*mathematics*

MDPI

*Article*

# *BlockCrime:* Blockchain and Deep Learning-Based Collaborative Intelligence Framework to Detect Malicious Activities for Public Safety

**Dev Patel [1], Harshil Sanghvi [1], Nilesh Kumar Jadav [1], Rajesh Gupta [1], Sudeep Tanwar [1,*], Bogdan Cristian Florea [2,*], Dragos Daniel Taralunga [2], Ahmed Altameem [3], Torki Altameem [3] and Ravi Sharma [4]**

[1] Department of Computer Science and Engineering, Institute of Technology, Nirma University, Ahmedabad 382481, India
[2] Department of Applied Electronics and Information Engineering, Faculty of Electronics, Telecommunications and Information Technology, Politehnica University of Bucharest, 061071 Bucharest, Romania
[3] Computer Science Department, Community College, King Saud University, Riyadh 11437, Saudi Arabia
[4] Centre for Inter-Disciplinary Research and Innovation, University of Petroleum and Energy Studies, P.O. Bidholi Via-Prem Nagar, Dehradun 248001, India
* Correspondence: sudeep.tanwar@nirmauni.ac.in (S.T.); bogdan.florea@upb.ro (B.C.F.)

**Abstract:** Detecting malicious activity in advance has become increasingly important for public safety, economic stability, and national security. However, the disparity in living standards incites the minds of certain undesirable members of society to commit crimes, which may disrupt society's stability and mental calm. Breakthroughs in deep learning (DL) make it feasible to address such challenges and construct a complete intelligent framework that automatically detects such malicious behaviors. Motivated by this, we propose a convolutional neural network (CNN)-based Xception model, i.e., BlockCrime, to detect crimes and improve public safety. Furthermore, we integrate blockchain technology to securely store the detected crime scene locations and alert the nearest law enforcement authorities. Due to the scarcity of the dataset, transfer learning has been preferred, in which a CNN-based Xception model is used. The redesigned Xception architecture is evaluated against various assessment measures, including accuracy, F1 score, precision, and recall, where it outperforms existing CNN architectures in terms of train accuracy, i.e., 96.57%.

**Keywords:** convolutional neural network; deep learning; transfer learning; blockchain; smart contracts; public safety

**MSC:** 68T07



**Citation:** Patel, D.; Sanghvi, H.; Jadav, N.K.; Gupta, R.; Tanwar, S.; Florea, B.C.; Taralunga, D.D.; Altameem, A.; Altameem, T.; Sharma, R. *BlockCrime:* Blockchain and Deep Learning-Based Collaborative Intelligence Framework to Detect Malicious Activities for Public Safety. *Mathematics* **2022**, *10*, 3195. https://doi.org/10.3390/math10173195

Academic Editor: Emilio Abad Segura

Received: 5 August 2022
Accepted: 1 September 2022
Published: 4 September 2022



## 1. Introduction

In recent times, threats to public safety have been increased unceasingly, which has prompted the federal agencies to adopt various devising solutions, such as blockchain, artificial intelligence (AI), computer vision, and many more that significantly restrain such threats. People living in constant fear of threat and insecurity make it more urgent to preclude these threats before they come into action. Most of the dreadful activities annotated as a threat to public safety involve handheld arms such as guns, knives, pistols, etc. Apart from this, one more common characterization of such activities is masked faces, wherein the disturbing elements of society tend to commit their atrocities by hiding their faces under the mask [1].

Undoubtedly, the modern surveillance system can assist in finding the imminent threats by recognizing them through the feeds of surveillance devices and warning the law enforcement agencies, hence, ensuring public safety [2,3]. With time, surveillance devices have become common in any nation, which help in remote video monitoring to detect thefts

and burglaries, facility monitoring to prevent the facility's perimeter from intruders, and employee monitoring to observe their day-to-day activities. The number of such devices is increasing at a booming rate; however, it is practically infeasible to monitor them manually, and at the same time, alert authorities of all sorts of imminent threats [4]. Moreover, it also allows for security lapses that can arise due to human errors and inefficiencies caused by various reasons such as disliking a job, not serious towards its job responsibilities, personal issues, etc. In such scenarios, it is more favorable if there can be a way of making this entire process automatic by instructing computers to recognize such threats from the visuals of surveillance devices and prompting the law enforcement agencies beforehand.

Figure 1 shows the indication of the magnitude of crime occurring throughout the world. The crime index shown in the Figure 1 is the number of crimes committed per hundred thousand people in that nation [5]. Such anarchy levels demonstrate the critical need for a system that can prevent such incidents. To overcome such law-disturbing activities, there is a need for an automated system that can alert the appropriate law enforcement agencies of these wrongdoings before the atrocities are committed. Therefore, machine learning can be adopted as a favorable solution to tackle imminent threats by deploying them in the surveillance system. Nevertheless, machine learning is a cumbersome task because of its explicit feature handling and slow training data processing. Consequently, it leads to unwanted errors and sluggish implementation, hence, it becomes an incompetent system. To solve this problem, individuals divert to deep learning (DL) where various models are present that do not require the manual setting of features and they learn those features by themselves [6]. With the increasing computational power, deep learning has proven to be very effective and proficient in computer vision applications [7].

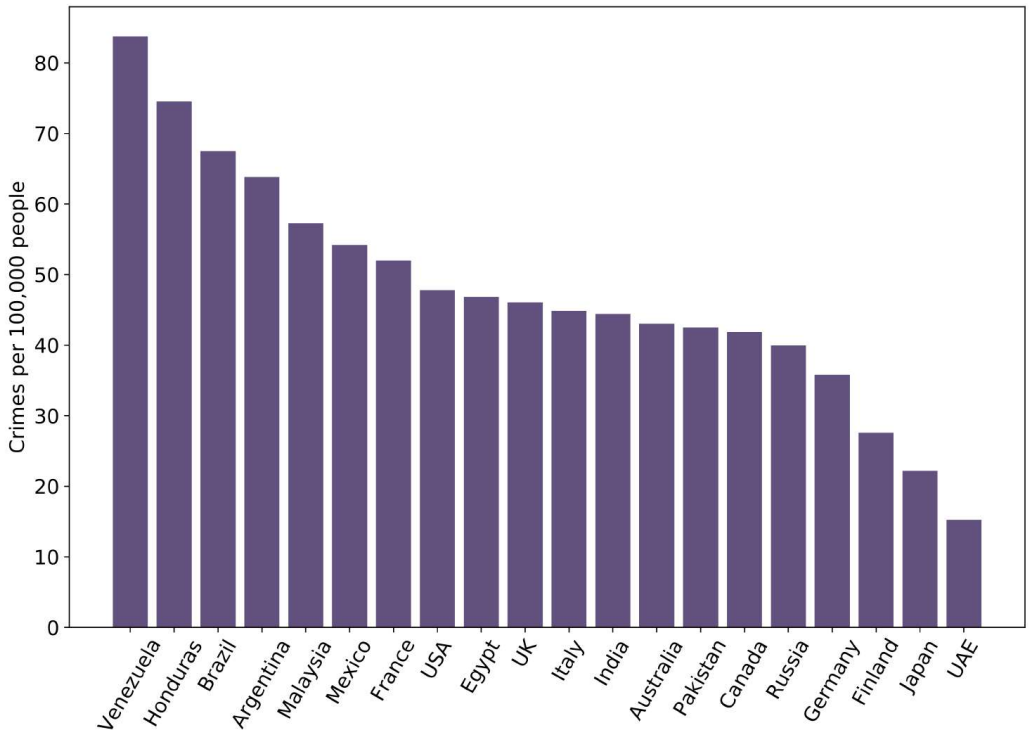



**Figure 1.** Global crime index [5].

To overcome the earlier aforementioned issues, many machine learning algorithms were used for the image retrieval/classification task. However, to improve the performance of image retrieval by reducing the "semantic gap", DL was introduced for image classification in 2006 by Hinton et al. [8]. Since then, for computer vision applications, such as digital recognition [9,10], image classification [11,12], and face recognition [13], convolutional neural networks (CNNs) and deep belief networks (DBN) have chartered

unparalleled success. The proficiency of CNN architectures such as LeNet, ResNet50, ResNet101, ReNet152, ResNet50 V2, VGG16, VGG19, and Xception has been tested by making use of transfer learning. The most effective model is then chosen and its pre-trained version is fine-tuned, which is explained in further sections. The objective behind the use of pre-trained models of famous architectures is to tackle the problem arising due to the scarcity of data. In DL, the availability of a large dataset is crucial, otherwise, the model fails to deliver high accuracy, and thus, fails in the task for which it is trained. As a savior, the concept of pre-trained models came into picture, which do not require training on a large dataset comprising of a large number of general objects. This makes the model classification-ready. By making use of their pre-trained weights, we can fine-tune them to fit our classification purposes.

