is collectively organized. However, colony sizes vary greatly from ant species to ant species: some ants are just a few ants that live on tree branches, while others are super colonies with millions of worker ants. Even a single ant colony can show significant seasonal variation. For example, in the ant Dolichoderus mariae, a colony can migrate from about 300 workers in summer to more than 2000 workers per queen in winter. Variation may be greater. Zooming out further, the differences within related populations of different ant species can be enormous. Formica yesensis is reported to have colonies of 306 million workers, whereas colonies of Formica fusca sometimes have only 500 workers.

3.6. Automobile Malicious Code Verification Composite Diagram

3.6.1. Automobile Malicious Code Verification Composite Diagram

I drew a composite dilagram of the automobile malicious code verification source code. Objects of AutomobileBlock's VisitUnAssistance and VisitVariable and classes of TypeCheckingBlock, CodeGeneratingBlock, BlockNode, AssignmentBlock, and VariableRef-Block were designed (Figure 9).

3.6.2. Automobile Malware Verification Source Code

Some codes of automobile malicious code verification source codes were inserted. The source code put Algorithms A1.pyc into the Appendix A.

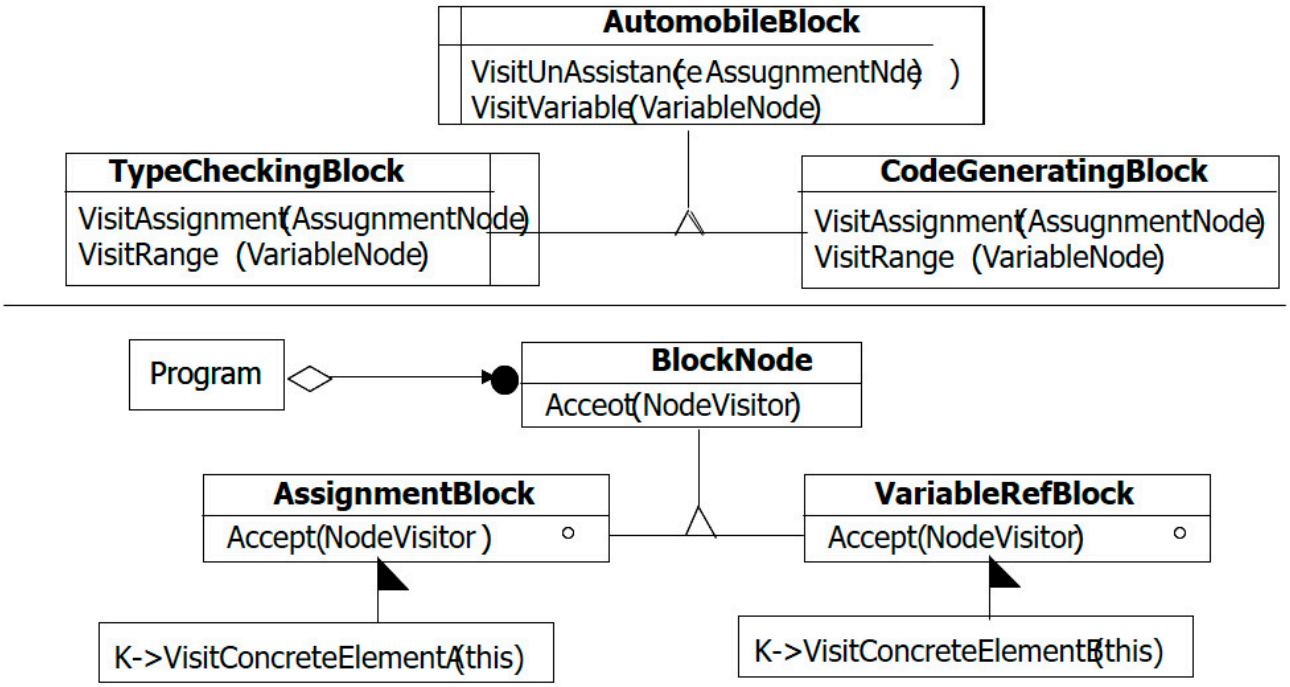

Figure 9. Automobile Malicious Code Verification Composite Diagram.

4. Experiments and Results

For the environment for conducting the experiment and the existing experimental environment, ML conducted the experiment using the Beta version. In addition, as a method for multinomial experiments, the functions of Agent and Academy were separated from the existing Unity ML, and the TPS of the Deep Learning and Blockchain consensus algorithm was also composed of an experimental environment.

4.1. Experiment Environment and Data Set

The experiment of this paper is as follows.

