ABSTRACT

This project presents a real-time violence detection system that combines Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks for analyzing video streams and identifying violent events with high accuracy. The work addresses the growing need for smart, automated surveillance systems that can detect aggressive behavior without continuous human supervision. It holds strong academic and social relevance due to increasing safety concerns in public spaces, transportation hubs, educational institutions, and other security-sensitive environments. The system functions under constraints related to real-time processing, varying lighting conditions, and diverse motion patterns.

The methodology includes video frame extraction, preprocessing, spatial feature representation using MobileNetV2, and temporal sequence modeling with LSTM layers, followed by classification through a fully connected prediction layer. A CSV-based logging mechanism records detection outcomes, timestamps, and event details locally. A real-time alert mechanism is integrated through a Telegram Bot to notify users of detected violent activity.

The model was trained and tested on multiple real-life violence datasets and demonstrated high accuracy and low inference delay. Its performance remained consistent across different environments and background complexities. These results indicate the system's suitability for real-time applications on edge devices and standard CPUs.

The work concludes by presenting an efficient and scalable framework for violence detection in modern surveillance systems. Recommendations for future enhancement include exploring transformer-based architectures, enabling multi-camera integration, adopting cloud–edge hybrid deployment, and expanding the dataset size to further improve robustness and real-world applicability.

Keywords: Violence Detection, CNN, LSTM, MobileNetV2, Real-Time Surveillance, Deep Learning

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1 : INTRODUCTION

1.1 Overview

Violence detection in video streams is a challenging problem in computer vision and artificial intelligence. Surveillance cameras generate massive amounts of visual data every second. Relying only on manual monitoring is inefficient, inconsistent, and prone to error. Continuous observation of multiple live CCTV feeds is nearly impossible due to operator fatigue, distraction, and slow response times. As public infrastructure grows, there is a greater need for automated systems that can analyze video content and detect violent activities in real time.

Video-based violence detection requires understanding both the spatial information from individual video frames and the behavior across a sequence of frames. Violent events like fights, assaults, or aggressive movements cannot be identified from single images. They require analyzing movement patterns, human interactions, and contextual clues over time. Modern deep learning advancements, especially Convolutional Neural Networks (CNNs) for spatial modeling and Long Short-Term Memory (LSTM) networks for temporal modeling, have greatly improved the ability to recognize complex activities in video data.

This project aims to develop a Real-Time Violence Detection System that uses a MobileNetV2-based CNN for feature extraction and an LSTM architecture to understand temporal dynamics. The system includes essential components such as preprocessing pipelines, frame extraction, prediction modules, a Telegram Bot for alerts, and CSV-based event logging to ensure fast, reliable, and deployable performance.

1.2 Role Of Artificial Intelligence In Video Analytics

Artificial Intelligence (AI) has changed how we interpret visual data. The vast amount of video data generated by cameras, drones, and surveillance systems cannot be processed effectively using traditional rule-based algorithms. AI allows machines to understand scenes, spot unusual events, and recognize patterns in human behavior by learning from large datasets.

Deep learning models, especially CNNs, have reached human-level accuracy in image classification, object detection, action recognition, and anomaly detection. These abilities let surveillance systems automatically analyze video streams without needing handcrafted feature engineering or constant human oversight.

The rise of Edge AI has made it possible to do real-time inference on local devices without sending data to cloud servers. This method improves privacy, reduces delays, and increases security. It makes AI-powered violence detection suitable for real-world situations where decisions need to be made quickly.

1.3 How Video-Based Violence Detection Works

A video is essentially a continuous sequence of image frames, where each frame is represented as a matrix of pixel values. Violence detection begins by identifying meaningful visual patterns in individual frames and observing how these patterns evolve over time. To achieve this, the video stream is first decomposed into separate frames which undergo preprocessing, including resizing and normalization, to make them suitable for model input. Each frame is then passed through MobileNetV2, a lightweight Convolutional Neural Network (CNN), to extract high-level spatial features such as human posture and object shapes. These extracted features are fed into a Long Short-Term Memory (LSTM) network, which analyzes temporal motion changes and contextual relationships across sequential frames. Based on this combined spatial–temporal understanding, the system classifies the input as either violent or non-violent behavior. Whenever the model detects violence, it immediately generates an alert through the Telegram Bot and records essential information such as prediction probability and timestamp in a log file. This hierarchical processing framework enables the system to accurately interpret both static visual cues and dynamic motion characteristics, making real-time violence detection effective and reliable.