Motivated by the aforementioned benefits, in this paper, we propose a framework called *BlockCrime* that intends to sense the imminent crime activities by detecting the characteristics braced by visuals using the DL algorithm and blockchain technology [14]. In case of any malicious activity, the framework flags this information to the nearby law enforcement agencies in real-time. This collaborative intelligent framework helps to detect malicious activities, thus, ensuring public safety. As it necessitates a high level of prudence and monitoring, this work is restricted to technical and analytical issues and does not examine social factors. With the availability of a big and diverse dataset, this work may be improved even more. A large corpus will also aid in the use of various DL algorithms for more accurate results [15].

### *1.1. Research Motivation*

The motivation of the proposed framework is elaborated as follows:

- The existing state-of-the-art crime detection approaches only perform crime detection but do not alert the nearby law enforcement agencies about the detected crime [16–24]. Thus, there is a need for an end-to-end system that detects crime from surveillance equipment and alerts nearby law enforcement agencies with the coordinates of crime occurrence.
- Existing baseline approaches can detect only knives and no other objects of crime that suffer from the generalization problem [21,23]. For instance, if the algorithm detects only knives, it can miss out on many potential criminal activities. This is because criminal activities involve many other weapons and not only knives. Therefore, there is a requirement for an intelligent system that detects a wide array of weapons that can be used to commit a crime.
- Many existing state-of-the-art approaches use conventional computer vision and image processing algorithms [19,20,22]. However, the exceptional power of AI, such as transfer learning, has not been explored to its potential in these approaches [24]. Thus, there is a need for a technology that harnesses the power of AI and fully capitalizes it by using transfer learning.
- Motivated by this, we have explored integrating the AI model and blockchain technology to detect malicious activities in real-time and securely communicate their coordinates, enhancing public safety.

### *1.2. Research Contributions*

The substantial contributions of the paper are as follows:

- We propose an intelligent framework based on deep learning and blockchain to detect crimes and provide immediate assistance by alerting the nearest law enforcement agencies.
- To train the dataset images, a CNN-based Xception model is used, which detects whether the image is associated with a crime scene or not. The dataset is created by gathering crime images from the web and a few images are extracted from surveillance video feed systems.

- We design customized blockchain smart contracts to enhance the security and privacy of the proposed *BlockCrime* framework. It validates the detected crime scene's location and securely stores it within the blockchain network.
- We then evaluate the proposed framework using performance metrics such as accuracy score, confusion matrix, F1 score, precision, and recall.

### *1.3. Paper Organization*

The following is the flow of the paper: Section 2 describes the state-of-the-art works in the domain. Section 3 discusses the system model and problem formulation. The proposed framework for the given problem statement is described in Section 4. Section 5 presents the results and discussion. Finally, the paper is concluded in Section 6.

## 2. Related Works

This section discusses various state-of-the-art approaches presented by the researchers worldwide on crime detection towards the accomplishment of public safety. For example, Olmos et al. [21] proposed an automatic handgun detection system from the video feed to offer a high crime-detection rate even in the low-quality videos. This has been achieved using a faster region-based convolutional neural network (FRCNN) and region-based CNN (RCNN) models. However, its limitation is to detect only guns as a signal of crime. Dever et al. [25] have used a skeleton silhouette algorithm to identify robbery where a knife has been used as an object of threat. The issue with this approach is that it can recognize only a knife as an object of threat. Furthermore, they have not leveraged the benefits of AI models. Further, Buckchash et al. [23] proposed a novel knife detection scheme to detect knives at all positions and of different sizes (scale). For that, they utilize features from accelerated segment test (FAST) for feature detection and multi-resolution analysis (MRA) for classification. However, the aforementioned papers have not explored real-time crime detection.

With the goal of detecting firearms in the picture and accurately classifying them, Lai et al. [24] used a Tensorflow-based implementation of the Overfeat network in the form of an integrated network. They just focused on weaponry as a marker and did not fully capitalized the power of transfer learning in CNN. Sivakumar et al. [18] put forward a DL-based real-time crime detection technique that watches real-time videos and made the nearby cybercrime admin aware of the unfortunate incident. The YOLO model has been utilized by the authors for object detection. Although they have put forward the idea of alerting the nearby cybercrime admin, they have not presented how exactly it will be carried out, and the additional underlying security of blockchain is not provided. Apoorva et al. [19] presented a real-time face recognition system which makes use of the Haar classifier to track faces on the OpenCV platform. The idea behind this approach was to compare the image with other images which are stored and trained. The drawback of their method is that it does not outline any framework to alert a nearby law enforcement agency in case of any threat and it neither makes use of blockchain for the communication of such alerts [26]. Chhoriya et al. utilized the Haar feature-based cascade classifier for automated facial recognition of malicious people [20]. Here, an automated surveillance camera is used for the real-time face recognition. However, there is no discussion about alerting nearby law enforcement agencies and making use of blockchain. The authors Elrefaei et al. provided a framework for real-time facial recognition monitoring [22]. It is a client-server video-based framework with a major focus on client-side face identification and tracking for Android handsets. The OpenCV library was used to perform the face detection operation.

Jan et al. [16] explores 2D and 3D CNNs to recognize assault, fighting, shooting, and vandalism from video sequences from CCTV cameras. Their framework claims to achieve 91.2% accuracy in detecting the crime activities and also reduces the false alarms. However, no information is provided about how to alert the authorities when crime is detected. Kumar et al. [17] explored a novel lion-based deep belief neural paradigm to

identify criminals by their behavioral activities. Table 1 shows the relative comparison of the proposed approach with various state-of-the-art approaches in crime detection for public safety.

**Table 1.** Relative comparison of the proposed approach with various state-of-the-art approaches in crime detection for public safety.

| Author | Year | Objective | Pros | Cons |
|---|---|---|---|---|
| Proposed work | 2022 | Detect crime using surveillance camera feeds and notify local law enforcement authorities of the crime scene utilizing blockchain's security. | For crime detection, uses state-of-the-art CNN architecture with transfer learning and issues the coordinates of the crime location through blockchain, assuring no manipulation and increased security. | - |
| Jan et al. [16] | 2022 | Identification of assault, fighting, shooting, and vandalism from video sequence using deep 2D and 3D CNN. | The 2D CNN, even with lesser parameters, achieved a promising classification accuracy of 91.2%. The system also reduces false alarm rate. | The system does not mention how to alert the authority on detecting crime. |
| Kumar et al. [17] | 2022 | Identify criminals by their activities and handling tools | Proposes a novel lion-based deep belief neural paradigm | The only purpose is to detect and not what to do with the detected results. |
| Sivakumar et al. [18] | 2021 | Monitor real-time video and notify the local cyber-crime Admin of the crime's incidence and location. | Extremely fast architecture that can process image in real-time at 45 fps. | Trained on a dataset created using high-res images. No results regarding performance on low-res images which will be the more common scenario. |
| Apoorva et al. [19] | 2019 | malicious identification by face recognition in real-time using an automated surveillance cameras. | Can recognize more than one face robustly in real-time environment. | Not capable of identifying potential malicious people by detecting objects of crime such as knife, gun, etc. |
| Chhoriya et al. [20] | 2019 | Identify and distinguish malicious people's faces in a real-time video feed collected from a camera. | Makes use of location and scale invariant detector and efficient selection of features. | Can just identify malicious people whose facial data are present in the database and cannot spot potential malicious people. |
| Olmos et al. [21] | 2018 | Detect Guns as a signal of crime. | Detects crime even in low-quality video. | Algorithm can only detect guns and no other object of crime, such as knives. |
| Elrefaei et al. [22] | 2017 | Recognize the face of a malicious person or a suspect. | Makes use of the Viola–Jones algorithm that is not affected by illuminations.<br>Designed to work on low hardware device such as android smart phones. | Does not experiment Optical Flow with CAM Shift algorithm which can handle fast and large movements, hence, improving overall accuracy. |
| Buckchash et al. [23] | 2017 | Their proposed approach can detect knives in video feed. | Knife detection at all position and different scales. | Can only detect knives for malicious activity. |
| Lai et al. [24] | 2017 | Detecting firearms and classifying them. | Used Tensor Flow based Overfeat network. | Did not focus on transfer learning techniques. |

## 3. System Model and Problem Formulation

This section presents the system model and problem formulation for the proposed work. It efficiently detects the crime and quickly alerts the nearby police station, thereby improving the public safety.

### *3.1. System Model*

Figure 2 shows the proposed system model, i.e., *BlockCrime*, which comprises surveillance equipments deployed in the city to ensure the safety of its residents. These equipments are seen heavily in use, especially where the stringent monitoring and critical protection is required from the delinquents; while continuous live video feed capturing, the surveillance systems capture the image frames at regular intervals. Then, the image frame information is passed to the trained CNN architecture, where CNN performs its forward propagation to detect any threat or crime. If the CNN model detects any unusual activity, it raises a flag for the crime. As a result, the blockchain-based smart contract gets activated, where it stores the location (coordinates) of the crime scene. Then, with appropriate distance computations between the crime scene and police station, the nearest police stations are pinpointed. Finally, the nearby police station is alerted about the crime scene through a secure channel and is bound to arrive at the location immediately.

