Building a Security System Using a Deep Learning and Blockchain-Based Approach for Abnormal Event Detection and Data Security

Maysam Majid Sabri, Haider Kadhim Hoommod and Khalid Ali Hussein
Department of Computer Science, College of Education, Mustansiriyah University, 10052 Baghdad, Iraq
maysam_majid@uomustansiriyah.edu.iq, *drhjnew@gmail.com, dr.khalid.ali68@gmail.com*

Keywords: Deep Learning, Blockchain Technology, ResNet50 Model, Densenet201 Model, Efficientnetv2 Model.

Abstract: Building surveillance systems in smart cities face significant challenges in ensuring security, detecting anomalies, and processing large volumes of real-time data. This paper proposes an enhanced surveillance system that integrates lightweight deep learning models with blockchain technology to improve authentication and authorization. The system processes real-time data from surveillance cameras, using a dataset of 46,297 samples divided into training, validation, and testing subsets. It operates in two main stages. In the authentication stage, images from surveillance footage are processed by three lightweight CNN models to predict class values, which are then hashed using the SHA-256 algorithm. These hashes are structured into a Merkle tree and stored in a blockchain network. In the authorization stage, the system verifies whether a user has blockchain-registered access to surveillance data for a specific time period. If authorized, the system retrieves data from all cameras and uses a SoftMax function to classify activity as normal or abnormal. Among the CNN models tested, DenseNet201 showed superior performance with high accuracy and low loss in training and validation. In testing, DenseNet201 achieved the best results with a precision of 0.951, recall of 0.946, F1 score of 0.948, and overall accuracy of 0.936. These findings highlight the model's effectiveness in identifying abnormal behavior and the potential of integrating deep learning with blockchain to secure and manage smart building surveillance systems.

1 INTRODUCTION

The key areas of study for contemporary intelligent video surveillance systems include anomalous detection and virtual reality. The use of anomalous detection is significant.It has garnered a lot of interest in the previous 20 years. abnormalities seen in videos is essential to provide safety in both indoor areas and the exterior of buildings such as campuses, waiting areas, and retail shopping centers. Due to the large number of cameras installed, the duty of managing several monitors by security personnel grows considerably more challenging when the humans become less focused and tired [1], [2] . Blockchain technology based on deep learning models was used, Blockchain's quick development has had a huge impact on many aspects of society; thus, it is important to carefully evaluate the complexity that improves our daily lives and plays a crucial part in community development [3] Blockchain is a technology that enables the recording of information in a manner that is challenging or almost impossible to modify, breach, or deceive [4]. Blockchain also offers the capability to control participant interactions without the need for a middleman or reliable third party through smart contracts [5]. The blockchain is made up of several information-containing blocks that are joined together by a cryptographic process known as a hashing function. The result is an unbreakable chain. Blockchain technology, which is a rapidly expanding security cryptography system with decentralized solutions outperforming many security techniques, is characterized by its nodes competing and working together to maintain an accurate ledger; its data is immutable, meaning that safety, security, strength, and resistance outweigh many security techniques [6]. On the other hand, Deep learning (DL) is one of the most often utilized machine learning algorithms in these kinds of applications. CNN is the most widely utilized network because, in contrast to its predecessors, it automatically recognizes important elements without human oversight The internal model parameters (weights and deviations), which are used to generate the output values, are crucial for training neural network models and improving results. These network parameters have an impact on both model training and model output [7]. Pre-trained CNN

models, such as AlexNet, GoogleNet, ResNet, etc., are used in deep learning on massive datasets like ImageNet to recognize images. Then, without needing to retrain, these models can be used to recognize a different task. Additionally, other than a few taught elements, the weights stay the same. These models are highly helpful when there are insufficient data samples [8]. Transfer learning can also be used to refine a pre-trained model. To adapt a pre-trained model to a new task, this method entails training the model on a new dataset. Typically, this is accomplished by retraining the pre-trained model with the fresh dataset after unfreezing a few of its layers [9].

Despite the advances in video surveillance, ensuring both real-time abnormal event detection and data integrity remains a challenge. This research aims to address this gap using a hybrid CNN–Blockchain approach.

We start first in the authentication process is the real-time data collection from various security cameras. Following the conversion of the video data into frame images, three contemporary lightweight CNN models process the data to produce prediction values for each class. The 256-SHA algorithm is then applied to produce a 256-bit hash using these predicted values. Blockchain networks will then be created by constructing a Merkle tree. The authorization phase's main goal is to guarantee safe access to surveillance footage for users who need to keep an eye on activities within the facility for a set amount of time.

Contributions:

1) Proposing a hybrid CNN-Blockchain framework for secure abnormal event detection.
2) Leveraging Merkle Tree structure for lightweight hash verification.
3) Evaluating and fusing three CNN models for robust decision-making.

2 RELATED WORKS

In this section, we mainly describe the related works, blockchain, video surveillance systems, and blockchain technology with deep learning.

Xiang Yu et al. (2019) The researchers discussed how to diagnose breast problems using the DenseNet201 model, which has a high diagnosis accuracy of 92.73%. The MINI-MIAS dataset, which includes 208 normal pictures and 114 images of aberrant tissue, was employed in the study. The paper also looked at the diagnostic results of networks trained using freezing and fine-tuning techniques. It was found that the sensitivity went up as the density of the retrained clusters went up. The report's consideration of potential future system enhancements, such as creating a CAD system for thorough breast cancer detection and diagnosis and grouping anomalies into specific categories, comes to a close [10] Sajja Tulasi Krishna. et al. (2019) the researchers concentrate on deep learning and transfer learning techniques for image classification. They use convolutional neural networks (CNNs), deep neural networks, and pre-trained structures to solve the issue of increasing classification accuracy. The study employed several picture datasets, including MNIST, CIFAR10, CIFAR100, Caltech101, and others. Image-based datasets are utilized. It took deep learning models like VGG16, ResNet50, VGG19, GoogleNet, LeNet, and AlexNet 99.72% of the time to get the MNIST dataset right and 89.23% of the time to get the CIFAR10 dataset right [11] Muhammad Asad Arshed et al. (2022) The researchers employed a convolutional neural network (CNN) architecture, especially ResNet 50, to create a basic deep learning model for recognising and categorising plant leaves. By better classifying different plant leaves, the study hopes to advance botanical research and our knowledge of plant diversity. Images gathered from actual settings make up the dataset. They employed the ResNet-50 model, which reduces the vanishing gradient issue and increases training efficiency by leveraging deep residual learning. The model's great performance in leaf classification was demonstrated by its high training accuracy of 98.3% and test accuracy of 92.5%, which were its most significant outcomes [12] Moolikagedara K et al. (2023) The researchers concentrated on using video blockchain technology to improve car cameras' security in smart cities. To create a secure system for connecting car cameras in smart cities, the Schnorr signature and the SHA-256 hash function were used to guarantee the safe transfer of surveillance data. The technique used a descriptive research style, main and supplementary video data, and a video blockchain architecture using Merkle trees to support the study's hypotheses. Videos from car cameras were among the data used in the experiment, which primarily used picture extraction and video capture as data collection methods. Because it required gathering original recordings from car cameras, the study is part of the primary dataset. The video blockchain architecture significantly enhances data security and integrity, while the proposed method effectively lowers potential risks [3]