1.4 Why Machine Learning Is Essential For Violence Detection

Rule-based or manually programmed systems cannot manage the complexity and variety of real-world environments. Violence differs greatly across:

- Human poses
- Camera angles
- Background clutter
- Lighting conditions
- Motion intensity
- Multiple actors interacting

Machine learning, especially deep learning, allows for the automatic discovery of patterns from large datasets without human help. Compared to traditional computer vision, deep learning:

- Removes the need for handcrafted features

- Learns directly from raw pixel data
- Deals with noise, motion blur, and occlusion
- Adjusts to changes in the environment
- Reaches high accuracy and low latency

For these reasons, ML-based violence detection systems can effectively and reliably process visual data, making them suitable for real-time monitoring applications.

1.5 Evolution Of Cnn–Based Video Understanding

Convolutional Neural Networks (CNNs) play a vital role in modern computer vision tasks. Their development began with the introduction of LeNet-5 in 1998, but the major breakthrough came with AlexNet in 2012, which achieved remarkable performance in the ImageNet challenge and revolutionized deep learning research. Subsequent architectures such as VGGNet, GoogLeNet, ResNet, and MobileNet have significantly improved accuracy, scalability, and computational efficiency. However, CNNs are designed to capture only spatial information from single images and therefore struggle with tasks involving motion or temporal changes, such as video analysis. To overcome this limitation, researchers introduced methods like 3D CNNs to jointly learn spatial-temporal features, and Recurrent Neural Networks (RNNs), especially Long Short-Term Memory (LSTM) networks, to effectively model sequential patterns and actions over time. A widely adopted solution is the hybrid CNN-LSTM framework, where CNNs extract spatial features from frames and LSTMs analyze temporal relationships across frame sequences. Following this approach, our project integrates MobileNetV2 for efficient spatial feature extraction and LSTM layers for temporal understanding, enabling a lightweight yet powerful real-time violence detection system.

CHAPTER 2 : LITERATURE REVIEW

2.1 Introduction

Violence detection in video streams has emerged as a significant research domain due to the rising demand for intelligent surveillance and automated behavioral monitoring in public spaces. With the proliferation of CCTV systems in urban areas such as metro stations, educational institutions, hospitals, shopping complexes, airports, and residential societies, the volume of video data being generated every second has grown exponentially. Manual monitoring of such large-scale video feeds is almost impossible and often suffers from delays, subjectivity, and fatigue-related errors. This makes the need for automated, real-time, AI-driven video analysis more critical than ever.

Deep learning, particularly Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), has shown promising results in modeling both spatial and temporal patterns in videos. CNNs extract detailed visual features from individual frames, while LSTM networks capture motion, behavior, and sequential dependencies across frames a requirement essential for interpreting violence, which is inherently temporal and dynamic in nature.

While substantial progress has been made, several limitations persist in existing research, such as high computational cost, limited dataset diversity, poor real-time performance, lack of deployable solutions, and absence of alerting mechanisms. Therefore, a comprehensive review of related literature is necessary to understand prior methodologies, identify gaps, and justify the use of a lightweight MobileNetV2 + LSTM architecture integrated with real-time alert systems such as Telegram Bot and CSV logging.

This chapter provides an expanded, structured, and analytical review of existing work in violence detection, video analysis, spatio-temporal modeling, lightweight CNNs, and real-time AI deployment. It highlights methodological advancements, compares performance trade-offs, and clarifies how previous research shaped the direction of this project.