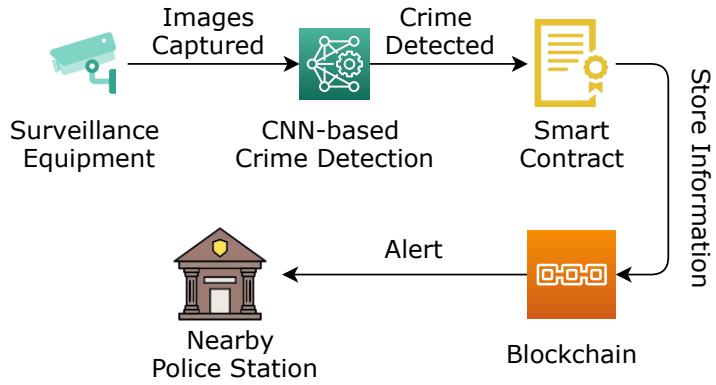



**Figure 2.** *BlockCrime* system model.

### *3.2. Problem Formulation*

In this subsection, we formulate the problem scenario for *BlockCrime* to detect malicious activity for public safety. Let $E = \{E_1, E_2, E_3, ..., E_i..., E_M\}$ be the surveillance equipments set-up (i.e., surveillance devices) around the city for constant monitoring where $1 \leq i \leq M$, and $M$ represents the total units of equipment deployed. Suppose each gadget captures image frames $I = \{I_1, I_2, I_3, ..., I_j, ..., I_N\}$ over some time where $1 \leq j \leq N$ and N represent total image frames captured. Let $P$ be a person in the perimeter of the surveillance equipment $E_i$ who is being robbed on the tip of a gun. The surveillance equipment $E_i$ capture that scene as an image frame $I_j$. All the captured image frames are continuously transferred to a trained CNN architecture for crime detection.

$$E_i(I_j) \xrightarrow{\text{captured}} g(\theta), \tag{1}$$

where, g($\theta$) denotes the functions of CNN having parameters $\theta$, such as weight, bias, and kernel size.

The image frame $I_j$, when tested on the CNN model, detects a threatening act depending on the $I_j$.

$$SC \xleftarrow{\text{satisfy}} h(g(\theta)), \tag{2}$$

where h(g($\theta$)) represents the predicted output of the CNN model. If it stands to be 1, the image frame $I_j$ captured by $E_i$ indicates a potential crime, otherwise not. The $I_j$ has potential information about the crime, such as location (coordinates) of the crime scene, possible witnesses, a weapon used in the crime, and many more. Hence, it is essential to secure this information without being manipulated by the attackers. This is because if the attackers find this information, they certainly try to modify it to escape from the judicial try. Therefore, the blockchain technology is integrated into the proposed system model, which initially prompts the information to validate its authenticity via smart contract *SC*. If all the conditions of *SC* are satisfied, the crime information (collected from surveillance systems and detected by a CNN model) is securely stored inside the immutable distributed ledger, i.e., blockchain. In addition, the secured information is only accessible to the crime investigation agencies who are destined to respond to the situation swiftly.

$$\psi_k, \omega_k \xrightarrow{stored} B, \tag{3}$$

where $\psi_k$ and $\omega_k$ represent the latitude and longitude of the crime scene, which is arbitrarily tagged as $k$. $B$ represents the blockchain onto which the coordinates are stored. The nearby police quarters are computed and alerted by the prototype application using the Haversine formula, which is represented mathematically in Equation (4) [27].

$$d = 2r \sin^{-1}\left(\sqrt{\sin^2\left(\frac{\psi_2 - \psi_1}{2}\right) + cos(\psi_1) \cdot cos(\psi_2) \cdot \sin^2\left(\frac{\omega_2 - \omega_1}{2}\right)}\right), \tag{4}$$

where $d$ is the calculated distance, $\psi_1$, $\omega_1$ and $\psi_2$, $\omega_2$ are the latitudes and longitudes of two points, respectively. The calculation assumes that the Earth is a perfect sphere while it is an oblate spheroid; therefore, this is not an accurate measurement. However, it returns a result that is about equivalent to the real distance.

$$min\{d(k, P_z)\} \xrightarrow{alert} P_n, \tag{5}$$

where $z$ and $n \in [1, Q]$ and $P_n$ denotes the nearest police station. $Q$ represents the total number of police quarters. Thus, we compute the distance of stored coordinates on the blockchain with the coordinates of all the police stations $P_z$. Assuming that $P_n$ is the nearest police station detected, we then alert that police station.

## 4. *BlockCrime* Framework

Figure 3 shows that the proposed system architecture consists of four layers, i.e., the cognitive, AI, surveillance, and defense layers. The cognitive layer has a dataset that comprises multiple image frames gathered from numerous crime sources. Then, the cognitive layer's pre-processed dataset is fed as an input to the AI layer, where it trains the neural network architecture on the dataset and prepares it to make predictions on unprocessed data. In addition, we also consider the crime scene from the video feed of the surveillance system deployed in various areas. Hence, the surveillance layer also collects data from multiple sources and delivers it to the AI layer for prediction. Once the CNN model detects the crime, it activates the defense layer to alert the neighboring law enforcement agencies and notifies them with the location of the crime scene through the smart contract deployed on blockchain. The following is the detailed description of the four-layered architecture:

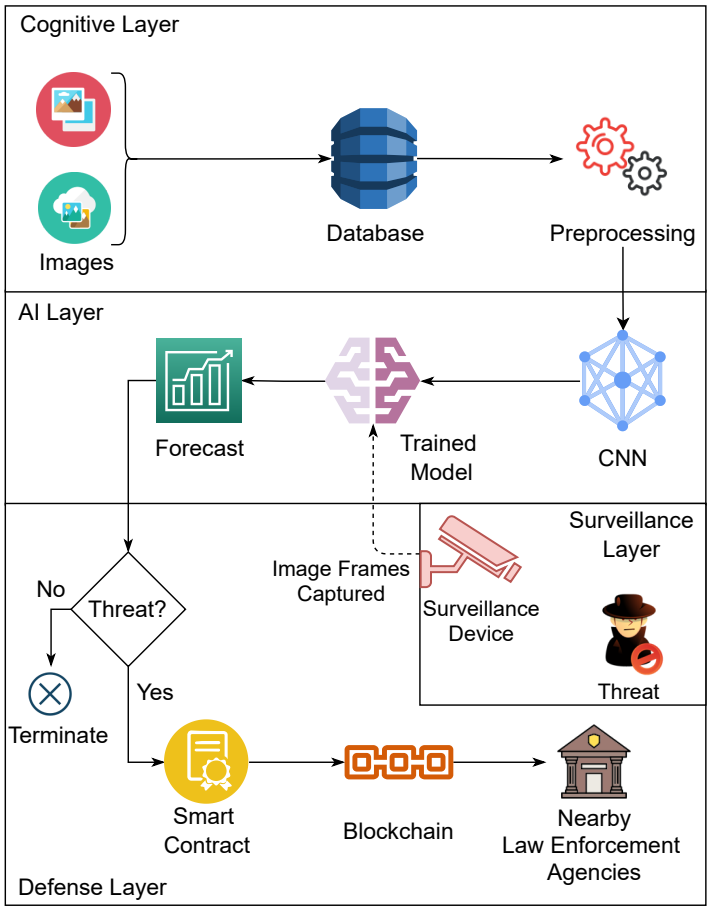



**Figure 3.** The proposed *BlockCrime* framework.

### *4.1. Cognitive Layer*

The cognitive layer accumulates crime images from numerous sources, such as social media, video-sharing websites, trusted sources such as law enforcement agencies, and many more. The collected images are then saved in the database and subsequently processed for high-quality output. We have gathered images of crime scenes from the aforementioned sources and created our own dataset consisting of normal images and crime images. Our original images in the dataset consist of red, green, and blue (RGB) channels with values between 0 and 255, which would be quite high for the proposed model to analyze. To solve this problem, all the images from the training and the testing set are rescaled to a range between 0 and 1 by scaling with a 1/255 factor. Further, to enhance the performance of the AI detection, the images are pre-processed by employing data augmentation and image transforming techniques. First, a shear intensity of 0.2 is applied to slant the training images for better analysis. Next, the images are zoom out and zoom in for better viewing by initializing the zoom range between (0.8 and 1.2). Lastly, training photos are randomly flipped horizontally by setting the horizontal flip to "True."