3 PROPOSED SYSTEM

A proposed building monitoring system aims to enhance the authentication and authorization of smart buildings based on lightweight deep learning models and parallel blockchain networks. This system, called BMS-LDBC, includes two main stages: authentication and authorization. The general block diagram of the proposed system is shown in Figure 1. The authentication stage includes multiple components, starting with collecting real-time data from various surveillance cameras. The video data is then converted into frame images and processed by three modern Lightweight CNN models to generate prediction values for each class. These prediction values are subsequently used to create a 256-bit hash via a 256-SHA algorithm. Then, the Merkel tree will be built, and blockchain networks will be made. The core concept of the authorization stage is to ensure secure access to surveillance data when a user needs to monitor activity in the building during a specific timestamp. When the user requests data from all surveillance cameras for that timestamp, the system first verifies if the user is a member of the blockchain and authorized to access the requested data. If the user is approved, the data recorded by all cameras during the specified timestamp is retrieved. The system then checks for any abnormal behavior during that period. This stage involves the following steps: the user submits a request with a specified timestamp, the system verifies the user's authorization based on the timestamp, and if authorized, the data is retrieved from all cameras and passed through a SoftMax function to classify it into a binary class: normal or abnormal behavior.

3.1 Acquiring Surveillance Cameras Dataset Step

The researcher collected approximately 92 video clips from surveillance cameras installed in buildings managed by a specific security department. The camera type, referred to as" camscan", recorded video in AVI format, with each clip ranging from 15 to 20 seconds. The dataset videos were categorized into 70 clips depicting abnormal behaviors and 22 clips showing normal behaviors.

3.2 Convert Video into Frames Step

This stage involves transforming each video into a set of frames. First, the video file opens using the OpenCV library. Then, the video is read frame by frame within a loop using "video_capture. read ()". Each frame is subsequently saved as an image file, as illustrated in Figure 2. Unusual events include unauthorized entry into the building during hours after official working hours or holidays. Events include any person, group of people, or vehicle entering or approaching the building in a suspicious manner.

3.3 Lightweight CNN Detection Models Step

Video frames are processed using three lightweight CNN models (ResNet50, DenseNet201, and EfficientNetV2) fine-tuned for surveillance data (Fig. 3). The models exclude the SoftMax function to produce floating-point prediction values between 0 and 1. Pre-trained weights are used, retaining initial layers for basic features, while the final layers are replaced. A dataset of 46,297 samples is divided into 70% training, 10% validation, and 20% testing to optimize performance.

3.3.1 Authentication Stage

The authentication stage is designed to create secure data and a 256-bit hash, which are then utilized within the blockchain. As illustrated in Figure 4, which represents the structure of Lightweight CNN models with Fine-Tuning Layer.

3.3.1.1 ResNet50 Model with Fine-Tuning

Resnet50 architecture is a CNN model with 50 layers that uses residual learning to improve training performance. Fine-tuning a ResNet50 model involves adapting the pre-trained model to a specific surveillance dataset – the details of the ResNet50 Model with fine-tuning are shown in Algorithm 1. Firstly, load the original ResNet 50, then apply fine-tuning layers, including global average pooling, 2 layers of batch normalization, 2 layers of dropout, and 2 layers of dense.

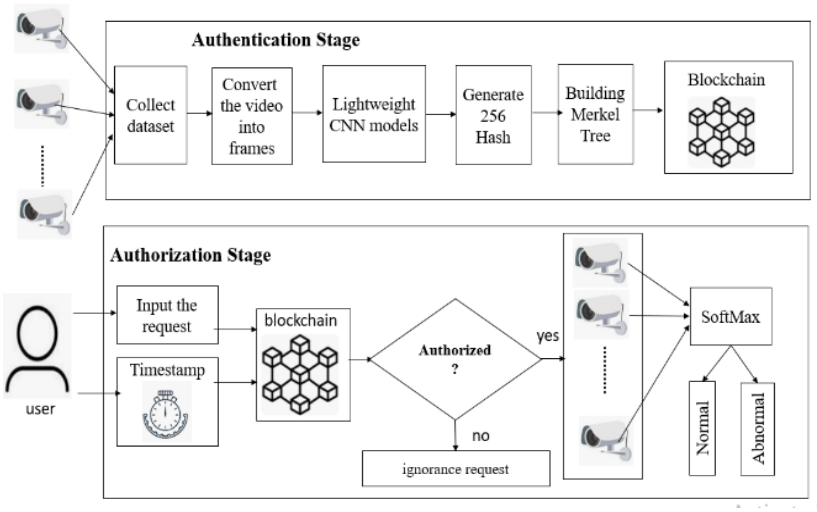

Figure 1: General block diagram of the proposed BMS-LDBC system.

