Pertanika J. Sci. & Technol. 34 (S2): 209 - 259 (2026)

PERTANIKA JOURNALS

SCIENCE & TECHNOLOGY

Journal homepage: http://www.pertanika.upm.edu.my/

SentryNet: Fusion of CNN-LSTM Architecture for Robust Suspicious Activity Detection in Video Surveillance Using Novel Hybrid Optimisation

Roopa R* and Humera Khanam M

Department of Computer Science and Engineering, Sri Venkateswara University College of Engineering, Sri Venkateswara University, 517502 Tirupati, Andhra Pradesh, India

ABSTRACT

The exponential expansion of digital data has increased the need for automated systems to spot suspicious activity in surveillance and cybersecurity, for example. Despite numerous studies, reducing computational complexity, lowering biases, and improving model generalisation remain complex tasks. For automated detection of suspicious activity in surveillance videos, this study presents SentryNet. This unique framework combines Long Short-Term Memory (LSTM) networks and Convolutional Neural Networks (CNNs). The exponential expansion of digital data has increased the need for automated systems to spot suspicious activity in surveillance and cybersecurity. Reducing computational complexity, lowering biases, and improving model generalisation remain difficult tasks despite numerous studies. To detect suspicious activity in surveillance video data in real-time, this study presents SentryNet. This unique framework combines LSTM networks and CNNs. Furthermore, to overcome the drawbacks of single optimisers and enhance convergence rates and model stability, we suggest a hybrid optimisation strategy that combines the best features of Adam and Stochastic Gradient Descent (SGD). The suggested CNN+LSTM architecture outperformed other well-known models in the literature, achieving remarkable accuracy rates of 98.85% on the HBD21 dataset and 99.57% on the UCF-Crime dataset. Moreover, our hybrid optimisation approach improved overall performance while lowering computational complexity.

Keywords: Deep learning, HBD21, hybrid model, hybrid optimiser, SentryNet, suspicious activity, UCF-Crime, video surveillance

ARTICLE INFO

Article history:
Received: 03 August 2026
Accepted: 06 October 2026
Published: 20 November 2026

DOI: https://doi.org/10.47836/pjst.34.S3.10

E-mail addresses:
roopa509@gmail.com (Roopa R)
humera.svec@gmail.com (Humera Khanam M)
* Corresponding author

INTRODUCTION

Identifying suspicious or criminal activity using surveillance footage has become an essential element of public safety in an increasingly digital and networked

e-ISSN: 2231-8526

© Universiti Putra Malaysia Press

environment (Ahmed & Yousaf, 2024). Urbanisation and the spread of surveillance technology have spurred an ever-growing demand for smart systems capable of instantly identifying suspicious criminal activity. Manual monitoring has proven unworkable given the sheer volume of data generated by public and private video surveillance cameras (Şengönül et al., 2023). Conventional manual monitoring methods, dependent on human operators, are inefficient and error-prone, especially in high-density settings where suspicious behaviours may be subtle or concealed (Wu et al., 2026). While trying to identify unusual activity across several video streams, security operators can struggle with cognitive overload, fatigue, and distractions, which can lead to delayed reactions and missed incidents (Stewart, 2022). These difficulties have created a pressing need for automated suspicious crime detection systems that can operate effectively in various challenging situations, including low illumination, occlusion, and high crowd density. Conventional machine learning (ML) methods, including Support Vector Machines (SVMs), K-Nearest Neighbours (KNN), and Gaussian Mixture Models (GMMs), have been employed for anomaly identification in surveillance footage. These systems generally depend on manually designed criteria, such as item trajectories or motion patterns, to categorise actions as normal or abnormal. Though these methods provided early success in controlled contexts, they have significant limitations in more complex, real-world situations.

A main drawback of traditional ML methods is their dependence on feature extraction. Usually used to train the model, these techniques sometimes require manual identification of important elements, such as pixel intensities or motion vectors (Turaga et al., 2008). This procedure may fail to detect intricate patterns in video data and could be labour-intensive. These models sometimes struggle to scale properly, as their performance suffers on larger datasets. Moreover, conventional ML models are constrained in their capacity to process temporal information (Taye, 2023). Anomalies in video data generally manifest over time, and in the absence of a model for these temporal dependencies, such methods frequently yield elevated false-positive rates. Deep learning (DL) models such as CNNs and Recurrent Neural Networks (RNNs) have been well-suited to address these constraints due to their capacity to automatically extract hierarchical features from video data and simulate long-term temporal correlations (Vosta & Yow, 2022).

Identifying suspicious objects in surveillance footage is essential for improving public safety and decreasing crime rates (Mahalakshmi & Thenmalar, 2023). Spotting abnormalities in surveillance videos is still tricky for numerous reasons related to real-world environments. Occlusion from elevated crowd density hinders anomaly detection by obscuring suspicious activities, complicating models' ability to effectively differentiate between normal and abnormal behaviours. Moreover, inadequate lighting in video frames, particularly during low-light or nocturnal surveillance, diminishes the quality of spatial characteristics, adversely affecting the model's efficacy. Complicating these problems,

anomalies in everyday things might cause misclassification—that is, slight differences between the two can lead normal behaviours or items to be wrongly labelled as suspicious (Ali, 2023). Moreover, given the complexity of real-world surveillance data, current methods comprising DL models such as CNNs, LSTMs, or hybrid approaches struggle to achieve high accuracy, efficiency, and robustness. For example, Kumar and Biswas reported a low accuracy of 49.04% when evaluating a hybrid DL model on the UCF-Crime dataset (Kumar & Biswas, 2024). Dependence on a single optimiser, such as SGD or Adam, which can, especially in complex video datasets, lead to overfitting or underfitting, is one of the causes of this. Patwal et al. (2023) achieved an accuracy of 86.63% using a CNN based on the DenseNet121 architecture, optimised with the SGD optimiser. Likewise, Kirichenko et al. (2022) used the Adam Optimizer with their suggested neural network architecture. However, their 93% accuracy was insufficient to support the system's use in detecting criminal activity. The model's performance can be sensitive to variations in the optimiser's parameters; hence, utilising only one type can miss the full data variance, thereby lowering accuracy in practical settings. Widely used to assess these algorithms, the UCF Crime dataset is a significant benchmark for anomaly identification in videos, as it comprises diverse criminal behaviours. Current methods often suffer from limited generalisation across diverse surveillance environments and insufficient detection performance in low-resolution or occluded settings, even with advanced models such as attention-based LSTM networks and residual learning. Our work aims to address these challenges using a novel hybrid fusion model that combines CNNs with transfer learning and RNN approaches, tuned for detecting suspicious objects on two different datasets. The suggested model will combine the spatial learning proficiencies of CNNs with the temporal sequence analysis advantages of other neural networks to efficiently capture spatial characteristics and temporal dynamics. Additionally, by examining hybrid optimisation techniques combined with RMSprop, SGD, or Adam, we want to determine the most effective method for improving model performance regarding accuracy, precision, recall, and F1-score. The research objectives and contributions are described later.

Research Objective

The key objectives of this research are delineated as follows:

- Handling complex surveillance environments: Our research aims to solve detection difficulties in poor lighting and obstructed and congested areas in suspicious crime detection, therefore affecting the accuracy of current surveillance systems.
- Model generalisation: To ensure good performance on unseen data, our research attempts to create a well-fitted model that minimises both overfitting and underfitting. To build a more resilient model that can adapt to varied circumstances, we focus on

generalisation across a range of monitoring scenarios. In addition, the model will be built to perform well across different datasets, increasing its adaptability and ensuring it produces accurate results beyond the training data's specific constraints.

- Boosting classification accuracy while reducing complexity: Mitigating computing complexity while improving model correctness is one of the research's other main goals. DL models can be resource-intensive and risk having their performance affected by class bias. To reduce bias and improve class balance, our study will focus on enhancing the model's overall performance across all evaluation criteria. To make the model architecture more computationally efficient and ensure accuracy is not compromised for practical, scalable real-world deployment, we will also create a simplified model with fewer parameters.

Contribution

Following the above-described research goals, our major contributions to the field have been listed here:

- The research suggested a hybrid CNN+RNN model for suspicious activity detection to address demanding situations, including low-light, occlusion, and high-density crowd scenarios. In challenging surveillance settings, this raises detection accuracy. The study demonstrates how effectively CNNs and transfer learning identify suspicious activity in video footage. We expedite training and increase feature extraction using pre-trained models, improving detection performance.
- In this research, we propose a novel SentryNet model, which combines temporal and spatial features through a hybrid architecture. By leveraging CNNs for spatial feature extraction and RNNs (LSTM/GRU) for capturing temporal dependencies, SentryNet effectively fuses these features to provide a comprehensive analysis. This fusion of spatial and temporal information enhances the model's ability to detect suspicious activities with greater accuracy and robustness.
- In this study, we present a novel hybrid optimisation method that improves training by fusing the advantages of Adam and SGD. Initially, the optimiser uses Adam's flexible learning-rate schedule to handle noisy gradients and accelerate convergence. As the model develops, it transitions to momentum-assisted SGD, thereby stabilising learning and improving generalisation. This hybrid strategy produces a more resilient and optimal model by balancing steady fine-tuning and quick early training.
- The hybrid optimiser improves the proposed model (CNN+LSTM), which exhibits well-generalised and robust performance and successfully mitigates overfitting and underfitting problems. The model demonstrated remarkable robustness when tested on the UCF Crime dataset, achieving accuracy, precision, recall, and F-score of 99.57%.

In a similar vein, the model continued to perform well on the HBD21 dataset, with accuracy, F-score, precision, and recall all hitting 98.85%. These outcomes confirm the model's efficacy and versatility across many datasets.

The paper is organised into several parts. The overall workflow of the research is shown in Figure 1.

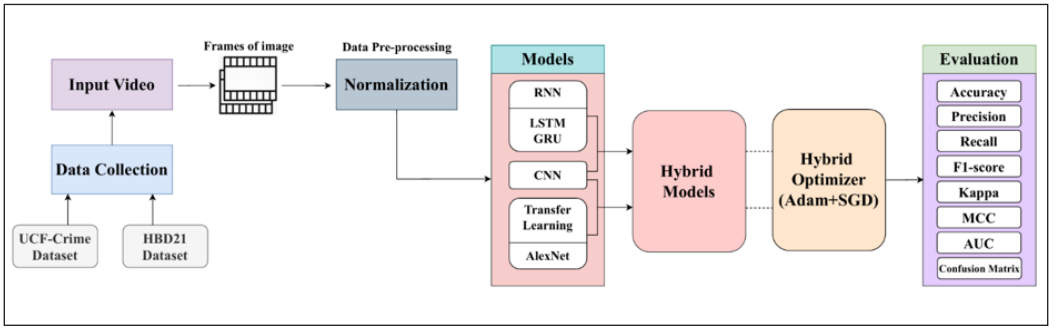


Figure 1. Workflow of the SentryNet framework

LITERATURE REVIEW

Recent improvements in video surveillance utilise DL methodologies to improve detection and identification skills. CNNs and their variants are widely used for spatial feature extraction, while hybrid models combine CNNs with recurrent networks or attention mechanisms to capture temporal dynamics. Attention mechanisms and transformer-based models prioritise essential video frames to enhance precision. Multi-stream models concurrently analyse many data types to capture geographical and temporal characteristics. Privacy-preserving methodologies employ federated learning to safeguard sensitive information while facilitating collaborative model training. Moreover, specific architectures for crime prediction and weakly supervised learning methodologies tackle anomaly detection and action recognition issues with few annotations. Every strategy enhances the precision and scalability of video analysis systems. The reviewed articles present articulated methodologies for effectively tackling these challenges. Methods for identifying anomalies and human interactions need a comprehensive study of departures from standard patterns, with research suggesting approaches for successfully collecting patterns and measuring deviations.

Hybrid CNN Architecture

Qasim and Verdú (2023) combine ResNet CNNs with Simple Recurrent Units (SRUs) for spatial and temporal coherence in anomaly detection. The hybrid CNN-LSTM model, in

which the CNN handles spatial features and the LSTM manages temporal dependencies for detecting human behavioural anomalies, is described by Pallewar et al. (2024). The approach integrates a 3D CNN with a ConvLSTM to extract spatial and temporal features and is particularly well-suited for dynamic surveillance scenarios, as described by Mahareek et al. (2024). Additionally, it employs YOLOv5, a CNN-based framework for object recognition, and LSTM for time-series analysis to detect suspicious activities in retail environments, following the method outlined by Nazir et al. (2023). Furthermore, the system leverages a deep neural network (DNN), a variant of CNN, alongside ResNet to detect criminal behaviour in video footage, according to the approach presented by Mehsen (2023). Two systems have been created to improve real-time surveillance and criminal monitoring by integrating several models for comprehensive analysis. Mukto et al. (2024) propose a system that integrates MobileNetV2 for behaviour detection, YOLOv5 for weapon detection, and LBPH for facial recognition, offering a comprehensive solution for real-time surveillance and criminal monitoring. Likewise, a crime prediction system developed by Cesario et al. (2024) employs YOLOv5 for weapon detection, MobileNetV2 for behavioural analysis, and LBPH for facial recognition, with a special focus on densely populated metropolitan regions. Both systems utilise a multi-model strategy to efficiently tackle many facets of surveillance and crime prevention.

CNNs with RNN or Transfer Learning Models

Qasim Gandapur and Verdú (2023) merge a ResNet-50 CNN for spatial feature extraction with a ConvGRU to model temporal dynamics, primarily developed for real-time surveillance applications. Another approach employs a two-stream architecture, where one stream collects RGB-based spatial information. At the same time, the other analyses optical flow for temporal motion, with Bi-LSTM fusing the streams for action identification (Kurchaniya & Kumar, 2023). An alternative model combines ConvLSTM and GRUs to capture spatiotemporal characteristics, focusing on the anomalous human behaviour detection study by Ghosh et al. (2022). A model blends ResNet50 with ConvLSTM to capture spatiotemporal sequences and utilises multi-head attention to optimise feature extraction for violence detection, as described by Vosta and Yow (2024). These hybrid models balance spatial and temporal analysis for diverse anomaly detection applications. The method presented by Al-Lahham et al. (2024) is a federated learning architecture enabling numerous users to cooperatively train localised models while preserving the confidentiality of raw data. This method employs pseudo-label refinement and federated averaging for unsupervised anomaly detection, ensuring data privacy and enhancing detection accuracy. A distinct framework integrates multiple CNN architectures, including MobileNetV2, InceptionV3, InceptionResNetV2, and Xception, to provide a scalable and adaptable solution for anomaly detection across diverse datasets. This paradigm integrates

privacy-preserving methodologies to guarantee data security by Jebur et al. (2025). Both techniques signify substantial progress in anomaly detection by including privacy and scalability into their frameworks.

Attention Mechanisms and Transformer-Based Models

The model proposed by Kumar et al. (2023) combines a CNN with a Residual Attention Bi-LSTM, utilising attention processes to emphasise the most significant segments of the video, hence enhancing the accuracy of anomalous behaviour detection. Another technique employs Vision Transformers (ViTs) to record spatiotemporal correlations, whereas masked autoencoders improve group activity detection in movies, as approached by Yadav et al. (2024). A lightweight CNN combined with LSTM employs Bahdanau attention mechanisms to enhance anomaly classification by focusing on the most pertinent aspects of video sequences, as proposed by Ul Amin et al. (2023). These models illustrate how attention processes may emphasise pertinent characteristics for anomaly detection and action identification. Inception-v3, a CNN variant, is utilised with transfer learning to improve anomaly detection in surveillance videos, as observed by Jebur et al. (2023). The system employs Inflated 3D CNNs (I3D) to capture spatiotemporal characteristics, augmented with attention mechanisms to improve action recognition, following Gowada et al. (2023).