### *4.2. Surveillance Layer*

This layer comprises of surveillance technology scattered around the city anywhere continual monitoring is warranted. These devices capture image frames frequently and feed them to the trained neural network present in the AI layer, which uses them to foresee any indication of crime or danger identified. These systems are essentially comprised of closed-circuit television cameras, commonly known as CCTV cameras. These cameras record real-time events that occur inside their field of view. However, manual monitoring with these devices often results in inaccuracy and non-identification of prospective dangers.

In such a case, the proposed surveillance layer passes the captured frames of the feed to the AI layer, which accurately detects the crime. These frames are taken from the real-time stream on a regular basis. The *OpenCV* library is used to extract pictures from real-time CCTV video [28]. The live video capture is passed into the $VideoCapture()$ function of *OpenCV* by setting the parameter to zero. The frames are then sampled from the video stream using the $read()$ function. The frames collected via this process are then used for detection by being sent to the AI layer. The flow of the pipeline is addressed further in the framework's succeeding levels.

### *4.3. AI Layer*

The CNN model is used in this layer to detect crimes from the data gathered from the cognitive layer. Several experiments with multiple CNN models determined that Xception (CNN-based model) achieves a balance of classification accuracy and area under the curve (AUC) score. Xception is a deep CNN architecture developed by Google researcher François Chollet and reported in [29]. Xception is an efficient model that depends on two major points [30]:

- Depth-wise separable convolutions.
- Shortcuts between convolution blocks similar to in ResNet.

The model's uniqueness comes from the idea that the mapping of cross-channel correlations and spatial correlations in CNN feature maps can be separated.

On the other hand, Xception uses filters on each depth map and then compresses the input space using $1 \times 1$ convolution. The methodology included here closely mimics the method known as depth-wise separable convolutions.

The availability of fewer data to train a full-scale model from scratch motivated us to take advantage of transfer learning and prevent overfitting. The AI layer comprises Xception as the basic model to leverage the features learned on the ImageNet dataset and fine-tune them for our objective. Algorithm 1 shows the entire flow of Xception model to detect the crime. A comprehensive description of the architecture is as follows:

- The core base model comprises the CNN-based Xception model. It contains 36 convolutional layers comprising the feature extraction basis of the network. All these layers are restricted to avoid deleting any type of information during future learning.
- On top of the base model, the following trainable layers are added:
  - A flatten layer which flattens the input, i.e., $4 \times 4 \times 2048$ size input is converted into 32,768;
  - A dense layer consisting of 4096 neurons with ReLU activation;
  - Dropout layer consisting of 4096 neurons with a rate equal to 0.5;
  - A dense layer consisting of 4096 neurons with ReLU activation;
  - Dropout layer consisting of 4096 neurons with a rate equal to 0.5;
  - A dense layer consisting of 1 neuron with Sigmoid activation.
- These new layers, which are built on top of the existing model, transform the old characteristics into predictions on the target dataset. The proposed architecture can be seen in Figure 4.
- Except for the first and final modules, the 36 convolutional layers that comprise the Xception model are organized into 14 modules, all of which are surrounded by linear residual connections [29].
- Formally, Xception architecture is a rectilinear pile of depth-wise separable convolutional layers with residual connections.
- On top of this architecture, we added a few more layers which consist of a combination of fully linked and dropout layers. These layers use characteristics retrieved by the Xception CNN model and categorize the input into one of the target classes.

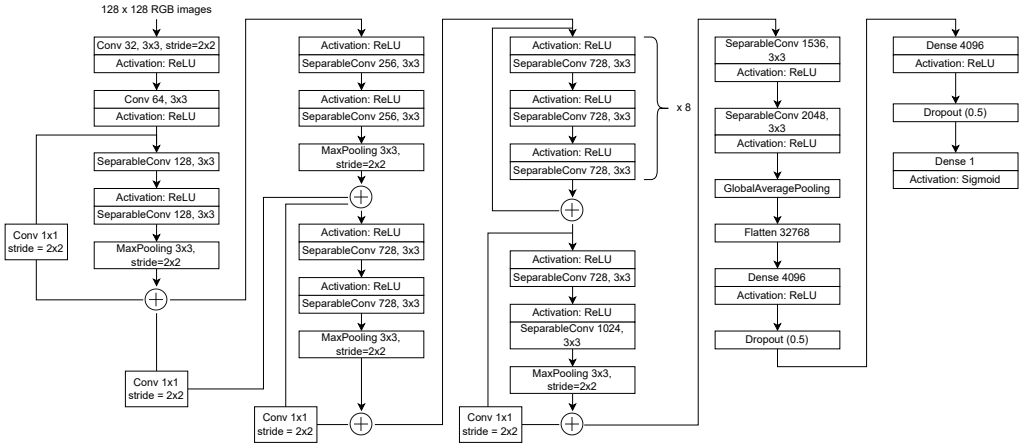



**Figure 4.** Xception ImageNet pre-trained model with transfer learning.

We have used ReLU as an activation function in the hidden layers. The computation of ReLU is carried out as shown in Equation (6).

$$a(z) = max(0, z). \tag{6}$$

For the final layer, classification is carried out by using the Sigmoid activation function, which is computed according to the formula given in Equation (7).

$$y(z) = \frac{1}{1 + e^{-z}}. \tag{7}$$

During the learning phase, backpropagation is used for adjusting weights and tuning them as per our classification task [31]. For implementing backpropagation, the derivatives of activation functions are used. The derivative of the ReLU function is represented in Equation (8) and the Sigmoid function is represented in Equation (9).

$$\begin{aligned} a^{'}(z) &= 1, z >= 0 \\ &= 0, z < 0 \end{aligned} \tag{8}$$

$$y^{'}(z) = (\frac{1}{1 + e^{-z}}) * (\frac{e^{-z}}{1 + e^{-z}}). \tag{9}$$

For the optimization of the proposed neural network, we utilize an optimizer, i.e., the Adam optimizer [32]. It takes momentum [33] and RMSprop [34] and combines them. Typically, a mini-batch gradient descent [34] is used, followed by the momentum exponentially weighted average. This method combines the effects of gradient descent and RMSprop. There are various hyperparameters used in this method, such as the learning rate ($\alpha$), batch size, epochs, and many more, to optimize the result of our AI model. Equation (10) denotes exponentially weighted averages for weights and bias.

$$\begin{aligned} V_{dW} &= \beta_1 * V_{dW} + (1 - \beta_1) * dW \\ V_{db} &= \beta_1 * V_{db} + (1 - \beta_1) * db \end{aligned}. \tag{10}$$

Here, $dW$ represents the gradient of weight and $db$ represents the gradient of bias, $V_{g(t)}$ is the mean (first moment) of gradients where $g(t)$ stands for $dW$ and $db$ [34]. $\beta_1$ is a momentum-related term whose default value is chosen as 0.9. $\beta_1$ computes the exponentially weighted average of the derivatives [34].

$\beta_1$ is referred to as the first moment, while $\beta_2$ is used to calculate an exponentially weighted average of the squared gradients, as seen in Equation (11) [34], which is referred to as the second moment.

$$\begin{aligned} S_{dW} &= \beta_2 * S_{dW} + (1 - \beta_2) * dW^2 \\ S_{db} &= \beta_2 * S_{db} + (1 - \beta_2) * db^2 \end{aligned} . \tag{11}$$

Here, $S_{g(t)}$ is the uncentered variance (second moment) of gradients where $g(t)$ stands for $dW$ and $db$ [34]. The authors of the Adam paper advocate a value of 0.999 for the hyperparameter $\beta_2$ in [32].

As a result, the term "adaptive moment estimate" was coined. A bias correction is implemented in the standard Adam implementation, which can be seen in Equations (12) and (13) [32].

$$V_{dW}^{corrected} = \frac{V_{dW}}{1 - \beta_1^t}, \quad V_{db}^{corrected} = \frac{V_{db}}{1 - \beta_1^t} \tag{12}$$

$$S_{dW}^{corrected} = \frac{S_{dW}}{1 - \beta_2^t}, \quad S_{db}^{corrected} = \frac{S_{db}}{1 - \beta_2^t}. \tag{13}$$

After computing these moments and performing bias correction, the parameters are updated using the update algorithm described in Equations (14) and (15).

$$W = W - \alpha * \frac{V_{dW}^{corrected}}{\sqrt{S_{dW}^{corrected}} + \epsilon} \tag{14}$$

$$b = b - \alpha * \frac{V_{db}^{corrected}}{\sqrt{S_{db}^{corrected}} + \epsilon} \tag{15}$$

where, $W$ and $b$ represent the weight matrix and bias vector of the CNN architecture, respectively. The choice of $\epsilon$ is not important, however, the authors of the Adam study advocate a $10^{-8}$, but one does not need to set it, and it has little effect on performance [32]. However, when implementing Adam, it is common to utilize the default values of $\beta_1$ and $\beta_2$, as well as $\epsilon$.