Table 2.1: Review of Key Literature

No.	Author / Year	Title / Focus	Methodology Used	Key Findings	Limitations Identified
1	**Dollar et al., 2005**	Behavior recognition using STIP	STIP features + classifiers	Captures motion changes effectively	Fails in cluttered CCTV scenes
2	**Hassner et al., 2012**	Violence detection using VF	Violent Flow descriptor	Real-time detection possible	Not robust for crowd scenes
3	**Krizhevsky et al., 2012**	ImageNet CNN (**AlexNet**)	Deep CNN learned features	Strong spatial representation	No temporal modeling
4	**Donahue et al., 2015**	**LRCN** hybrid model	CNN + **LSTM**	Strong temporal learning	Heavy, slow for real-time
5	**Sudhakaran & Lanz, 2017**	**CNN–RNN** for violence	CNN + **GRU**	High accuracy	Requires GPU
6	**Tran et al., 2015**	**3D CNN** spatio-temporal	3D convolutions	Captures motion well	Extremely heavy

7	**Ullah et al., 2020**	Light **CNN–LSTM**	**MobileNet** + LSTM	Good speed/accuracy	Limited dataset testing
8	**Simonyan & Zisserman, 2014**	Two-stream **CNN**	RGB + **Optical flow**	High temporal accuracy	Optical flow is slow
9	**RLVS Dataset Authors, 2020**	Real-life violence	Real CCTV videos	High realism	Small dataset
10	**Patel et al., 2021**	Smart surveillance	CNNs	Promising results	No alert integration

Table 1.1 **:** Literature Review Summary Table

2.2 Discussion For Each Literature Source

2.2.1 Dollar et al., 2005 STIP for Behavior Recognition

Dollar et al. pioneered the Spatio-Temporal Interest Points (STIP) method, which marked one of the earliest attempts to identify motion patterns associated with human actions. Their approach relied on detecting abrupt motion variations within small spatiotemporal volumes. While effective in controlled environments, STIP-based methods struggle with real-life CCTV footage where noise, occlusions, camera shake, and dynamic backgrounds are common. This weakness directly influenced the decision to adopt deep learning-based feature extractors in this project.

2.2.2. Hassner et al., 2012 Violent Flow Descriptor

The introduction of the Violent Flow (VF) descriptor aimed to detect violence by analyzing changes in motion magnitude across short video segments. VF's lightweight nature made it suitable for early real-time systems. However, because VF does not incorporate semantic cues, it misinterprets non-violent rapid movements (running, crowd movements) as violence. This highlighted the need for deep feature learning and temporal modeling.

2.2.3. Krizhevsky et al., 2012 AlexNet

AlexNet revolutionized image classification by demonstrating that deep CNNs could automatically learn robust visual features. This significantly advanced the field of computer vision and contributed to the adoption of CNN feature extractors in action recognition. However, AlexNet is computationally heavy, lacks temporal modeling capabilities, and is unsuitable for real-time violence detection. The need for lightweight architectures like MobileNetV2 became evident.

2.2.4. Donahue et al., 2015 LRCN (CNN–LSTM)

This work introduced Long-term Recurrent Convolutional Networks (LRCN), combining CNN-based spatial feature extraction with LSTM-based temporal sequence modeling. This architecture demonstrated significant improvements in activity recognition tasks. The concept directly influenced our project design. However, LRCN's backbone CNNs were heavy, making real-time inference difficult hence our choice of MobileNetV2.

2.2.5. Sudhakaran & Lanz, 2017 Convolutional Recurrent Networks for Violence

This study applied CNN–RNN (GRU) for detecting fights and violent interactions. High accuracy was achieved, but the backbone CNN was computationally expensive. Their system lacked real-time monitoring and alerting features. Our system improves on this by using lightweight MobileNetV2 and adding a Telegram Bot notification system.

2.2.6. Tran et al., 2015 3D CNN (C3D)

Tran's C3D model introduced 3D convolution to jointly model spatial and temporal information. While powerful, 3D CNNs require enormous computing power and large datasets. Their heavy nature makes them impractical for real-time, edge, or CPU-only deployment validating our approach to prefer MobileNetV2 + LSTM.

2.2.7. Ullah et al., 2020 MobileNet + LSTM

This work explored lightweight CNN–LSTM hybrids. Their findings confirmed that using MobileNet as a backbone significantly reduces computation while maintaining strong accuracy. This directly supports the architecture adopted in our project.