Multi-Stream Models for Spatio-Temporal Analysis

Several advanced models have been developed to enhance video anomaly detection through innovative multi-stream approaches. One model, AnomalyNet, introduces dual parallel networks: one for optical flow and the other for RGB frames, aiming to detect anomalies in surveillance videos (Mumtaz et al., 2024). Additionally, Jan and Khan (2023) proposed a dual-stream approach in which a 3D CNN processes motion and object data, while a 2D CNN captures spatial features, with Softmax used for classification. Lastly, a multi-stream model leverages BERT for action recognition, integrating RGB and optical flow streams to perform spatiotemporal classification for video anomaly detection, as reported by Tan et al. (2024). These models balance spatial and temporal information to enable accurate anomaly detection. Two advanced models have been developed to enhance anomaly detection in video analysis via multi-instance learning (MIL) and weak supervision techniques. An approach uses Multiple Instance Learning with pseudo-labelling to identify irregularities at the frame level without necessitating significant annotations. This model utilises 3D convolution and I3D feature extraction for effective anomaly identification, as demonstrated by Nayak and Chaudhari (2023). An alternative method integrates BERT with MIL ranking and binary cross-entropy to detect anomalies in video segments. It enhances feature extraction by combining MIL with ranking loss, hence improving the

model's ability to detect anomalies with less supervision, as noted by Tan et al. (2024). Both strategies are utilised for anomaly detection, employing MIL and feature extraction techniques to operate on reduced annotated data.

Limitations and Scopes

The analysed articles disclose several significant flaws across various methodologies. The I3D-based model faces accuracy challenges and legal and ethical dilemmas in its use. The CNN + SRU model has difficulties with anomalies without observable individuals and may produce false positives. The 3DCNN + ConvLSTM model is prone to overfitting and requires significant computing resources, whereas DenseNet-121 has challenges with false positives due to shadows and scalability limitations. Attention-based models exhibit suboptimal performance in swiftly evolving or distorted environments, whereas the ConvGRU-CNN model fails to detect transient abnormalities and lacks auditory processing capabilities. The CNN-LSTM model requires prolonged training and addresses only a limited range of abnormalities.

Two-stream networks are highly dependent on optical flow, which limits their effectiveness in complex situations. Transfer learning with Inception-v3 is constrained by pre-trained features, limiting flexibility [34], whereas criminal detection models face scaling challenges due to a limited number of crime categories. AnomalyNet encounters difficulties with temporal context in both rapid and slow actions and is computationally intensive. The Masked Autoencoder approach fails to identify essential items such as weapons, leading to ambiguity in analogous tasks, whereas the hybrid detection model is sensitive to video quality, thereby constraining its real-time applicability.

Conv-LSTM-GRU and LRCN models require substantial computational resources and struggle with intricate movements. The attention-based detection approach, while precise, is computationally intensive and reliant on extensive datasets. The real-time criminal monitoring system has challenges with concealed weapons, and lightweight structures diminish accuracy. Likewise, the multi-density crime predictor has difficulty with obstructed objects and incurs substantial computational requirements. The lightweight design of MobileNetV2 limits its ability to handle intricate scenarios, whereas richer models increase computational complexity. Collaborative learning algorithms face challenges in scaling to distributed data and are adversely affected by noisy pseudo-labels, whereas BERT-based models exhibit variable performance across datasets. The unethical action recognition model suffers from misclassification and training delays due to its complexity, whereas KianNet's dual-attention mechanisms increase computational costs and do not incorporate auditory input processing.

A hybrid fusion method that integrates CNN and RNN effectively mitigates the shortcomings of existing anomaly detection models. This approach leverages the CNN's

capacity to extract spatial information from video frames, while the RNN addresses temporal dependencies, providing a more thorough examination of both spatial and temporal dynamics. The CNN-RNN hybrid fusion method improves detection accuracy by more effectively identifying intricate behaviours in surveillance footage, particularly in challenging conditions such as poor lighting, occlusion, or high crowd density. Furthermore, implementing a hybrid training approach that integrates several optimisation strategies enables the model to optimise performance, mitigate overfitting, and enhance generalisation across diverse datasets. This connection enhances the model's proficiency in managing real-world surveillance data and bolsters its capability to identify nuanced or ambiguous behaviours. This method utilises the advantages of both CNN and RNN, together with enhanced training procedures, to provide more efficient and precise anomaly detection in real-time surveillance systems.

METHODOLOGY

The explosion of digital data in recent years has increased the demand for automated systems that can identify suspicious activity in surveillance and cybersecurity, among other domains. CNNs are one type of DL model that has successfully extracted spatial features from photos. Nevertheless, temporal dynamics—a critical component in identifying questionable activity in video data—are not captured by CNNs. LSTM and Gated Recurrent Unit (GRU) are two examples of RNNs that excel at learning temporal relationships but struggle with spatial dependencies. To overcome these drawbacks, hybrid models integrating RNNs and CNNs have become increasingly popular. Such architectures enable a thorough study of spatial and temporal cues by using CNNs to extract spatial characteristics from images, which are then processed by LSTMs or GRUs to capture temporal patterns. Transfer learning has been helpful in further improving these hybrid models, particularly when using pre-trained CNNs such as AlexNet. Using pre-trained weights on massive datasets such as ImageNet, AlexNet can enhance spatial feature extraction. The hybrid technique builds a robust framework for identifying intricate patterns of suspicious behaviour by combining CNNs for spatial learning, RNNs for temporal learning, and AlexNet for transfer learning. The proposed hybrid model in this work integrates three key components: CNNs, RNNs (LSTM/-GRU), and transfer learning (AlexNet). First, a CNN is employed to extract rich spatial features from each input data frame. Then these features are fed into an RNN (LSTM/GRU) to capture temporal dependencies and model sequential patterns across multiple frames. Incorporating AlexNet as a transfer learning component further boosts the model's generalisation ability by efficiently leveraging pre-learned weights to extract hierarchical spatial features. This architecture enables the hybrid model to detect suspicious activities more accurately by leveraging spatial and temporal cues. This research makes a significant contribution by combining multiple hybrid designs to produce a more reliable

and accurate detection system. Each part of the suggested model—CNN, LSTM/GRU, and AlexNet—contributes differently to learning and feature extraction. The outputs of the various architectures are combined during fusion to produce a single representation that captures sequential dependencies and both high- and low-level features. This hybrid model architecture uses a fusion strategy that combines the temporal features learned by LSTM/GRU with the spatial features retrieved from CNN and AlexNet. Concatenation or weighted average of feature maps and activations from various model branches is used to accomplish this. We hope to improve the system's overall discriminative capabilities by combining the outputs of these models, enabling the hybrid model to more effectively and broadly distinguish between regular and suspicious activity. The fused output is subjected to fully connected and classification layers to produce the final forecast. Through end-to-end fusion, the model's accuracy increases and its resilience is ensured across a variety of datasets, enhancing the hybrid system's adaptability to real-world applications where the detection of suspicious activity is crucial. The algorithm of the proposed hybrid model is shown in Algorithm 1.

Algorithm 1 Hybrid Model with CNN and RNN Fusion

Input: A sequence of frames or images $X = \{x_1, x_2, \dots, x_T\}$
Output: A classification label indicating different types of suspicious activity
Initialisation:
Set hyperparameters: learning rate α, number of epochs E, batch size B, optimiser Opt
Initialise CNN weights W_{CNN}, RNN weights W_{RNN}, fusion layer weights W_{fusion}
Spatial Feature Extraction (CNN):
for each frame $x_t \in X$ for $t = 1,2, \dots, T$ do
    Pass each frame through the CNN using the specified pretrained initialisation.
    Extract spatial features: $f_T^{CNN} = CNN(x_t)$
end for
Collect spatial features: $\{f_1^{CNN}, f_2^{CNN}, \dots, f_T^{CNN}\}$
Temporal Feature Learning (RNN - LSTM/GRU):
Initialise h_o and c_o for LSTM (or h_o for GRU)
for each spatial feature $f_t^{CNN} \in F_{CNN}$ do
    Pass the spatial feature vector into the RNN
    Capture temporal dependencies:
    $h_t^{RNN}, c_t = LSTM(f_t^{CNN}, h_{t-1}, c_{t-1})$ (or)
    $h_t^{GRU} = GRU(f_t^{CNN}, h_{t-1})$
end for
Store hidden states: $H_{RNN} = \{h_1^{RNN}, h_2^{RNN}, \dots, h_T^{RNN}\}$
Fusion of Spatial and Temporal Features:
Apply feature fusion:
F_{fusion} = Fusion (F_{CNN}, H_{RNN}) (or)
Perform fusion via concatenation or addition:
$F_{fusion} = [F_{CNN} \oplus H_{RNN}]$

Classification:
Pass F_{fusion} through a fully connected layer:
$z = W_{fusion} F_{fusion} + b_{fusion}$
Apply softmax or sigmoid to obtain output:
$\hat{y} = Softmax(z)$ (for multi-class) or
Loss Calculation and Backpropagation:
Compute loss $L = Loss(\hat{y}, y)$
Backpropagate the loss through the network, updating weights W_{CNN}, W_{RNN}, W_{fusion}
Training Loop:
for each epoch e = 1, 2, . . ., E do
    Split data into batches of size B
    for each batch do
        Perform forward pass and backpropagation
        Update model parameters
    end for
end for=0

The following describes the branches of the hybrid model that combine CNN and RNN (LSTM, GRU) or CNN with Transfer learning. The outputs from the two branches—RNN and CNN—are fused together in the suggested hybrid CNN-RNN model by concatenating their feature vectors. While the RNN branch captures temporal dependencies, the CNN branch analyses spatial information in video frames. The outputs from both branches, designated as F_{CNN} for the CNN and F_{RNN} for the RNN, are combined into a single feature vector Ffinal once they have finished their individual jobs. Mathematically, this combination of temporal and spatial properties is expressed as shown in Equation 1:

$$F_{final} = concat(F_{CNN}, F_{RNN}) \quad [1]$$

The final feature vector obtained from the combined features is passed through a fully connected (dense) layer with softmax activation. This layer ensures that the model produces a valid probability distribution over the potential classes and outputs probabilities for each class, such as normal or suspicious activity. The last layer's softmax function is defined as follows in Equation 2:

$$P(y_i) = \frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_j}} \quad [2]$$

where z_i represents the output score for each class i, and $P(y_i)$ gives the probability of class i. The final output is a set of probabilities, with the class having the highest probability being the predicted activity. The block diagram of the proposed hybrid model schema is shown in Figure 2.

CNN Model

CNN functions as a standard feedforward neural network, developing many filters to extract features from the input data (Liu et al., 2021). The design of our suggested CNN model consists of 16 layers, starting with an input layer that handles grayscale images with dimensions $50 \times 50 \times 1$. This layer receives the image's unprocessed pixel values, with each pixel's intensity represented as a single channel. The input is subsequently transmitted to the initial convolutional layer for preliminary feature extraction. This layer is essential for preparing picture data for further analysis by normalising and organising it into a format suitable for processing by succeeding levels.

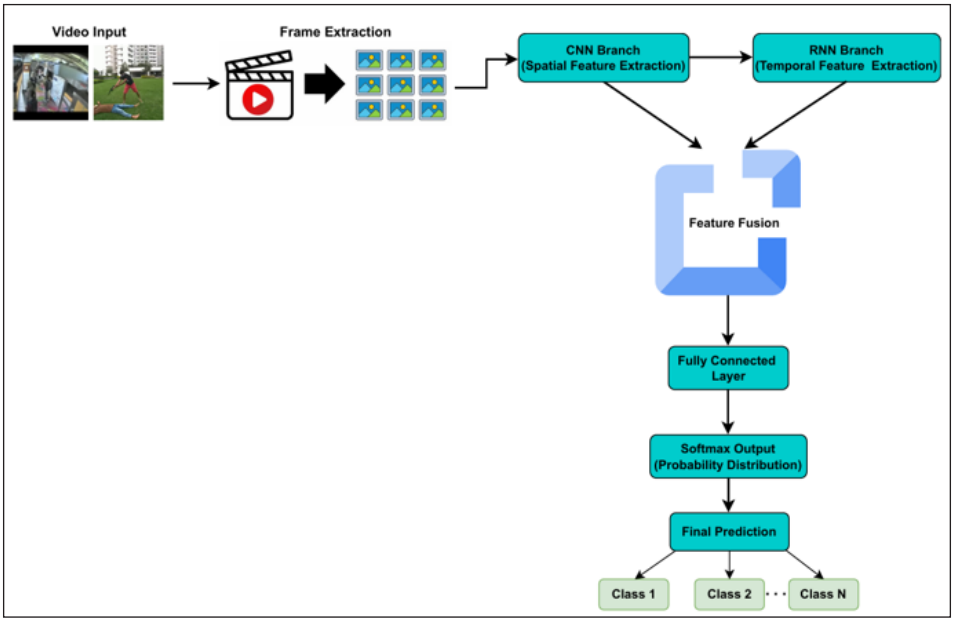


Figure 2. Proposed SentryNet block diagram

This network involves several 2D convolutional layers, each followed by max-pooling and dropout for regularisation. Following convolution and pooling, the CNN architecture incorporates a flatten layer that transforms the multidimensional feature maps into a one-dimensional vector, followed by a fully connected layer with 256 neurons and a dropout rate of 0.5. This model captures the fundamental features step by step using convolution and pooling. These abstractions, which provide translation, scaling, and rotation invariance, help the network deepen. Mathematically, the 2D convolution operation of this network is expressed as Equation 3:

$$x_j^l = f\left(\sum_{i \in M_j} x_i^{l-1} \times k_{ij}^l + B_j^l\right) \quad [3]$$

where x_j^l represents the *j*-th feature map of the l-th layer, k_{ij}^l is the kernel applied for convolution, and B_j^l is the bias term. The model guarantees scale invariance while keeping the most important features by using max-pooling to lower dimensionality. The max-pooling operation can be formally expressed as Equation 4:

$$p_j^l = \max\left(q_j^{l-1}(t)\right), \qquad t \in [(j-1)w, jw] \tag{4}$$

where p_j^l is the output after pooling the feature map, and w is the width of the pooling window. Following multiple rounds of convolution and max-pooling, convolution layer feature maps are flattened into a one-dimensional vector and subsequently fed into the fully connected, or dense layers. If the final feature map has dimensions $m \times n \times d$, flattening converts it to a vector of size ($m \times n \times d \rightarrow mnd$). The operation of the dense layer can be articulated as follows Equation 5:

$$z_j^l = \sum_i w_{ij}^j . x_i^l . b_j^l \tag{5}$$

where w_{ij}^j is the weight connecting neuron *i* to neuron *j* in the dense layer, x_i^l is the input from the previous layer, b_j^l is the bias term for the neuron *j*, and z_j^l is the output of the dense layer.

The dropout layer is employed after the fully connected layers to mitigate overfitting by randomly deactivating a proportion of neurons during training. The dropout operation can be characterised as Equation 6:

$$y_i^l = f(x) = \begin{cases} x_i^l, & with\ probability\ p \\ 0, & with\ probability\ 1-p \end{cases} \tag{6}$$

where p is the dropout rate (0.5 in our model), and x_i^l is the output of the neuron *i* in the dense layer.