**Algorithm 1** Xception to detect crime

**Initialize:**

- X = Xception base model with transfer learning
- Loss = Binary_CrossEntropy
- Optimizer = Adam

$I_{m \times n} \xrightarrow{\text{rescale}} I_{128 \times 128}$
$a_{<32768>} = \text{X}(I_{128 \times 128})$
$i \leftarrow 1$
**while** $i \leq 2$ **do**
  $a_{<32768>} \xrightarrow{Dense(units=4096)} a'_{<4096>}$
  $a_{<4096>} \xrightarrow{Activation(relu)} a''_{<4096>}$
  $dropout(50\%)$
**end while**
$a''_{<4096>} \xrightarrow{Dense(units=1)} y_z$
$\hat{y} = \frac{1}{1+e^{-y_z}}$
$loss = Binary\_CrossEntropy(y, \hat{y})$
Backpropogate and update weights & biases
**During testing**:

$$Output = \begin{cases} No - crime & \hat{y} < 0.5 \\ Crime & \hat{y} \geq 0.5 \end{cases}$$

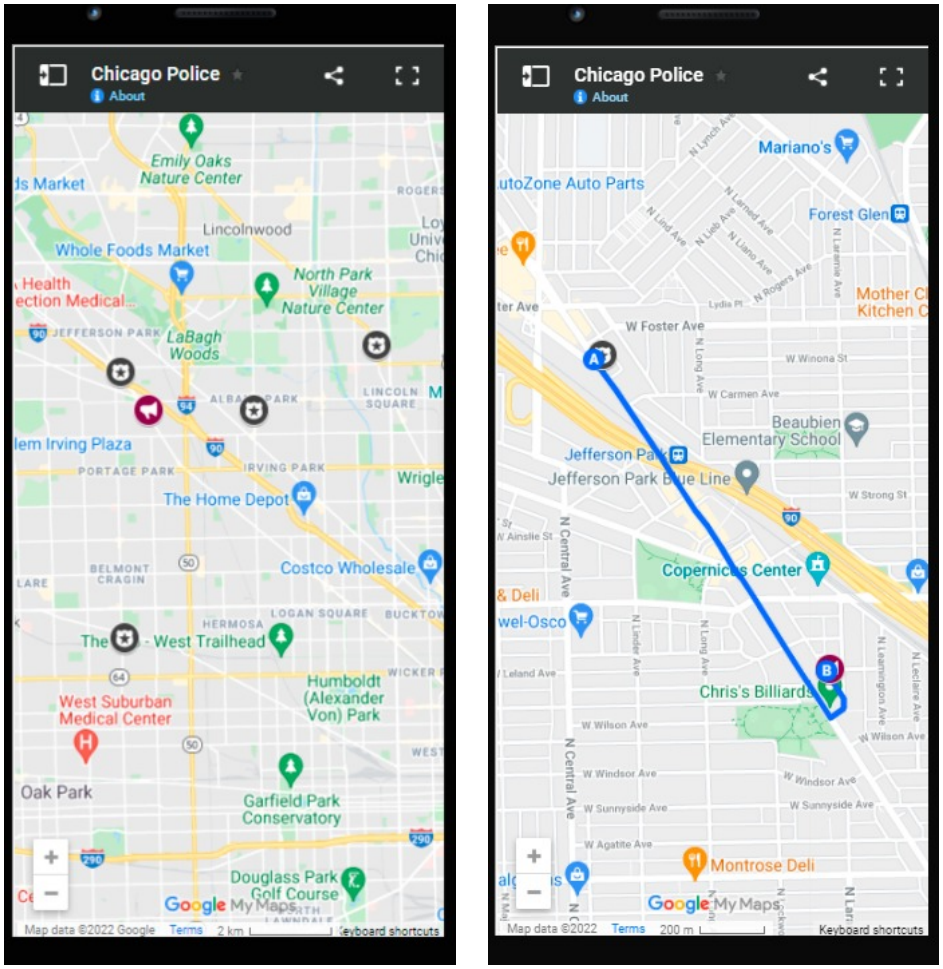
### *4.4. Defense Layer*

The trained neural network can forecast whether the image captured by the surveillance system poses a threat to society or not. If the CNN model classifies the activity as a malicious activity from the captured image, the coordinates of the location where the image is captured are stored into the blockchain network. This location information is very sensitive and if posed under an attack, it can be modified or tampered. This raises a question of data integrity. The need for a mechanism to store sensitive information securely is required. Blockchain poses to be a solution to such problem. Before storing the location data inside the blockchain, they are first verified by the smart contracts. They are the software programs that execute when specific criteria are satisfied. For example, it allows trustworthy transactions and contractual agreements to be executed among divergent, anonymous parties without the requirement for centralized authority, a legal system, or an external enforcement mechanism. Only authorized law enforcement agencies and judicial employees would be able to access the information stored on the blockchain under the proposed architecture's smart contracts. In the proposed framework, a smart contract having various functions to validate the incoming data has been developed. These functions are described in detail in Section 5.3. Once the data are verified, it is securely stored inside the blockchain, where the essential characteristics of blockchain, such as immutability, transparency, distributed ledger, and decentralization network [35], secure the location data against any data integrity attacks. It stores data in blocks that are connected and secured by cryptographic public–private key pairs. After successfully adding data, all blocks are updated and shared with the authorized blockchain members to build transparency in the blockchain network. The decentralized blockchain is irreversible, which means that once the information is added, it cannot be modified later. If someone tries to modify a block, this modification will be seen by every member of the blockchain and soon eradicated from the trusted blockchain network. In the proposed architecture, police can use blockchain technology to gain secure access to the coordinates of the crime location.

The proposed framework would be dependent on 5G network as a communication channel. The performance of 5G network is far better than traditional networks. It shows low latency ($<$1 ms), high data rate ($\leq$10 Gbps), and high efficiency (99.99%). The aforementioned properties lead to a ultra-low latency system; thus, if any crime is detected, it quickly reaches the nearby crime agency with a minimum end-to-end delay.

### *4.5. Prototype Implementation*

Figure 5 depicts the prototype application that would help the police officers to access the coordinates of the crime location from the blockchain. Figure 5a shows the police headquarters near the crime location. The black icon depicts the police headquarters, while the dark magenta icon shows the crime place. The police officer responding to the alert may find the directions to the location, as displayed in Figure 5b.




(**a**) (**b**)

**Figure 5.** Prototype application for finding the location of the nearest police station. (**a**) Nearby police quarters. (**b**) Direction to the crime location.

### *4.6. System Configuration*

The CNN architecture is designed over Python v3.7.12 and Tensorflow v2.7.0 on a Google Colab with 12.69 GB of RAM. The Tesla K80 GPU was used to speed up the training process. The Remix Integrated Development Environment (IDE) v0.21.4 is used to deploy the smart contract, which is built in Solidity v0.8.7.

### *4.7. Dataset Description*

The dataset utilized for the study comprises a total of 715 images. Out of which, 100 images are used for testing purposes, while the remaining 615 images are used for training the model. The train set includes 263 photos classified as *No crime* and 352 images marked as *Crime*. A total of 34 photos classified as *No crime* and 66 images marked as *Crime* make up the test set. The images labeled as *Crime* contains people carrying vulnerable objects such as a knife or a gun, using them to attack or rob civilians. Furthermore, few images are of people looking similar to terrorists. The images labeled as *No Crime* contain frames of crowds and people under normal conditions, as captured from a surveillance feed. By using this dataset, a model is trained that can classify whether the provided image pertains to a threat or not. Further development of the dataset can lead to improved accuracy and, consequently, better outcomes in the future.

Algorithm 2 shows the pre-processing steps applied to the image data. Each pixel of image $(I_p)$ has been normalized in the range from 0 to 1. Further, data augmentation techniques such as shear, zoom, and directional flip have been applied.

**Algorithm 2** Real-time data augmentation and pre-processing on the image data

**for** $I_p \in \mathcal{I}$ **do**
$I_p \leftarrow I_p/255$
**end for**
$I_p^1 \xleftarrow{\text{Shear Intensity of 0.2}} I_p$
$I_p^2 \xleftarrow{\text{Random zoom [0.8, 1.2]}} I_p$
$I_p^3 \xleftarrow{\text{Flip horizontally}} I_p$

## 5. Results and Discussions

This section evaluates the CNN models used for classification purposes in the proposed framework, i.e., *BlockCrime*. It is evaluated on the basis of different performance metrics, such as accuracy, precision, recall, and F1 score. These metrics are critical for selecting a final model for the problem.