- CPU Intel Core i9-9900
- GPU GeForce RTX 2070
- MEMORY 64GB
- EngineVersion Unity 2019.2,13
- Tensorflow Version 2.2.2
- Unity ML ML release 2

The experimental environment was constructed together. Moreover, the existing Unity ML was shared as Beta version, but since the recent release of the official version, Release 1, it has continued to release and serve the official version. The structure is slightly different from the previous beta version, and the biggest is the deletion of Academy and Brain. In the existing Unity ML, the functions of Agent and Academy were separated, and the subject of learning and the non-subject of learning were divided. However, from the release version, the Brain file was deleted, and the nn file was added while managing it integrally. The nn file is a model file derived after learning through machine learning. It also shows the difference between the ML Agent beta version and the official version. In addition, 50,000 automobile malicious codes were used as test data. I observe the environment, make decisions based on that environment, and act based on that decision. The changed state due to the actions taken has rewards and a new environment. It returns to observing this new environment and repeats this process, and as a result, the agent will learn to take action in the direction of maximizing the reward. After importing Unity.MLAgents, the custom agent class included therein must be defined. Here, the agent is the object that acts.

If you define the class as an agent and look at the file imported using the visual studio's see definition, you can see under what name the agent and functions are defined by. This file contains all the functions that perform all of the above processes. By overriding these functions, we are learning AI to suit our problems. Environmental observation can be performed by overriding MLAgent's CollectObservation, which first requires an observer and defines what that subject should observe. The subject to be observed is observed through sensors defined in MLAgents.Sensors. By registering the values that we want to observe in the sensor, we can observe them according to the problem.

In summary, sensors define how agents look at the environment and depending on the problem, other variables to be observed must be defined and registered according to the problem with this sensor. For example, let us check this problem in which the character approaches the ball without scribbling. In order for the character to approach the ball without scribbling, it is necessary to know the position of the character and the position of the ball. Let us add this variable to the sensor and specify that the sensor observes these variables.

You may think that the location of each component only needs to receive two variables to observe, but it does not. Because the position consists of x, y, and z, the space size must be set to 6. Additionally, if you want to learn using the previous environment value together, you can configure the stacked vector as a value other than 1. This value registers transform in the component and registers through the sensor's addObservation in the above function. Now that you have registered the sensor let us define in OnActinoReceived what value you will receive and what action you will take when determining your behavior.

The OnActinoReceived function may receive action as ActionBuffers and set a logic to determine the "decision" when the action is performed. Here, MLAgent can recognize all values, including all actions/rewards, only by number and cannot be understood by actions, so let us remind ourselves in advance that we have to define what values and how each action changes according to the problem. If you look at the script in which the OnActinoReceived function is defined through Unity Inspector, you will be able to set up the Behind Parameters.

Here, the beach parameters is a parameter that defines an event when an action is received and may define the type of behavior value to be received or the number of inputs to be received, and the branch I size of the value to be received. ActionBuffers, which we checked earlier, have different member buffers depending on the type of behavior value, and if it is discrete, it should be approached with actions. Discrete actions.

4.2. Experimental Evaluation Method

The LSTM blockchain consensus node network consists of 50,000 processes. To this end, Amazon cloud services are needed. For now, it is desirable to implement an experiment using only a few processes on several servers. After the implementation is stabilized, the experiment is conducted on the cloud. In addition, a separate NTP server and performance monitor are used for performance measurements, and clock synchronization of NTP-based nodes is performed to perform accurate verification. In addition, the clock of each node is synchronized and tested using s network time protocol (NTP).

For this purpose, a separate clock synchronization server is used, and according to the NTP protocol, the clock error of each node is smaller than and the clock value is recorded when transmitting and receiving messages at all nodes to process the time and latency required for agreement. Further, throughput and TPS, which are throughput, are measured, and performance measurement is implemented. TIMERjoin, TIMERmessage_delay, and TIMERview_change are determined, and the clock values written in the message are analyzed to calculate the latency and throughput required for consensus.

4.3. Experimental Results and Evaluation

The experiment first looked at the experimental result values for the existing malicious code verification and detection techniques. In addition, performance comparisons were

made in the paper based on the artificial intelligence LSTM algorithm and the blockchain consensus node for algorithms in which malicious codes are detected in communication between automobiles. In addition, an experiment was conducted to obtain the optimal weight to compare the accuracy of the system between vehicles moving objects considering the weight and the simple recommendation system. Finally, the excellence of the system proposed in this paper is proved through a comparison between the mobile system, the fixed system, and the mobile object security performance system with weights.

4.4. Experimental Evaluation Method

For the first time, 100 Grouped Tx, 500 Channels were tested for performance. For the first time, the malicious code detection rate of the existing system was verified. Accelerator, Multichannel, Sharding, Raiden, Plasma, and Trubit values were verified, and values of approximately 15,000 to 50,000 (Figure 10).