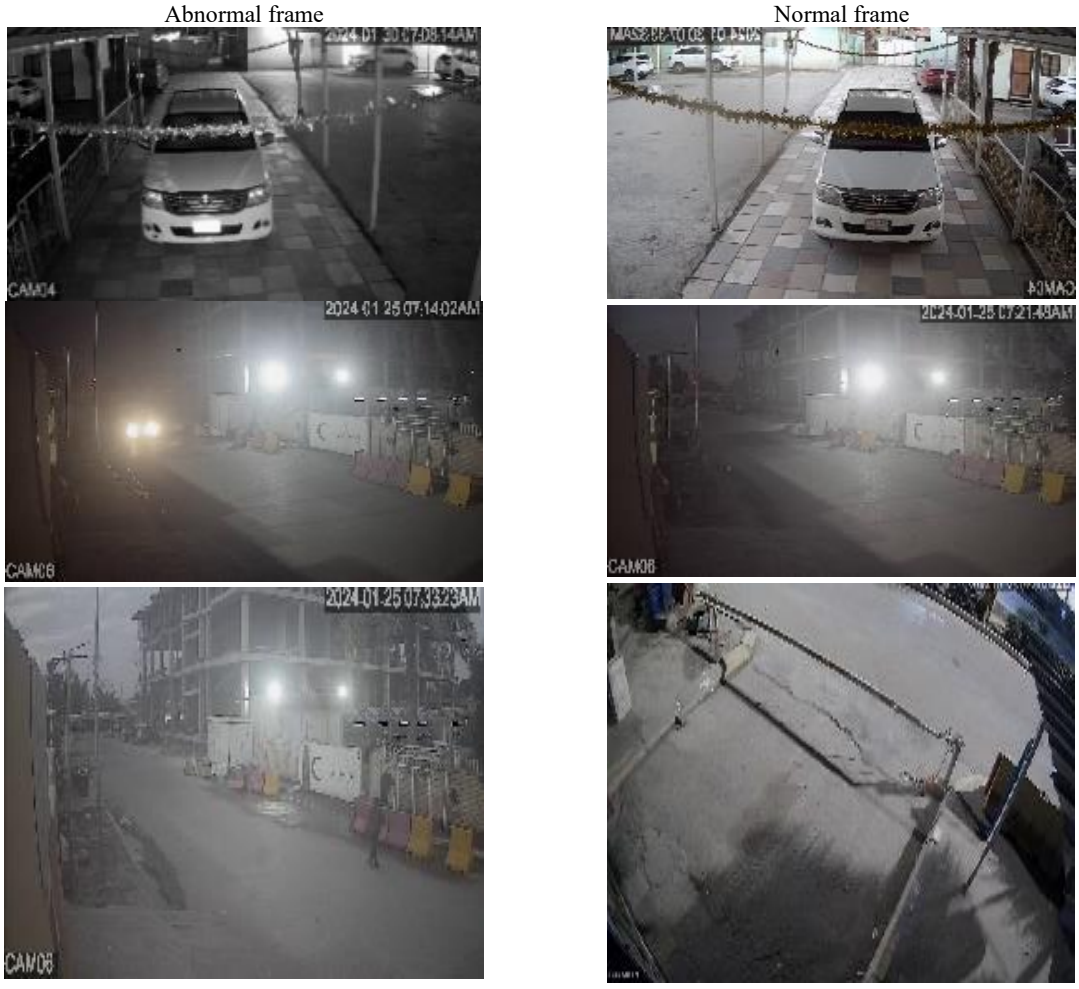

Figure 2: Examples of video frames.

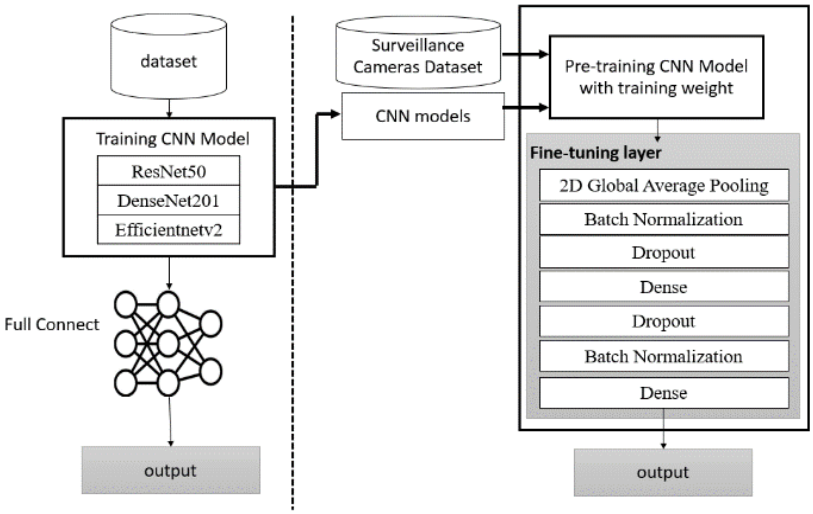

Figure 3: Lightweight CNN models with fine-tuning layer.

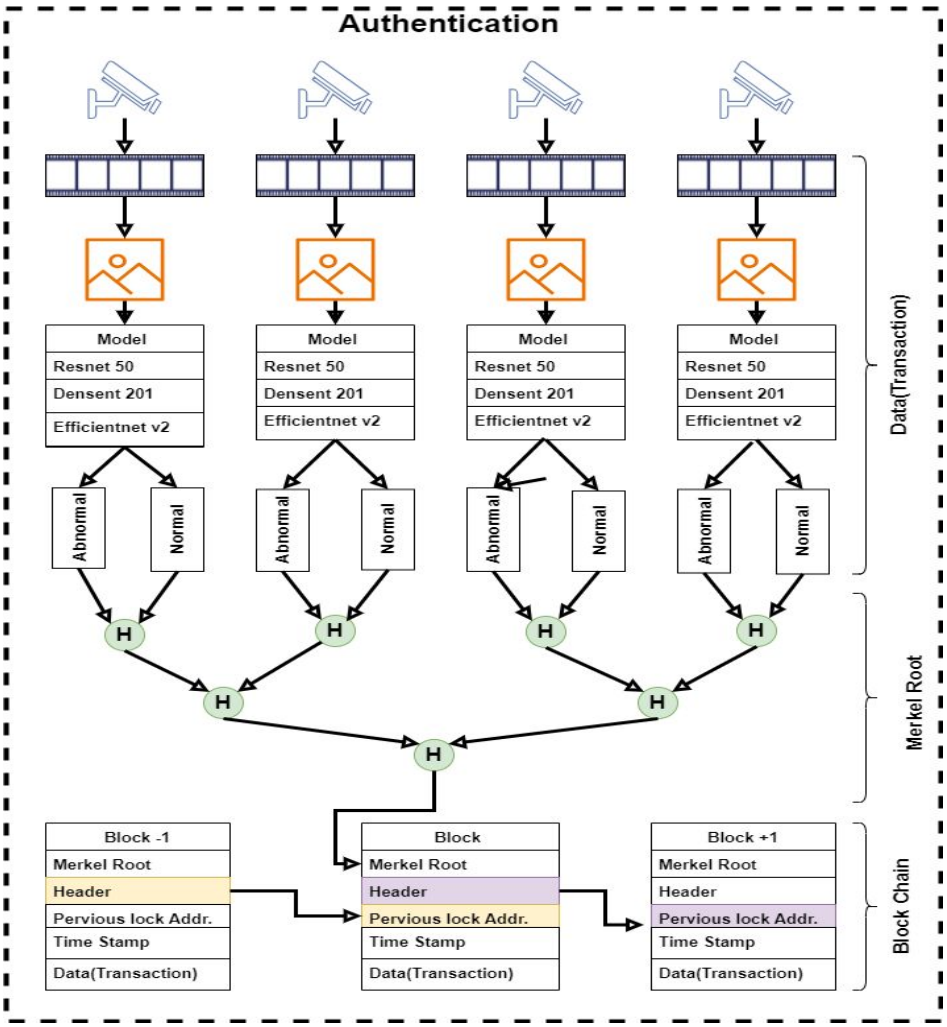

Figure 4: Framework of the authentication stage in the proposed BMS-LDBC system.