### *5.1. Experiment Setup and Simulation Analysis*

We developed digital smart contracts for blockchain using the Remix IDE version 0.12.4 and the solidity programming language version 0.8.7, which eliminates the need for intermediary third-party services and enhances the performance of the blockchain network. Additionally, by safely storing the data in the immutable ledger, the blockchain protects the data connection between the AI and the defense layer. Python v3.7.12 was used as the environment for orchestrating the entire framework. Tensorflow v2.7.0 was used to design the deep learning model and perform transfer learning. To handle the dataset and perform floating point operations, Pandas v1.4.1 and NumPy v1.22.1 were used. The proposed architecture was evaluated against performance metrics, such as accuracy, recall, precision, F1 score, and ROC curve. For this evaluation, scikit-learn v1.0.1 was used. For exploratory data analysis and results visualization, matplotlib v3.5.1 was taken into action. The entire architecture was simulated on a cloud computing facility named Google colab with configurations such as Intel(R) Xeon(R) CPU @ 2.20 GHz, 12.69 GB RAM, and Tesla K80 graphic card for ten epochs and batch size 32. This setup was the bare minimum required to simulate the proposed framework due to the use of Remix IDE and the computationally expensive CNN architecture. Further, the 5G communication was established by maneuvering the 5G communication toolbox of the MATLAB tool. The authors explored various literature papers on 5G communication, and as per that, the 5G parameters were included in the 5G communication toolbox of the MATLAB. The various simulation settings utilized in the proposed framework are also shown in Table 2.

**Table 2.** Simulation parameters of the proposed framework.

| Parameters | Values |
|---|---|
| Wireless network layer parameters | |
| Frequency range | 28 GHz |
| Channel bandwidth | 110 MHz |
| Subcarrier spacing | 30 KHz |
| Modultation | OFDMA |
| Channel coding | Linear coding |
| Fading channel | Rayleigh Channel |

**Table 2.** *Cont.*

| Parameters | Values |
|---|---|
| AI layer parameters | |
| Epochs | 10 |
| Batch size | 32 |
| Optimizer | Adam |
| Loss function | Binary Cross Entropy |
| $\alpha$ | 0.001 |
| $\beta_1$ | 0.9 |
| $\beta_2$ | 0.999 |
| $\epsilon$ | $10^{-8}$ |
| Defense layer parameters | |
| Solidity compiler | v0.8.7 |
| Remix IDE | v0.21.4 |
| Gas limit | 3,000,000 |

### *5.2. Evaluation Metrics*

The evaluation metrics give a quick comparison between the different models. In addition, they provide us with an in-depth view of how well the model is trained on the data and how well it performs on unseen data. Various metrics used in selecting the final CNN model are discussed below.

#### 5.2.1. Confusion Matrix

A confusion matrix is a term used to describe a table that summarizes and helps visualize the performance of a classification algorithm. The confusion matrix consists of four primary features that are utilized to define the evaluation metrics of the classifier. These four features are.

1. True Positive ($\tau$): Number of positive labels that are correctly classified as positive.
2. False Positive ($\phi$): Number of negative labels that are wrongly classified as positive.
3. True Negative ($\upsilon$): Number of negative labels that are correctly classified as negative.
4. False Negative ($\chi$): Number of positive labels that are wrongly classified as negative.

A confusion matrix for pre-trained Xception architecture is shown in Figure 6. Here, detecting crime is the primary objective, so we consider crime as a positive label and detecting no crime as a negative label.

#### 5.2.2. Accuracy

Accuracy is an important metric to measure the classification performance, which is defined as a ratio between the successful predictions to the total number of samples under study [36,37]. This is a critical measurement since misidentifying a non-threat image as a threat may cause disruption and failing to recognize the true malicious. The mathematical representation of accuracy is stated below.

$$Accuracy = \frac{\tau + \upsilon}{\tau + \upsilon + \phi + \chi}. \tag{16}$$

Equation (16) shows the ratio of correctly detected samples to the total number of samples. Accuracy tells how good a model performs and helps in comparing different models.

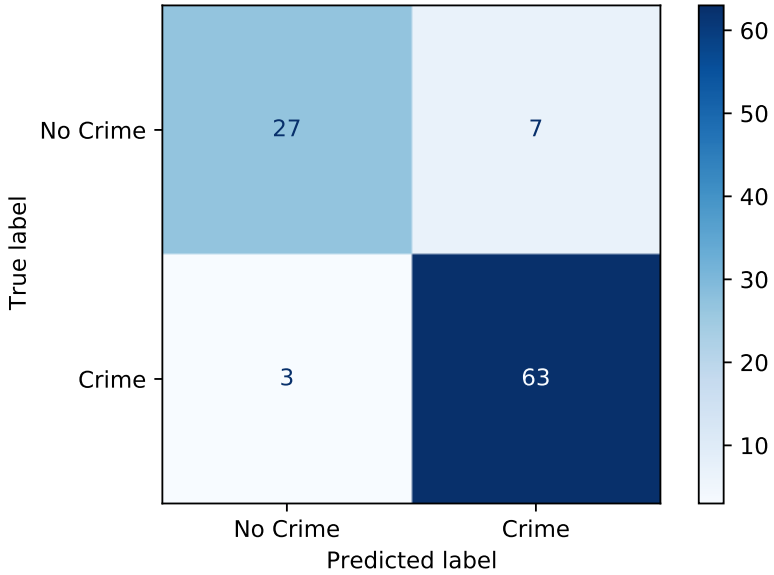
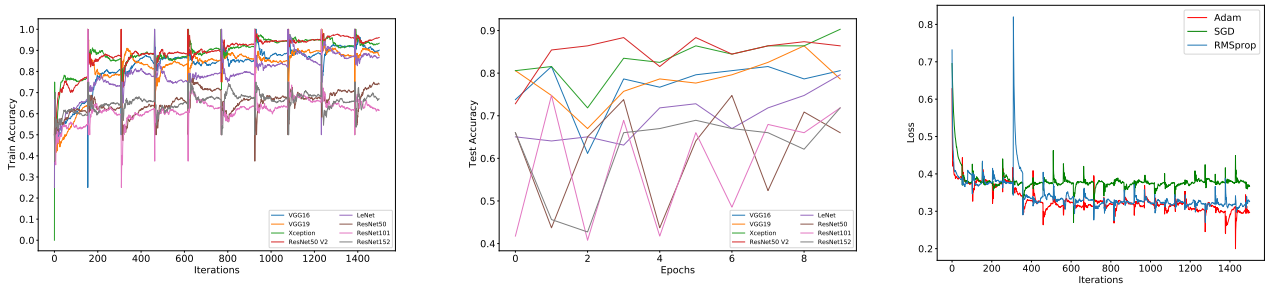


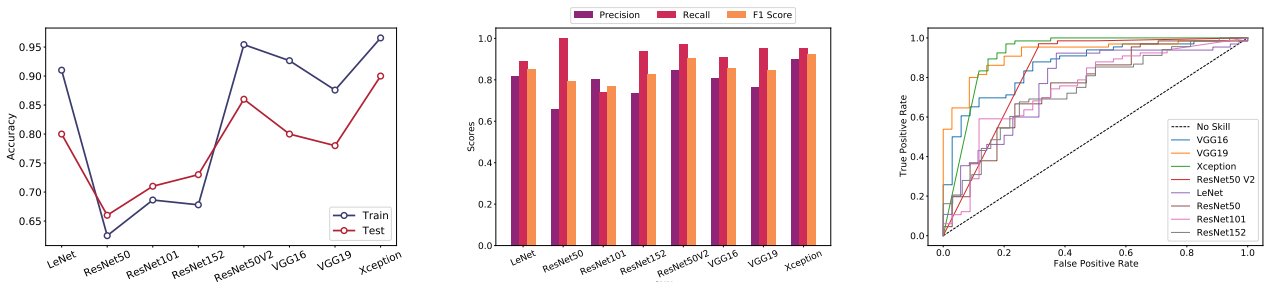

**Figure 6.** Confusion matrix for Xception on the test dataset.

Figure 7a,b shows the train and validation accuracies of different CNN architectures throughout training and validation. Figure 8a helps to visualize the train and test accuracy for various CNN architectures taken into consideration for crime detection. In addition, Table 3 also shows the train test accuracies of different CNN architectures. Needless to say, the majority of the CNN architecture achieves high train accuracy but lesser test accuracy, indicating overfitting. Contrary, the Xception architecture outperforms other architectures by delivering high train accuracy and test accuracy with an acceptable difference between them.




(**a**) (**b**) (**c**)

**Figure 7.** Performance of CNN models during training and validation. (**a**) Training accuracy during iterations. (**b**) Validation accuracy during epochs. (**c**) Comparison of optimizer loss during iterations.




(**a**) (**b**) (**c**)

**Figure 8.** Comparison of CNN architectures on different evaluation metrics. (**a**) Train and test accuracy curve. (**b**) Precision, recall, and F1 score. (**c**) ROC-AUC curves.