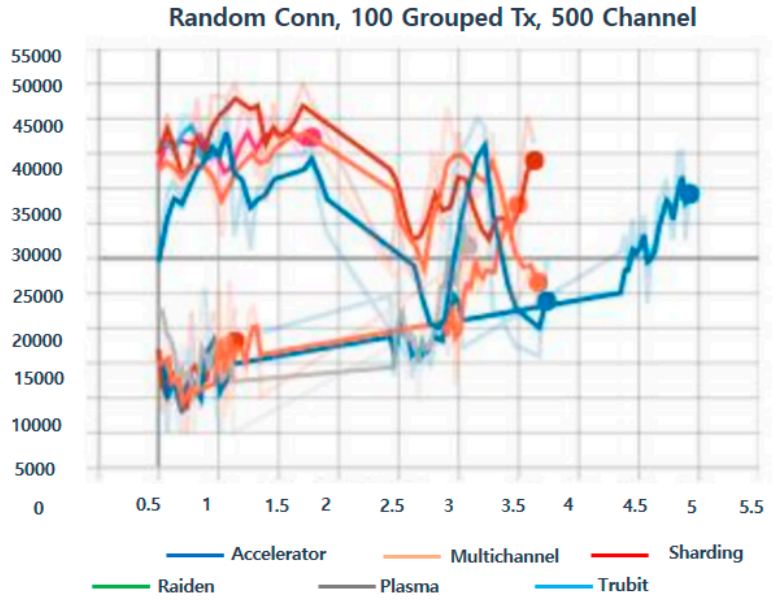

Figure 10. 100 Grouped Tx, 500 Channel.

In addition, verification was conducted based on artificial intelligence LSTM with 1000 grouped Tx and 5000 channels. Further, the blockchain consensus algorithm was improved and verified. The range ranged from 0 to 45,000 with even Accelerator, Multichannel, Sharding, Raiden, Plasma, and Trubit values. In addition, the performance was very stable (Figure 11).

In addition, the existing research method was verified by not using the LSTM method of artificial intelligence and by inserting artificial intelligence. This showed a very meaningful result value. As a result of measuring in the existing method, the verification was the highest when the experiment was carried out for about 300 min, and when the measurement was carried out with a new artificial intelligence mounted algorithm, the optimal verification model was formed when it was about 0 to 60. This means that the accuracy of verification increases in order to achieve optimal performance (Figure 12).

I also tried to verify more things. New, Next, Existing, Phase, and Basic were analyzed. This resulted in a very interesting result. New was optimized for verification between 0 and 43. When the experiment was conducted, Next was 0–45, Existing was 0–48, Phase was 0–55, and Basic was 0–60, indicating that the verification value is the highest (Figure 13).

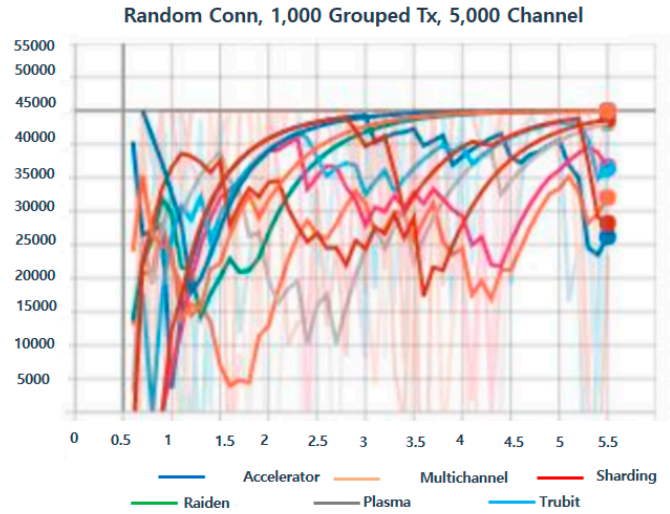

Figure 11. 1000 Grouped Tx, 5000 Channel.

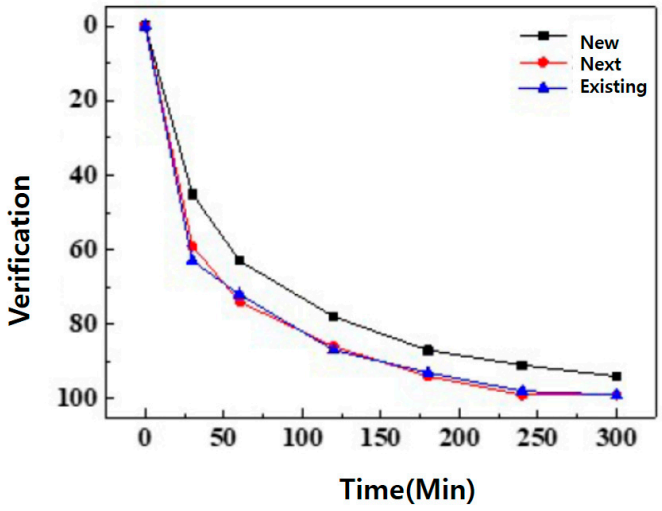

Figure 12. The new from the existing method.