Algorithm 1. ResNet50 Model with fine-tuning:

```
Inputs:
    ▪ Load ResNet50 model
    ▪ Load Pre-training weight
Output:
predication value
Begin
Step 1: Load the pre-trained ResNet50 model with the output shape from the last convolution block (none,5,5,2048);
Step 2: Apply Fine Tuning Layers:
   2-1 apply 2d global average pooling on feature maps, resulting in out shape (none,2048);
   2-2 apply batch normalization with output shape (none,2048);
   2-3 Add a dropout layer to reduce overfitting with output shape (none,2048);
   2-4 add a dense layer with 4096 units with output shape(none,4096);
   2-5 Apply dropout layer with output shape (none,4096);
   2-6 Apply batch normalization layer with output shape (none,4096);
   2-7 Add a final dense layer with output prediction value for each class;
End
```

3.3.1.2 DenseNet201 Model with FineTuning

The denseNet201 architecture is a type of CNN with 201 layers that follows the concept of Dense Connections, where each layer is connected to every other layer in a feed-forward manner. This architecture ensures maximum information flow between layers, as each layer receives the feature maps of all preceding layers as input. Algorithm 2 illustrates the steps of training the fine-tuning model. Algorithm 2 provides a detailed overview of the densenet201 architecture with fine-tuning. The process begins by loading the pre-trained denseNet201 model and applying fine-tuning layers to adopt the model for produced prediction values.

Algorithm 2. denseNet201 Model with fine-tuning predication value:

```
Inputs:
    ▪ Load Densenet201 model.
    ▪ Load Pre-training weight.
Output:
Begin:
Step 1: Load the pre-trained DenseNet201 Model with the output shape from the last convolution block (none,7,7,512)
Step 2: Apply Fine Tuning Layers
   2-1 apply 2d global average pooling on feature maps, resulting in out shape (none, 512)
   2-2 apply batch normalization with output shape (none, 512)
   2-3 Add a dropout layer to reduce overfitting with output shape (none, 512)
   2-4 add a dense layer with 4096 units with output shape(none,4096)
   2-5 Apply dropout layer with output shape (none,4096)
   2-6 Apply batch normalization layer with output shape (none,4096)
   2-7 Add a final dense layer with output prediction value for each class
End
```

3.3.1.3 Efficientnet v2 Model

EfficientNetV2 is an improved EfficientNet architecture focusing on scaling neural networks while optimizing performance and efficiency. In this work, the type of model used is a small EfficientNetV2.

After applying EfficientNetv2, the SoftMax function is freeze, and fine-tuning layers are introduced, as detailed algorithm 3.4.

Algorithm 3 EfficientNetV2 Model with fine-tuning:

```
Inputs:
    ▪ Load EfficientNetV2 model
    ▪ Load Pre-training weight
Output:
predication value
Begin
Step1: Load the pre-trained EfficientNetV2 model with the output shape from the last convolution block (none,7,7,1280)
Step 2: Apply Fine Tuning Layers
   2-1 apply convolution without shape (none, 7,7,8)
   2-2 apply dropout without shape (none,7,7,8)
   2-3 apply batch normalization with output shape ((none,7,7,8)
   2-4 apply global average pooling without shape(none,8)
   2-5 add a dense layer with 1152 units with output shape(none,128)
   2-6 Add a final dense layer with output prediction value for each class
End
```

3.3.2 Generating 256-Bits Hash and Constructing Transaction Camera

This step focuses on generating a list of the transaction hashes and creating a transaction camera. A transaction is a secure record of an exchange of data stored within a blockchain. In this work, the transaction data consists of the normal and abnormal prediction values obtained from CNN models, reflecting the behavior detection by the monitoring system.

The list of the transaction hashes contains a set of hashed (normal and abnormal prediction values), which will be used to construct the Merkel tree. To ensure data integrity and security, each predication is hashed using a 256-bit SHA algorithm, producing a unique hash for each prediction.

The framework of generating a 256-bit hash and constructing a transaction camera is presented in Figure 5. Algorithm 4 shows details of developing a list of the transaction hashes and creating a transaction camera. Predicted float values were serialized into byte streams before applying SHA-256 hashing, ensuring compatibility with the Merkle Tree structure.

Algorithm 4. Generating 256 bits using 256-SHA:

```
   Inputs: predication value from Camera
1, Camera 2, Camera n)
   Output: return list transection hash
   Begin
   Step1: create transaction
          list transaction Camera=[]
        for each predication value in
Camera do
        value normal from predication
normal using algorithms (3.1,3.2, and
3.3)
        value Abnormal from predication
Abnormal using algorithms (3.1,3.2, and
3.3)
     result =str(value normal)+str(value
Abnormal)
     list transection Camera.add(result)
     End for
   Step2: generation hash for all
transection camera
            list transection hashes=[]
            for each value in list
transection do
                  result =
hash256(value) using Gs hash265
                list transection
hashes.add(result)
            End for
   return list transection hash
   End
```

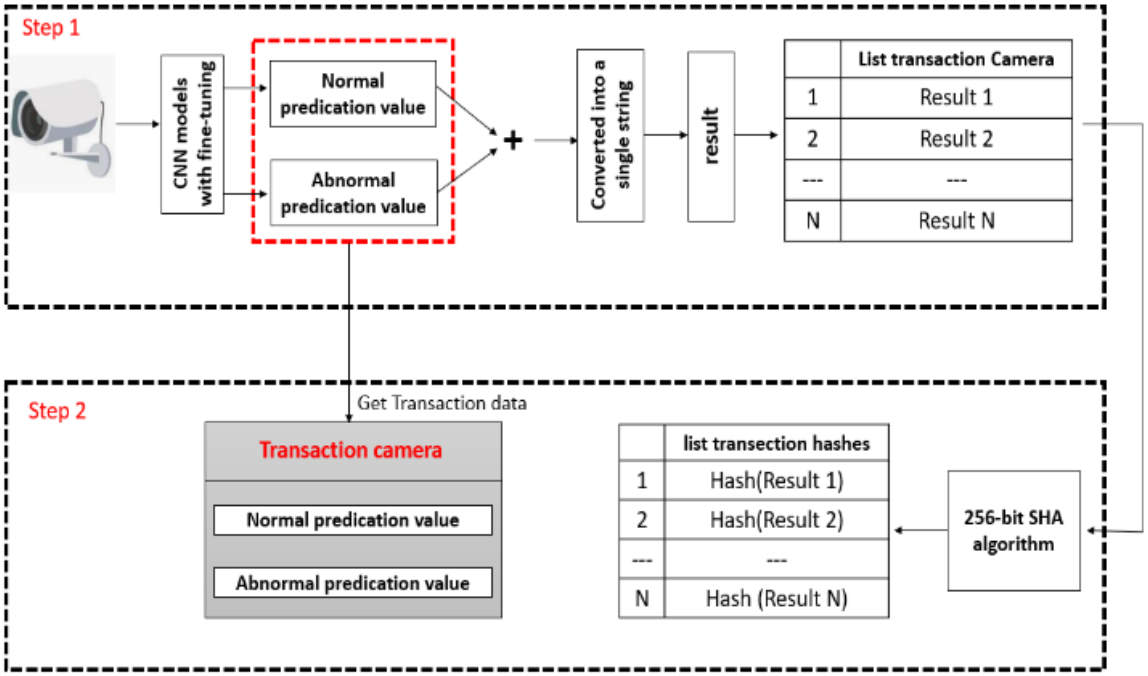

Figure 5: The frame work of the generating 256-bits hash and constructing transaction camera.

