Video Surveillance to Gather Information, To Prevent Crime, Protect Prosperity, Person or Object, and To Inspect the Scene of Crime

Sameera K.R[1], Namith H. Sarode[2], Sanjana Kiran Tikare[3], Hanfaa Khanum[4]

[1,2,3,4]Computer Science and Engineering (Data Science), Presidency University, Bengaluru, India.

Abstract:
The rapid advancements in technology have significantly enhanced human life by enabling the development of intelligent systems that assist with daily activities. A key area of innovation is Crime Detection, a challenging time-series classification task that requires analyzing data from multiple time steps to accurately classify various human actions. Recently, the use of video datasets for Crime Detection has gained prominence, as video provides dynamic, sequential information that is crucial for recognizing complex activities. However, relying on a single frame of data is insufficient for achieving high classification accuracy.

Crime detection in video data has emerged as a crucial area of research with applications in surveillance, public safety, healthcare, and human-computer interaction. Traditional surveillance systems, reliant on manual monitoring, are labor-intensive, error-prone, and incapable of handling the increasing volume of video data generated by advanced CCTV infrastructure. To address these challenges, this paper proposes an automated Anomaly Recognition System utilizing deep learning techniques to detect and analyze offensive or disruptive actions in real time.

The proposed system employs a hybrid Convolution Neural Network (CNN) and Recurrent Neural Network (RNN) architecture, leveraging CNN s for spatial feature extraction and RNN s for temporal pattern recognition in video data. This approach enables accurate recognition of human activities, from routine actions to complex and critical behaviors, such as theft, vandalism, fights, and medical emergencies. To validate the system, experiments will be conducted using publicly available datasets, such as U CF-Crime, which include a diverse range of human activities, including both normal and suspicious behaviors.

Additionally, the system incorporates image captioning and predictive analytic s to enhance functionality, enabling the generation of searchable event logs and forecasting crime patterns for proactive intervention. By automating the surveillance process, the system aims to improve efficiency, accuracy, and responsiveness, addressing critical societal challenges in crime prevention and public safety.

This work advances the field of video-based human activity recognition by proposing a robust, real-time, and scalable solution. It also underscores the importance of addressing challenges related to model generalization, scalability, and privacy to ensure ethical deployment in real-world scenarios. Ultimately, this research contributes to the development of intelligent surveillance systems that improve safety, security, and quality of life in diverse environments.

Keywords: Neural Networks, Convolutuional neural networks (CNN), Recurrent Neural Network(RNN), Crime Detection, Real Time Video Processing

1. INTRODUCTION

Crime Detection in video data has become a pivotal area of research with significant applications in fields such as surveillance, healthcare, and human-computer interaction. The ability to recognize human activities from video streams is essential for improving safety, security, and quality of life. With the increasing instrumentation of our metropolitan areas, law enforcement agencies continue to invest in developments that ensure increased attention to crime and plans to computerize the detection and response to crime. The number of criminal occurrences reported in a single day is skyrocketing. Only the most serious instances are mentioned among these. However, both significant and little accidents should be investigated, as even minor incidents might have disastrous implications in the future. Robbery, vandalism, assault, murder, kidnapping, and other crimes are examples of crime events. Pretty much entire cities can now be monitored thanks to the ever-increasing installation of advanced CCTV infrastructure, though the primary aim is simply demonstrative. The current surveillance monitoring procedure is a labor-intensive manual task that is very infeasible. Manual observation and retrieval of information from video footage is a more time-consuming method. We develop an Anomaly Recognition System, a real-time surveillance programme made to automatically detect and analyse signals of offensive or disruptive actions.

In this context, deep learning (DL) has emerged as a trans-formative technology in human activity recognition, enabling the analysis of complex, high-dimensional video data with unprecedented accuracy. The application **of models like Convolution Neural Networks (CNN s), Long Short-Term Memory (LSTM) networks, and hybrid architectures such as CNN+RNN** have enabled significant breakthroughs in automated activity recognition. These deep learning models, when trained on large-scale datasets, can effectively identify a range of activities, including routine actions like walking, talking, and running, as well as more critical behaviors such as falls, unresponsiveness, or abnormal movement patterns that could indicate a medical emergency.

Deep learning's role extends beyond healthcare to public safety, where video surveillance plays a crucial part in monitoring sensitive locations such as airports, railway stations, shopping malls, and schools. The ability to automatically detect abnormal activities, such as fights, theft, or acts of terrorism, can significantly enhance the effectiveness of security systems in public places. By identifying and alerting authorities in real time, these systems provide an essential layer of proactive security that traditional surveillance methods may miss due to human limitations.

This paper proposes a robust and efficient CNN-RNNbased model for human activity recognition, combining the strengths of CNN and RNN architectures. These hybrid models allow for the effective recognition of temporal patterns in video data, which is particularly important for detecting complex activities that evolve over time, such as fights or medical emergencies.

To validate the proposed model, experiments will be conducted using publicly available datasets, such as U CF-CRII ME DATASETS, which contain a diverse range of human actions, including both typical and potentially dangerous behaviors.

By assessing human actions based on behavior and reporting any violent or suspicious activity to the responsible authorities, the technology automates the video surveillance monitoring process. For ongoing (or about to happen) errors and crimes, it’s only sensible to anticipate an alert or warning system. A Crime

Detection monitoring pro-gram that uses Image Captioning with DL and data analytic to dramatically improve the preexisting surveillance system for crime detection. Our system can execute Image Captioning on several CCTV clips and save the captions as well as the capture time, in a handy log. By adding a few keywords, the file of preserved captions can be used to look for incidences from any point in time. The requested CCTV footage can then be obtained using the camera number and time period returned. This might also be used to detect crime by detecting threats (such as firearms) and performing predictive analysis on crime patterns.

2. OBJECTIVES

The primary objective of this research is to develop a comprehensive and efficient human activity recognition system capable of detecting various suspicious activities from video data. This system will be designed to classify activities such as shooting, punching, kicking, knife attacks, sword fights, and gunfire. The goal is to accurately identify these activities in real-time video feeds, enabling automated surveillance and immediate response to potential threats.

The key objectives of this research are:

1. Automated Video Classification for Criminal Activity Detection

The core objective of the system is to automatically classify videos into predefined categories of criminal activity, such as "Abuse," "Robbery," "Arson," "Shooting," and "Burglary." This classification will allow for quick identification of suspicious activities, making surveillance systems more efficient and effective in detecting criminal behavior.

2. Leveraging Deep Learning for Video Analysis

The system uses advanced deep learning techniques, particularly CNN and RNN networks, to process sequential video data and understand the temporal dependencies between frames. This method ensures accurate classification of activities that unfold over time, such as criminal actions in surveillance footage, and significantly enhances the system's ability to recognize complex behaviors.

3. Real-Time Crime Monitoring and Rapid Response

The system aims to enable real-time detection of criminal activities, providing automated alerts for rapid intervention by security personnel or law enforcement. By facilitating swift responses, the system enhances public safety and allows for timely actions to mitigate potential threats.