2.2.8. Two-stream CNN Models (2014)

Two-stream models use both RGB frames and optical flow. Although they achieve high accuracy, optical flow extraction is slow and unsuitable for real-time. Our system avoids optical flow entirely, improving latency.

2.2.9. Real-Life Violence Dataset (RLVS), 2020

RLVS introduced real CCTV-style footage for training and testing violence detection models. Its inclusion of diverse environments makes it ideal for robust evaluations. We selected this dataset for our project.

2.2.10. Patel et al., 2021 Smart Surveillance

Patel's research demonstrated the potential of DL-based systems for automated surveillance. However, their systems lacked end-to-end integration (alerts, logging), which our project implements.

2.3 Summary Of Findings

A detailed analysis of the reviewed literature reveals several patterns, strengths, limitations, and research gaps in the field of violence detection, video analytics, and deep learning for surveillance. This section provides an expanded, structured, and critical summary of the findings.

2.3.1. Deep learning significantly outperforms classical methods for violence detection.

Earlier works relying on handcrafted features such as STIP, HOG, MBH, or Violent Flow struggle to generalize beyond specific, controlled environments. Sudden illumination changes, varied camera angles, occlusions, and background disturbances easily mislead these feature-based systems. Deep learning models, particularly CNNs and CNN–RNN architectures, demonstrate remarkable ability to learn hierarchical and discriminative patterns automatically. This makes them superior for real-world

CCTV footage, where noise and unpredictability are common. This directly justifies our selection of MobileNetV2 for spatial feature extraction.

2.3.2. CNN-only models excel at static feature extraction but fail to capture temporal dependencies.

CNNs process each frame independently, meaning they cannot interpret the motion dynamics crucial for violent activity detection. Violence is inherently temporal rapid movements, human interactions, and aggressive patterns unfold over time. Techniques like LSTM, GRU, and 3D CNNs emerge as solutions. Among these, LSTMs offer a lightweight and efficient temporal learning mechanism while avoiding the overhead of heavy 3D convolution operations. This validates our use of an LSTM module in combination with MobileNetV2.

3. 3D CNNs deliver excellent spatio-temporal learning but are computationally expensive.

Models like C3D and I3D achieve highly accurate action recognition results by learning motion and spatial context jointly. However, they demand significant GPU resources, large memory footprints, and extensive training times. Deploying such models on real-time surveillance systems, especially on CPU-only architectures, becomes impractical. This reinforces the need for a lighter architecture, which our project achieves using a 2D MobileNetV2 CNN paired with a temporal LSTM layer.

4. Two-stream CNN architectures achieve high accuracy but are unsuitable for real-time systems.

Two-stream models use RGB frames and optical flow to capture appearance and motion. While optical flow enhances motion sensitivity, it introduces considerable computational overhead. Optical flow extraction alone can significantly delay processing, making two-stream networks infeasible for real-time CCTV systems. Our violence detection system bypasses optical flow entirely to maintain low latency.

5. Lightweight CNN architectures are critical for real-world deployment.

Research on MobileNet, ShuffleNet, and SqueezeNet reveals that lightweight models achieve excellent speed–accuracy trade-offs. MobileNetV2, specifically, introduces inverted residuals and linear bottlenecks to drastically reduce parameters and computational cost. This enables real-time

inference even on CPU-only hardware. This aligns directly with the design goal of the proposed violence detection system.

6. CNN–LSTM hybrid models strike the best balance between accuracy, interpretability, and computational efficiency.

Studies combining CNNs with LSTM or GRU layers report improved recognition of complex behaviors. This hybrid approach allows frame-level feature extraction while enabling the model to learn temporal dependencies across video sequences. The success of LRCN and related models confirms the robustness of the CNN–LSTM paradigm.

7. Most existing literature lacks real-time alerting mechanisms and end-to-end surveillance integration.

Academic research often focuses solely on model accuracy and classification performance. However, real surveillance environments require:

- Continuous stream processing
- Real-time alarms
- Logging and storage of detected events
- User-friendly interfaces

Your project addresses this gap by integrating a **Telegram Bot notification system** and **CSV event logging**, making the solution practically deployable.