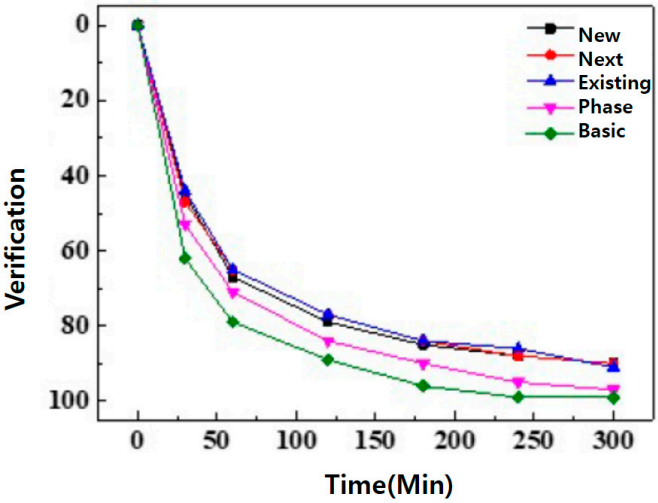

Figure 13. The difference from the existing method.

5. Conclusions and Future Work

In this paper, we prove that if any commissioner I determines block X as the nth block, the block that any other commissioner j determines as the nth block is always X. It is also proved using the return method. Suppose that the block that commissioner j determines as the nth block is Y, and Y is different from X. For Y to be determined, Q = f + ((n – f + 1)/2 or more vote (commit message) is required for agreement. An important property of Q is that the sets of commission members above Q always have normal commission members as intersections. Therefore, one or more normal commission members who have voted to X must also vote to Y, which is only possible if the normal node violates the consensus rule. If a Byzantine error is detected during the consensus process, AQ is used instead of Q, and the above proof is valid even if AQ is used. In addition, it proves that it is possible to reach an agreement only with normal committee members even if abnormal committee members do not cooperate with the agreement to prove their vitality. If the block constructor is abnormal, the block constructor does not send a prepared message to all commissioners.

In the case, and the case where the block creator prepares different blocks may be classified. In this case, it is possible to replace it with a normal block generator through a view change. In addition, if the block constructor is normal and f abnormal committee members do not send messages within, all f abnormal committee members can detect Byzantine errors, in which case b = f at AQ = f − b + (n − f + 1)/2 to secure AQ = (n − f + 1)/2 or higher. If the block constructor is normal, and f abnormal commission members vote for multiple different blocks, it is possible to detect a Byzantine error in all f abnormal commission members vote to multiple blocks, in which case AQ = f − b + (n − f + 1)/2 to secure AQ = ((n − f +/21) or more. In addition, to achieve the development of optimized consensus algorithms, high-expansion and high-trust consensus algorithms are developed, and the maximum transaction throughput, 5000 TPS, transaction final processing time, and maximum number of users remain at around 50,000, and the core properties of the consensus algorithm were proved theoretically and mathematically. The safety and vitality of the studied consensus algorithm were proved. In addition, it is expected that the high-performance and high-reliability consensus algorithm, which will be developed using high-performance and high-reliability consensus algorithms in the future, can be applied to new mainnet development or existing mainnet to raise the performance and reliability of blockchain to practicality. In addition, IDC expects the global blockchain market to reach about 100 trillion won by 2022 and predicts that the market will be formed around finance, economy, government, public, medical, health, retail and distribution. Considering that the most important core technology of the blockchain is the consensus algorithm, the result of this task is expected to have a huge economic ripple effect. In addition, if the consensus algorithm evaluation technology, which is an output for the use of blockchain evaluation methodology and tools, is expanded to evaluation and verification technology for blockchain platforms and smart contracts, the technical and industrial ripple effect will be very high.

Furthermore, as a result of the experiment for the algorithm for vehicle verification, the performance test was performed on 100 Grouped Tx, 500 Channel for the first time. First, the malware detection rate of the existing system was verified. Accelerator, Multichannel, Sharding, Raiden, Plasma, and Trubit values were verified, and values between 15,000 and 50,000 were obtained.

In addition, it was verified based on artificial intelligence LSTM with 1000 Grouped Tx and 5000 Channels, and the blockchain consensus algorithm was improved and verified.

Funding: This paper was supported by Joongbu University Research & Development Fund, in 2020.

Conflicts of Interest: The author declares no conflict of interest.