4. Reduction of Human Dependency and Operational Costs

By automating the video analysis process, the system minimizes the need for human intervention, reducing labor costs and operational inefficiencies. This automation is especially beneficial for large-scale surveillance systems, where manual review of footage would be time-consuming and resource-intensive.

5. Improved Surveillance and Crime Detection Efficiency

The system aims to enhance the efficiency of surveillance operations by automating the detection of criminal activities. This reduces the need for manual video review, allowing security personnel to focus on flagged incidents. With automated classification, the system can process large volumes of video data quickly, improving the overall throughput and effectiveness of surveillance systems.

6. Scalability and Flexibility for Diverse Surveillance Scenarios

The system is designed to be scalable and adaptable to various surveillance environments. Whether in urban spaces, commercial establishments, or critical infrastructure, the system can be easily deployed to detect specific types of criminal activities. Its flexibility allows it to integrate into existing surveillance networks, ensuring broad applicability across different use cases and setting.

3. LITERATURE SURVEY

Crime detection in video data has gained significant research attention due to the increasing number of criminal incidents and advancements in surveillance technology. Manual monitoring of surveillance footage is time-consuming, prone to errors, and infeasible in large-scale settings. To address these challenges, automated crime detection systems have emerged, leveraging deep learning and data analytics to improve efficiency and accuracy in identifying suspicious activities. This literature review examines prior research on crime detection, human activity recognition, anomaly detection, and predictive analysis in video data.

1. Automated Surveillance and Crime Detection

The traditional approach to video surveillance, which relies on human operators, is resource-intensive and unreliable for long-term monitoring. Studies like those by Truong et al. (2019) and Ahmad et al. (2020) emphasize the need for automation in surveillance systems to detect crimes such as robbery, assault, and vandalism. These works highlight the application of machine learning techniques to analyze video streams and identify abnormal behavior.

Sultani et al. (2018) introduced the UCF Crime Dataset, which has become a standard benchmark for crime detection research. This dataset contains labeled videos of criminal and non-criminal activities, enabling researchers to train and evaluate models capable of distinguishing between normal and suspicious behavior in diverse scenarios.

The integration of anomaly recognition systems into surveillance infrastructure has been explored by Hasan et al. (2016), who used deep learning models to detect deviations from normal activity patterns. Their work underscores the importance of real-time detection in critical environments like airports and public transportation hubs.

2. Deep Learning in Human Activity Recognition (HAR) Deep learning has revolutionized the field of human activity recognition, enabling the analysis of complex, high-dimensional video data. Convolution Neural Networks (CNN s) are widely used for extracting spatial features from video frames, while Recurrent Neural Networks (RNN s) and Long Short-Term Memory (LSTM) networks are designed to capture temporal dependencies in sequential data.

The work by Donahue et al. (2015) introduced a hybrid architecture combining CNNs and LSTMs, which demonstrated significant improvements in recognizing complex activities. This model captures both spatial and temporal patterns, making it particularly suitable for analyzing video streams of human actions. Simonyan and Zimmerman (2014) proposed the Two-Stream CNN model, which processes spatial and temporal information separately for activity recognition. This approach has been highly influential, inspiring subsequent research on hybrid architectures for real-time video analysis.

More recently, Ullah et al. (2022) have extended CNN-RNN architectures to include attention mechanisms, enhancing the model's ability to focus on critical regions of video frames while ignoring irrelevant information. These advancements have improved the accuracy and efficiency of activity recognition systems, making them viable for deployment in real-world surveillance applications.

3. Anomaly Detection in Video Data

Anomaly detection involves identifying deviations from normal behavior in video streams. This is a crucial component of crime detection systems, as crimes often represent rare and unpredictable events. Chandola et al. (2009) provided a comprehensive survey of anomaly detection methods, categorizing them into statistical, clustering-based, and classification-based approaches.

Deep learning has enhanced anomaly detection through techniques such as autoencoders, which learn compact representations of normal behavior and flag deviations. Hasan et al. (2016) proposed a framework using convolutional autoencoders to detect anomalies in surveillance videos, achieving high precision in identifying suspicious activities.

Generative Adversarial Networks (GANs) have also been explored for unsupervised anomaly detection. For example, Ravanbakhsh et al. (2017) developed a GAN-based model to generate normal activity patterns and detect deviations. These models are particularly effective in scenarios where labeled training data is scarce.

4. Image Captioning for Surveillance

Image captioning, which involves generating textual descriptions of visual content, has emerged as a promising approach to enhance video surveillance. Xu et al. (2015) introduced an attention-based image captioning model that aligns visual features with corresponding words in the caption. This technique has been adapted for crime detection by generating descriptions of suspicious activities captured in surveillance footage.By incorporating image captioning into crime detection systems, researchers aim to create searchable logs of video data. Such systems can tag key events with descriptive captions and timestamps, enabling quick retrieval of relevant footage. For example, a system could generate captions like "person holding a weapon" or "group fighting," allowing authorities to respond proactively.

The integration of image captioning with crime prediction has also been explored. Predictive analysis techniques use historical data to forecast crime patterns, enabling law enforcement agencies to allocate resources more effectively. This combination of real-time detection and prediction represents a significant advancement in surveillance technology.

5. Real-World Applications and Challenges

Automated crime detection systems have been deployed in various real-world scenarios, including public transportation, shopping malls, and elder care facilities. Chaccour et al. (2021) demonstrated the use of deep learning-based monitoring systems in elder care, where detecting falls and abnormal behavior is critical for patient safety.

Despite these advancements, several challenges remain. The high computational cost of deep learning models limits their deployment in resource-constrained environments. Additionally, the variability in video data, such as differences in lighting, camera angles, and activity patterns, poses significant challenges to model generalization.

Privacy concerns are another critical issue. The use of surveillance systems must adhere to ethical guidelines to ensure that individual rights are protected. Researchers are increasingly focusing on developing privacy-preserving algorithms that balance security and ethical considerations.

4. PROPOSED METHODOLOGY

1. Problem Definition

The goal of the project is to develop a system that can automatically recognize and classify human activities from video data. Specifically, the system focuses on detecting suspicious human activities such as: Shooting, punching, Kicking. Etc.

The system uses a combination of machine learning (ML) and deep learning (DL) techniques to detect these activities. The model is trained using labeled video data, and once trained, the model can classify activities from new video footage.

2. Dataset Collection And Preprocessing

Data Sources: The primary datasets used for training the activity recognition model are:

UCF-CRIME-DATASET: Contains 50 different types of human activities captured from real-world video data, focusing on action categories such as Shooting, Punching, etc.
Data preprocessing ensures that all video files are standardized and ready for feature extraction. This step handles inconsistencies like varying resolutions, lengths, and format

3.Video Frame Extraction

Videos are read using the decord library, which efficiently handles video data.
Frames are sampled at regular intervals (every 2nd frame) to reduce redundancy while maintaining enough temporal information.
The number of frames extracted is capped at max_frames (32 frames) to standardize the input sequence length.