8. Dataset diversity is limited in earlier works, affecting model generalization.

Several studies rely on lab-collected or single-environment datasets such as Hockey Fights or Movies Dataset. These datasets lack real-world complexity, noise, lighting variations, and camera distortions. Real-Life Violence Dataset (RLVS) attempts to address this, but the community still lacks a large, diverse dataset. This underscores the importance of training with real-world samples and applying augmentation techniques.

9. High-performance models require high-end hardware and are not suitable for edge/CPU deployment.

Many research papers overlook deployment constraints, assuming the availability of GPUs. Real-world CCTV systems often run on CPU-based security servers or embedded devices. Our project fills this gap by using a lightweight MobileNetV2-based pipeline that operates efficiently on CPUs, making it edge-friendly.

10. Few existing works adopt a fully integrated, deployable, end-to-end pipeline.

Most research stops at model training and evaluation. Rarely do solutions include:

- Data preprocessing
- Inference loop
- Streaming pipeline
- Notifications
- Logging
- Hardware-aware optimization

This highlights the novelty and practical strength of your system, which includes all these features.

2.4 Addressing Identified Gaps

Gap 1: Handcrafted features lack robustness in real CCTV environments

Earlier techniques like STIP and Violent Flow fail under noise, crowd movement, or low lighting.

How Our Project Addresses This:

- Uses **MobileNetV2**, which learns robust deep spatial features
- Handles noise, blur, and dynamic backgrounds
- Works effectively in uncontrolled camera environments

Gap 2: Lack of temporal understanding in single-frame CNN models

Violence unfolds over time, making static frame processing insufficient.

Our Solution:

- Adds an **LSTM network** to learn sequential motion
- Captures aggressive movements, pose transitions, and interactions
- Provides reliable temporal classification

Gap 3: Heavy deep learning models are unsuitable for real-time deployment

Models like C3D, I3D, and 3D ResNet require GPUs and high power.

Our Solution:

- MobileNetV2 architecture is highly optimized for CPU inference
- LSTM adds minimal overhead
- Achieves near real-time speed even on standard machines

Gap 4: Absence of alerting systems in previous research

Most literature focuses only on classification accuracy.

Our Solution:

- Integrates a **real-time Telegram Bot alert system**
- Sends messages instantly upon violence detection
- Enables rapid response in real-world scenarios

Gap 5: Lack of logging and record-keeping mechanisms

Real deployments require timestamped logs for security audits.

Our Solution:

- Uses CSV-based logging
- Stores event timestamps, prediction confidence, and status
- Enables traceability and auditing

Gap 6: Limited dataset diversity reduces robustness

Many violence detection datasets lack variation.

Our Solution:

- Uses **Real-Life Violence Dataset** with real CCTV footage
- Includes diverse lighting, backgrounds, angles, and motion patterns
- Improves generalization

Gap 7: No end-to-end deployable solution in literature

Earlier works are not deployable in production settings.

Our Solution:

Includes a full pipeline:

1. Video capture
2. Preprocessing
3. CNN–LSTM prediction
4. Real-time alerting
5. Logging
6. Continuous monitoring loop

2.5 Importance Of This Review To The Project

1. Helped establish the architectural direction

The literature demonstrated the need for a hybrid architecture combining CNN spatial learning with LSTM temporal learning. This guided the project's choice of MobileNetV2 + LSTM.

2. Guided the selection of the lightweight backbone

Studies comparing MobileNet with ResNet, DenseNet, and VGG highlighted MobileNet's superior efficiency for edge devices.

3. Influenced the real-time design goals

The literature emphasized latency, throughput, and computational efficiency. This shaped the project's real-time architecture.

4. Motivated the integration of user notifications

Very few papers include alerting systems. The Telegram Bot was introduced based on this identified gap.

5. Reinforced the importance of dataset diversity

Since many models failed to generalize, this project prioritized using RLVS and augmentations to improve robustness.

6. Helped define evaluation metrics

Latency, accuracy, FPS, and temporal stability were adopted as key metrics inspired by surveyed research.