The figure above illustrates that the first step involves generating a list of transaction cameras. We require N cameras to capture image frames, which are then processed using CNN models. These models generate prediction values (normal and abnormal behaviors). These values are concatenated into a single string, stored in a variable called "Result", and added to a list referred to as "list transaction cameras".

In the second step, the values from the "list transaction camera" are processed by the 256 SHA algorithm to generate their corresponding hash. The hashed results are stored in a specifi c list called "list transaction camera hash." Also, it constructs the transaction camera by getting prediction values for both standard and abnormal and recording them in the transaction.

3.3.3 Building Merkle Tree

The proposed BMS-BLCNN mechanism's builder Merkle tree creates a dependable environment by summarizing all of the transactions inside a block and saving it as the current hash in the block header.

Merkle Root is a full binary tree in which every leaf considers the hashed value of the associated authentication user transaction T. The two children's series hashes are represented by the branches. Until the top of the tree is reached, also referred to as the "Root Hash" as explained in the algorithm (5), the process of rehashing the concatenation among the child nodes to generate the parent node is carried out continually.

Algorithm 5. Building Merkle Tree:

```
Inputs:  Input:  n  //  number  of
transactions;
T // transaction node;
Hash // transaction data.
Output:256-bits hash Merkle tree;
Begin:
Step1: // Initialize leaf nodes with
data hashes
For each transaction in data blocks do
    leafNodes = Compute hash of data
(block)
End for
Step 3: Check leftChild and rightChild
in Leafnodes
             If length (leafNode) % 2 !=
0 then
                Get last leafNodes call
rightChild
                Add    rightChild    to
leafNodes
Step 2: Recursively build internal nodes
       while length(leafNodes) > 1:
            newLevel = []
            for   each   index      in
range(leng(  leafNodes) skip 2
                leftChild            =
leafNodes[i]
                rightChild           =
leafNodes[i + 1]
                createInternalNode
=hash(leftChild + rightChild)

newLevel.append(createInternalNode)
           endfor
          leafNodes = newLeve
        end while
step3:// The first element is the Merkle
root
    return leafNodes[0]
End
```

If the number of transactions T is even, the value of the parent node's hash is equal to the hash of two T leaf nodes, as shown in Figure 6, since algorithm (2) extracts a hash value for each authenticated user transaction T and stores it in array hash[k]. In the case where transaction T is odd as shown in Figure 7 , the proposed system replicated the last leaf node and calculated the parent hash in order to maintain the Merkle tree's balance.

3.3.4 Building Blockchain Network

The blockchain is composed of N blocks, each block contains a header and body, as illustrated in Figure 8.

A) The Block header consists from:
   1) Merkle Tree Hash (MTH). Single hash value that represents all transactions in this block.
   2) Time-Stamp. Indicates the date and time when the transaction was enrolled in the block.
   3) Previous -hash. Indicates the previous block's SHA-256 using (3.4) algorithm. Specifically, the first block on the Blockchain has a value of "0" since it was not included in the previous hash.
   4) Current-hash: represents the current block header's hash value.

B) The Block body consists from: The block body represents data transactions and includes prediction values for normal and abnormal classes.

Two values were computed at the block creation and they are:

1) Computed current hash value by copying the contents of the block header (time-stamp value, previous hash value, MTH value), and then applying the SHA-256 hash algorithm (4) to the contents of the block header to get a single hash value.
2) It computed a time stamp that records the time the current block is created.

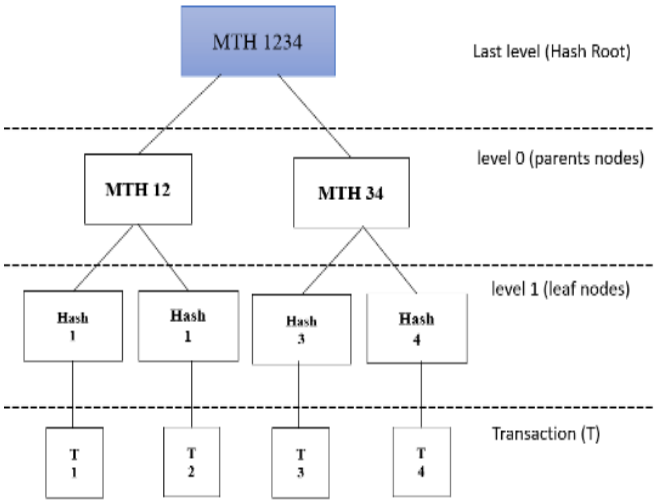

Figure 6: Building Merkle tree with the case of the number of transactions is even.

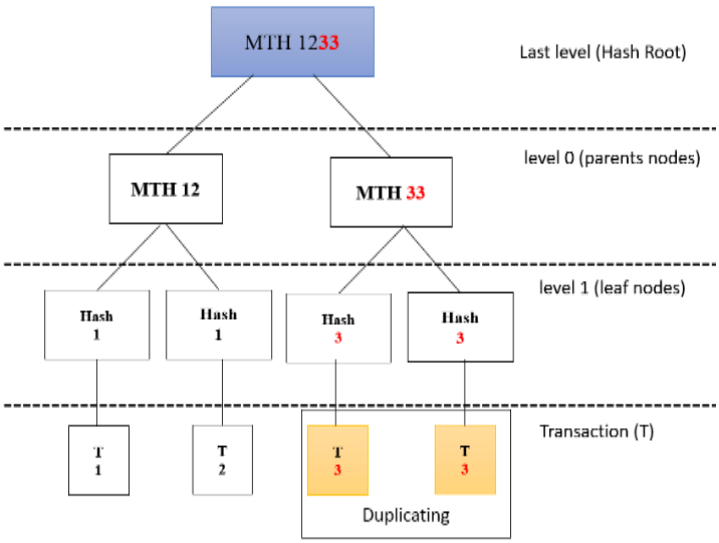

Figure 7: Building Merkle tree with the case of number of the transaction is odd.