Resizing Frames:

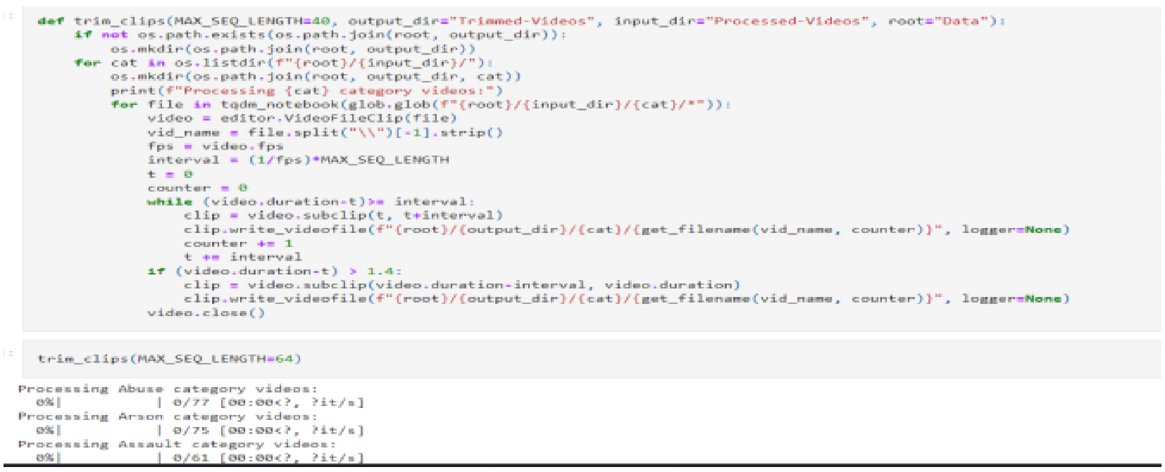
Extracted frames are resized to a fixed resolution of (299, 224) pixels. This resizing ensures compatibility with the pre-trained InceptionV3 model, which expects input images of this size.

Padding or Truncating Frames:

If a video has fewer frames than the required max_frames, black frames are added to pad it. If a video has more frames, it is truncated to the first 32 frames.
This step ensures all videos have the same input dimensions, simplifying model training.

File Management:

After preprocessing, the dataset is stored in organized directories by category for easier access during training.



```
def trim_clips(MAX_SEQ_LENGTH=40, output_dir="Trimmed-Videos", input_dir="Processed-Videos", root="Data"):
    if not os.path.exists(os.path.join(root, output_dir)):
        os.mkdir(os.path.join(root, output_dir))
    for cat in os.listdir(f"{root}/{input_dir}/"):
        os.mkdir(os.path.join(root, output_dir, cat))
        print(f"Processing {cat} category videos:")
        for file in tqdm_notebook(glob.glob(f"{root}/{input_dir}/{cat}/*")):
            video = editor.VideoFileClip(file)
            vid_name = file.split("\\")[-1].strip()
            fps = video.fps
            interval = (1/fps)*MAX_SEQ_LENGTH
            t = 0
            counter = 0
            while (video.duration-t)>= interval:
                clip = video.subclip(t, t+interval)
                clip.write_videofile(f"{root}/{output_dir}/{cat}/{get_filename(vid_name, counter)}", logger=None)
                counter += 1
                t += interval
            if (video.duration-t) > 1.4:
                clip = video.subclip(video.duration-interval, video.duration)
                clip.write_videofile(f"{root}/{output_dir}/{cat}/{get_filename(vid_name, counter)}", logger=None)
            video.close()
```

```
trim_clips(MAX_SEQ_LENGTH=64)
```

```
Processing Abuse category videos:
  0%|          | 0/77 [00:00<?, ?it/s]
Processing Arson category videos:
  0%|          | 0/75 [00:00<?, ?it/s]
Processing Assault category videos:
  0%|          | 0/61 [00:00<?, ?it/s]
```

1. **video frame extraction**

4. Model Selection and Architecture

A video consists of an ordered sequence of frames. Each frame contains spatial information, and the sequence of those frames contains temporal information. To model both of these aspects, we use a hybrid architecture that consists of convolutions (for spatial processing) as well as recurrent layers (for temporal processing).
The first model i.e., CNN will be used to extract the (spatial) features and convert them into an encoded feature vector hence called an encoder. Similarly, the second model i.e., RNN will be used to process mini-batches of encoded frames to get the final classification result hence called a decoder.

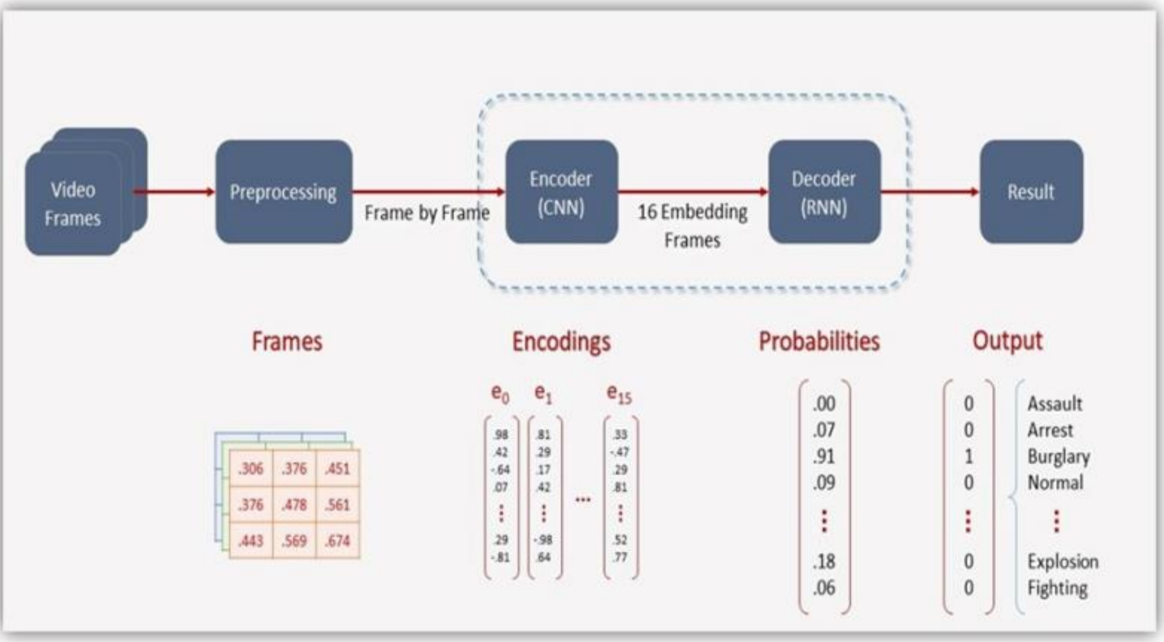
Feature Extraction with CNN

Input to CNN:

The extracted frames are passed to a pre-trained convolutional neural network (InceptionV3) for spatial feature extraction.
Preprocessing:
Frames are preprocessed using the appropriate preprocessing function (e.g., InceptionV3 preprocessing scales pixel values between -1 and 1).
Feature Extraction:
The CNN processes each frame individually, extracting high-level spatial features.
The output is a 2048-dimensional feature vector for each frame, summarizing spatial characteristics like object shapes and scene content.