Abbreviations

AI	Artificial Intelligence
BPTT	Back Propagation Through Time
CAMP	Crash Avoidance. Metrics Partnership
CAN	Car Area Network
CTC	Connectionist Temporal Classification
DIS	Draft International Standard
DoS	Denial-of-Service
ECU	Electronic Control Unit
GPS	Global Positioning System
ICBM	IoT, Cloud, Big Data, Mobile
HIS	Internatinal Hybrid System
IT	Information Technology
IoT	Internet of Things
IDS	Intrusion Detection System
LSM	Linux security module
LSTM	Long short-term memory
NV	Night Vision
PCA	Principal Component Analysis
PoS	Proof of Stake
PoW	Proof of Work
RNN	Recurrent neural network
SCMS	Security Credential Management System
SOTA/FOTA	Software-Over-The-Air/Firmware-Over-The-Air
PKI	Public Key Infrastructure

Appendix A

Algorithms A1.pyc

```
def lossFun(inputs, targets, hprev, cprev):
xs, hs, cs, is_, fs, os, gs, ys, ps= {}, {}, {}, {}, {}, {}, {}, {}, {}, {}
hs[−1] = np.copy(hprev)
cs[−1] = np.copy(cprev)
loss = 0
H = hidden_size
# forward pass
for t in range(len(inputs)):
xs[t] = np.zeros((vocab_size, 1))
xs[t][inputs[t]] = 1
tmp = np.dot(Wxh, xs[t]) + np.dot(Whh, hs[t − 1]) + bh # hidden state
is_[t] = sigmoid(tmp[:H])
fs[t] = sigmoid(tmp[H:2 * H])
os[t] = sigmoid(tmp[2 * H: 3 * H])
gs[t] = np.tanh(tmp[3 * H:])
cs[t] = fs[t] * cs[t − 1] + is_[t] * gs[t]
hs[t] = os[t] * np.tanh(cs[t])

# compute loss
for i in range(len(targets)):
idx = len(inputs) − len(targets) + i
ys[idx] = np.dot(Why, hs[idx]) + by # unnormalized log probabilities for next chars
ps[idx] = np.exp(ys[idx]) / np.sum(np.exp(ys[idx])) # probabilities for next chars
loss += −np.log(ps[idx][targets[i], 0]) # softmax (cross-entropy loss)

# backward pass: compute gradients going backwards
dWxh, dWhh, dWhy = np.zeros_like(Wxh), np.zeros_like(Whh), np.zeros_like(Why)
dbh, dby = np.zeros_like(bh), np.zeros_like(by)
dhnext, dcnext = np.zeros_like(hs[0]), np.zeros_like(cs[0])
n = 1
a = len(targets) − 1
for t in reversed(range(len(inputs))):
if n > len(targets):
continue
dy = np.copy(ps[t])
dy[targets[a]] −= 1 # backprop into y
dWhy += np.dot(dy, hs[t].T)
dby += dy
dh = np.dot(Why.T, dy) + dhnext # backprop into h
dc = dcnext + (1 − np.tanh(cs[t]) * np.tanh(cs[t])) * dh * os[t] # backprop through tanh nonlinearity
dcnext = dc * fs[t]
di = dc * gs[t]
df = dc * cs[t − 1]
do = dh * np.tanh(cs[t])
dg = dc * is_[t]
ddi = (1 − is_[t]) * is_[t] * di
ddf = (1 − fs[t]) * fs[t] * df
ddo = (1 − os[t]) * os[t] * do
ddg = (1 − gs[t]^2) * dg
da = np.hstack((ddi.ravel(),ddf.ravel(),ddo.ravel(),ddg.ravel()))
dWxh += np.dot(da[:,np.newaxis],xs[t].T)
dWhh += np.dot(da[:,np.newaxis],hs[t − 1].T)
dbh += da[:, np.newaxis]
dhnext = np.dot(Whh.T, da[:, np.newaxis])
n += 1
a −= 1
for dparam in [dWxh, dWhh, dWhy, dbh, dby]:
np.clip(dparam, −5, 5, out = dparam) # clip to mitigate exploding gradients
return loss, dWxh, dWhh, dWhy, dbh, dby, hs[len(inputs) − 1], cs[len(inputs) − 1]
```