#### 5.2.3. Precision

Precision is the ratio of correctly detected positive samples to the total number of positive expected samples. In this case, precision would aid in determining the model's classification ability for the correctly predicted *Crime* labels out of the total predicted *Crime* labels. A better precision assures that the suggested system generates a higher number of true alerts. Equation (17) represents precision in mathematical format [36,38].

$$\rho = \frac{\tau}{\tau + \phi}, \tag{17}$$

where $\rho$ denotes precision. Here in our case, if a model has low precision, it means that the model gives more false alarms.

#### 5.2.4. Recall

A classifier's recall indicates the proportion of correctly classified positive samples to the total number of positive samples. The recall is used to determine the model's classification performance for correctly predicted *Crime* labels out of the total number of actual *Crime* labels. A higher recall reflects the classification model's dependability or how well it identifies threat images from genuine threat images. It is evaluated according to Equation (18) [39].

$$\gamma = \frac{\tau}{\tau + \chi}, \tag{18}$$

where $\gamma$ denotes recall. Here in our case, if a model has low recall it means that the model gives less true alarms.

#### 5.2.5. F1 Score

The F1 score is defined as the harmonic mean of precision and recall, as shown in Equation (19) [36].

$$F1 - Score = \frac{2 * \rho * \gamma}{\rho + \gamma}. \tag{19}$$

It is used to contrast models with varying accuracy and recall. The F1 score plays an essential role when two models are having same precision and recall scores. The primary objective of the F1 score is to provide the best model when precision and recall scores fail in evaluating the model. Moreover, it has been intended to perform effectively on unbalanced data. The F1 score has a range of values from 0 to 1, with higher values indicating better classification ability [36]. Figure 8b and Table 3 illustrate the comparison of different performance metrics, such as precision, recall, and F1 score for all CNN architecture. It is evident from the graph that the *Xception* demonstrates the highest F1 score, hence being selected as the final model.

#### 5.2.6. ROC Curve

The receiver operating characteristic-area under the curve (ROC-AUC) curve is a two-dimensional plot between the true positive rate (TPR) and false-positive rate (FPR) [40]. The higher the AUC score, the better the performance of the model. Figure 8c shows the ROC curve for CNN architectures considered under the study. AUC for all architectures has been mentioned in Table 3. The highest AUC is portrayed by VGG19 architecture—91.87%. However, due to its low test accuracy and F1 score, Xception is considered a promising CNN architecture because it has an AUC score of 91.84%, which helps in accomplishing the aim of the proposed system model *BlockCrime*.

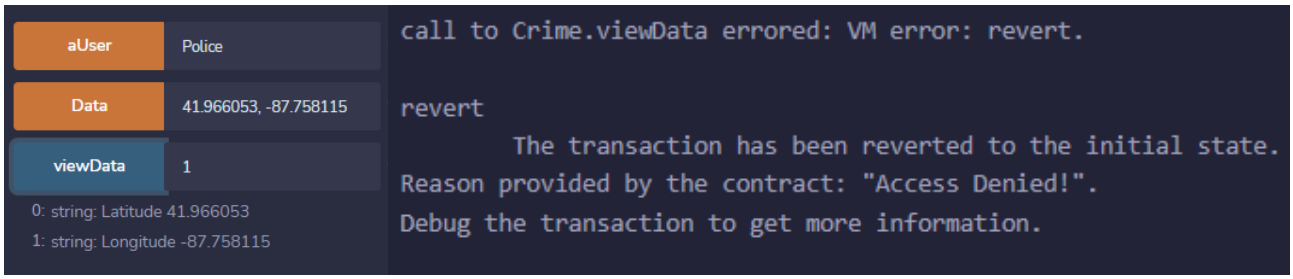
We can never compare different models based on only one evaluation parameter. Thus, here we compare various parameters to pick out the best DL network for better classification.

**Table 3.** Evaluation metrics expressed as %.

| Architecture | Train Accuracy | Test Accuracy | Precision | Recall | F1 Score | AUC |
|---|---|---|---|---|---|---|
| LeNet | 91.01 | 88.00 | 81.69 | 89.23 | 85.29 | 77.10 |
| ResNet50 | 62.48 | 66.00 | 66.00 | 100.00 | 79.52 | 74.95 |
| ResNet101 | 68.63 | 71.00 | 80.33 | 74.24 | 77.17 | 74.78 |
| ResNet152 | 67.80 | 73.00 | 73.56 | 94.12 | 82.57 | 73.94 |
| ResNet50 V2 | 95.42 | 86.00 | 84.62 | 97.06 | 90.41 | 82.95 |
| VGG16 | 92.65 | 80.00 | 81.08 | 90.91 | 85.71 | 86.00 |
| VGG19 | 87.58 | 78.00 | 76.54 | 95.38 | 84.93 | 91.87 |
| Xception | 96.57 | 90.00 | 90.00 | 95.45 | 92.64 | 91.84 |

### *5.3. Implementation of Smart Contract*

The smart contract is designed using a programming language, i.e., solidity—a contract-oriented, high-level programming language as shown in Figure 9. Figure 9a shows the *aUser, Data*, and *viewData* functions that are included in the smart contract. *aUser* function determines if a user is authorized, such as a police officer or an undesirable invader. The *Data* function is responsible for collecting position coordinates and maintaining them on the blockchain. Finally, *viewData* facilitates the retrieval of position coordinates. With the use of a navigation system, these coordinates may be utilized to go to the crime scene. Moreover, Figure 9b shows that when an attacker tries to get access to the data stored in the blockchain, it gets prompted with a revert error message. This implies that the blockchain is reverted back to its original state because an unauthorized user explicitly tries to access the secure stored data.




(**a**) (**b**)

**Figure 9.** Prototype application for finding the location of the nearest police quarters. (**a**) Data accessed by an authorized official. (**b**) Warning message for an intruder.

## 6. Conclusions

In this paper, a novel collaborative intelligence framework is proposed to identify malicious activities using DL and blockchain technology. The proposed framework employs a CNN architecture, i.e., Xception, which has been fine-tuned utilizing the transfer learning to cater the needs. With the aid of smart contracts deployed in the blockchain, the proposed framework identifies the activities that constitute a danger to public safety and provides a warning message to the law enforcement agencies. The proposed framework has been tested against the test set and produced excellent results on the limited dataset with training and testing accuracies as 96.57% and 90%, respectively, by comparing it with different CNN architectures such as LeNet, ResNet50, ResNet101, ResNet152, ResNet50 v2, VGG16, and VGG19. The current challenge of this framework includes the scarcity of the dataset images. Suppose we can access data with a good amount of volume and variety related to criminal activities; in that case, the DL network can be more accurate and reduce false alarms. Therefore, in future work, we can experiment by combining different state-of-the-art CNN

architectures (hybrid model) such as CoCa [41], Model soups [42], and CoAtNet-7 [43] on a wholesome dataset for more accurate results.

**Author Contributions:** Conceptualization: R.G., D.P., B.C.F. and S.T.; Writing—original draft preparation: D.P., R.G., H.S. and N.K.J.; Methodology: S.T., A.A., D.D.T. and R.S.; Writing—review and editing: S.T., T.A., B.C.F., D.D.T. and R.S.; Investigation: R.G., S.T. and N.K.J.; Supervision: S.T., B.C.F., R.S. and D.D.T.; Visualization; S.T., T.A., H.S. and N.K.J.; Software; N.K.J., A.A., H.S., D.P. and R.G. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work is funded by Researchers Supporting Project number (RSP-2022R503), King Saud University, Riyadh, Saudi Arabia.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** No data are associated with this research.