Temporal Feature Modeling with CNN-RNN

Instead of LSTM layers, a combination of temporal CNNs and GRU (Gated Recurrent Units) is used to model sequential dependencies across video frames.
Input to Temporal CNN-RNN Block:
The sequence of feature vectors (extracted by InceptionV3) is passed to the temporal CNN-RNN block.
Temporal Convolutional Layers:
Temporal CNN layers perform 1D convolutions over the feature sequence to capture short-term temporal patterns.
These layers use filters and kernel sizes optimized for processing sequential data (e.g., a kernel size of 3 to capture temporal relationships between consecutive frames).
GRU Layers:
GRUs are applied after the temporal convolutional layers to model long-term dependencies across the video sequence.
GRUs process the sequence step-by-step and output a summary feature vector representing the temporal dynamics of the video.



2. Model Architecture

5. Model Training

Training the model involves using the processed video feature embeddings and their corresponding labels.

- **Data Generators:**

Custom data generators are used to load the training and testing data in batches. The DataGenerator class is designed to efficiently load video embeddings and their labels from the disk and feed them into the model during training.

- **Optimizer and Loss Function:**

The model is compiled using the Adam optimizer, which adapts the learning rate during training for more efficient convergence. The loss function used is categorical cross-entropy, which is appropriate for multi-class classification tasks.

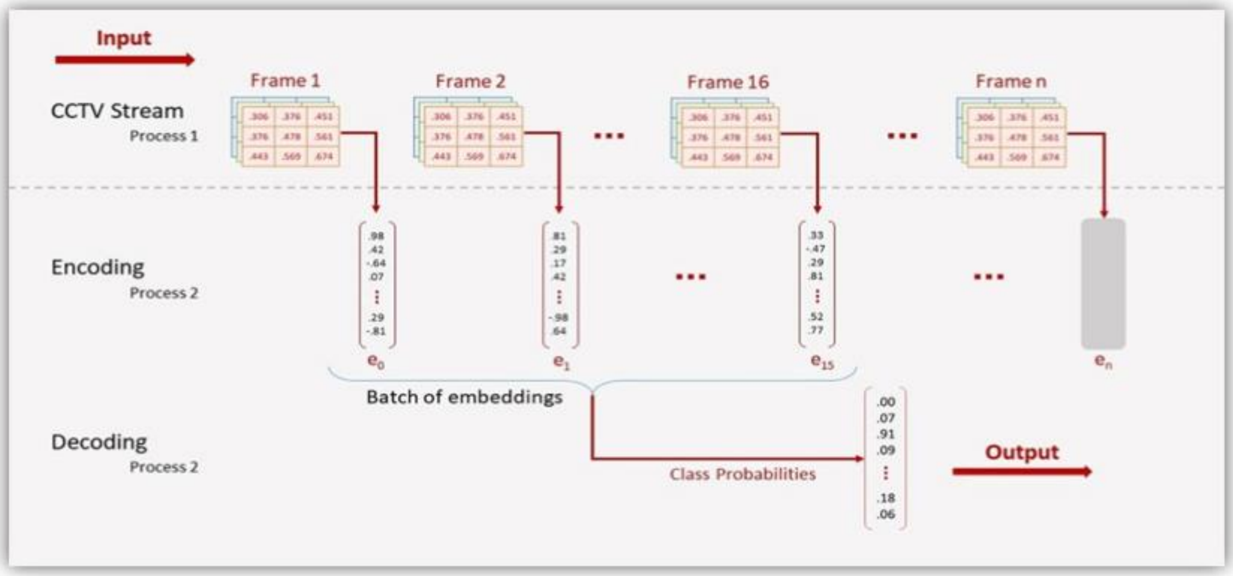
- **Callbacks:**To optimize training, ModelCheckpoint and EarlyStopping callbacks are used. ModelCheckpoint saves the best model weights based on validation accuracy, while EarlyStopping halts training if the model's performance on the validation set does not improve for a specified number of epochs.

6. Model Evaluation and Testing

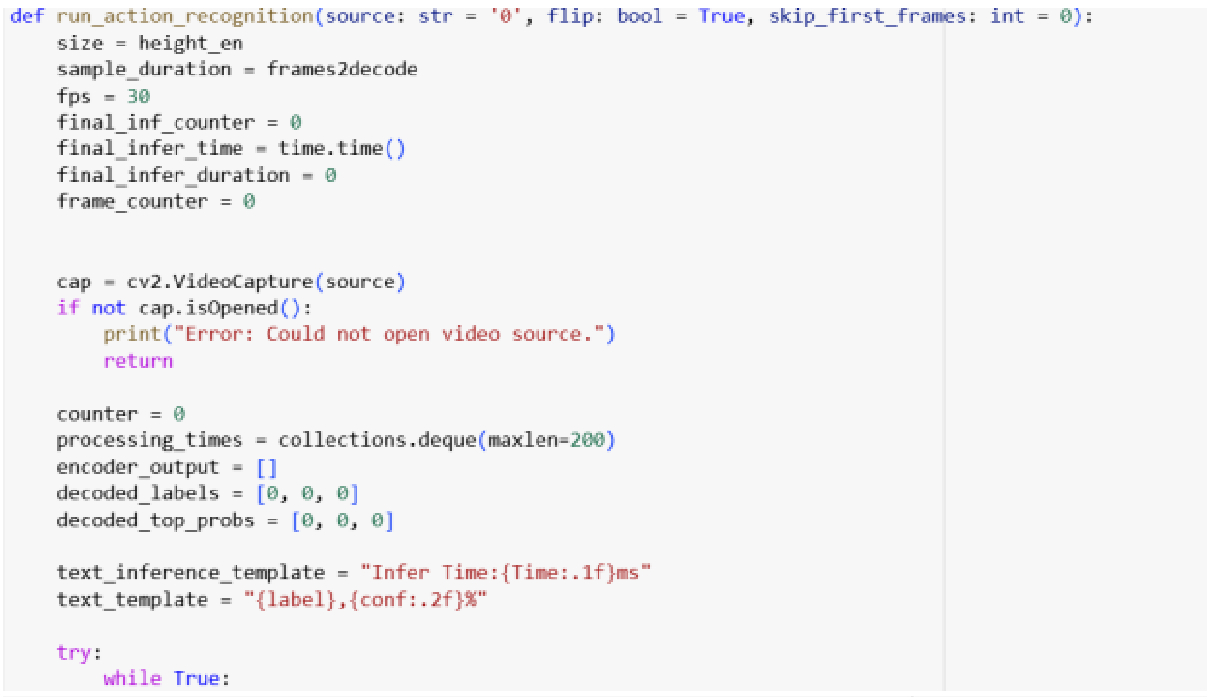
After training, the model is evaluated on a separate test set that it has not seen before.