LSTM Model

CNN-based methods are limited to tackling fixed, short-sequence classification tasks, so they are unsuitable for complex, long-term time-series applications such as crime activity detection, despite their strong resilience and efficiency (Ullah et al., 2022). Analysing trends over time to detect odd or suspicious behaviour in crime activity calls for a model adept at handling sequential data with varying temporal dependencies. Under such circumstances, RNNs fit quite well. Designed specifically to process sequential data, RNNs can efficiently identify temporal correlations and hidden patterns in time-series data. This makes them perfect for jobs like crime activity detection, as identifying trends suggestive of criminal

activity depends on studying occurrences over time. RNNs will enable us to more accurately investigate and detect illegal conduct, hence improving predictive ability over more traditional CNN techniques. Usually, the RNN network predicts the future sequence by concatenating current information from both spatial and temporal dimensions with past information, thereby analysing the underlying sequential pattern (Khan et al., 2021). LSTM's internal design consists of multiple gates—input, forget, and output gates—where each gate processes the input from the previous gate and passes it to the next gate, thereby regulating the flow of information towards the ultimate output (Ullah et al., 2021)

Usually, a sigmoid or *tanh* activation function controls all gates; for example, the input gate is in charge of updating the information. When necessary, the forget gate removes the information from the current state C_t and processes the input information from the input gate i_t and the state of the previous cell C_{t-1}. Whereas the output gate o_t stores the output value for the next sequence prediction and transfers the last output to the next LSTM unit. Conversely, the recurrent unit C_t estimates the current input value x_t using the tanh activation function and the state of the previous cell C_{t-1}. Whereas the scalar product of o_t and $tanh$ of C_t. allows one to determine the value of h_t. At last, feeding h_t to the softmax classifier will provide the ultimate result. Mathematically, the above-listed gates' operations can be stated as follows from Equations 7 to 13:

$$f_t = \Phi\left(\widehat{W}_f \cdot [h_{t-1}, x_t]\right) + B_f \quad [7]$$

$$i_t = \Phi\left(\widehat{W}_i \cdot [h_{t-1}, x_t]\right) + B_i \quad [8]$$

$$C_t = \tanh\left(\left(\widehat{W}_C \cdot [h_{t-1}, x_t]\right) + B_C \right. \quad [9]$$

$$C_t = f_t \cdot C_{t-1} + i_t \cdot C_t \quad [10]$$

$$o_t = \Phi\left(\widehat{W}_o \cdot [h_{t-1}, x_t]\right) + B_o \quad [11]$$

$$h_t = o_t \cdot \tanh\left(\Phi(C_t)\right) \quad [12]$$

$$Output = softmax(h_t) \quad [13]$$

GRU Model

A particular kind of RNN, called the GRU, was created to overcome the difficulties of discovering long-term dependencies in sequential data. The GRU, proposed by Cho et al. (2014), improves on the capabilities of conventional RNNs, which often encounter problems such as the vanishing gradient problem, particularly when handling long sequences. The GRU is especially useful for tasks because it addresses this issue by introducing gating

mechanisms that control the flow of input. The update gate (z_t) and the reset gate (r_t) are the two main gates of the GRU. The update gate controls the amount of historical data that is required for the present state by deciding how much of the previous hidden state (h_{t-1}) should be retained. The model can ignore unnecessary data from earlier time steps because the reset gate controls how much of the previous hidden state to discard. GRUs merge cell and hidden states into a single hidden state, whereas LSTM networks maintain separate cell and hidden states. This results in fewer parameters and quicker training and inference times. The GRU operations can be mathematically defined through a series of equations; The reset gate is calculated as shown in Equation 14:

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r) \quad [14]$$

where σ is the sigmoid activation function, W_r and U_r are weight matrices for the input x_t and the previous hidden

state h_{t-1}, and b_r is the bias term for the reset gate. The update gate is computed as shown in Equation 15:

$$z_t = \sigma(W_z x_z + U_z h_{t-1} + b_z) \quad [15]$$

where z_t is the update gate, W_z and U_z are weight matrices, and b_z is the bias term. The new memory content, representing the candidate hidden state, is given as shown in Equation 16:

$$\widetilde{h_t} = \tanh(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h) \quad [16]$$

where $\widetilde{h_t}$ is the candidate hidden state, W_h and U_h are the weight matrices, b_h is the bias term, and $\odot$ denotes element-wise multiplication. Finally, the hidden state at time t is calculated as shown in Equation 17:

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \widetilde{h_t} \quad [17]$$

where h_t is the hidden state. Overall, the gating methods and the GRU's integrated hidden state enable it to process sequential input effectively while addressing issues with conventional RNNs. Language modelling, sentiment analysis, and time-series forecasting are just a few of the applications that choose it for its ability to handle long-range dependencies with lower computational complexity. The GRU improves neural network performance on tasks involving sequential patterns by efficiently balancing information retention and amnesia. One of the main differences between LSTM and GRU networks is that LSTM networks rely on multiple hidden states, whereas GRUs have only one.

Faster computations and lower memory usage result from this integration's architectural simplification and reduced parameter count. Combining the forget and input gate functions into a single update gate substantially improves the computational efficiency of the GRU by eliminating the need for separate cell states. Consequently, the GRU maintains strong performance in a wide range of applications while achieving faster training and inference durations. The basic structure of GRU and LSTM is shown in Figure 3.

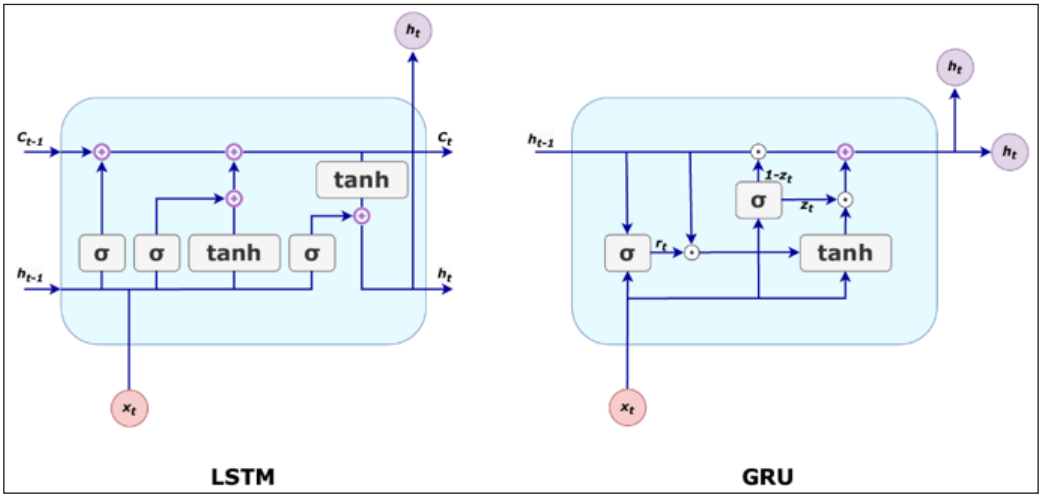


Figure 3. Basic architecture of LSTM and GRU

Proposed CNN+LSTM Hybrid Model

For efficient feature extraction and sequence modelling, we provide a hybrid method that combines the capabilities of LSTM networks and CNNs. Starting with two 2D convolutional layers, the first utilises 64 filters and the second uses 128 filters, both with a kernel size of 3x3, thereby effectively managing negative activations. LeakyReLU activation function is used in both layers. Each convolutional layer is followed by a dropout layer to reduce overfitting and a 2x2 Max Pooling layer. Subsequently, the extracted features are fed into the LSTM network, which consists of two layers, each with eight LSTM units, and aims to capture temporal relationships in the data. Based on these properties, a final Dense layer with a softmax activation function forecasts class labels. Originally using the Adam optimiser with a learning rate of 0.001, the model first utilises a hybrid of Adam and SGD, then switches to SGD with a learning rate of 0.001 and momentum of 0.9 for better convergence and optimal results. This hybrid model shows promise for robust data modelling, as it efficiently captures temporal and spatial relationships. The architecture is shown in Figure 4.

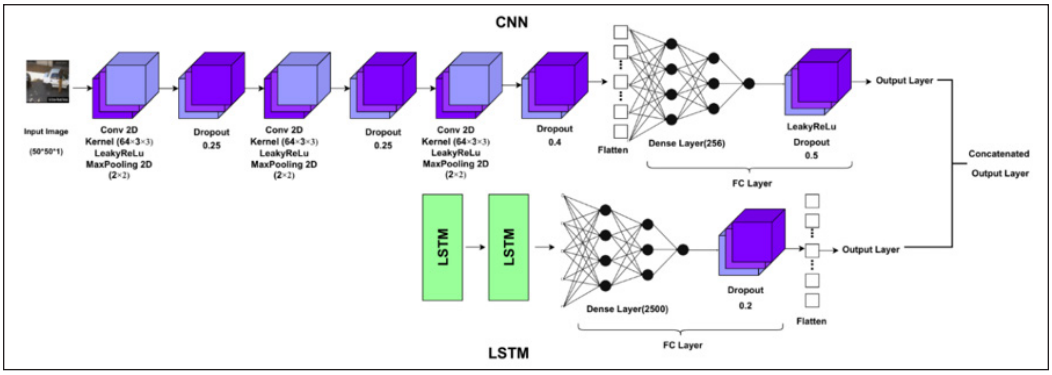


Figure 4. Architecture of the proposed CNN+LSTM model

Proposed CNN+GRU Hybrid Model

The CNN-based models were initialised using the pretrained weights specified for their respective architectures. AlexNet was used as a transfer-learning component with weights pretrained on ImageNet. The pretrained convolutional representations were used to extract hierarchical spatial features from the surveillance frames, after which the corresponding recurrent or classification layers were trained for the target activity-classification task. This initialisation strategy is distinct from training the entire network with randomly initialised weights. The MaxPooling layers downsample the feature maps, reducing spatial dimensions and computational cost, while the LeakyReLU activation function adds non-linearity, enabling the network to learn more complex representations. To improve generalisation and avoid overfitting, dropout layers are used. To prepare the data for integration with the GRU network, the flattening phase converts the 2D feature maps into a 1D vector. At this point, the CNN has successfully converted the input's spatial structure into a feature representation that highlights the most important elements of the data. Two GRU layers are employed in the model, one of which is configured to return sequences of True values, enabling the network to output a series of hidden states. For sequential tasks, where the contribution of each timestep is crucial to the final prediction, this setup is advantageous. The GRU maintains strong performance while being computationally inexpensive thanks to its very small number of parameters and its ability to preserve critical information over time steps. Concatenating the feature representations from both components after they have processed their respective inputs yields the final output in the hybrid CNN-GRU model. While the GRU processes the sequential input and outputs the hidden state h_t, the CNN processes the spatial data and produces a flattened output V. Concatenating these feature vectors results in a combined representation, as shown in Equation 18:

$$H_{concat} = concat(V, h_t) \quad [18]$$

After passing through a fully connected layer with this concatenated feature vector, the final output is calculated using Equation 19

$$y_{pred} = softmax(W_o H_{concat} + b_o) \quad [19]$$

Here, W_o is the weight matrix, b_o is the bias term, and y_{pred} represents the predicted class probabilities. The integration of CNN and GRU in this manner allows the model to capture both spatial and temporal features, enhancing its ability to perform classification tasks. The architecture of the proposed model is shown in Figure 5.

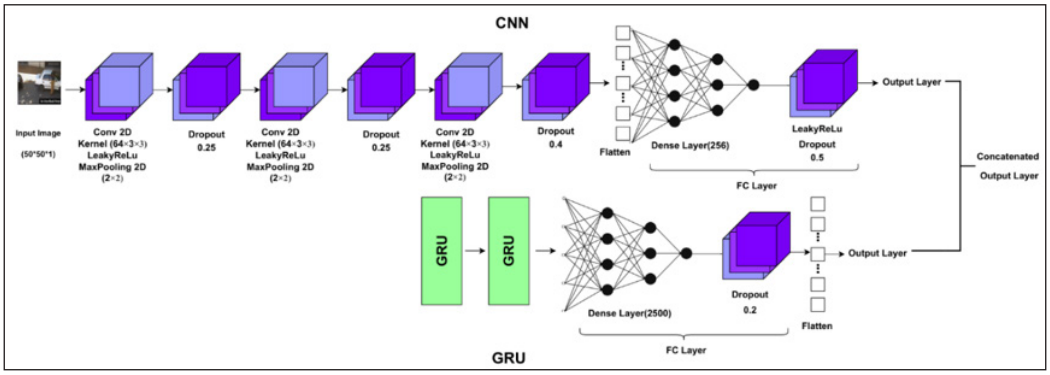


Figure 5. Architecture of the proposed CNN+GRU model

EXPERIMENTAL SETUP

Implementation

In our research, we employed Python as the primary programming language and executed the whole process in Google Colab Pro, utilising its superior computing capabilities and GPU acceleration to expedite training and execution. We utilised TensorFlow 2.15, Scikit-Learn, and Keras to develop and train DL models, while ImageDataGenerator facilitated real-time data augmentation, enhancing generalisation by generating diverse training examples.

We incorporated OpenCV for image preprocessing to perform tasks such as scaling and formatting the input data, thereby ensuring uniformity across the dataset. The Scikit-learn library offers essential utilities, such as train_test_split for dataset splitting and evaluation metrics like classification_report and confusion_matrix, for a complete assessment of model performance. We utilised Pickle for efficient storage of intermediate results, and TQDM improved code readability by including progress bars in our loop operations.

We incorporated convolutional layers and other critical elements for spatial feature extraction, followed by layers for temporal analysis, thereby capturing both spatial and

sequential characteristics. Utilising Colab Pro facilitated the rapid training of models and performance monitoring via ModelCheckpoint and CSVLogger, thereby enabling ongoing progress tracking and ensuring repeatability. Visualisation packages like Matplotlib were essential for graphing performance indicators, facilitating the analysis and optimisation of the model. The integration of these libraries, together with the computational benefits of Colab Pro, has markedly enhanced the productivity and performance of our research. The number of parameters of each model for both datasets is shown in Table 1.

Table 1
Parameters and values of all models

Dataset	Model	Parameter	Size (Megabyte)	Training Time (Seconds)
Dataset 1	CNN+AlexNet	32396948	123.58	105.98
	CNN+LSTM	3705000	14.13	124.88
	CNN+GRU	3705168	14.13	127.66
Dataset 2	CNN+AlexNet	62396948	86.3	156.32
	CNN+LSTM	17411208	66.42	136.7
	CNN+GRU	17411040	66.42	143.86

Data Preprocessing

Using Python's OpenCV library, frames were extracted from videos as part of the video preprocessing in this investigation. The frames were initially placed in an empty list, and the total number of frames in each video was obtained by accessing it. To ensure that every aspect of the video material was covered, frames were systematically extracted at predetermined intervals. The predefined sequence length, S_{length}, was divided by the total frame count, F_{total}, to find the interval. The result was shown in Equation 20:

$$I = \frac{F_{total}}{S_{length}} \quad [20]$$

where the interval at which frames were extracted is denoted by I. Following extraction, frames were resized to standard sizes (e.g., 50 × 50 and 128× 128 pixels) in accordance with the analysis specifications, guaranteeing uniformity throughout all samples. The 50 × 50 was used for dataset 1, and the 128x128 was used for dataset 2. After resizing the frames, colour histograms were computed for each frame to capture the distribution of pixel intensities across colour channels. The colour histogram, denoted by H, was computed over the three colour channels (Red, Green, Blue) using 8 bins per channel, producing a total of 512 bins as shown in Equation 21:

$$H(c_1, c_2, c_3) = \sum_{x=0}^{W-1} \sum_{y=0}^{H-1} \mathbb{I}\big(I_{x,y}(R) = c_1\big)\mathbb{I}\big(I_{x,y}(G) = c_2\big)\mathbb{I}\big(I_{x,y}(B) = c_3\big) \quad [21]$$

where:

-$H(c_1, c_2, c_3)$ is the histogram bin value for a particular combination of colour intensities c_1 (Red), c_2 (Green), and c_3 (Blue). - W and H represent the width and height of the image, respectively. -𝕀 is an indicator function that checks if the pixel at coordinates (x, y) has the specific colour values c_1, c_2, and c_3.

This resulted in a compact feature vector representing the overall colour distribution of the frame, which was then flattened for use in the model. The colour histogram method is computationally efficient and captures important colour patterns, which is useful for distinguishing different visual content.

Later, the normalised pixel intensity I_{norm} for every pixel p in each image was computed using min-max normalisation as follows shown in Equation 22:

$$I_{norm} = \frac{I_p - I_{\min}}{I_{max} - I_{min}} \quad [22]$$

where I_p represents the pixel value, and I_{min} and I_{max} denote the minimum and maximum pixel intensities, respectively. where the minimum and maximum pixel intensities are indicated by Imax and I_{min} accordingly, and the pixel value is represented by I_p. To standardise the frames for later model input, this phase ensured that the pixel values, initially in the range [0, 255], were scaled to [0, 1]. By lessening the sensitivity to changes in pixel intensity, this preprocessing step improved the convergence and generalisation of DL models. The prepared frames were fed into the suggested model for additional analysis and training after normalisation. For each video, frames were sampled at approximately uniform temporal intervals to construct an ordered sequence of frames. Let F_{total} denote the total number of frames in a video and S_{length} denote the predefined sequence length. The sampling interval was determined as

$$I = \frac{F_{total}}{S_{length}}$$

where I represent the approximate interval between two consecutive sampled frames. The selected frames preserve their original temporal order and form a sequence

$$\left\{X_1, X_2, \ldots\ldots\ldots\ldots, X_{s\,lengt\,h}\right\}$$

Each frame was independently resized and normalised before being processed by the CNN. Thus, the CNN extracts spatial features from individual frames, and the resulting ordered sequence of features is fed into the LSTM to model temporal dependencies across the video segment.