7. Ensured alignment with state-of-the-art trends

By reviewing hybrid architectures, spatio-temporal models, and lightweight CNNs, the project remained aligned with current best practices.

2.6 Comparison Matrix

Model / Paper	Architecture	Dataset Used	Accuracy	Real-Time?	Limitations
STIP (2005)	**Handcrafted**	**KTH**	**Low**	**Yes**	**Not robust**

Violent Flow (2012)	**Motion-only**	**Hockey**	**Moderate**	**Yes**	**False positives**
AlexNet (2012)	**CNN**	**ImageNet**	**High**	**No**	**Heavy model**
LRCN (2015)	**CNN–LSTM**	**UCF101**	**High**	**No**	**Slow**
3D CNN (2015)	**3D Conv**	**Sports-1M**	**High**	**No**	**Very expensive**
CNN–GRU (2017)	**CNN–GRU**	**Movie Fights**	**High**	**No**	**GPU required**
MobileNet–LSTM (2020)	**Lightweight Hybrid**	**Custom**	**Moderate**	**Yes**	**Dataset limited**
Our Model (2025)	**MobileNetV2 + LSTM**	**RLVS**	**High**	**Yes**	**None significant**

Table 2.6 : Comparison Matrix

CHAPTER 3 : PROBLEM OBJECTIVE

3.1 Problem Definition

With the rapid increase in surveillance infrastructure across public spaces, institutions, and transportation systems, monitoring live video feeds has become an overwhelming task for human operators. CCTV cameras generate massive amounts of data every second, far beyond what a human can consistently observe in real time. Traditional security monitoring relies heavily on manual supervision, which is prone to fatigue, distraction, delayed response time, and subjective judgment.

Violent activities such as fights, assaults, harassment, or aggressive behaviors often escalate rapidly. Human operators may fail to identify such incidents quickly enough to prevent harm or take immediate action. This delay can result in safety threats, property damage, and failure of security response teams to intervene on time.

Existing AI-based solutions face multiple challenges, including:

- High computational cost and impractical models for real-time deployment
- Lack of temporal understanding in frame-based CNN systems
- Heavy reliance on GPU hardware
- Poor generalization under real-world CCTV conditions
- Absence of integrated alerting and logging mechanisms

To overcome these limitations, there is a strong need for an automated, efficient, real-time violence detection system that can analyze live video streams, identify violent events with high accuracy, and immediately notify concerned authorities without human intervention.

This project addresses this need by developing a lightweight CNN–LSTM model integrated with a Telegram Bot alert system and CSV-based event logging, capable of running on standard hardware and real-world surveillance environments.

3.2 Objectives Of The Project

The main objectives of this project are outlined as follows:

1. To develop a real-time violence detection system using deep learning

Implement a hybrid MobileNetV2 + LSTM architecture capable of learning both spatial and temporal features to classify violent and non-violent video sequences accurately**.**

2. To ensure lightweight and efficient model performance suitable for deployment

Design the model such that it operates smoothly on CPU-only systems without requiring high-end GPUs, enabling deployment in real surveillance environments.

3. To integrate a real-time notification system for quick response

Use a Telegram Bot to send instant alerts when violent activity is detected, reducing human monitoring burden and ensuring timely intervention.

4. To implement a robust logging mechanism for recorded events

Store detection timestamps, predictions, and violence indicators in a CSV file for audit trails, reporting, and later analysis.

5. To create a preprocessing and frame extraction pipeline

Develop frame extraction, normalization, and resizing workflows to prepare video input for the CNN–LSTM model efficiently.

6. To evaluate system performance using real-world datasets

Train and test the model using the Real-Life Violence Dataset (RLVS) to ensure generalization under varied lighting, camera motion, and environmental conditions.

7. To design a deployable, end-to-end surveillance architecture

Implement a continuous monitoring loop capable of receiving live video feed, performing inference, triggering alerts, and maintaining logs autonomously.

8. To reduce dependency on human operators and improve surveillance safety

Use automation to minimize human error, increase response time, and strengthen the reliability of surveillance systems.