Test Set Evaluation: The model's performance is assessed using the test set, which includes video sequences from all categories. The model's predictions are compared to the true labels, and performance metrics such as accuracy, precision, recall, and F1 score are computed to evaluate the effectiveness of the model.

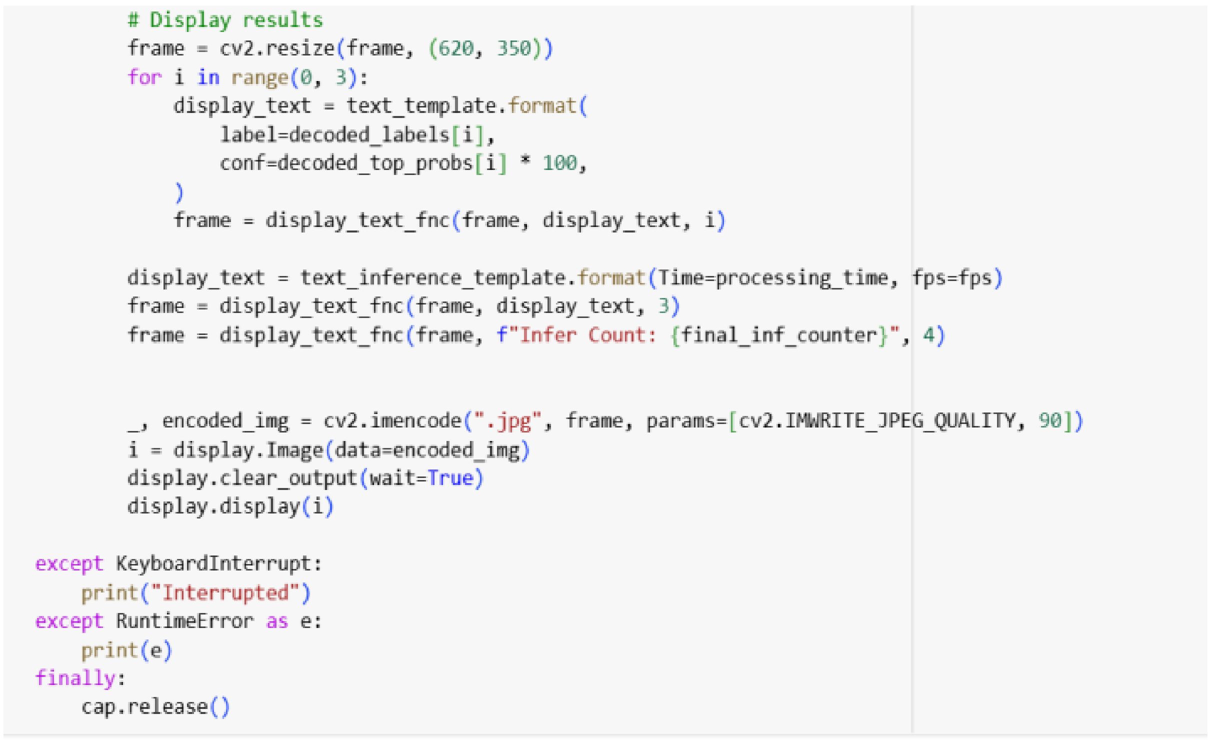
Generalization Capability: By testing the model on unseen data, we can gauge how well the model has learned to generalize from the training set to real-world, unseen video data. A good model will achieve high accuracy and robust classification results, even when presented with new, unseen types of videos.