**Acknowledgments:** This work was supported by a grant from the Ministry of Research, Innovation and Digitization, CNCS/CCCDI – UEFISCDI, project number PN-III-P1-1.1-TE-2021-0393, within PNCDI III. In addition, this work is also supported by Researchers Supporting Project number (RSP-2022R503), King Saud University, Riyadh, Saudi Arabia.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Bu, W.; Xiao, J.; Zhou, C.; Yang, M.; Peng, C. A cascade framework for masked face detection. In Proceedings of the 2017 IEEE International Conference on Cybernetics and Intelligent Systems (CIS) and IEEE Conference on Robotics, Automation and Mechatronics (RAM), Ningbo, China, 19–21 November 2017; pp. 458–462.
2. Liang, Y.M.; Shih, S.W.; Shih, A.C.C. Human action segmentation and classification based on the Isomap algorithm. *Multimed. Tools Appl.* **2013**, *62*, 561–580.
3. Velastin, S.A.; Boghossian, B.A.; Vicencio-Silva, M.A. A motion-based image processing system for detecting potentially dangerous situations in underground railway stations. *Transp. Res. Part C Emerg. Technol.* **2006**, *14*, 96–113.
4. Nakib, M.; Khan, R.T.; Hasan, M.S.; Uddin, J. Crime scene prediction by detecting threatening objects using convolutional neural network. In Proceedings of the 2018 International Conference on Computer, Communication, Chemical, Material and Electronic Engineering (IC4ME2), Rajshahi, Kazla, Bangladesh, 8–9 February 2018; pp. 1–4.
5. Crime Rate by Country 2022. Available online: https://worldpopulationreview.com/country-rankings/crime-rate-by-country (accessed on 23 July 2022).
6. Kumar, R.; Kumar, P.; Tripathi, R.; Gupta, G.P.; Kumar, N.; Hassan, M.M. A Privacy-Preserving-Based Secure Framework Using Blockchain-Enabled Deep-Learning in Cooperative Intelligent Transport System. *IEEE Trans. Intell. Transp. Syst.* **2021**, 1–12. [CrossRef]
7. Kumar, P.; Kumar, R.; Kumar, A.; Franklin, A.A.; Jolfaei, A. Blockchain and Deep Learning Empowered Secure Data Sharing Framework for Softwarized UAVs. In Proceedings of the 2022 IEEE International Conference on Communications Workshops (ICC Workshops), Foshan, China, 11–13 August 2022; pp. 770–775. [CrossRef]
8. Hinton, G.E.; Salakhutdinov, R.R. Reducing the dimensionality of data with neural networks. *Science* **2006**, *313*, 504–507. [PubMed]
9. Ciregan, D.; Meier, U.; Schmidhuber, J. Multi-column deep neural networks for image classification. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, 16–21 June 2012; pp. 3642–3649.
10. LeCun, Y.; Jackel, L.D.; Bottou, L.; Cortes, C.; Denker, J.S.; Drucker, H.; Guyon, I.; Muller, U.A.; Sackinger, E.; Simard, P.; et al. Learning algorithms for classification: A comparison on handwritten digit recognition. *Neural Netw. Stat. Mech. Perspect.* **1995**, *261*, 2.
11. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **2012**, *25*, 1097–1105.
12. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
13. Taigman, Y.; Yang, M.; Ranzato, M.; Wolf, L. Deepface: Closing the gap to human-level performance in face verification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 1701–1708.
14. Kakkar, R.; Gupta, R.; Tanwar, S.; Rodrigues, J.J.P.C. Coalition Game and Blockchain-Based Optimal Data Pricing Scheme for Ride Sharing Beyond 5G. *IEEE Syst. J.* **2021**, 1–10. [CrossRef]

15. Verma, H.; Lotia, S.; Singh, A. Convolutional Neural Network Based Criminal Detection. In Proceedings of the 2020 IEEE Region 10 Conference (TENCON), Osaka, Japan, 16–19 November 2020; pp. 1124–1129.
16. Jan, A.; Khan, G.M. Deep Vigilante: A deep learning network for real-world crime detection. *J. Intell. Fuzzy Syst.* **2022**, *42*, 1–13.
17. Kumar, K.K.; Venkateswara Reddy, H. Crime activities prediction system in video surveillance by an optimized deep learning framework. *Concurr. Comput. Pract. Exp.* **2022**, *34*, e6852.
18. Sivakumar, P. Real Time Crime Detection Using Deep Learning Algorithm. In Proceedings of the 2021 International Conference on System, Computation, Automation and Networking (ICSCAN), Dalian, China, 22–24 October 2021; pp. 1–5.
19. Apoorva, P.; Impana, H.; Siri, S.; Varshitha, M.; Ramesh, B. Automated Criminal Identification by Face Recognition using Open Computer Vision Classifiers. In Proceedings of the 2019 3rd International Conference on Computing Methodologies and Communication (ICCMC), Erode, India, 27–29 March 2019; pp. 775–778. [CrossRef]
20. Chhoriya, P. Automated criminal identification system using face detection and recognition. *Int. Res. J. Eng. Technol. (IRJET)* **2019**, *6*, 910–914.
21. Olmos, R.; Tabik, S.; Herrera, F. Automatic handgun detection alarm in videos using deep learning. *Neurocomputing* **2018**, *275*, 66–72.
22. Elrefaei, L.A.; Alharthi, A.; Alamoudi, H.; Almutairi, S.; Al-rammah, F. Real-time face detection and tracking on mobile phones for criminal detection. In Proceedings of the 2017 2nd International Conference on Anti-Cyber Crimes (ICACC), Abha, Saudi Arabia, 26–27 March 2017; pp. 75–80.
23. Buckchash, H.; Raman, B. A robust object detector: Application to detection of visual knives. In Proceedings of the 2017 IEEE International Conference on Multimedia & Expo Workshops (ICMEW), Hong Kong, China, 10–14 July 2017; pp. 633–638.
24. Lai, J.; Maples, S. Developing a real-time gun detection classifier. In *Course: CS231n*; Stanford University: Stanford, CA, USA, 2017.
25. Dever, J.; da Vitoria Lobo, N.; Shah, M. Automatic visual recognition of armed robbery. In Proceedings of the 2002 International Conference on Pattern Recognition, Quebec City, QC, Canada, 11–15 August 2002; pp. 451–455.
26. Kumari, A.; Gupta, R.; Tanwar, S.; Kumar, N. A taxonomy of blockchain-enabled softwarization for secure UAV network. *Comput. Commun.* **2020**, *161*, 304–323. [CrossRef]
27. Haversine Formula to Find Distance between Two Points on a Sphere. Available online: https://www.geeksforgeeks.org/haversine-formula-to-find-distance-between-two-points-on-a-sphere/ (accessed on 9 June 2022).
28. Bradski, G.; Kaehler, A. *Learning OpenCV: Computer Vision with the OpenCV Library*; O'Reilly Media, Inc.: Sebastopol, CA, USA, 2008.
29. Chollet, F. Xception: Deep learning with depthwise separable convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1251–1258.
30. Fabien, M. XCeption Model and Depthwise Separable Convolutions. Available online: https://maelfabien.github.io/deeplearning/xception/# (accessed on 15 June 2022).
31. Svoboda, P.B.J.; Goldmann, T. Application for Recognition of People by Face. *Adv. Neural Inf. Process. Syst.* **2014**, *15*, 1988–1996.
32. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
33. Qian, N. On the momentum term in gradient descent learning algorithms. *Neural Netw.* **1999**, *12*, 145–151. [PubMed]
34. Ruder, S. An overview of gradient descent optimization algorithms. *arXiv* **2016**, arXiv:1609.04747.
35. Kumari, A.; Gupta, R.; Tanwar, S. Amalgamation of blockchain and IoT for smart cities underlying 6G communication: A comprehensive review. *Comput. Commun.* **2021**, *172*, 102–118. [CrossRef]
36. Sokolova, M.; Japkowicz, N.; Szpakowicz, S. Beyond accuracy, F-score and ROC: A family of discriminant measures for performance evaluation. In *Australasian Joint Conference on Artificial Intelligence*; Springer: Berlin/Heidelberg, Germany, 2006; pp. 1015–1021.
37. Tharwat, A. Classification Assessment Methods: A detailed tutorial. *Appl. Comput. Inform.* **2018**, 1–13. [CrossRef]
38. Kumar, P.; Gupta, G.P.; Tripathi, R.; Garg, S.; Hassan, M.M. DLTIF: Deep Learning-Driven Cyber Threat Intelligence Modeling and Identification Framework in IoT-Enabled Maritime Transportation Systems. *IEEE Trans. Intell. Transp. Syst.* **2021**, 1–10. [CrossRef]
39. Jadav, N.K.; Gupta, R.; Alshehri, M.D.; Mankodiya, H.; Tanwar, S.; Kumar, N. Deep Learning and Onion Routing-based Collaborative Intelligence Framework for Smart Homes underlying 6G Networks. *IEEE Trans. Netw. Serv. Manag.* **2022**, 1. [CrossRef]
40. Muhammad, Y.; Alshehri, M.D.; Alenazy, W.M.; Hoang, V.T.; Alturki, R. Identification of Pneumonia Disease Applying an Intelligent Computational Framework Based on Deep Learning and Machine Learning Techniques. *Mob. Inf. Syst.* **2021**, *2021*, 9989237:1–9989237:20. [CrossRef]
41. Yu, J.; Wang, Z.; Vasudevan, V.; Yeung, L.; Seyedhosseini, M.; Wu, Y. Coca: Contrastive captioners are image-text foundation models. *arXiv* **2022**, arXiv:2205.01917.

42. Wortsman, M.; Ilharco, G.; Gadre, S.Y.; Roelofs, R.; Gontijo-Lopes, R.; Morcos, A.S.; Namkoong, H.; Farhadi, A.; Carmon, Y.; Kornblith, S.; et al. Model soups: Averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In Proceedings of the International Conference on Machine Learning, PMLR, Paris, France, 29–30 April 2022; pp. 23965–23998.
43. Dai, Z.; Liu, H.; Le, Q.V.; Tan, M. Coatnet: Marrying convolution and attention for all data sizes. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 3965–3977.