References

1. Tellegen, A.; Watson, D.; Clark, L.A. On the Dimensional and Hierarchical Structure of Affect. *Psychol. Sci.* **1999**, *10*, 297–303. [CrossRef]
2. Patra, B.G.; Maitra, P.; Das, D.; Bandyopadhyay, S. Mediaeval 2015: Music emotion recognition based on Feed-forward neural network. In Proceedings of the MediaEval 2015 Workshop, Wurzen, Germany, 14–15 September 2015.
3. Chen, S.-H.; Lee, Y.-S.; Hsieh, W.-C.; Wang, J.-C. Music emotion recognition using deep Gaussian process. In Proceedings of the 2015 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA), Hong Kong, China, 16–19 December 2015; pp. 495–498.
4. Bargaje, M. Emotion recognition and emotion based classification of audio using genetic algorithm-an opti-mized approach. In Proceedings of the 2015 International Conference on Industrial Instrumentation and Con-trol, Pune, India, 28–30 May 2015; pp. 562–567.
5. Malik, M.; Adavanne, S.; Drossos, K.; Virtanen, T.; Ticha, D.; Jarina, R. Stacked Convolutional and recurrent neural networks for music Emotion Recognition. In Proceedings of the 14th Sound and Music Computing Conference, Espoo, Finland, 5–8 July 2017; pp. 208–213.
6. Wang, Z.; Peterson, J.L.; Rea, C.; Humphreys, D. Special Issue on Machine Learning, Data Science, and Artificial Intelligence in Plasma Research. *IEEE Trans. Plasma Sci.* **2020**, *48*, 1–2. [CrossRef]
7. Picard, R.W. Computer learning of subjectivity. *ACM Comput. Surv.* **1995**, *27*, 621–623. [CrossRef]
8. Darwin, C. *The Expression of Emotions in Animals and Man*; University of Chicago Press: London, UK, 2015.
9. Wang, Y.; Kwong, S.; Leung, H.; Lu, J.; Smith, M.H.; Trajkovic, L. Brain-Inspired Systems: A Transdisciplinary exploration on cognitive cybernetics, humanity, and systems science toward au-tonomous artificial intelligence. *IEEE Syst. Man Cybern. Mag.* **2020**, *6*, 6–13. [CrossRef]
10. Watkins, T. Cosmology of artificial intelligence project: Libraries, makerspaces, community and AI litera-cy. *ACM AI Matters* **2019**, *4*, 134–140. [CrossRef]
11. Seo, Y.-S.; Huh, J.-H. Automatic emotion-based music classification for supporting intelligent IoT applications. *Electronics* **2019**, *8*, 164. [CrossRef]
12. Lee, Y.; Rathore, S.; Park, J.H.; Park, J.H. A blockchain-based smart home gateway architecture for preventing data forgery. *Hum. Cent. Comput. Inf. Sci.* **2020**, *10*, 1–14. [CrossRef]
13. Seo, Y.-S.; Huh, J.-H. Context-aware auction solution of cooperative fish market monitoring system for intelligent user. *Hum. Cent. Comput. Inf. Sci.* **2020**, *10*, 1–36. [CrossRef]
14. Park, J.S.; Park, J.H. Advanced technologies in Blockchain, machine learning, and big data. *J. Inf. Processing Syst.* **2020**, *16*, 239–245.
15. Woo, H.; Jeong, S.-J.; Huh, J.-H. Improvement of ITSM it service efficiency in military electronic service. *J. Inf. Processing Syst.* **2020**, *16*, 246–260.
16. Rahmadika, S.; Noh, S.; Lee, K.; Kweka, B.J.; Rhee, K.H. The dilemma of parameterizing propagation time in blockchain P2P network. *J. Inf. Processing Syst.* **2020**, *16*, 699–717.
17. Yuan, Y.; Huh, J.-H. Automatic pattern setting system reacting to customer design. *J. Inf. Processing Syst.* **2019**, *15*, 1277–1295.
18. Salim, M.M.; Shanmuganathan, V.; Loia, V.; Park, J.H. Deep learning enabled secure IoT handover authentication for blockchain networks. *Hum. Cent. Comput. Inf. Sci.* **2021**, *11*, 21.
19. Kim, Y.; Chung, M.; Chung, A.M. An Approach to Hyperparameter Optimization for the Objective Function in Machine Learning. *Electronics* **2019**, *8*, 1267. [CrossRef]
20. Huh, J.-H.; Kim, S.-K. The Blockchain Consensus Algorithm for Viable Management of New and Renewable Energies. *Sustainability* **2019**, *11*, 3184. [CrossRef]
21. Kim, S.-K.; Kwon, H.-T.; Kim, Y.-K.; Park, Y.-P.; Keum, D.-W.; Kim, U.-M. A Study on Application Method for Automation Solution Using Blockchain dApp Platform. In *International Conference on Parallel and Distributed Computing: Applications and Technologies*; Springer: Singapore, 2019; pp. 444–458.
22. Bengio, Y.; Courville, A.; Vincent, P. Representation Learning: A Review and New Perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *35*, 1798–1828. [CrossRef]
23. Luo, W.; Lu, J.; Li, X.; Chen, L.; Liu, K. Rethinking Motivation of Deep Neural Architectures. *IEEE Circuits Syst. Mag.* **2020**, *20*, 65–76. [CrossRef]
24. Wang, X.; Lin, X.; Dang, X. Supervised learning in spiking neural networks: A review of algorithms and evaluations. *Neural Netw.* **2020**, *125*, 258–280. [CrossRef]
25. Enríquez-Gaytán, J.; Gómez-Castañeda, F.; Flores-Nava, L.; Moreno-Cadenas, J. Spiking neural network approaches PCA with metaheuristics. *Electron. Lett.* **2020**, *56*, 488–490. [CrossRef]
26. Liu, C.; Shen, W.; Zhang, L.; Du, Y.; Yuan, Z. Spike Neural Network Learning Algorithm Based on an Evolutionary Membrane Algorithm. *IEEE Access* **2021**, *9*, 17071–17082. [CrossRef]
27. Al-Garadi, M.A.; Mohamed, A.; Al-Ali, A.K.; Du, X.; Ali, I.; Guizani, M. A Survey of Machine and Deep Learning Methods for Internet of Things (IoT) Security. *IEEE Commun. Surv. Tutor.* **2018**, *22*, 1646–1685. [CrossRef]
28. Zhou, Z.; Chen, X.; Li, E.; Zeng, L.; Luo, K.; Zhang, J. Edge intelligence: Paving the last mile of artificial intelli-gence with edge computing. *Proc. IEEE* **2019**, *107*, 1738–1762. [CrossRef]