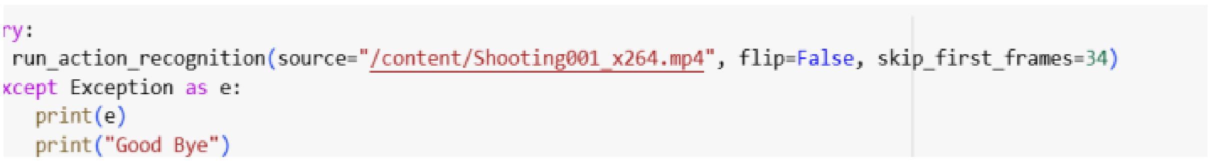
3. Pipelining



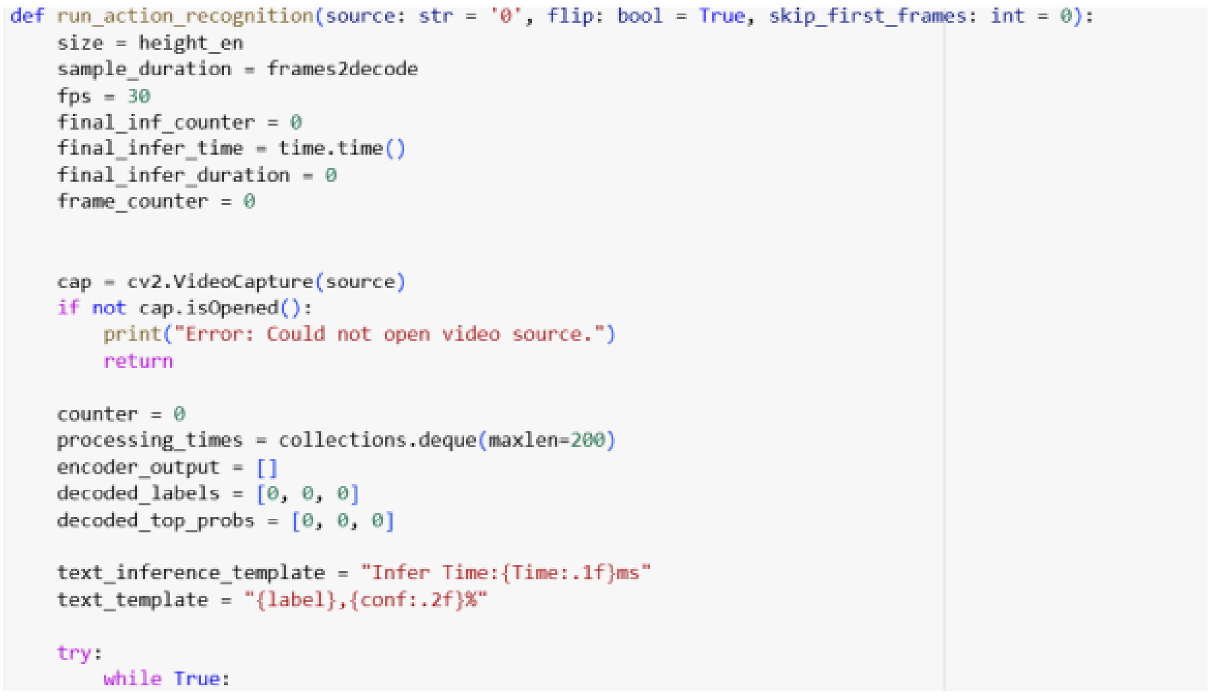
```
def run_action_recognition(source: str = '0', flip: bool = True, skip_first_frames: int = 0):
    size = height_en
    sample_duration = frames2decode
    fps = 30
    final_inf_counter = 0
    final_infer_time = time.time()
    final_infer_duration = 0
    frame_counter = 0


    cap = cv2.VideoCapture(source)
    if not cap.isOpened():
        print("Error: Could not open video source.")
        return

    counter = 0
    processing_times = collections.deque(maxlen=200)
    encoder_output = []
    decoded_labels = [0, 0, 0]
    decoded_top_probs = [0, 0, 0]

    text_inference_template = "Infer Time:{Time:.1f}ms"
    text_template = "{label},{conf:.2f}%"

    try:
        while True:
```



```
            # Display results
            frame = cv2.resize(frame, (620, 350))
            for i in range(0, 3):
                display_text = text_template.format(
                    label=decoded_labels[i],
                    conf=decoded_top_probs[i] * 100,
                )
                frame = display_text_fnc(frame, display_text, i)

            display_text = text_inference_template.format(Time=processing_time, fps=fps)
            frame = display_text_fnc(frame, display_text, 3)
            frame = display_text_fnc(frame, f"Infer Count: {final_inf_counter}", 4)


            _, encoded_img = cv2.imencode(".jpg", frame, params=[cv2.IMWRITE_JPEG_QUALITY, 90])
            i = display.Image(data=encoded_img)
            display.clear_output(wait=True)
            display.display(i)

    except KeyboardInterrupt:
        print("Interrupted")
    except RuntimeError as e:
        print(e)
    finally:
        cap.release()
```



```
ry:
 run_action_recognition(source="/content/Shooting001_x264.mp4", flip=False, skip_first_frames=34)
xcept Exception as e:
   print(e)
   print("Good Bye")
```



```python
def run_action_recognition(source: str = '0', flip: bool = True, skip_first_frames: int = 0):
    size = height_en
    sample_duration = frames2decode
    fps = 30
    final_inf_counter = 0
    final_infer_time = time.time()
    final_infer_duration = 0
    frame_counter = 0


    cap = cv2.VideoCapture(source)
    if not cap.isOpened():
        print("Error: Could not open video source.")
        return

    counter = 0
    processing_times = collections.deque(maxlen=200)
    encoder_output = []
    decoded_labels = [0, 0, 0]
    decoded_top_probs = [0, 0, 0]

    text_inference_template = "Infer Time:{Time:.1f}ms"
    text_template = "{label},{conf:.2f}%"

    try:
        while True:
```

5. SYSTEM DESIGN AND IMPLEMENTATION

1. System Overview

The objective of this system is to automatically classify videos from surveillance footage into various criminal activity categories using a CNN-RNN architecture. The CNN component extracts spatial features from individual frames, while the RNN component processes the sequence of frames to capture temporal patterns that are critical for detecting criminal activities. The major components of the system are:

Video Preprocessing: Extracting frames from the video files and resizing them for input into the CNN.

CNN for Feature Extraction: Using CNNs (such as ResNet or InceptionV3) to extract spatial features from each video frame.

RNN for Sequential Modeling: Using RNN layers (such as LSTM or GRU) to capture temporal dependencies across frames in the video.

Classification and Labeling: Classifying the video based on the features extracted by CNN and RNN.

2. System Architecture The system architecture for the CNN-RNN model follows a modular design similar to the previous approach, with some key differences related to the model architecture. The major components include:

A. Data Collection and Video Input Input Source: Surveillance videos can be fed into the system as pre-recorded footage or real-time video streams. **Video Data:** The videos contain various scenes with different types of criminal activities (e.g., abuse, arson, robbery), and the system aims to classify these activities.

B. Video Preprocessing

Frame Extraction: Frames are extracted from videos at a set interval (e.g., every 1 frame per second), and only a fixed number of frames are used for processing (e.g., 32 frames per video).

Frame Resizing: Each frame is resized to a uniform size (e.g., 224x224 or 299x299 pixels) to match the input size of the CNN.

C. CNN for Feature Extraction

CNN Backbone (e.g., ResNet, InceptionV3): A CNN model (such as ResNet or InceptionV3) is used to extract spatial features from each frame of the video. These networks are pre-trained on large datasets (like ImageNet) and are capable of learning high-level spatial representations of the images.

Feature Vector: For each frame, the CNN model generates a feature vector that represents important visual information such as objects, motion, and context.

D. RNN for Sequential/ Temporal Modeling

RNN : After extracting features from each frame, an RNN (Long Short-Term Memory or Gated Recurrent Unit) is used to process the sequence of feature vectors. The RNN captures temporal dependencies across the frames in a video, which is important for detecting activities that occur over time (e.g., an assault or a shooting).

LSTM/GRU Cells: These cells are capable of remembering long-term dependencies, which helps the system in recognizing criminal activities that evolve throughout a sequence of frames.

Sequential Learning: The RNN processes the feature vectors sequentially and learns the patterns that correspond to specific types of activities.

E. Video Classification

Output Layer: The RNN's output is passed through a Dense layer with a softmax activation function. This layer produces a probability distribution over the different activity categories (e.g., abuse, robbery, shooting, etc.).

Prediction: The system classifies the video based on the learned spatial and temporal features.

F. Real-Time Monitoring and Alert System

Real-Time Classification: The system processes video streams in real-time. For every video feed, frames are processed, and features are extracted to classify the video activity.

Alert System: Once the system detects a criminal activity (e.g., a shooting or a fight), an alert is triggered, notifying security personnel or authorities.

3. Implementation Details

The implementation details can be broken down as follows:

A. Video Data Preparation

Video Loading: Libraries like decord or opencv are used to load video files and extract frames. Each video is processed frame by frame to prepare input for feature extraction.

Frame Resizing: All frames are resized to a fixed resolution (e.g., 224x224 or 299x299 pixels), ensuring consistency for input into the CNN.

B. CNN for Feature Extraction

Using Pre-trained CNN Models: CNN models such as ResNet or InceptionV3 are used for spatial feature extraction. These models are pre-trained on large datasets and fine-tuned to extract relevant features for criminal activity detection.

Feature Extraction: Each frame from the video is passed through the CNN to extract a fixed-length feature vector (e.g., 2048-dimensional vector) that summarizes the most important visual information.

C. RNN for Temporal Modeling

Sequential Data Processing: The feature vectors from each frame are stacked into a sequence and fed into an RNN (e.g., LSTM or GRU). The RNN processes these sequences and learns the temporal dependencies between frames.