Dataset Details and Splitting

UCF-Crime and HBD21, two publicly accessible datasets, were used in this study. The two datasets, from reputable sources, were essential for evaluating the proposed hybrid model for identifying questionable activities. Table 2 presents a comprehensive overview of the datasets, outlining the research questions each dataset addresses and providing references for further investigation.

Table 2
Summary of datasets used

Dataset	Video Samples	Reference
UCF-Crime (Dataset 1)	1900	Sultani et al., 2018
HBD21 (Dataset 2)	463	Jayaswal & Dixit, 2021b

Each dataset extracts a specific number of images from videos. A total of 338,784 images are taken from 1,900 videos, and a total of 13890 images are taken from 463 videos for the UCF-Crime dataset, as shown in Table 3 and Table 4.

The UCF-Crime dataset consists of 12 classes— Abuse, Arrest, Arson, Assault, Burglary, Explosion, Fighting, Vandalism, Stealing, Robbery, Shooting, and Shoplifting—which we examined. Each class in the dataset has a specific number of images, for a total of 338,784 images extracted from 1,900 videos after the dataset was gathered.

Table 3
Training and testing set distribution after 80:20 split (UCF Crime dataset)

Class	Number of Images	
	Training Set	Testing Set
Abuse	15,557	3,816
Arrest	23,666	6,096
Arson	21,863	5,351
Assault	10,417	2,600
Burglary	37,766	9,395
Explosion	20,249	5,014
Fighting	20,623	5,292
Vandalism	11,798	2,939
Stealing	37,391	9,395
Robbery	33,850	8,478
Shooting	11,859	2,911
Shoplifting	25,988	6,470
Total	271,027	67,757

Using train_test_split from the sklearn model_selection module, we split our UCF Crime dataset at an 80:20 ratio before training to ensure an appropriate distribution of training and testing data. The training set will consist of 271,027 images, and the testing set will have 67,757 images; after splitting,

each class will have the following number of images for training and testing, as shown in Table 4.

Similarly, the Human Behaviour dataset (HBD21) contains four activity categories: assault violence, gun violence, normal actions, and sabotage violence. In total, 13,890 images were extracted from 463 videos. An 80:20 training-testing split was used, resulting in 11,112 training images and 2,778 testing images. The class-wise distributions are reported in Table 4.

Table 4
Training and testing set distribution after 80:20 split (HBD21)

Class	Number of Images	
	Training Set	**Testing Set**
assault_violence	2454	636
gun_violence	3469	851
normal_actions	2613	627
sabotage_violence	2576	664
Total	11112	2778

Table 5
Hyperparameter setup

Parameter	Value
Learning Rate	0.01, 0.001
Optimizer	Adam, SGD, Hybrid
Loss Function	CategoricalCrossentropy
Activation (CNN Hidden)	LeakyReLU (alpha=0.1)
Activation (Output)	Softmax
Epochs	50
Batch Size	1000
Verbose	1
Callbacks	ModelCheckpoint, CSVLogger, EarlyStopping
CNN Layers	3 Conv2D + MaxPooling2D + Dropout
Input Shape (CNN)	(50, 50, 1) (after reshaping)
Regularisation (CNN)	L2 (0.01)
Number of Classes	12

Hyperparameters

The experiments used a fixed maximum of 50 training epochs and a batch size of 1000. Learning-rate experiments considered values of 0.01 and 0.001, while the final configuration was selected based on the comparative experiments reported in Tables 6 and 7. For the hybrid optimiser, Adam was used during the initial training phase and was subsequently replaced by SGD with momentum at the predefined switching epoch. The momentum coefficient and switching epoch were fixed throughout the corresponding experiments. These settings are explicitly reported in Table 5 to improve reproducibility.

Learning Rate: In training, the learning rate is a crucial hyperparameter that regulates the size of weight updates during gradient descent. It regulates the rate at which the model learns by adjusting the weights to minimise the loss function. An elevated learning rate may lead to rapid convergence, risking overshooting the optimal solution, whereas a lower learning rate might result in slower convergence or entrapment in local minima. When training samples are very numerous, the learning rate becomes a concern as the underlying system loses its capacity for meaningful generalisation. This can also happen when the computational volume exceeds the input's

dimensionality in the vector space, and the network has too many neurons. Although it reduces training loss, it is well acknowledged that a deep network cannot be effectively trained with a small initial learning rate (Li et al., 2019). While reducing the memorisation of noisy data, an initially high learning rate slows the learning of complex patterns via degradation (You et al., 2019). The equation for weight updates in gradient descent is as follows in Equation 23:

$$W_{new} = W_{old} - \alpha \cdot \frac{\partial L}{\partial W} \quad [23]$$

where, $W_{new}$is the updated weight, W_{old} is the previous weight, α is the learning rate, $\frac{\partial L}{\partial W}$ is the gradient of the loss function L with respect to the weights W. Our research employed three learning rates—0.01, 0.001, and 0.0001—to investigate their impact on the training dynamics of our hybrid models. A learning rate of 0.01 facilitated rapid convergence through substantial weight updates; however, this expediency posed the risk of instability and the possibility of overshooting the ideal solution (Kingma & Ba, 2014). A learning rate of 0.001 achieved equilibrium, promoting consistent advancement with controllable updates and preventing excessive variability. The minimal learning rate of 0.0001 facilitated very steady and accurate modifications, enhancing model performance, but necessitating extended training durations. By manipulating these rates, we sought to balance convergence speed and model stability, ensuring both efficient training and successful outcomes.

Activation Function: A crucial element of neural network architecture that influences behaviour and computational complexity. Artificial neural networks employ activation functions to convert an input signal into an output signal, which is then supplied back as input to the next layer in the stack. We compute the sum of the products of inputs and their corresponding weights in an artificial neural network, then apply an activation function to produce the output of that layer, which is fed as input to the next layer. An activation function is used to transform the input signal into a simple linear function, which is equivalent to a degree-1 polynomial. Even though linear equations are simple and easy to solve, they are not very complex and can't learn or identify complex mappings from data. In most cases, a neural network without activation functions behaves like a linear regression model, but it doesn't offer as much power or speed. It would be great if a neural network could do more than just learn and generate a linear function. For example, it should be able to model different types of data, such as images, videos, music, speech, and text (Sharma et al., 2020). We used the Softmax activation function in the output layer and the LeakyReLU activation function in the hidden layers for our work. The LeakyReLU effect assigns a variant to negative values to preserve neurons and thus the useful data. Unlike the ReLU function, which renders these values zero and kills the neurons, it can ignore part

of the important data. For massive volumes of data, this can greatly influence the learning process. Moreover, the LeakyReLU function offers a trade-off between performance and network sparsity. LeakyReLU is a variation of ReLU by assigning a non-zero output for negative input, shows in Equation 24:

$$f_{LeakyReLU}\ (x) = max(\alpha x, x) \quad [24]$$

where α is a predetermined value in the range of (0, 1). ReLU maps negative inputs to zero, whereas LeakyReLU compresses negative inputs using a predefined linear function. Compressing LeakyReLU helps preserve the negative part of the feature information. LeakyReLU therefore trades off its input knowledge with network sparsity. Moreover, ReLU and LeakyReLU are unbounded functions that solve the gradient vanishing problem in Softplus and ELU. On the other hand, A combination of multiple sigmoid functions is the softmax function. These values can be interpreted as the probabilities of data points belonging to a specific class, since a sigmoid function returns values in the range 0 to 1. In contrast to sigmoid functions, which are employed for binary classification, the softmax function can be employed for multiclass classification problems. The function returns the probability for each data point across all individual classes. The softmax activation function ensures that, as shown in Equation 25:

$$\sum_{k} P_k z_k = 1 \quad [25]$$

which is desirable in a classification problem with q classes and allows us to use the cross-entropy error function. However, by modelling $P(x|C)$, we can provide a better justification for the use of the softmax activation function. For a classification scheme using the sampling paradigm, $P(C_i|x)$ is modelled as shown in Equation 26:

$$P(C_i|x) = \frac{P(x|C_i)P(C_i)}{\sum_k P(x|C_k)\,P(C_k)} \quad [26]$$

where $P(C_i|x)$ is the posterior probability of class C_i given input x, $P(x|C_i)$ is the likelihood, and $P(C_i)$ is the prior probability of class C_i. The calculated probabilities are constrained by the Softmax function to ensure that they sum to 1 and that each probability lies between 0 and 1. The Softmax function is defined as shown in Equation 27,

$$\sigma(z_i) = \frac{e^{z_i}}{\sum_j e^{z_i}} \quad [27]$$

where $\sigma(z_i)$ represents the probability for class i, and the denominator is the sum of exponentials of all class scores z_j.

Loss Function: In mathematics, a loss function, also referred to as a cost function or objective function, is a mathematical function that measures the actual target value's difference from the expected output of an ML model. It gauges the model's performance: a smaller loss indicates that the model's predictions are more in line with the actual values, whereas a larger loss indicates inadequate forecasts. The aim of the training procedure is to minimise the loss function by adjusting model parameters (weights and biases) via gradient descent. Conversely, the loss functions of neural networks often limit their accuracy [64]. In this study, we used the categorical cross-entropy loss function, which is commonly used for multiclass classification problems. This loss function evaluates the degree of alignment between the predicted probability distribution and the actual distribution by comparing predicted class probabilities with the actual class identifiers [65] (typically represented in one-hot format). The model's classification accuracy is enhanced by encouraging it to assign higher probabilities to the correct class by minimising this loss during training. This loss function is defined as shown in Equation 28:

$$L(y, \hat{y}) = -\sum_{i=1}^{n} y_i \log(\hat{y}_i) \quad [28]$$

where y_i represents the true label for class i (typically in one-hot encoded format), and $\hat{y}_i$ denotes the predicted probability for class i. Given the predicted probabilities, this loss function computes the negative log-likelihood of the true labels. The objective of training is to reduce this loss, thereby enhancing the predicted probability of the correct class.

Optimisation is the process of minimising or maximising an objective function, also known as a cost or loss function. Attempts are made to identify optimal parameters θ that substantially reduce the cost function J(θ) in a training dataset for a deep neural network. To identify the optimal parameters for a deep neural network, gradient descent can be used. The accuracy of a deep neural network model is influenced by the optimiser's role. The optimisers used in this research are described below.

Adam

The advantages of momentum and RMSProp are combined in the popular DL technique known as the Adam (Adaptive Moment Estimation) optimiser. By keeping squared gradients and exponentially decaying averages of previous gradients, it modifies the learning rate for every parameter. Adam computes two estimates at each time step t:

The first moment (mean of gradients) m_t, updated as:

$$m_t = \beta_t m_{t-1} + (1 - \beta_1) g_t$$

The second moment (variance of gradients) v_t, updated as:

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$$

Bias correction terms are applied to m_t and v_t:

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}, \hat{v}_t = \frac{v_t}{1 - \beta_2^t}$$

The parameter update rule is:

$$\Theta_t = \Theta_{t-1} - \frac{\alpha \hat{m}_t}{\sqrt{\hat{v}_t} + \varepsilon}$$

where:
α is the learning rate,
ε is a small constant added for numerical stability.

SGD

When training neural networks, SGD is a fundamental optimisation technique frequently utilised, particularly for large datasets. By calculating the gradient of the loss function with respect to the parameters over a single or small batch of data, the model parameters are updated incrementally, facilitating quicker and more effective training. The usual SGD update rule is written as shown in Equation 29:

$$\Theta_{t+1} = \Theta_t - \eta \nabla_{\Theta_t} J(\Theta) \quad [29]$$

where η is the learning rate, and $\nabla_{\Theta_t} J(\Theta)$ is the gradient of the loss function $J(\Theta)$ with respect to the parameters Θ. Momentum can be added to SGD to improve its performance by reducing oscillations and hastening convergence in the desired direction. The definition of the momentum-enhanced variant of SGD is shown in Equations 30 and 31:

$$v_{t+1} = \mu v_t - \eta \nabla_{\Theta_t} J(\Theta) \quad [30]$$

$$\Theta_{t+1} = \Theta_t + v_{t+1} \quad [31]$$

where μ is the momentum coefficient, typically set to a value between 0.8 and 0.9. This approach allows the optimiser to build velocity in directions with consistent gradients, leading to more efficient and stable convergence.

Hybrid Optimiser

In the hybrid optimisation approach, the optimiser initially employs Adam to leverage its adaptive learning rate, which excels at handling noisy gradients and efficiently navigating the early stages of training. This phase is characterised by rapid loss reduction, thanks to Adam's ability to adjust the learning rate for each parameter based on the first and second moments of the gradients. Mathematically, the Adam update rule for epochs $t < T_{switch}$ (where T_{switch} is the predefined switch epoch) is shown in Equation 32:

$$\Theta_{t+1} = \Theta_t - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t + \varepsilon}} \quad [32]$$

Here, $\hat{m}_t$ and $\hat{v}_t$ represent the bias-corrected first and second moments of the gradients, and α is the learning rate, with ε being a small constant for numerical stability.

As training progresses and reaches the epoch $t \geq T_{switch}$, the optimiser switches to SGD with momentum, which is known for its stable convergence and superior generalisation performance in the later stages of training. The momentum update rule is already given in Equations 30 and 31. This phase helps the model fine-tune and converge to a more robust solution, avoiding sharp minima and improving generalisation.

The rationale for combining Adam and SGD in a hybrid strategy is to leverage the strengths of both optimisers. Adam accelerates convergence during the noisy, early phase of training, while SGD with momentum provides stability and better generalisation as the model approaches the global minima. This hybrid approach balances fast early training and stable, efficient convergence, ensuring a more optimal and robust solution. Mathematically, the hybrid optimiser's behaviour is described by Equation 33:

$$\Theta_{t+1} = \begin{cases} \Theta_t - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t + \varepsilon}}, & if\ t < T_{switch} \\ \Theta_t + v_{t+1}, & if\ t \geq T_{switch} \end{cases} \quad [33]$$

This combination of rapid initial optimisation with Adam and fine-tuned convergence with SGD offers a powerful, flexible strategy for training the proposed models.

Epoch

A single pass through the entire training dataset is called an epoch. The model adjusts its weights based on the computed loss for each training example during each epoch. The duration of model training depends on the number of epochs. An insufficient number of epochs can lead to underfitting, in which the model fails to learn the data patterns, while an excessive number of epochs can result in overfitting, in which the model learns noise and irrelevant details; the number of epochs ensures that the model converges to a satisfactory solution by striking a balance between model training time and performance. The learning algorithm typically operates for a significant number of epochs, often in the hundreds or thousands, until the model's error is adequately reduced (Brownlee, 2022). In our studies, we trained the model for 50 epochs—a decision that provided a well-balanced approach to learning and generalisation. Training for over 50 epochs gave the model ample opportunities to learn intricate patterns and characteristics from the dataset, thereby ensuring it could faithfully capture the fundamental structure and relationships in the data. This length was carefully chosen to prevent overfitting, a common problem when the model performs well on training data but poorly on unseen test data. We sought to balance sufficient learning with avoiding excessive memorisation of the training instances by setting the number of epochs to 50. This method ensured that the model could efficiently generalise to new data, thereby improving its overall dependability and performance in practical settings. Furthermore, this number of epochs allowed for complete training without the risk of overly long training durations, which can lead to diminishing returns in model accuracy and higher computational costs.

EVALUATION METRICS

Evaluation metrics, which offer measurable indicators of a model's performance on a particular task, are crucial for evaluating ML models' effectiveness. To assess the models' performance in our tests, we used metrics such as accuracy, precision, recall, F1-score, classification error, AUC-ROC, confusion matrix, and jaccard_score. Among the most frequently utilised performance criteria in prior research is accuracy. In this example, accuracy might be useful if the dataset has the same number of cases per class. If not, as the model forecasts the value of the majority class, accuracy cannot be a useful metric. Considered the average value of recall and accuracy, the F1-score comprises essential findings about the performance of classifiers in every class. Hence, it is helpful in situations with diverse class distributions. True positives (TP), false positives (FP), true negatives (TN), and false negatives (FN) are fundamental elements of assessment metrics used to gauge a model's performance. While FP refers to cases where the model mistakenly predicts a positive class for a negative instance, TP refers to cases where the model correctly predicts the positive class. TN shows examples where the model correctly forecasts the

negative class; FN results when the model fails to find a positive occurrence and incorrectly categorises it as negative. These indicators provide a more comprehensive picture of how well a model works by helping to estimate other important performance measures, including accuracy, precision, recall, and F1-score.