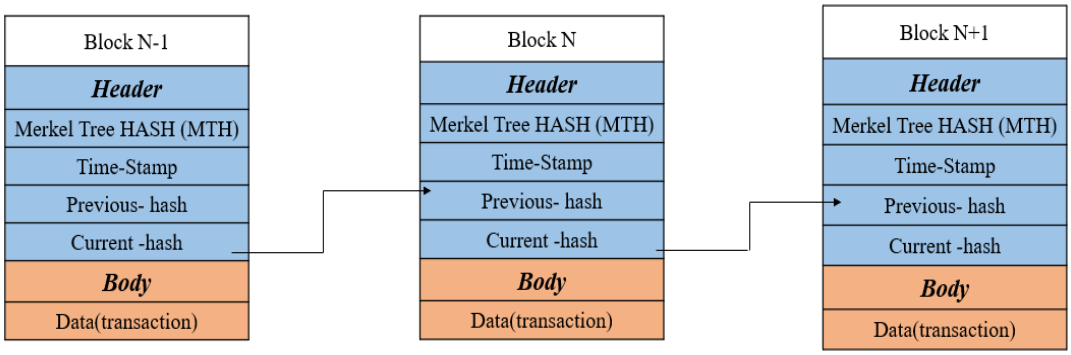

Figure 8: Structure of blockchain in the proposed BMS-BLCNN.

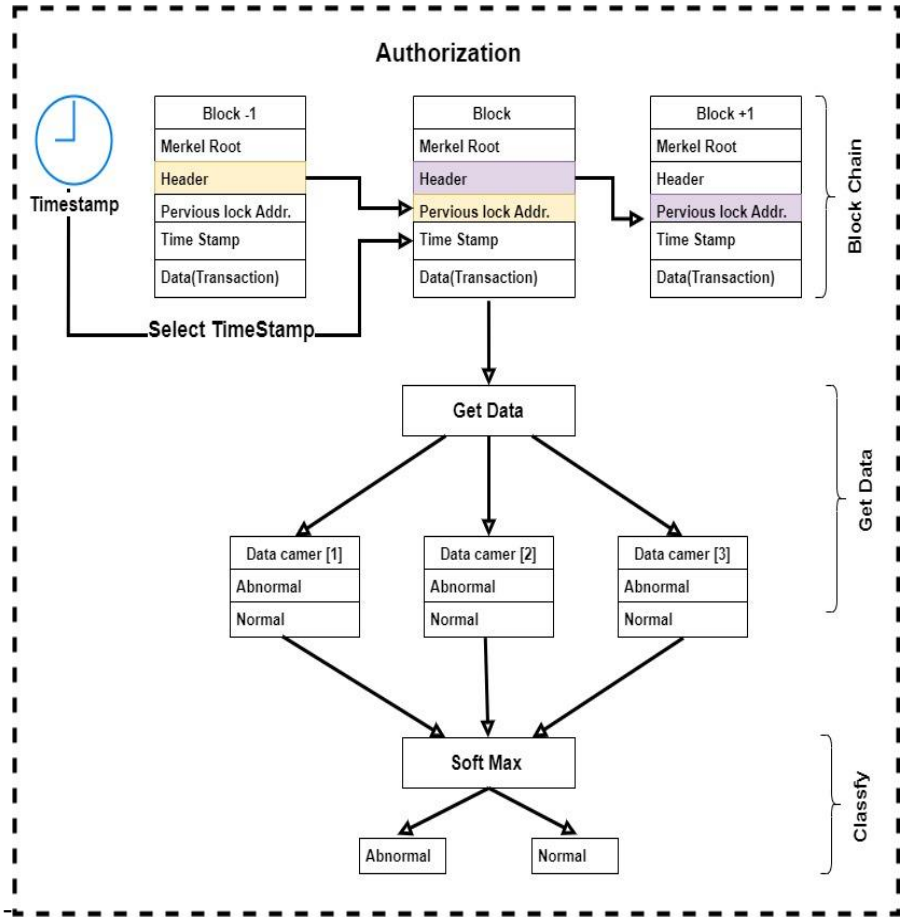

Figure 9: Framework of the authorization stage in the proposed BMS-BLCNN system.

The blockchain-based authorization mechanism described in Algorithm 3.6 ensures that only authenticated users can access camera data to identify abnormal behavior and determine the responsible camera (Fig. 9). The process begins with the user sending an authentication request containing a timestamp and camera-generated hash to the blockchain network. All nodes verify the request by checking the timestamp against block headers. If a match is found, the node retrieves the previous block's hash and constructs a Merkle tree using the camera hash and other transaction hashes, producing a Merkle root hash. A temporary hash is then generated by combining the Merkle root with the previous block's hash. If this temporary hash matches the expected value, the user is authorized to access the relevant block data; otherwise, access is denied. Upon authorization, the node extracts prediction results (normal or abnormal) and separates them by camera source. Finally, the SoftMax function processes these results to identify the camera with the highest probability of detecting abnormal behavior and determines whether the detected behavior is normal or abnormal.

Algorithm 6. Authorization process:

```
Input: Time-stamp, hash from camera
Output: select camera
Begin
Step 1: The user sends a request to
the blockchain network
Step 2: All nodes in the blockchain
network receive user requests and start
to verify them.
Step3: each node searches to check
Time-stamp in header block
            If Time-stamp in matches
then go to step4
             Else continue
Step4: get previse hash from this
block
Step5: create hash block based on
Merkle Tree from hash from camera and
rest hash in this block call hash_
Merkle Tree
```

```
   Step6  get temporary hash based on
hash(hash_ Merkle Tree, previse hash)
   Step7: if temporary is equal next
hash block
   Then return " user is authorization
" then go to step 8
   Else return " user is not
authorized" then go to step 9
   End if
   Step 8:  get data from this block
   Step9: split data base on data
cameras in this block
   Step10 get cameras data all [normal]
pass cameras data [normal] through
SoftMax function using (2) and get
index camera
   Step11 get cameras data [abnormal]
pass cameras data [abnormal] through
SoftMax function using (2.) and get
index camera
   Step12: appley SoftMax function from
step10 and Step11 to determine: normal
or abnormal and get index camera return
results of SoftMax: normal or abnormal
and select camera.
   End
```

4 RESULTS

In this section, the most important results obtained from training the convolutional neural network on pre-trained models and on real video data are reviewed, as well as the most important results and interfaces of blockchain technology for designing a security system for monitoring buildings.

4.1 Results of Lightweight CNN Detection Models

The proposed BMS-LDBC system used three Lightweight CNN models based on a combination of fine-tuning techniques with each model to enhance their performance. The CNN models used in this work are (DenseNet201, EfficientNetV2, and ResNet50).

The 46,297-sample input dataset will be split into three subsets before CNN models are applied: 70% for "training "(39,407 samples), 10% for "validation" (1,258 samples), and 20% for "testing" (5,632 samples.

Table 1 displays the numerical values of ten evaluation metrics for three (CNN) models throughout the testing process. The metrics include "precision", "recall"," F1-Score", and "accuracy".

Table 1: Compare three lightweight CNN models based on accuracy and loss during training and validation process.