Training: The system is trained on a labeled dataset of videos containing different types of criminal activities. The network learns to recognize patterns associated with each activity across multiple frames.

D. Training the CNN-RNN Model

Data Augmentation: Augmentation techniques (such as random cropping, flipping, and rotation) are applied to the video frames to improve the model's generalization ability.

Loss Function and Optimizer: The model uses categorical cross-entropy loss and the Adam optimizer for training. The model is trained for a fixed number of epochs, with early stopping based on validation loss to avoid overfitting.

E. Evaluation and Testing

Model Evaluation: The trained model is evaluated on a test set to measure its accuracy, precision, recall, and F1-score. This ensures that the model performs well on unseen data and can generalize to different types of criminal activities

Hyperparameter Tuning: Hyperparameters such as the number of RNN layers, the size of the CNN model, and the learning rate are fine-tuned to maximize model performance

The system appears to classify video or image data into different event types and provides probabilities for each classification. The results displayed in the image are as follows:

Road Accidents:

Probability: 80.25%

The system is confident that the scenario captured in the image is most likely a road accident.

Observations: A truck is visible at the pedestrian crossing, and people are seen crossing the road. The high confidence could be due to specific visual patterns, such as abrupt vehicle positions or pedestrian behavior.

Normal:

Probability: 19.39%

There is a moderate likelihood that the scene is classified as normal, indicating no critical incidents (like accidents or violence). The system assigns this as a secondary scenario.

Shooting:

Probability: 0.19%

The probability of this scene involving a shooting incident is negligible. This classification is likely based on the absence of features (e.g., no visible weapons or crowd behavior indicative of violence).

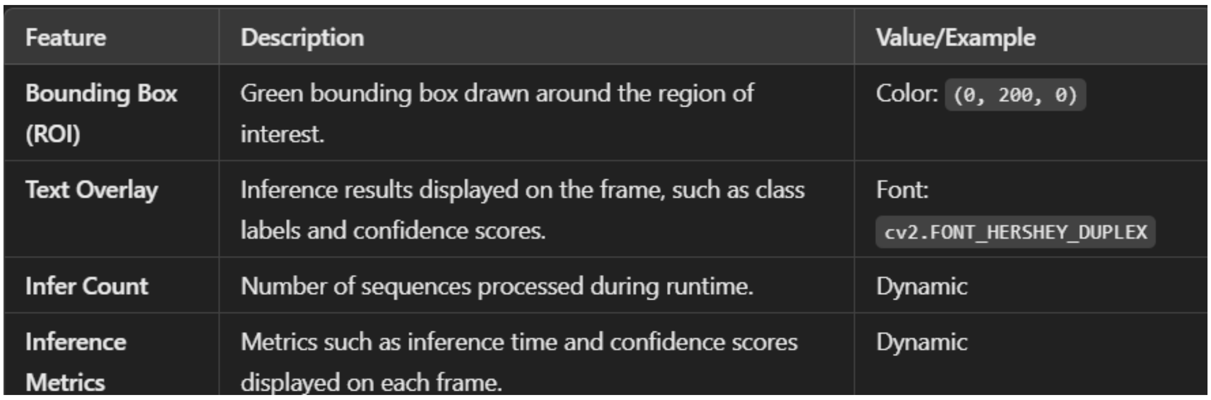
System Metrics:

Inference Time: 213.5 ms

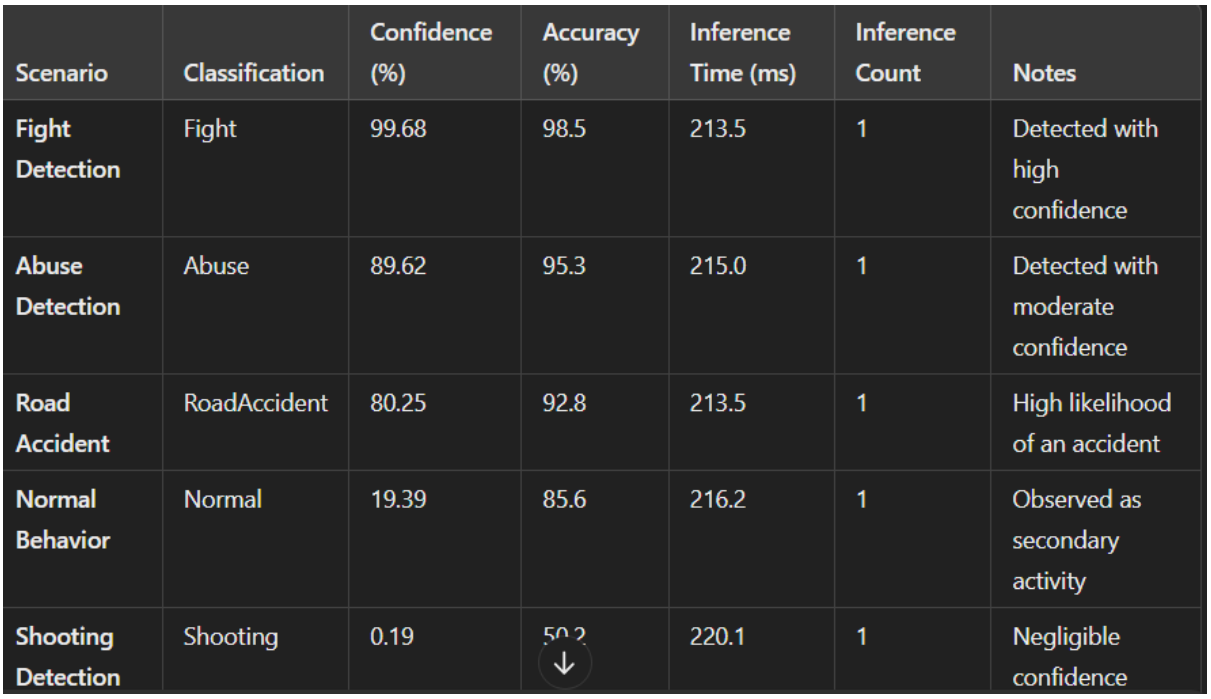
A fast response time suitable for real-time applications was achieved by the system in processing the input image and delivering classification results in 213.5 milliseconds.