Accuracy

In most research, the accuracy metric has been used primarily to estimate the classification accuracy of models. As demonstrated in the following Equation 34, the accuracy metric consists of the measurements of the confusion matrix in our hybrid models,

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad [34]$$

Precision

It indicates the quantity of relevant selected data elements. In other words, of the observations that an algorithm has classified as positive, how many are genuinely positive. As per Equation 35, precision is defined as the ratio of true positives to the total of true positives and false positives,

$$Precision = \frac{TP}{TP + FP} \quad [35]$$

Recall

It provides "what number of relevant data items are selected". Out of the observations that are positive, how many the algorithm has predicted. Equation 36 claims that the recall is equal to the number of true positives divided by the total of true positives and false negatives:

$$Recall = \frac{TP}{TP + FN} \quad [36]$$

F1-Score

Calculating an algorithm's performance uses accuracy and recall under consideration using this statistic, sometimes referred to as F1-score or F-measure. Mathematically, it is the harmonic mean of recall and precision, developed as follows Equation 37:

$$F1 - score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \quad [37]$$

Confusion Matrix

Accuracy and correctness of an ML algorithm can be assessed using this matrix, one of the clearest and most understandable measures available. Its primary use is in classification problems where two or more classes might be produced. ROC-AUC score: ROC curve—receiver operating characteristic curve—which shows the relationship between false positive rate (1-specificity) and actual positive rate—aka sensitivity or recall—is used to compute this statistic. Notably, the ROC-AUC score can be a good performance indicator if the numbers of positive and negative samples are equal.

Kappa

Cohen's kappa is computed using the predicted degree of agreement resulting from chance alone and the observed agreement among raters. Cohen's kappa lets researchers assess the extent to which raters' assessments are consistent and trustworthy, rather than only the outcome of random chance, by comparing these two values [80]. The expression for kappa can be given by Equation 38.

$$p_e = \frac{(TP + FP)(TP + FN)}{(TP + TN + FP + FN)^2} + \frac{(TN + FP)(TN + FN)}{(TP + TN + FP + FN)^2} \quad [38]$$

MCC

For imbalanced data, the Matthews correlation coefficient (MCC) measures the precision of binary classifications. It also considers all of TP, TN, FP, and FN and is considered a balanced indicator, applicable even when class sizes vary. This can be mathematically expressed as Equation 39,

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \quad [39]$$

OPTIMISER BASED RESULT

The optimisation technique applied in this study stimulates the performance evaluation of several hybrid models integrating CNNs with transfer learning models and RNNs. Optimisers are essential for modifying model parameters to reduce the loss function and enhance overall performance. This study used Adam and SGD optimisers to evaluate their effects on model convergence and performance. In this research, we utilised two datasets: UCF-Crime and HBD21. We focused solely on Dataset 1 (UCF-Crime) to establish a baseline for our optimisation approach. The performance metrics of the baseline optimiser for Dataset 1 are detailed in the following section.

Performance of Adam Optimiser

With a learning rate of 0.001, CNN+AlexNet ranks first among models, achieving 98.61% on all measures—accuracy, precision, recall, and F1-score. This reveals a high degree of consistency and balance in spotting illicit behaviour. By leveraging its deep architecture and fine-tuning capability with the Adam optimiser, CNN+AlexNet may maximise performance at reduced learning rates. Though they do significantly less, CNN+LSTM and CNN+GRU yield consistent, strong results. CNN+LSTM exhibits good performance with 97.54% across all evaluation measures—precision, recall, and F1-score. With its 97.52% across all tests, CNN+GRU is thus also a consistent model. Though they generalise well, the small differences between these two models suggest that their architectures are not as strong at this learning rate as CNN+AlexNet.

Every model shows a small performance decline at the higher learning rate of 0.01. Leading with 97.82% accuracy, 97.53% precision, 97.32% recall, and a 97.81% F1-score, CNN+AlexNet still leads. The model still maintains high generalisation capacity, even when these values are somewhat lower than the findings obtained at 0.001; this holds even when the learning rate is raised. CNN+LSTM performs worse still with 96.84% accuracy, 96.53% precision, 96.53% recall, and an F1-score of 96.29%. This suggests that CNN+LSTM is more sensitive to increasing learning rates, thereby resulting in a larger performance reduction than CNN+AlexNet. CNN+GRU demonstrates higher resilience than CNN+LSTM, achieving 97.65% accuracy, 97.43% precision, 96.52% recall, and an F1-score of 96.53%; however, it shows a slight drop.

For all three hybrid models, 0.001 is the optimal learning rate compared with the other two rates (Table 6). With the highest precision, recall, and F1 scores, the CNN+AlexNet model proves its supremacy and is the most effective model overall. CNN+LSTM and CNN+GRU perform well, particularly at 0.001. Still, their performance drops more sharply at 0.01, confirming that a lower learning rate is preferable for maintaining consistent model accuracy and preventing overfitting or underfitting across different models. When CNN+LSTM and CNN+GRU are trained at the same learning rate, CNN+AlexNet achieves

Table 6
Comparison of hybrid models using adam optimiser with different learning rates

Learning Rate	Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
0.001	CNN+AlexNet	98.61	98.62	98.61	98.61
	CNN+LSTM	97.54	97.54	97.54	97.54
	CNN+GRU	97.52	97.52	97.52	97.52
0.01	CNN+AlexNet	97.82	97.53	97.32	97.81
	CNN+LSTM	96.84	96.53	96.54	96.29
	CNN+GRU	97.65	97.43	96.52	96.53

the best results at a learning rate of 0.001 using the Adam optimiser. The overall outcomes are shown in Figure 6.

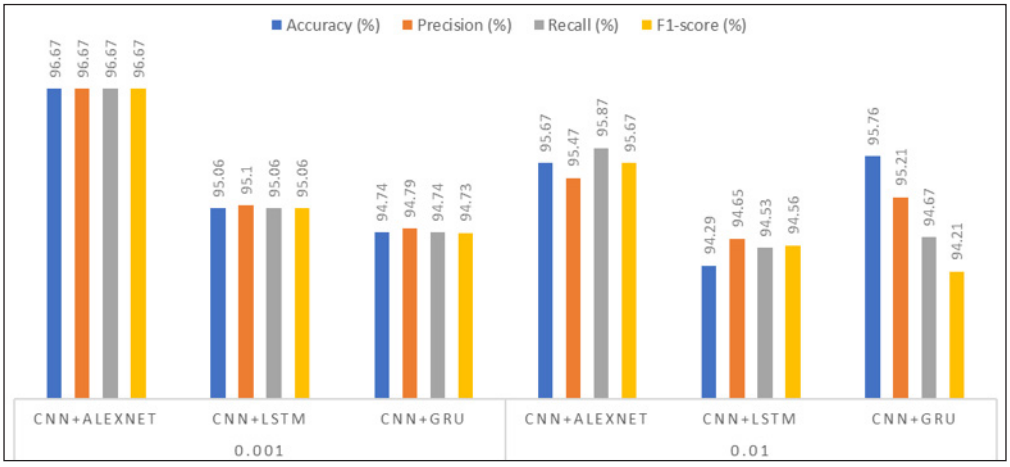


Figure 6. Performance comparison of three models using the Adam optimiser with varying learning rates

Performance of SGD Optimiser

This work finds that the performance of hybrid models combining CNN with AlexNet, LSTM, and GRU depends critically on the SGD optimiser. SGD has proven to be efficient as a conventional optimisation method for guiding these models toward convergence, especially in situations with large datasets, as shown in Table 7 for crime activity detection.

Table 7
Comparison of hybrid models using SGD optimiser with different learning rates

Learning Rate	Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
0.001	CNN+AlexNet	96.67	96.67	96.67	96.67
	CNN+LSTM	95.06	95.1	95.06	95.06
	CNN+GRU	94.74	94.79	94.74	94.73
0.01	CNN+AlexNet	95.67	95.47	95.87	95.67
	CNN+LSTM	94.29	94.65	94.53	94.56
	CNN+GRU	95.76	95.21	94.67	94.21

Of the three hybrid models using the SGD optimiser at a learning rate of 0.001, CNN+AlexNet ranks highest. The model shows steady performance across metrics, with accuracy, recall, and F1-score all reaching 96.67%. CNN+AlexNet thus detects relevant features and effectively balances predictions for positive and negative classes. Excellent alignment of these criteria suggests that the model generalises well, free from evidence of

overfitting or underfitting. The CNN + LSTM model achieves 95.10% accuracy, recall, and F1-score. Though it lags considerably behind CNN+AlexNet, this performance indicates good generalisation. At this learning rate, CNN+GRU achieves the lowest metrics: 94.79% accuracy, 94.74% recall, and an F1-score of 94.73%. The similarity of these values across the LSTM and GRU models indicates stable performance, even though they do not match CNN+AlexNet's precision and generalisation capability.

All models experience some performance degradation at a higher learning rate of 0.01. With precision, recall, and F1-score at 95.47%, 95.87%, and 95.67%, respectively, CNN+AlexNet still ranks first. The model remains robust even as the learning rate increases; though its performance declines somewhat, it still maintains a strong balance across these criteria. CNN+LSTM and CNN+GRU show more precise cuts. CNN+LSTM's F1-score falls to 94.65%, 94.53%, and 94.56%, respectively, for precision and recall. CNN+GRU also shows a similar trend, with an F1 Score of 94.21%, 95.21% accuracy, and 94.67% recall. This suggests that these models are more sensitive to fluctuations in the learning rate, thereby perhaps reducing ideal performance as the rate increases. The 0.001 learning rate helps all three hybrid models perform better than the 0.001 rate. CNN+AlexNet scored first among models tested at both learning speeds, maintaining extraordinary accuracy, recall, and F1-score. Conversely, the more significant declines in performance for CNN+LSTM and CNN+GRU at 0.01 imply that they are more prone to performance degradation as the learning rate increases. The SGD optimiser works best for these models at a learning rate of 0.001 overall; it primarily benefits CNN+AlexNet, which consistently produces balanced, high-precision predictions across numerous tasks. The comparative performance is shown in Figure 7.

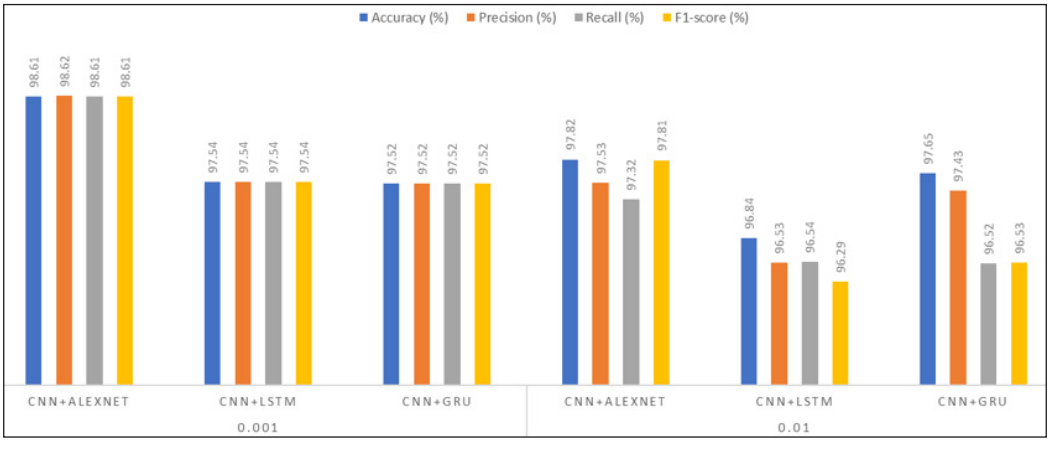


Figure 7. Performance comparison of three models using the SGD optimiser with varying learning rates

Proposed Hybrid Optimiser Performance Analysis

A thorough evaluation was conducted of the suggested hybrid optimiser across various settings to ensure its generalisability and robustness. First, the model's generalisation capacity is examined, with training and testing accuracy as essential benchmarks. This is a critical stage in determining the model's performance on hypothetical data. Precision, recall, and F-score metrics are then used in both datasets to calculate TP, FP, TN, and FN, which serve as indicators of classification performance. These metrics provide a thorough understanding of the model's classification performance, illuminating how well it differentiates between classes across various scenarios. Statistical measures such as Kappa and MCC, in addition to these fundamental classification metrics, are used to further assess the model's performance by analysing the agreement between anticipated and actual classifications. These measures are crucial for providing a more complex view of the model's dependability, mainly when datasets are unbalanced, or accuracy is insufficient to convey the subtleties of performance. Through a thorough evaluation of the model across these varied criteria (generalisation, classification, and statistical agreement), the comprehensive examination ensures that the suggested approach is efficient in terms of raw performance and robust, dependable, and appropriate for practical use. The outcomes are discussed in detail below.

Generalisation Analysis

In Table 8, models using the hybrid optimiser clearly benefit compared with those using single optimisers such as Adam and SGD. First, although the CNN+LSTM model reached an outstanding accuracy of 98.61%, models such as CNN+AlexNet and CNN+GRU fared considerably worse, hovering between 97% and 98% accuracy using the Adam optimiser. Conversely, the hybrid optimiser drove all models to surpass 99% accuracy; CNN+LSTM achieved an impressive 99.57%, thereby demonstrating an apparent gain in overall performance.

Second, the models struggled to attain the same performance levels using the SGD optimiser at different learning rates. While some, like CNN+GRU and CNN+AlexNet, dipped to the 94-95% range, even the best-performing CNN+LSTM model only attained 96.67% accuracy. With model accuracy exceeding 99%, the hybrid optimiser clearly outperformed these findings, demonstrating its ability to improve both precision and recall across all models.

Table 8
Training and testing accuracy for the hybrid optimiser utilised in the UCF-Crime dataset

Model	Accuracy (%)	
	Training	Testing
CNN+AlexNet	99.62	99.24
CNN+GRU	99.72	99.5
CNN+LSTM	99.69	99.57

With the best accuracy (99.57%), CNN+LSTM is the best-performing model in our analysis. Its hybrid optimiser and

deeper design enable it to more successfully capture complicated characteristics than CNN+AlexNet and CNN+GRU, thereby ranking among the best models in our work.

Table 9
Training and testing accuracy for the hybrid optimiser utilised in the HBD21 dataset

Model	Accuracy (%)	
	Training	Testing
CNN+AlexNet	99.21	98.78
CNN+GRU	99.64	98.7
CNN+LSTM	99.6	98.85

The hybrid optimiser's ability to balance its own constraints by combining the features of Adam and SGD optimisers helps explain the observed performance increase when utilising it. SGD is excellent for fine-tuning, but it may be slow to converge; Adam is well-known for its quick convergence, even if it may sometimes overshoot ideal solutions. Higher accuracy, as shown in Table 9, is achieved by models that benefit from both fast initial convergence and exact fine-tuning enabled by the hybrid optimiser. More crucially, the smallest difference between training and testing accuracy (99.69% vs 99.57%) for CNN+LSTM indicates that the models are well-generalised and not overfitted or underfitted, not merely memorising training data but consistently performing on unknown data. This improved generalisation guarantees that models are durable and can consistently manage new data, so the hybrid optimiser is better than Adam and SGD.

For Table 9, over both training and testing sets, the table shows the performance of the three hybrid models—CNN+AlexNet, CNN+GRU, and CNN+LSTM. The high accuracy ratings of these models demonstrate the benefit of using a hybrid optimiser.

CNN+LSTM outperforms CNN+AlexNet and CNN+GRU, achieving an amazing test accuracy of 98.85% when models utilising single optimisers such as Adam or SGD are compared. While all models exceeded 99% accuracy on the training set with the hybrid optimiser, CNN+LSTM models employing a single optimiser, such as Adam, have previously shown great performance in past trials.