CNN model	Process	Accuracy	Loss
DenseNet201	Train	0.9552	0.1058
	Valid	0.94617	0.11306
EfficientNetV3	Train	0.90629	0.24785
	Valid	0.92707	0.18702
ResNet50	Train	0.90649	0.23737
	Valid	0.92681	0.18991

Table 2: Results of accuracy metrics for three lightweight CNN models in testing process.

Metrics	DenseNet201	EfficientNetV3	ResNet50
Precision	0.951	0.928	0. 935
Recall	0.946	0.926	0.916
F1 Score	0.948	0.941	0.934
Accuracy	0.936	0.925	0.914

The Figure 10 shows a comparison based on the test values in the Table 2 to determine the best model in detecting abnormal behavior and its ability to distinguish between it and normal behavior.

DenseNet201 performs best in detecting abnormal behavior due to its dense connections, which enhance feature reuse and gradient flow for more effective learning. This improves feature extraction, allowing the model to capture subtle anomalies with greater accuracy. Additionally, its lightweight design optimizes parameter efficiency, making it highly effective for detecting abnormal patterns while maintaining computational efficiency.

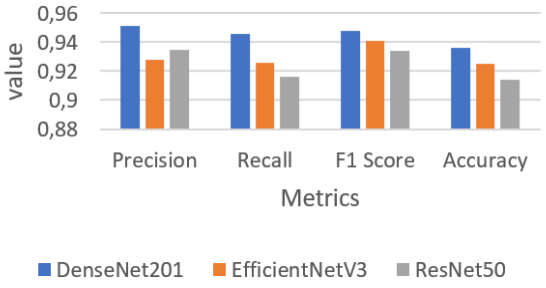

Figure 10: Comparison between three classification LightweightCNN Models based on testing results.

4.2 Results of Blockchain

This section will present the interfaces of the blockchain which is consists from authentication and authorization interfaces. These components ensure secure access control, verifying user identities and managing permissions within the blockchain system. Figure 11 shows the "initial parameters interface" for load train files; this interface includes two options:

1) Select best model: this option allows the user to choose a pre-trained model (DenseNet201, EfficientNetV3, and ResNet50).
2) Select best model with weights: this option enables the user to input the optimal weights for the selected model to enhancement its accuracy in abnormal behavior classification.

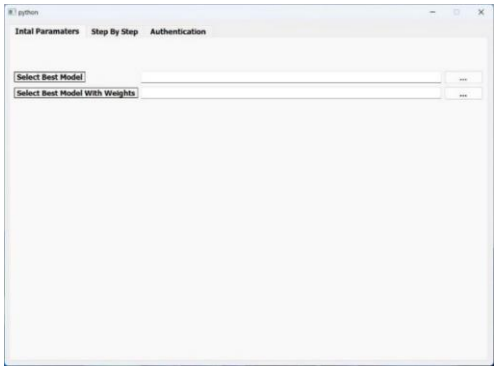

Figure 11: Initial parameters interface.

Figure 12 illustrated the interface of the Upload images from camera, extract rating value and store it in blockchain.

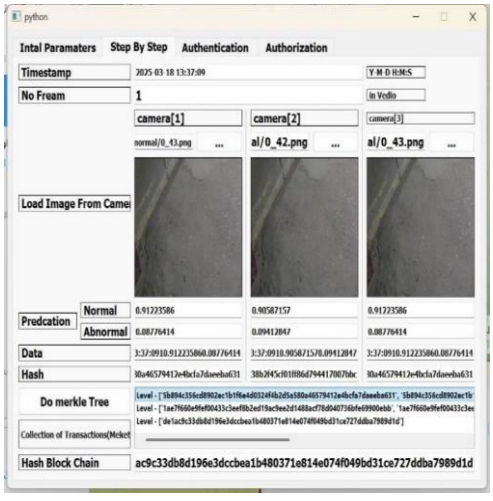

Figure 12: Interface of the upload images from camera, extract rating value and store it in blockchain.

4.3 Authentication Phase

Figure 13 and 14 shows the interface of create and saving the first block in the blockchain. The block consists from: time-stamp, Hash blockchain, Header block, Hash of previous block Header, and data.

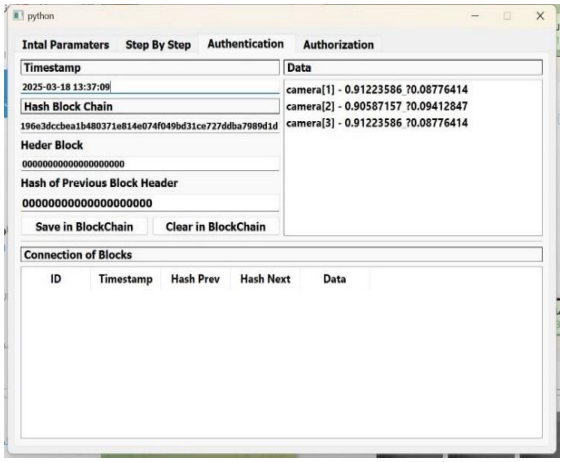

Figure 13: Interface of the create first block in the blockchain.

Figure 15 shows interface the save second block and linked with first block in the blockchain.

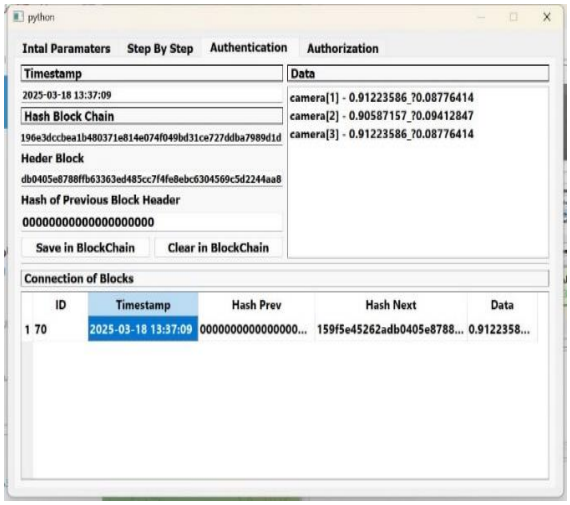

Figure 14: Interface of the save first block in the blockchain.

4.4 Authorization Phase

Figure 16 shows the interface of authorization in the blockchain. In this interface will be retrieval the first block and make decision.

Table 3: Comparison of the proposed system with related work based on dataset, detection models, and accuracy.