29. Bennis, F.; Bhattacharjya, R.K. *Nature-Inspired Methods for Metaheuristics Optimization: Algorithms and Applications in Science and Engineering*; Springer: Basingstoke, UK, 2020.
30. Richard, B.; Levin, P.W.; Holly, L. Betting Blockchain Will Change Everything–SEC and CFTC Regulation of Blockchain Technology. In *Handbook of Blockchain, Digital Finance, and Inclusion*; Elsevier: Amsterdam, The Netherlands, 2017; Volume 2, pp. 187–212.
31. Vogels, C.B.F.; Brito Anderson, F.; Wyllie, A.L.; Fauver, J.R.; Ott, I.M.; Kalinich, C.C.; Petrone, M.E.; Casano-vas-Massana, A.; Muenker, M.C.; Moore, A.J.; et al. Analytical sensitivity and efficiency comparisons of SARS-COV-2 qRT-PCR primer-probe sets. *Nat. Microbiol.* **2020**, *5*, 1299–1305. [CrossRef]
32. Zhang, Y.; Odiwuor, N.; Xiong, J.; Sun, L.; Nyaruaba, R.O.; Wei, H.; Tanner, N.A. Rapid Molecular Detection of SARS-CoV-2 (COVID-19) Virus RNA Using Colorimetric LAMP. *medRxiv* **2020**. [CrossRef]
33. Notomi, T.; Okayama, H.; Masubuchi, H.; Yonekawa, T.; Watanabe, K.; Amino, N.; Hase, T. Loop-mediated isothermal amplification of DNA. *Nucleic Acids Res.* **2000**, *28*, E63. [CrossRef] [PubMed]
34. Kashir, J.; Yaqinuddin, A. Loop mediated isothermal amplification (LAMP) assays as a rapid diagnostic for COVID-19. *Med. Hypotheses* **2020**, *141*, 109786. [CrossRef] [PubMed]
35. Rohaim, M.A.; Clayton, E.; Sahin, I.; Vilela, J.; Khalifa, M.E.; Al-Natour, M.Q.; Bayoumi, M.; Poirier, A.C.; Bra-navan, M.; Tharmakulasingam, M.; et al. Artificial intelligence-assisted loop mediated isothermal amplification (AI-LAMP) for rapid detection of SARS-CoV-2. *Viruses* **2020**, *12*, 972. [CrossRef]
36. Sodhro, A.H.; Pirbhulal, S.; de Albuquerque, V.H.C. Artificial intelligence-driven mecha-nism for edge computing-based industrial applications. *IEEE Trans. Ind. Inform.* **2019**, *15*, 4235–4243. [CrossRef]
37. Padmapriya, T.; Manikanthan, S.V. Implementation of Dual-Band Planar Inverted F- Antenna (PIFA) Using Machine Learning (ML) for 5G Mobile Applications. In Proceedings of the First International Conference on Computing, Communication and Control System, I3CAC 2021, Chennai, India, 7–8 June 2021.
38. Dewi, K.C.; Harjoko, A. Kid's Song Classification based on mood parameters using K-Nearest neighbor classification method and self organizing Map. In Proceedings of the 2010 International Conference on Distributed Frameworks for Multimedia Applications, Yogyakarta, Indonesia, 2–3 August 2010; pp. 1–5.
39. Han, B.J.; Rho, S.M.; Dannenberg, R.B.; Hwang, E.J. SMERS: Music Emotion Recognition Using Support Vector Regression. In Proceedings of the 10th International Society for Music Information Retrieval Conference, Kobe, Japan, 26–30 October 2009; pp. 651–656.
40. Lin, C.; Liu, M.; Hsiung, W.; Jhang, J. Music emotion recognition based on two-level support vector classification. In Proceedings of the 2016 International Conference on Machine Learning and Cybernetics, Jeju, Korea, 10–13 July 2016; pp. 375–389.
41. Kim, B.-G.; Park, D.-J. Novel target segmentation and tracking based on fuzzy membership dis-tribution for vision-based target tracking system. *Image Vis. Comput.* **2016**, *24*, 1319–1331. [CrossRef]
42. Huh, J.-H.; Seo, Y.-S. Understanding edge computing: Engineering evolution with artificial intelligence. *IEEE Access* **2019**, *7*, 164229–164245. [CrossRef]
43. Chen, M.; Mao, S.; Liu, Y. *Big Data: A Survey, Mobile Networks and Applications*; Springer: Berlin/Heidelberg, Germany, 2014; pp. 171–209.
44. Lee, S.; Woo, H.; Shin, Y. Study on Personal Information Leak Detection Based on Machine Learning. *Adv. Sci. Lett.* **2017**, *23*, 12818–12821. [CrossRef]
45. Huh, J.-H.; Otgonchimeg, S.; Seo, K. Advanced metering infrastructure design and test bed experiment using intelligent agents: Focusing on the PLC network base technology for smart grid system. *J. Supercomput.* **2016**, *72*, 1862–1877. [CrossRef]
46. Huang, X.; Yang, Q.; Qiao, H. Lightweight Two-Stream Convolutional Neural Network for SAR Target Recognition. *IEEE Geosci. Remote Sens. Lett.* **2021**, *18*, 667–671. [CrossRef]
47. Huh, J.-H. PLC-based design of monitoring system for ICT-integrated vertical fish farm. *Hum. Cent. Comput. Inf. Sci.* **2017**, *7*, 1–19. [CrossRef]
48. Huh, J.-H. *Smart Grid Test. Bed Using OPNET and Power Line Communication*; IGI Global: Seoul, Korea, 2018; pp. 1–425.
49. He, D.; Liu, C.; Quek, T.Q.S.; Wang, H. Transmit Antenna Selection in MIMO Wiretap Channels: A Machine Learning Approach. *IEEE Wirel. Commun. Lett.* **2018**, *7*, 634–637. [CrossRef]
50. Huh, J.-H.; Lee, D.-G.; Seo, K. Design and Implementation of the Basic Technology for Realtime Smart Metering System Using Power Line Communication for Smart Grid. In *Lecture Notes in Electrical Engineering*; Springer: Singapore, 2015; pp. 663–669.
51. Fraga-Lamas, P.; Fernandez-Carames, T.M. A Review on Blockchain Technologies for an Advanced and Cyber-Resilient Automotive Industry. *IEEE Access* **2019**, *7*, 17578–17598. [CrossRef]
52. Suciu, G.; Nadrag, C.; Istrate, C.; Vulpe, A.; Ditu, M.-C.; Subea, O. Comparative Analysis of Distributed Ledger Technologies. In Proceedings of the 2018 Global Wireless Summit (GWS), Chiang Rai, Thailand, 25–28 November 2018; pp. 370–373.
53. Praveen, G.; Chamola, V.; Hassija, V.; Kumar, N. Blockchain for 5G: A Prelude to Future Telecommunication. *IEEE Netw.* **2020**, *34*, 106–113. [CrossRef]
54. Tran, T.-T.-Q.; Tran, Q.-T.; Le, H.-S. An Empirical Study on Continuance Using Intention of OTT Apps with Young Generation. In *Lecture Notes in Electrical Engineering*; Springer: Singapore, 2020; pp. 219–229.
55. Seo, E.; Song, H.M.; Kim, H.K. GIDS: GAN based Intrusion Detection System for In-Vehicle Network. In Proceedings of the 16th Annual Conference on Privacy, Security and Trust, Belfast, UK, 28–30 August 2018; pp. 1–6. [CrossRef]