This shows the hybrid optimiser's greater generalisation ability, enabling improved performance on both the training and testing sets. Models trained using the hybrid optimiser show a smaller gap between training and testing accuracies, ensuring their generalisation to new data and reducing their tendency to overfit. Adam and SGD benefit from fast convergence during fine-tuning, making the hybrid optimiser an excellent tool for achieving top-notch results across models such as CNN+LSTM. As a result, the proposed CNN+LSTM is the best performer among others. The training and validation accuracy and loss across 50 epochs for both datasets are shown in Figures 8 and 9.

Classification Analysis

This work assessed the classification performance of the hybrid model across multiple criteria, including accuracy, precision, recall, and F1-score, using the hybrid optimiser.

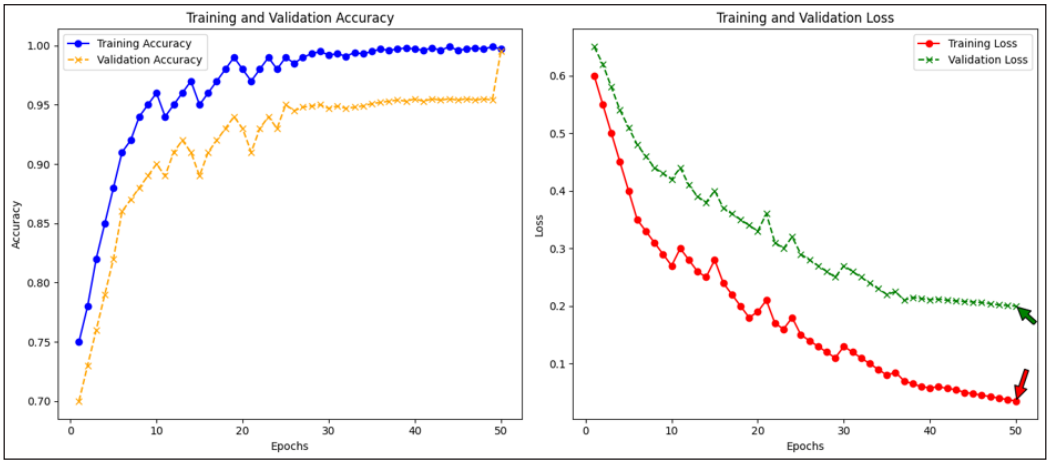


Figure 8. Training and validation accuracy and loss for the UCF-Crime dataset

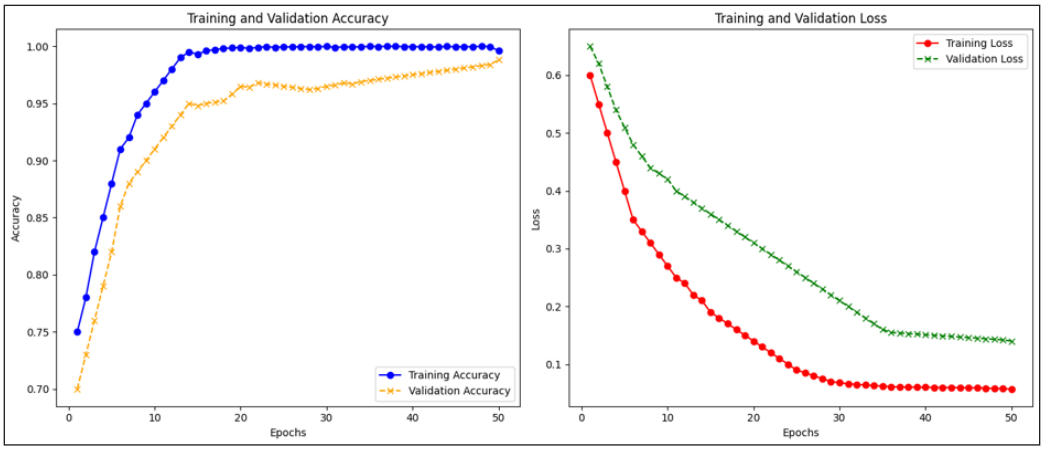


Figure 9. Training and validation accuracy and loss for the HBD21 dataset

Table 10 and Table 11 show the outcomes, which emphasise how effectively the hybrid optimiser increases the models' capacity to identify crime activities.

Evaluating the hybrid optimiser's performance for the UCF-Crime dataset with a remarkable balance between accuracy, recall, and F1-score, CNN+LSTM stands out as the

Table 10
Precision, Recall and F1-score for the hybrid optimiser utilised in the UCF-Crime dataset

Model	Optimiser	Precision (%)	Recall (%)	F1-Score (%)
CNN+AlexNet	Hybrid	99.25	99.24	99.25
CNN+GRU		99.59	99.5	99.5
CNN+LSTM		99.57	99.57	99.57

Table 11
Precision, Recall and F1-score for the Hybrid Optimiser utilised in the HBD21 dataset

Model	Optimiser	Precision (%)	Recall (%)	F1-Score (%)
CNN+AlexNet	Hybrid	98.78	98.78	98.77
CNN+GRU		98.71	98.7	98.71
CNN+LSTM		98.85	98.85	98.85

best performer. This model shows incredible consistency across all metrics: an F1 Score of 99.57%, a precision of 99.57%, and a recall of 99.57%. The near alignment of these values suggests that CNN+LSTM is well optimised for crime detection, handles edge cases effectively, and demonstrates strong robustness to both overfitting and underfitting. This clarifies its standing as the most consistent model for this set. With an F1-score of 99.25%, CNN+AlexNet performs well but shows a slight discrepancy between precision (99.25%) and recall (99.24%). Although these variations are small, they suggest that CNN+AlexNet may sometimes trade off recall—that is, the ability to properly detect genuine positives—in favour of precision, thereby increasing false positives at the expense of false negatives. CNN+GRU achieves the best precision (99.59%) but a somewhat lower recall (99.50%), resulting in an F1-score of 99.50%. This discrepancy between accuracy and recall suggests that although CNN+GRU minimises false positives well, it may occasionally miss real positives, making it somewhat less reliable than CNN+LSTM in preserving overall classification performance. The CNN+LSTM model proves superior in managing the UCF-Crime dataset overall, showing the best generalising and balanced performance across all measures.

Consistent with the results from the UCF-Crime dataset, CNN+LSTM once more ranks highest in the performance study on the HBD21 dataset. With almost perfect alignment across these key criteria, it achieves an F1-score, precision, and recall of 98.85%.

This harmony underpins CNN+LSTM's ability to generalise successfully without substantial trade-offs between false positives and false negatives. CNN+AlexNet follows closely with 98.78% accuracy, recall, and F1-score; it performs well but behind CNN+LSTM in terms of consistency. The slight variance in accuracy and recall here suggests that, while efficient, it may not always handle edge cases as well as CNN+LSTM does. CNN+GRU performs on par with CNN+AlexNet and shows minimal variance, generally with 98.83% precision, recall, and F1-score. Even though these values are somewhat comparable to those of CNN+LSTM, this hybrid model maintains a slight lead by exhibiting tighter alignment and a better balance across the measurements.

In both the UCF-Crime and HBD21 datasets, with minimal variation in precision, recall, and F1-score, CNN+LSTM frequently exhibits improved classification performance. This reveals its robustness in detecting real positives while reducing false positives and false

negatives. CNN+LSTM is the most reliable model for achieving consistent performance across these criteria in challenging classification tasks and crime activity detection. On the other hand, CNN+AlexNet and CNN+GRU show slight differences in recall and accuracy, even though they remain formidable opponents. These variations suggest they might be more likely to trade off among many classification methods, making them less reliable for handling demanding or edge-case scenarios than CNN+LSTM.

Statistical Analysis

According to observations from earlier generalisation and classification reports, the suggested hybrid model—CNN+LSTM with the hybrid optimiser—performed better on both datasets. The training and testing accuracy results showed that this model performed better than others in terms of generalisation and robustness when handling unknown input. The model's agreement between predicted and actual classifications was further evaluated statistically using Kappa and MCC. The CNN+LSTM model achieved nearly perfect scores of 0.99 for both Kappa and MCC on the UCF Crime dataset, as shown in Table 12. This shows a very high degree of agreement and few classification mistakes. This outcome held true for the HBD21 dataset as well, where the model obtained 0.97 for both metrics. While still functioning well, the CNN+GRU and CNN+AlexNet models did not achieve the same agreement, as evidenced by their significantly lower MCC and Kappa values. In summary, the proposed hybrid optimiser-enhanced CNN+LSTM model outperformed the other models in statistical agreement, classification, and generalisation, as demonstrated by the MCC and Kappa assessments. As these results confirm, the CNN+LSTM model is a highly successful method for tasks needing accurate prediction and performance across various circumstances. The resilient, well-generalised model can deliver high classification accuracy across varied datasets.

We performed ROC-AUC analysis on it, since CNN+LSTM was the best-performing model across both the UCF-Crime and HBD21 datasets in previous evaluations. Regarding precision, recall, and F1-score, this model displayed better generalisation capacity and stability. The CNN+LSTM model has remarkable classification abilities across all examined classes, as illustrated by the ROC-AUC curve in Figure 10 for the UCF-Crime dataset. Signifying perfect separation between real positives and false positives across all threshold

Table 12
Kappa and MCC scores for UCF Crime and HBD21 datasets across different models

Model	UCF Crime Dataset		HBD21 Dataset	
	Kappa	MCC	Kappa	MCC
CNN + AlexNet	0.96	0.96	0.97	0.97
CNN + LSTM	0.99	0.99	0.97	0.97
CNN + GRU	0.98	0.98	0.98	0.98

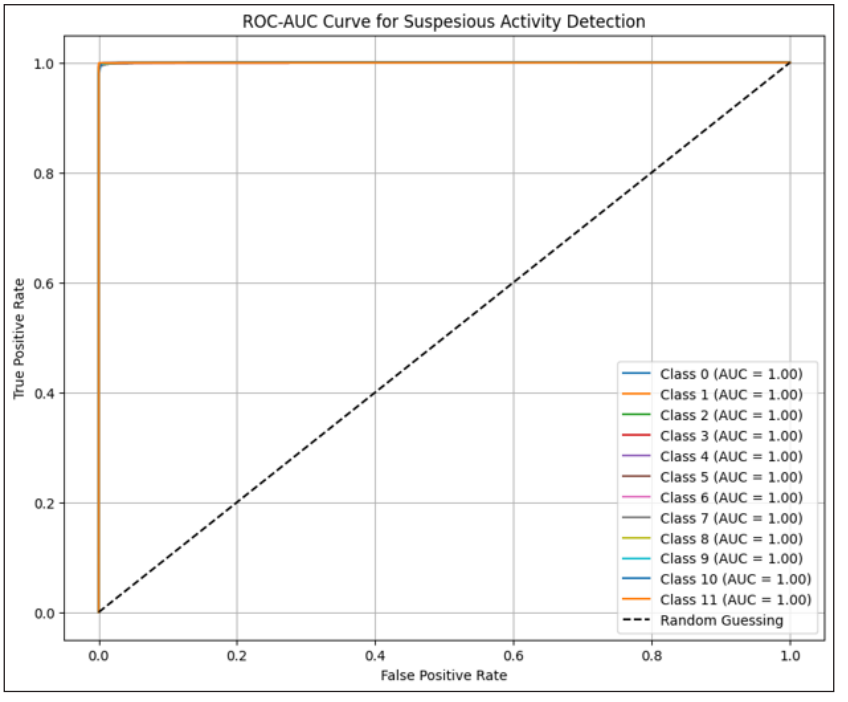


Figure 10. AUC values of all classes by the proposed hybrid optimisation technique (UCF-Crime dataset)

levels, the AUC values for every class consistently reach the ideal value of 1.00. This result underscores how effectively the model detects suspicious behaviour with minimal misclassifications—an essential capability for real-time criminal detection systems. This remarkable performance in all domains reveals the dependability and resilience of the CNN+LSTM model. The excellent balance between sensitivity and specificity, as indicated by the high AUC values, ensures the model accurately identifies a substantial number of actual suspicious behaviours, thereby reducing the likelihood of false positives. This feature is especially crucial in surveillance systems since false positives could lead to unnecessary interventions, while false negatives could cause hidden security problems. Applied to the HBD21 data, the ROC-AUC analysis supports the model's excellent generalisation across several domains. Figure 11 shows the AUC values for each of the twelve classes, which likewise consistently reach the maximum of 1.00, demonstrating the model’s exceptional accuracy in distinguishing various health-related events. This is important considering the complexity of the multi-class classification problem—where each class reflects a different disease or health abnormality. Achieving outstanding AUC scores across all classes demonstrates how well the CNN+LSTM model maintains high classification accuracy even with complex, advanced data. This performance is essential in healthcare applications because precise diagnosis and detection directly influence patient outcomes. The model's

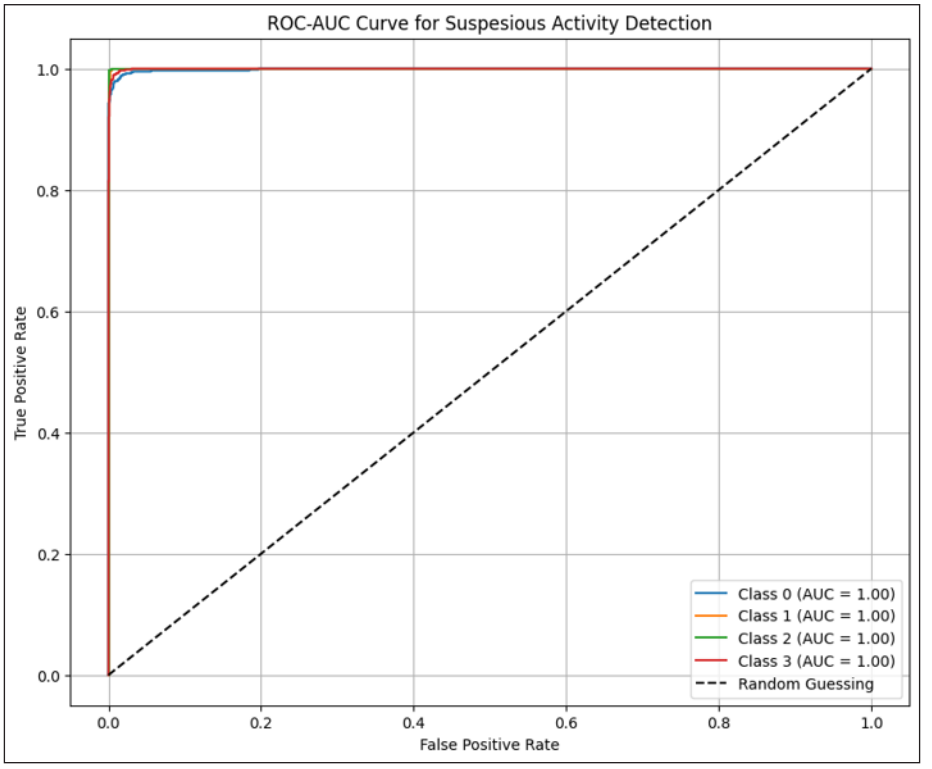


Figure 11. AUC values of all classes by the proposed hybrid optimisation technique (HBD21 dataset)

high sensitivity ensures successful detection of health issues, while its strong specificity reduces the likelihood of erroneous diagnoses, enabling accurate clinical decision-making.

The ROC-AUC analysis was used to evaluate the discriminative ability of the CNN+LSTM model for the individual activity classes. For the multiclass setting, ROC curves were computed in a one-versus-rest manner, and the area under each ROC curve was used as the class-wise AUC. The theoretical range of AUC is from 0 to 1, where a value of 1.0 represents perfect separation between the positive class and the remaining classes under the evaluated data. Therefore, the reported AUC values are interpreted as class-wise discriminative measures and should not be interpreted as evidence of real-world generalisation or deployment performance. Any value exceeding 1.0 would indicate a calculation or reporting error and is therefore not reported.