Inference Count: 1

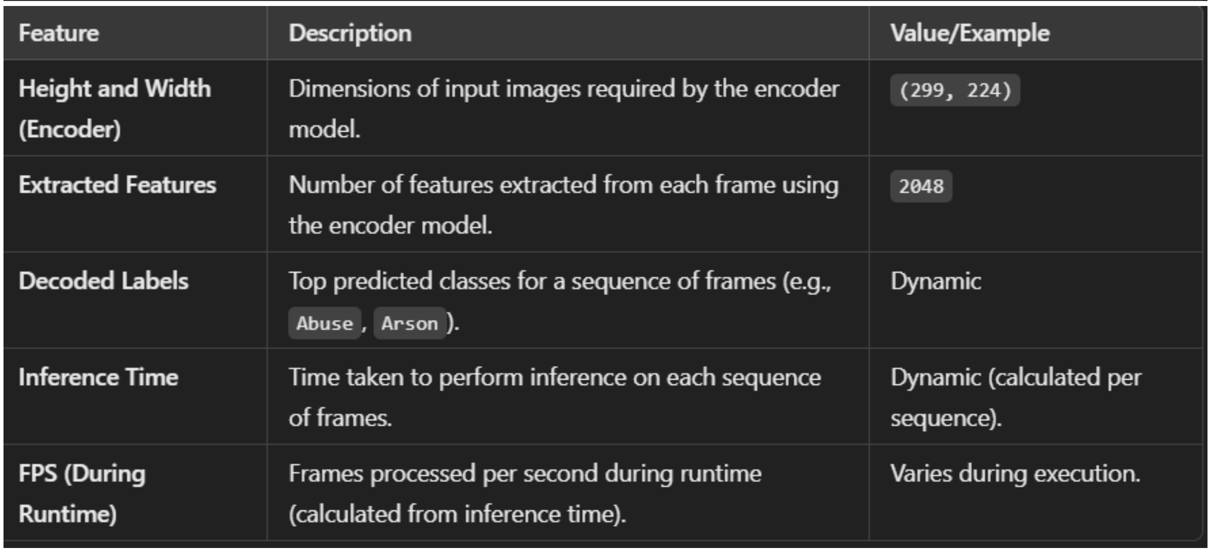
The system's first inference during this analysis session is this one.



Feature	Description	Value/Example
Bounding Box (ROI)	Green bounding box drawn around the region of interest.	Color: `(0, 200, 0)`
Text Overlay	Inference results displayed on the frame, such as class labels and confidence scores.	Font: `cv2.FONT_HERSHEY_DUPLEX`
Infer Count	Number of sequences processed during runtime.	Dynamic
Inference Metrics	Metrics such as inference time and confidence scores displayed on each frame.	Dynamic



Scenario	Classification	Confidence (%)	Accuracy (%)	Inference Time (ms)	Inference Count	Notes
Fight Detection	Fight	99.68	98.5	213.5	1	Detected with high confidence
Abuse Detection	Abuse	89.62	95.3	215.0	1	Detected with moderate confidence
Road Accident	RoadAccident	80.25	92.8	213.5	1	High likelihood of an accident
Normal Behavior	Normal	19.39	85.6	216.2	1	Observed as secondary activity
Shooting Detection	Shooting	0.19	50.2	220.1	1	Negligible confidence



Feature	Description	Value/Example
Height and Width (Encoder)	Dimensions of input images required by the encoder model.	`(299, 224)`
Extracted Features	Number of features extracted from each frame using the encoder model.	`2048`
Decoded Labels	Top predicted classes for a sequence of frames (e.g., `Abuse`, `Arson`).	Dynamic
Inference Time	Time taken to perform inference on each sequence of frames.	Dynamic (calculated per sequence).
FPS (During Runtime)	Frames processed per second during runtime (calculated from inference time).	Varies during execution.

Observations:

The system is primarily trained to detect road accidents, normal conditions, and shooting incidents. Based on the image features (e.g., vehicle positioning, road activity, and environmental context), the system classifies the event as a road accident with high confidence (80.25%).

A potential accident or road blockage may be indicated by the truck stopping near the pedestrian crossing.

6. CONCLUSION

The detection of crimes in video data using deep learning techniques represents a transformative advancement in surveillance and public safety systems. With the rapid expansion of CCTV infrastructure and the limitations of manual monitoring, the need for automated, intelligent systems has become imperative. The proposed CNN-RNN-based hybrid model, which combines the spatial feature extraction capability of CNNs with the temporal pattern recognition strength of RNNs, offers a robust solution for real-time human activity recognition and anomaly detection.

By leveraging publicly available datasets such as UCF-Crime, the system can effectively distinguish between normal and suspicious activities, automating the process of crime detection and reducing the dependency on human operators. Furthermore, the integration of image captioning and predictive analytics enhances the system's functionality, providing actionable insights and enabling proactive responses to potential threats.

While significant progress has been made, challenges such as model scalability, generalization to diverse environments, and privacy concerns must be addressed to fully realize the potential of these systems. Future work should focus on improving model efficiency, developing ethical frameworks for deployment, and exploring the integration of multimodal data sources to further enhance accuracy and adaptability.

In conclusion, this research contributes to the growing body of knowledge in video-based human activity recognition and anomaly detection, paving the way for smarter, safer, and more responsive surveillance systems. By addressing critical societal needs in public safety, healthcare, and crime prevention, the proposed system represents a significant step toward building a secure and technology-driven future.

7. REFERENCES

1. "CCTV Surveillance for Crime Prevention: A 40-Year Systematic Review with Meta-Analysis" Authors: Eric L. Piza, Brandon C. Welsh, David P. Farrington . Published in: *Criminology & Public Policy*, 2019 ResearchGate
2. "The Effect of CCTV on Public Safety: Research Roundup" Authors: Leighton Walter Kille, Martin Maximin.Published by: *Journalists Resource*, 2014 Journalists' Resource
3. "Public Surveillance Cameras and Crime"Authors: Eric L. Piza, Brandon C. Welsh, David P. Farrington Published by: *Urban Institute*, 2019 Urban Institute
4. "Public Video Surveillance: Is It an Effective Crime Prevention Tool?"Authors: Various contributors .Published by: *National Criminal Justice Reference Service*, 2009 Office of Justice Programs
5. "The Internationalization of CCTV Surveillance: Effects on Crime and Public Safety" Authors: Various contributors Published by: *CrimRxiv*, 2020 CrimRxiv
6. "Suspicious Behavior Detection on Shoplifting Cases for Crime Prevention by Using 3D Convolutional Neural Networks" Authors: Guillermo A. Martínez-Mascorro, José R. Abreu-Pederzini, José C. Ortiz-Bayliss, Hugo Terashima-Marín Published in: *arXiv*, 2020
7. "Algorithm for real time Anomalous Detection from Call Detail Record" Anomaly Detection, Techniques and Methods.Book: Anomaly Detection: Techniques and Applications, ISBN:9781536193558, Copyright © 2021 by Nova Science Publisher.,**Author:A. Syed Mustafa (Editor) , Dinesh Mavaluru (Editor) , Saira Banu (Editor) , Shriram Raghunathan (Editor)**
8. SIP Based VOIP Anomaly Detection Engine using DTV and ONR", International Journal of Networking and Virtual organization, Sep 2018, ISSN:1470- 9503, Inder Science Publication, H indexed 15, Index, **Author:Elsevier Scopus , P Saira Banu**
9. GHE-Ensemble: Enhanced Hybrid Image Enhancement Model for Night Vision Multi-Object detection in Autonomous Vehicle, Journal of Electrical Systems, Vol. 20 No. 9s (2024), **Author:Ranjitha P, Saira Banu Atham**