Reference	Detection Method	Data type	Data set	Accuracy
[10]	DnseNet201 model	Image	The research report used the MINI-MIAS dataset, which contains 114 images of abnormal tissue and 208 normal images.	92.73%
[11]	VGG16, ResNet50, VGG19, GoogleNet, LeNet, AlexNet	Image	MNIST CIFAR10 CIFAR100	99.72% 89.23% 66.70%
[12]	Resnet-50	Image	Images of a realistic environment	92.5%
Proposed BMS-LDBC	Blockchain and three lightweight CNN models	Image	Novel abnormal dataset with 46,297-sample	95 %

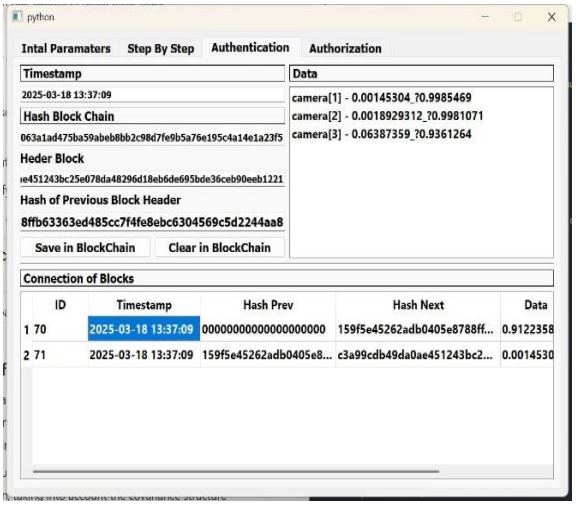

Figure 15: Save the second block in the blockchain.

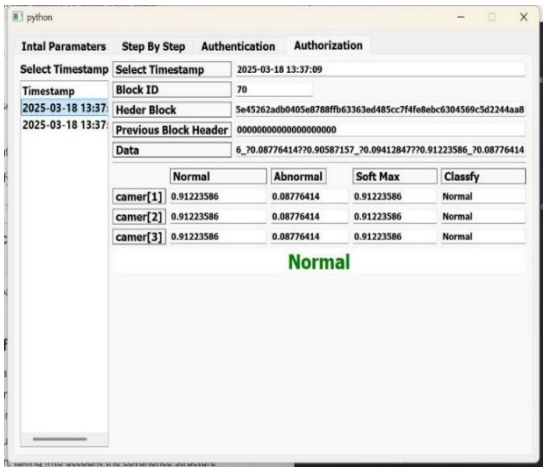

Figure 16: Interface of authorization in the blockchain

5 COMPARED THE PROPOSED SYSTEM WITH RELATED WORKS

This section compares the proposed system with related work, evaluating it based on several factors, including the type and name of the dataset, detection models, and accuracy results, as shown in Table 3. This comparison shows that while many studies have attempted to detect abnormal behaviors with comparable accuracy, the proposed system takes precedence in utilizing lightweight CNN models with blockchain to detect abnormal behaviors with higher accuracy. The proposed system is implemented on a real-time dataset, enhancing its flexibility and applicability in real-world scenarios.

6 CONCLUSIONS

The proposed BMS-LDBC system enhances smart building security by combining lightweight CNN models with parallel blockchain technology to ensure robust authentication and authorization and detect abnormal behavior. The system ensures efficient processing while maintaining low computational costs. The proposed BMS-LDBC system provides a secure and scalable surveillance environment.

The proposed system collects a realistic dataset from surveillance cameras, consisting of 46,297 samples representing normal and abnormal behaviors. This diverse dataset is used to train and test the proposed system using lightweight CNN models to effectively detect abnormal behaviors. The

proposed system uses a fine-tuning technique to optimize the performance of three lightweight CNN models (DenseNet201, EfficientNetV3, and ResNet50). This optimization improves their effectiveness in detecting abnormal behavior, making the system more robust and accurate in real-world surveillance applications. The results and implementation of the three lightweight CNN models through training and validation demonstrate that the DenseNet201 model performs best with accuracy (0.95552 and 0.94617) and loss (0.10958 and 0.11306) in training and validation, respectively.

The test results demonstrate that the proposed system achieves the best performance with the DenseNet201 model. This model outperforms the others, achieving accuracy (0.951), recall speed (0.946), F1 score (0.948), and precision (0.936), demonstrating its effectiveness in detecting abnormal behavior.

REFERENCES

[1] H. Mu, R. Sun, G. Yuan, and Y. Wang, "Abnormal human behavior detection in videos: A review," Information Technology and Control, vol. 50, no. 3, pp. 522–545, 2021, doi: 10.5755/j01.itc.50.3.27864.

[2] S. A. Jebur, K. A. Hussein, H. K. Hoomod, and L. Alzubaidi, "Novel deep feature fusion framework for multi-scenario violence detection," Computers, vol. 12, no. 9, p. 175, 2023, doi: 10.3390/computers12090175.

[3] K. Moolikagedara, M. Nguyen, W. Q. Yan, and X. J. Li, "Video blockchain: A decentralized approach for secure and sustainable networks with distributed video footage from vehicle-mounted cameras in smart cities," Electronics (Switzerland), vol. 12, no. 17, Sep. 2023, doi: 10.3390/electronics12173621.

[4] V. Plevris, N. D. Lagaros, and A. Zeytinci, "Blockchain in civil engineering, architecture and construction industry: State of the art, evolution, challenges and opportunities," Front. Built Environ., vol. 8, pp. 1–19, 2022, doi: 10.3389/fbuil.2022.840303.

[5] K. Salah, M. H. U. Rehman, N. Nizamuddin, and A. Al-Fuqaha, "Blockchain for AI: Review and open research challenges," IEEE Access, vol. 7, pp. 10127–10149, 2019, doi: 10.1109/ACCESS.2018.2890507.

[6] M. S. Mohammed and A. N. Hashim, "Protect medical records by using blockchain technology," Indonesian Journal of Electrical Engineering and Computer Science, vol. 32, no. 1, pp. 342–352, 2023, doi: 10.11591/ijeecs.v32.i1.pp342-352.

[7] L. Alzubaidi et al., "Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions," J. Big Data, vol. 8, no. 1, 2021, doi: 10.1186/s40537-021-00444-8.

[8] "Abnormal behavior detection in video surveillance using Inception-v3 transfer learning approaches," Iraqi Journal of Computer, Communication, Control and System Engineering, pp. 210–221, Jun. 2023, doi: 10.33103/uot.ijccce.23.2.16.

[9] X. Yu, N. Zeng, S. Liu, and Y. D. Zhang, "Utilization of DenseNet201 for diagnosis of breast abnormality," Mach. Vis. Appl., vol. 30, no. 7–8, pp. 1135–1144, Oct. 2019, doi: 10.1007/s00138-019-01042-8.

[10] S. T. Krishna and H. K. Kalluri, "Deep learning and transfer learning approaches for image classification," International Journal of Recent Technology and Engineering, vol. 7, no. 5, pp. 427–432, 2019.

[11] M. A. Arshed, H. Ghassan, M. Hussain, M. Hassan, A. Kanwal, and R. Fayyaz, "A lightweight deep learning model for real world plant identification," in Proc. 2022 2nd Int. Conf. Distributed Computing and High Performance Computing (DCHPC), pp. 40–45, 2022, doi: 10.1109/DCHPC55044.2022.9731841.