The confusion matrix for the UCF-Crime dataset, shown in Figure 12, covers a wide range of crime-related activities and provides insight into the CNN+LSTM model's performance. The Abuse class has 3809 test examples; among these, the model accurately identified 3787. Still, it misclassified 22 events—7 as Arrests and another 7 as Assaults. Arson class has 5346 test samples; the model correctly predicted 5302 cases. There are a

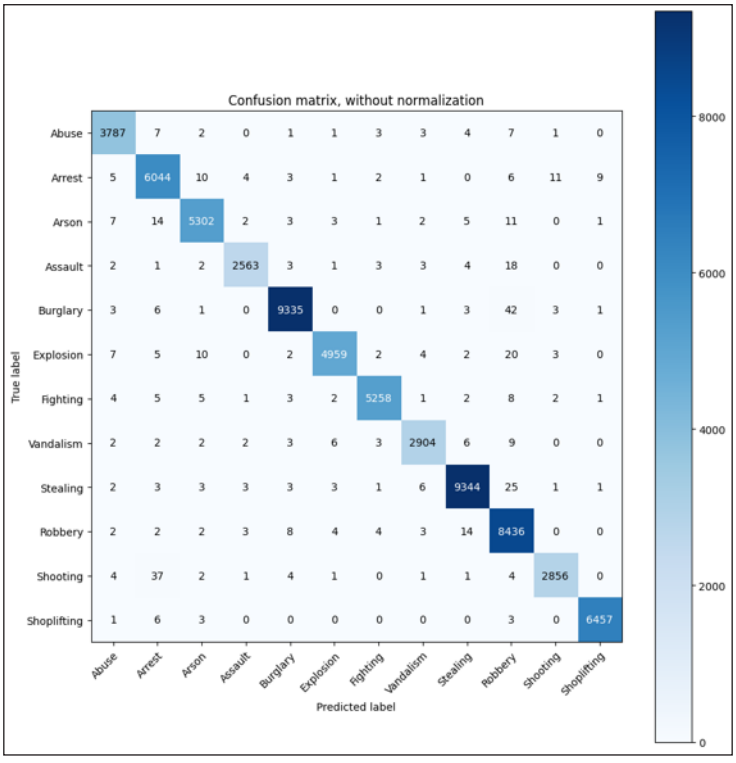


Figure 12. Confusion matrix of the proposed hybrid optimisation technique (UCF-Crime dataset)

few more misclassifications: two Vandalism and eleven Explosions. Having 9335 of 9389 test samples successfully classified, the model performed rather well in the Burglary class. The primary error was the misclassification of forty-two samples as robbery. Comprising 2925 test samples, the model struggled considerably in the Shooting class. It accurately categorised 2856 samples but misclassified 37 cases as Arrest and 14 as Robbery. Still, the model performed remarkably well in the Shoplifting class, precisely classifying all 6457 test instances. Overall, across the classes in the UCF-Crime dataset, the CNN+LSTM model performed relatively well, with an accuracy of almost 99.57%. Separating Shooting from Arrest in Robbery was the main challenge for the model, since dataset visual similarities or potentially overlapping features led to uncertainty.

CNN+LSTM model performance spanning four categories—attack violence, gun violence, everyday acts, and sabotage violence—is shown in the HBD21 dataset confusion matrix in Figure 13. Having 636 test samples, the Assault Violence class model correctly categorised 621 cases. Five were still misclassified as gun violence, one as normal behaviour, and nine as sabotage violence. The model did quite well with 851 observations in the Gun Violence class—predicting exactly 848 incidents. Two samples taken overall were

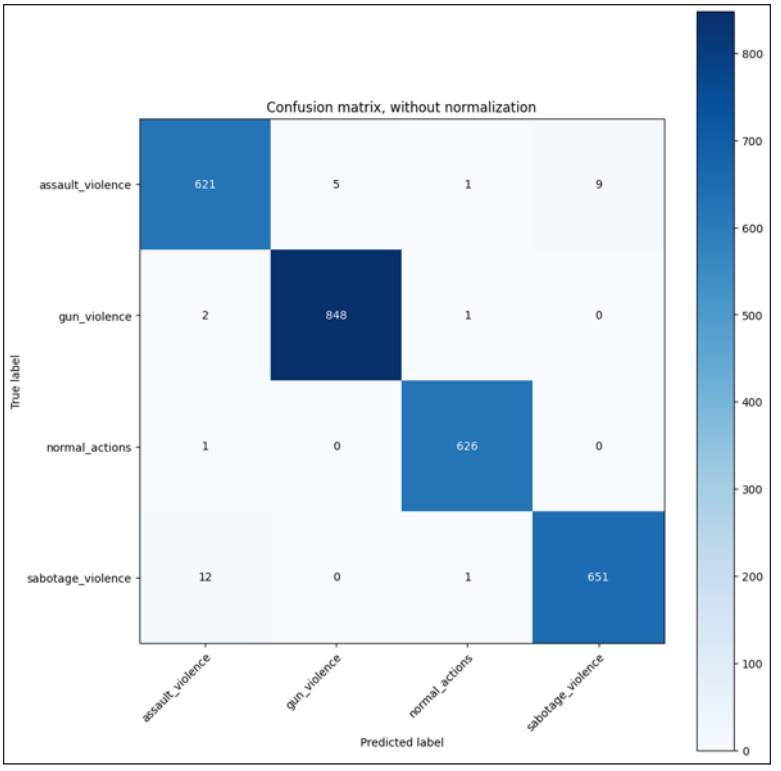


Figure 13. Confusion matrix of the proposed hybrid optimisation technique (HBD21 dataset)

misclassified as assault violence or expected behaviour. With just one incident misclassified as assault violence, the model exhibits great accuracy and correctly groups 626 out of 627 data into the Normal Actions class. For the model, the Sabotage Violence class presented notably more challenges. Of the 664 test samples, 651 were correctly identified; 12 were misclassified as attack violence, and one was misclassified as normal behaviour. On the HBD21 dataset, the accuracy exceeded 98.8%, with 2746 out of 2778 test samples correctly identified by the CNN+LSTM model. The leading cause of misclassification between attack and sabotage revealed possible overlap in attributes the model failed to detect.

The results of the proposed hybrid optimiser are summarised graphically in Figure 14 and Figure 15.

DISCUSSION

The suggested SentryNet concept is a watchful defender, constantly sifting through video footage to spot and address possible threats. In our experimental evaluations on both datasets, LSTM networks outperformed GRU and AlexNet among the RNN architectures tested. The LSTM's improved memory capacity, which enables it to remember information

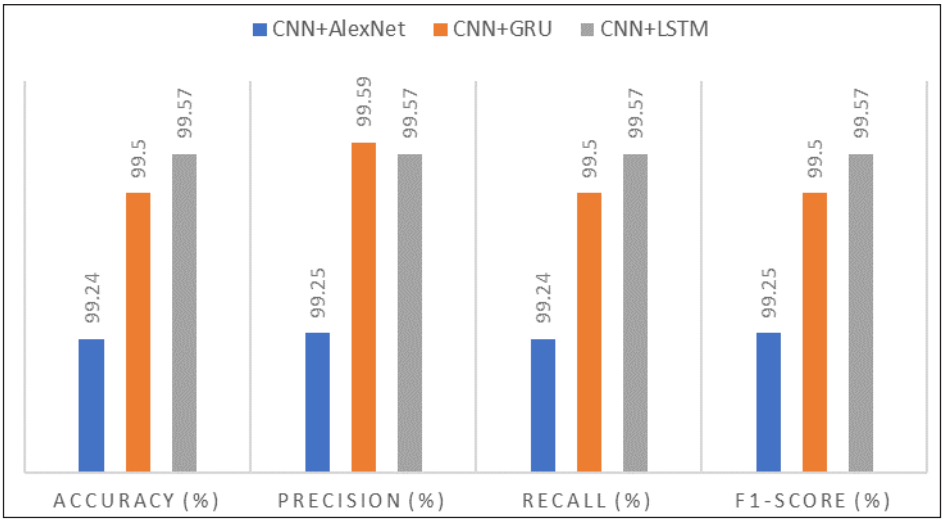


Figure 14. Summarisation of hybrid optimiser outcomes on UCF-Crime dataset

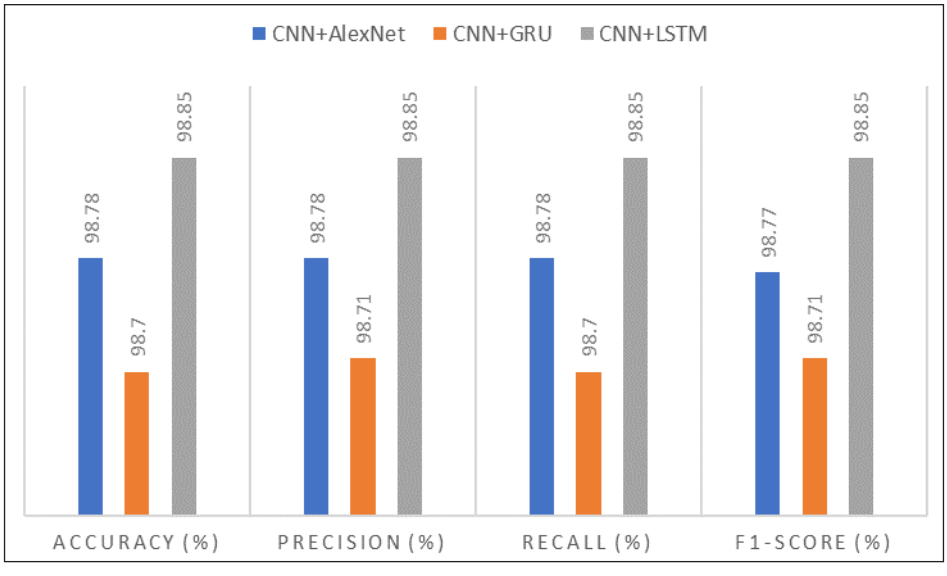


Figure 15. Summarisation of hybrid optimiser outcomes on HBD21 dataset

over longer sequences, is responsible for this better performance. Accurately detecting suspicious actions in surveillance scenarios is dependent on this ability. Furthermore, LSTMs have gating mechanisms that efficiently control information flow, ensuring that essential data are retained. In contrast, less important ones are ignored. They are excellent at capturing long-term dependencies. Additionally, their sturdy construction mitigates the vanishing-gradient issue that standard RNNs often face, enabling more efficient learning

from longer video sequences. Although AlexNet excels at extracting spatial features, it cannot model temporal dynamics, an area in which LSTMs excel. SentryNet is therefore a potent option for assessing complex behaviours in suspicious activity detection, owing to the combination of LSTM's capabilities and the spatial characteristics captured by CNNs. Based on Tables 7-11, our results show that the hybrid optimiser performs significantly better than the Adam and SGD optimisers. Because single optimisers are inherently incapable of managing the intricacies of training DL models, their performance is poor. For example, Adam performs well in the early training phases because of their adaptive learning rate and capacity to handle noisy gradients. However, later in the process, it might have trouble converging, which could lead to less-than-ideal outcomes. On the other hand, while SGD works well for generalisation and stable convergence, it often requires careful adjustment of its momentum and learning rate parameters to prevent it from crashing into local minima or converging too slowly. On the other hand, the hybrid optimiser leverages the advantages of Adam and SGD. The hybrid technique achieves a balanced optimisation strategy by using Adam for its fast convergence during the noisy early period of training, then transitioning to SGD with momentum for stability and fine-tuning. Compared with single optimisers, this synergy improves performance metrics by helping the model overcome training obstacles more effectively.

The hybrid optimiser offers a more flexible and reliable framework, improving model performance on a range of datasets. Reducing the computational complexity of the suggested models without sacrificing their ability to identify suspicious activity is one of our core research goals. We find considerable differences in resource requirements across model configurations, as indicated by parameter counts and training durations. With over 32.4 million parameters, the CNN+AlexNet model for Dataset 1 has the largest parameter count. This results in a significant computational load and a prolonged training time of 106 seconds. On the other hand, the CNN+LSTM and CNN+GRU models have smaller parameter counts, about 3.7 million, resulting in faster training times of about 125 and 128 seconds, respectively. This illustrates how successful learning with fewer parameters is possible when an LSTM or GRU is integrated with a CNN, thereby reducing computational complexity. With a training duration of 156 seconds, the CNN+AlexNet model also has the greatest parameter count in Dataset 2—roughly 62.4 million. With parameter counts of about 17.4 million and training times of 137 and 144 seconds, respectively, the CNN+LSTM and CNN+GRU models, in contrast, remain efficient. By strategically selecting architectures that balance performance and computational demands, our research aims to develop models that are effective in detecting suspicious activities and computationally efficient, facilitating their deployment in real-world scenarios. This is particularly important in applications where resource constraints and real-time processing capabilities are critical, enabling broader accessibility and practicality of DL solutions in surveillance systems.

Thus, the findings highlight the importance of selecting a model architecture to achieve optimal performance while minimising computational complexity. The analysis shows that the proposed CNN+LSTM is more computationally sound and boosts performance. The comparison in Tables 13 and 14 is intended to provide context for previously published results. The proposed model was evaluated using the 80:20 training-testing split described in this study. The experimental protocols, preprocessing procedures, data partitions, model configurations, and evaluation procedures used by the referenced studies were not necessarily identical to those used in the present work. Therefore, the reported accuracy values are compared descriptively and should not be interpreted as results from a strictly controlled, common evaluation protocol.

Table 13
Comparison of proposed model with state-of-the-art approaches on the UCF-Crime dataset

Reference	Model	Accuracy (%)	Optimiser
Kolaib and Waleed, 2024	Deep CNN	99.48	-
Park et al., 2023	Deep neural network with absorbing MC	84.94	Backpropagation
Kumar et al., 2024	CNN+RNN+Attention module	62.04	-
Sultana et al., 2023	2D Deep CNN	98.82	Adam
Taha et al., 2024	ViT B16 + Key Frame Extraction	95.87	SGD
Proposed method	CNN+LSTM	99.57	Hybrid (Adam+SGD)

Table 14
Comparison of proposed model with state-of-the-art approaches on the HBD21dataset

Reference	Model	Accuracy (%)	Optimiser
Kurchaniya and Kumar, 2024	Channel-wise attention with MOG and a pointwise ConvBi-LSTM	95.1	-
Jayaswal and Dixit, 2021a	Xception model+LSTM	97.25	Adam
Proposed model	CNN+LSTM	98.85	Hybrid (Adam+SGD)

Additionally, Table 13 compares the suggested model with cutting-edge models while accounting for the optimisers used in those investigations. Our CNN+LSTM architecture, which achieves greater accuracy and uses a hybrid optimisation technique combining Adam and SGD, is effective, as demonstrated by this comparison. Using a hybrid optimisation strategy that combines Adam and SGD, the proposed CNN+LSTM model achieves a remarkable accuracy of 99.57% in detecting suspicious activity on the UCF-Crime dataset. By contrast, the Deep CNN developed by Kolaib and Waleed (2024) achieved 99.48%

accuracy, although it lacked a robust optimisation plan, which might have limited its flexibility and led to lower evaluation metrics than those reported in the proposed research. Park et al. (2023) used a deep neural network with absorbing Markov Chains and achieved only 84.94%, indicating that the intricacies of activity detection proved too much for it to handle. Notably, there was a performance difference with Kumar et al. (2024), who obtained much lower accuracies of 49.04% and 62.04% with their respective models. Our model outperformed Sultana et al. (2023), achieving 98.82% using a 2D Deep CNN with Adam, by successfully integrating CNN and LSTM architectures to leverage both temporal and spatial information. Finally, Taha et al. (2024) used a Vision Transformer (ViT B16) with keyframe extraction and achieved 95.87%, but our hybrid optimisation method offers a more balanced and robust solution. Overall, the CNN+LSTM model achieved the highest reported testing accuracy among the evaluated hybrid models on both datasets. On UCF-Crime, it achieved 99.57% testing accuracy, while on HBD21 it achieved 98.85%. The corresponding precision, recall, and F1-score values were also closely aligned with the reported accuracy values. These results indicate consistent classification performance under the experimental settings used in this study.

As shown in Table 14, we evaluated the proposed model against cutting-edge methods on the HBD21 and UCF-Crime datasets. Our CNN+LSTM model, which used a hybrid optimisation technique combining Adam and SGD, obtained an outstanding accuracy of 98.85%. This outperforms the Xception model with LSTM by Jayaswal and Dixit (2021a) (97.25%) and the Channel-wise Attention with MOG and ConvBi-LSTM model by Kurchaniya and Kumar (2024) (95.10%). The results show that our proposed model performs very well in terms of accuracy and demonstrates the hybrid optimiser's efficacy, enhancing its resilience when handling complex surveillance tasks across various datasets. The comparison examination highlights the CNN+LSTM architecture's superior performance.