56. Hanselmann, M.; Strauss, T.; Dormann, K.; Ulmer, H. CANet: An Unsupervised Intrusion Detection System for High Dimensional CAN Bus Data. *IEEE Access* **2020**, *8*, 58194–58205. [CrossRef]
57. Lokman, S.F.; Othman, A.T.; Musa, S.; Abu Bakar, M.H. Deep Contractive Autoencoder-Based Anomaly Detection for In-Vehicle Controller Area Network (CAN). In *Progress in Engineering Technology*; Abu Bakar, M., Mohamad Sidik, M., Öchsner, A., Eds.; Advanced Structured Materials; Springer: Cham, Switzerland, 2019; pp. 195–205. [CrossRef]
58. Huh, J.H.; Kwak, S.Y.; Lee, S.Y.; Seo, K. A design of small-size smart trash separation box using ICT technology. In Proceedings of the Asia-pacific Proceedings of Applied Science and Engineering for Better Human Life; In Proceedings of the 10th 2016 International Interdisciplinary Workshop Series at Jeju National University, Jeju, Korea, 16–19 August 2016; pp. 141–144.
59. Kim, S.-K.; Huh, J.-H. Autochain platform: Expert automatic algorithm Blockchain technology for house rental dApp image application model. *EURASIP J. Image Video Process.* **2020**, *2020*, 1–23. [CrossRef]