CONCLUSION

The SentryNet framework successfully meets the urgent need for sophisticated automated systems in the cybersecurity and surveillance domains to identify suspicious activity. Our method combines the advantages of both architectures—LSTM networks and CNNs—to ensure reliable extraction of spatial and temporal features. The long-term memory capacity and efficient gating mechanisms of the LSTM model have contributed to its higher performance compared to other standard recurrent architectures, such as GRUs and AlexNet. Our empirical results demonstrate that SentryNet achieves remarkable accuracy rates of 98.85% on the HBD21 dataset and 99.57% on the UCF-Crime dataset, outperforming established models in the literature. Additionally, the proposed hybrid optimisation strategy, combining Adam and SGD, effectively mitigates the limitations

of each optimiser, thereby enhancing convergence rates and model stability. Overall, the suggested architecture is a viable option for real-time suspicious activity detection since it minimises computational cost, improves model generalisation, and lowers biases. Future studies could combine cutting-edge technologies such as attention mechanisms, real-time data collection, and dynamic optimisation procedures to further improve the model's performance and scalability. These modifications will increase the model's real-time applicability and robustness, enabling its adaptation to different surroundings. Moreover, resolving constraints, including computational limitations and dataset biases, more effectively would improve the model's generalisability and scalability. This study intends to advance the field of automated crime detection by carefully navigating these advancements, enabling more accurate and efficient surveillance systems and, in turn, enhancing public safety and security. The present study does not include dedicated measurements of end-to-end inference latency or frames per second; therefore, real-time deployment performance cannot be established solely from the reported training and classification results. Future work should evaluate inference speed, memory consumption, and deployment performance on representative surveillance hardware.

ACKNOWLEDGEMENT

The authors would like to extend their sincere thanks to their respective organisations for providing an academic environment and other requirements for conducting research. The authors also thank HBD21 and UCF-Crime datasets which were used for experimentation purposes as a base for proposed SentryNet system development. Moreover, the authors would also like to thank those researchers whose works are very valuable in the areas of video surveillance, deep learning, and hybrid optimisation.

REFERENCES

Ahmed, W., & Yousaf, M. H. (2024). A deep autoencoder-based approach for suspicious action recognition in surveillance videos. *Arabian Journal for Science and Engineering, 49*(3), 3517-3532. https://doi.org/10.1007/s13369-023-08038-7

Ali, M. M. (2023). Real-time video anomaly detection for smart surveillance. *IET Image Processing*, *17*(5), 1375-1388.

Al-Lahham, A., Zaheer, M. Z., Tastan, N., & Nandakumar, K. (2024). Collaborative learning of anomalies with privacy (CLAP) for unsupervised video anomaly detection: A new baseline. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 12416-12425). IEEE. https://doi.org/10.1109/cvpr52733.2024.01180

Brownlee, J. (2022, August 15). *Difference between a batch and an epoch in a neural network*. Machine Learning Mastery. https://machinelearningmastery.com/difference-between-a-batch-and-an-epoch/

Cesario, E., Lindia, P., & Vinci, A. (2024). Multi-density crime predictor: An approach to forecast criminal activities in multi-density crime hotspots. *Journal of Big Data, 11*(1), Article 75. https://doi.org/10.1186/s40537-024-00935-4

Cho, K., van Merriënboer, B., Gülçehre, Ç., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 1724-1734). Association for Computational Linguistics. https://doi.org/10.3115/v1/d14-1179

Ghosh, P., Bose, S., Roy, S., & Mondal, A. (2022). Detection of abnormal human behavior using deep learning. *BOHR International Journal of Smart Computing and Information Technology, 3*(1), 59-68. https://journals.bohrpub.com/index.php/bijscit/article/view/302

Gowada, R., Pawar, D., & Barman, B. (2023). Unethical human action recognition using deep learning based hybrid model for video forensics. *Multimedia Tools and Applications, 82*(19), 28713-28738. https://doi.org/10.1007/s11042-023-14508-9

Jan, A., & Khan, G. M. (2023). Real world anomalous scene detection and classification using multilayer deep neural networks. *International Journal of Interactive Multimedia and Artificial Intelligence, 8*(2), 158-167. https://doi.org/10.9781/ijimai.2021.10.010

Jayaswal, R., & Dixit, M. (2021a). A framework for anomaly classification using deep transfer learning approach. *Revue d'Intelligence Artificielle, 35*(3), 255-263. https://doi.org/10.18280/ria.350309

Jayaswal, R., & Dixit, M. (2021b). *Human Behavior dataset (HBD21)* (Version 1) [Data set]. Mendeley Data. https://doi.org/10.17632/xh9pgb3w8c.1

Jebur, S. A., Alzubaidi, L., Saihood, A., Hussein, K. A., Hoomod, H. K., & Gu, Y. (2025). A scalable and generalised deep learning framework for anomaly detection in surveillance videos. *International Journal of Intelligent Systems, 2025*(1), Article 1947582. https://doi.org/10.1155/int/1947582

Jebur, S. A., Hussein, K. A., & Hoomod, H. K. (2023). Abnormal behavior detection in video surveillance using Inception-v3 transfer learning approaches. *Iraqi Journal of Computers, Communications, Control and Systems Engineering, 23*(2), 210-221. https://doi.org/10.33103/uot.ijccce.23.2.16

Khan, N., Ullah, F. U. M., Haq, I. U., Khan, S. U., Lee, M. Y., & Baik, S. W. (2021). AB-Net: A novel deep learning assisted framework for renewable energy generation forecasting. *Mathematics, 9*(19), Article 2456. https://doi.org/10.3390/math9192456

Kingma, D. P., & Ba, J. (2014). *Adam: A method for stochastic optimization* [Preprint]. arXiv. https://doi.org/10.48550/arXiv.1412.6980

Kirichenko, L., Radivilova, T., Sydorenko, B., & Yakovlev, S. (2022). Detection of shoplifting on video using a hybrid network. *Computation, 10*(11), Article 199. https://doi.org/10.3390/computation10110199

Kolaib, R. J., & Waleed, J. (2024). Crime activity detection in surveillance videos based on developed deep learning approach. *Diyala Journal of Engineering Sciences, 17*(3), 98-114. https://doi.org/10.24237/djes.2024.17307

Kumar, M., & Biswas, M. (2024). Abnormal human activity detection by convolutional recurrent neural network using fuzzy logic. *Multimedia Tools and Applications, 83*(22), 61843-61859. https://doi.org/10.1007/s11042-023-15904-x

Kumar, M., Patel, A. K., & Biswas, M. (2024). Real-time detection of abnormal human activity using deep learning and temporal attention mechanism in video surveillance. *Multimedia Tools and Applications, 83*(18), 55981-55997. https://doi.org/10.1007/s11042-023-17748-x

Kumar, M., Patel, A. K., Biswas, M., & Shitharth, S. (2023). Attention-based bidirectional-long short-term memory for abnormal human activity detection. *Scientific Reports, 13*(1), Article 14442. https://doi.org/10.1038/s41598-023-41231-0

Kurchaniya, D., & Kumar, S. (2023). Two stream deep neural network based framework to detect abnormal human activities. *Journal of Electronic Imaging, 32*(4), Article 043021. https://doi.org/10.1117/1.jei.32.4.043021

Kurchaniya, D., & Kumar, S. (2024). D-SCAN: Dual stream spatiotemporal channel-wise attention network with point-wise ConvBi-LSTM for activity recognition. *IEEE Transactions on Consumer Electronics, 70*(4), 7244-7251. https://doi.org/10.1109/tce.2024.3425610

Li, Y., Wei, C., & Ma, T. (2019). Towards explaining the regularization effect of initial large learning rate in training neural networks. *Advances in Neural Information Processing Systems, 32*. https://papers.neurips.cc/paper_files/paper/2019/hash/bce9abf229ffd7e570818476ee5d7dde-Abstract.html

Liu, Y., Pu, H., & Sun, D.-W. (2021). Efficient extraction of deep image features using convolutional neural network (CNN) for applications in detecting and analysing complex food matrices. *Trends in Food Science & Technology, 113*, 193-204. https://doi.org/10.1016/j.tifs.2021.04.042

Mahalakshmi, P., & Thenmalar, S. (2023). Feature based training for crime detection using deep learning techniques. In *2023 International Conference on Research Methodologies in Knowledge Management, Artificial Intelligence and Telecommunication Engineering (RMKMATE)* (pp. 1-4). IEEE. https://doi.org/10.1109/rmkmate59243.2023.10369315

Mahareek, E. A., ElSayed, E. K., ElDesouky, N. M., & ElDahshan, K. A. (2024). Detecting anomalies in security cameras with 3d-convolutional neural network and convolutional long short-term memory. *International Journal of Electrical and Computer Engineering, 14*(1), 993-1004. https://doi.org/10.11591/ijece.v14i1.pp993-1004

Mehsen, R. S. (2023). Deep learning algorithm for detecting and analyzing criminal activity. *International Journal of Computing, 22*(2), 248-253. https://doi.org/10.47839/ijc.22.2.3095

Mukto, M. M., Hasan, M., Al Mahmud, M. M., Haque, I., Ahmed, M. A., Jabid, T., Ali, M. S., Rashid, M. R. A., Islam, M. M., & Islam, M. (2024). Design of a real-time crime monitoring system using deep learning techniques. *Intelligent Systems with Applications, 21*, Article 200311. https://doi.org/10.1016/j.iswa.2023.200311

Mumtaz, A., Sargano, A. B., & Habib, Z. (2024). AnomalyNet: A spatiotemporal motion-aware CNN approach for detecting anomalies in real-world autonomous surveillance. *The Visual Computer, 40*(11), 7823-7844. https://doi.org/10.1007/s00371-023-03210-4

Nayak, R., & Chaudhari, J. P. (2023). Real-world anomaly detection in video using spatio-temporal features analysis for weakly labelled data with auto label generation. *International journal of electrical and computer engineering systems, 14*(5), 565-573. https://doi.org/10.32985/ijeces.14.5.8

Nazir, A., Mitra, R., Sulieman, H., & Kamalov, F. (2023). Suspicious behavior detection with temporal feature extraction and time-series classification for shoplifting crime prevention. *Sensors, 23*(13), Article 5811. https://doi.org/10.3390/s23135811

Pallewar, M. G., Pawar, V. R., & Gaikwad, A. N. (2024). Human anomalous activity detection with CNN-LSTM approach. *Journal of Integrated Science and Technology, 12*(1), Article 704. https://pubs.thesciencein.org/journal/index.php/jist/article/view/a704

Park, J., Kim, J., & Han, B. (2023). End-to-end learning for weakly supervised video anomaly detection using absorbing markov chain. *Computer Vision and Image Understanding, 236*, Article 103798. https://doi.org/10.1016/j.cviu.2023.103798

Patwal, A., Diwakar, M., Tripathi, V., & Singh, P. (2023). An investigation of videos for abnormal behavior detection. *Procedia Computer Science, 218*, 2264-2272. https://doi.org/10.1016/j.procs.2023.01.202

Qasim, M., & Verdú, E. (2023). Video anomaly detection system using deep convolutional and recurrent models. *Results in Engineering, 18*, Article 101026. https://doi.org/10.1016/j.rineng.2023.101026

Qasim Gandapur, M., & Verdú, E. (2023). ConvGRU-CNN: Spatiotemporal deep learning for real-world anomaly detection in video surveillance system. *International Journal of Interactive Multimedia and Artificial Intelligence, 8*(4), 88-95. https://doi.org/10.9781/ijimai.2023.05.006

Şengönül, E., Samet, R., Abu Al-Haija, Q., Alqahtani, A., Alturki, B., & Alsulami, A. A. (2023). An analysis of artificial intelligence techniques in surveillance video anomaly detection: A comprehensive survey. *Applied Sciences, 13*(8), Article 4956. https://doi.org/10.3390/app13084956

Sharma, S., Sharma, S., & Athaiya, A. (2020). Activation functions in neural networks. *International Journal of Engineering Applied Sciences and Technology, 4*(12), 310-316. https://doi.org/10.33564/ijeast.2020.v04i12.054

Stewart, T. E., Sr. (2022). *Strategies cybersecurity operators need to improve situation awareness during continuous monitoring performance* [Doctoral dissertation, Colorado Technical University]. ProQuest Dissertations & Theses Global. https://www.proquest.com/dissertations-theses/strategies-cybersecurity-operators-need-improve/docview/2805415237/se-2

Sultana, R., Das, P., Zaman, R., Mazumdar, P., Tasneem, S., Karmakar, A., & Rokunuzzaman, M. (2023). *Intelligent vision-based human activity recognition for surveillance using deep learning algorithm* [Conference paper]. 7th International Conference on Mechanical Engineering and Renewable Energy (ICMERE2023), Chattogram, Bangladesh. https://math.ruet.ac.bd/prangon/bibliography_text

Sultani, W., Chen, C., & Shah, M. (2018). Real-world anomaly detection in surveillance videos. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 6479-6488). IEEE. https://doi.org/10.1109/CVPR.2018.00678

Taha, R. A., Youssif, A. A.-H., & Fouad, M. M. (2024). Transfer learning model for anomalous event recognition in Big Video Data. *Scientific Reports, 14*(1), Article 27868. https://doi.org/10.1038/s41598-024-78414-2

Tan, W., Yao, Q., & Liu, J. (2024). Overlooked video classification in weakly supervised video anomaly detection. In *2024 IEEE/CVF Winter Conference on Applications of Computer Vision Workshops (WACVW)* (pp. 202-210). IEEE. https://doi.org/10.1109/wacvw60836.2024.00029

Taye, M. M. (2023). Understanding of machine learning with deep learning: Architectures, workflow, applications and future directions. *Computers, 12*(5), Article 91. https://doi.org/10.3390/computers12050091

Turaga, P., Chellappa, R., Subrahmanian, V. S., & Udrea, O. (2008). Machine recognition of human activities: A survey. *IEEE Transactions on Circuits and Systems for Video Technology, 18*(11), 1473-1488. https://doi.org/10.1109/tcsvt.2008.2005594

Ul Amin, S., Kim, Y., Sami, I., Park, S., & Seo, S. (2023). An efficient attention-based strategy for anomaly detection in surveillance video. *Computer Systems Science and Engineering, 46*(3), 3939-3958. https://doi.org/10.32604/csse.2023.034805

Ullah, F. U. M., Khan, N., Hussain, T., Lee, M. Y., & Baik, S. W. (2021). Diving deep into short-term electricity load forecasting: Comparative analysis and a novel framework. *Mathematics, 9*(6), Article 611. https://doi.org/10.3390/math9060611

Ullah, W., Ullah, A., Hussain, T., Muhammad, K., Heidari, A. A., Del Ser, J., Baik, S. W., & De Albuquerque, V. H. C. (2022). Artificial intelligence of things-assisted two-stream neural network for anomaly detection in surveillance Big Video Data. *Future Generation Computer Systems, 129*, 286-297. https://doi.org/10.1016/j.future.2021.10.033

Vosta, S., & Yow, K.-C. (2022). A CNN-RNN combined structure for real-world violence detection in surveillance cameras. *Applied Sciences, 12*(3), Article 1021. https://doi.org/10.3390/app12031021

Vosta, S., & Yow, K.-C. (2024). KianNet: A violence detection model using an attention-based CNN-LSTM structure. *IEEE Access, 12*, 2198-2209. https://doi.org/10.1109/access.2023.3339379

Wu, P., Pan, C., Yan, Y., Pang, G., Yan, Q., Wang, P., & Zhang, Y. (2026). Deep learning for video anomaly detection: A review. *IEEE Transactions on Neural Networks and Learning Systems, 37*(7), 3010-3030. https://doi.org/10.1109/tnnls.2025.3647892

Yadav, R., Halder, R., & Banda, G. (2024). Masked autoencoders for spatial–temporal relationship in video-based group activity recognition. *IEEE Access, 12*, 132084-132095. https://doi.org/10.1109/access.2024.3457024

You, K., Long, M., Wang, J., & Jordan, M. I. (2019). *How does learning rate decay help modern neural networks?* [Preprint]. arXiv. https://doi.org/10.48550/arXiv.1908.01878