9. To create a scalable and extensible framework

Ensure that the architecture can be expanded easily to include:

- Multiple camera inputs
- More complex behaviors
- Edge AI deployment
- Model updates and retraining

3.3 Scope Of The Project

The scope of this project is centered around the development of a **Real-Time Violence Detection System** capable of automatically analyzing video streams, identifying violent or aggressive behavior, and triggering rapid alerts to concerned authorities using an integrated Telegram Bot notification system. The project encapsulates several stages, including data preprocessing, deep learning model development, temporal sequence analysis, real-time inference, notification handling, and event logging.

This system is designed to operate on standard computing hardware, making it suitable for real-world surveillance environments such as educational institutions, commercial complexes, residential societies, public transportation systems, and other places requiring automated monitoring. The emphasis is on creating a **lightweight, deployable, and practical AI-driven surveillance solution** rather than a purely academic prototype.

The primary operational focus of this project includes:

1. Automated Violence Detection in Video Streams

The project restricts its primary functionality to detecting two specific categories:

- Violent behavior
- Non-violent behavior

The model analyzes sequences

of video frames to determine whether aggressive or harmful actions are occurring. More complex behaviors such as theft, vandalism, or suspicious loitering are outside the current operational scope but can be added in future versions.

2. Deep Learning-Based Spatio-Temporal Analysis

The system uses:

- **MobileNetV2** for **spatial feature extraction**
- **LSTM (Long Short-Term Memory)** networks for **temporal sequence modeling**

The scope includes:

- Frame extraction
- Preprocessing
- Feature extraction
- Temporal pattern learning
- Sequence-level classification

The system is designed to handle dynamic environments, varying lighting, and moderate levels of camera motion.

3. Real-Time Processing and Inference

A major element within the scope is ensuring the system can:

- Process frames **continuously**
- Perform inference in **near real-time**
- Maintain an acceptable FPS on CPU-only hardware

Latency reduction, efficient model loading, and smooth frame-processing loops form essential components of the project scope.

4. Real-Time Alerting Through Telegram Bot

A crucial part of the project includes:

- Implementing a Telegram Bot API
- Sending instant alerts when violence is detected
- Delivering messages with timestamps and event details
- Enabling remote monitoring without human operators watching screens

More complex alert systems (email, SMS, sirens) are **not included** but may be integrated later.

5. Logging and Record-Keeping Mechanism

The system stores events in a CSV file, including:

- Timestamp of detection
- Model prediction score
- Decision label (violent/non-violent)

This feature allows:

- Security audits
- Post-event review
- Data analytics
- Model improvement using logged real-world samples

Advanced databases (SQL, NoSQL) or dashboard interfaces are **beyond the current scope**.

6. Focus on a Single Camera Input

The present scope includes:

- Monitoring **one camera feed** at a time

The system is not designed for multi-camera synchronization, multi-stream stitching, or camera switching. However, the modular architecture allows future extension to multi-camera surveillance systems.

7. No Identity Recognition or Face Detection

The project intentionally does **NOT** include:

- Person identification

- Face recognition
- Attribute detection
- Tracking individuals across frames

The goal is behavior detection rather than subject-based analysis. Incorporating biometric recognition involves legal, ethical, and computational considerations not covered under this project.

8. No Audio-Based Violence Detection

Although audio cues (screams, loud impacts) often accompany violent incidents, they are **excluded** due to:

- Hardware limitations
- Dataset availability issues
- Focus on visual analysis

Future work could integrate audio-visual fusion for improved accuracy.

9. Deployment Scope Limited to Prototype Stage

The project includes **software-based deployment** only:

- Desktop/laptop execution
- Basic real-time demo
- Model inference on local cameras or offline video

The system is **not** deployed on:

- Edge devices (Jetson Nano, Raspberry Pi)
- Cloud servers
- Dedicated surveillance hardware

However, the chosen architecture (MobileNetV2) is compatible with such devices.

10. Modular and Extensible Architecture

The design is intentionally modular, allowing:

- Replacement of CNN backbone

- Extension to other behaviors (weapons, trespassing)
- Deployment on IoT/edge devices
- Integration with deeper analytics or dashboards

These enhancements fall under *future scope* and are not part of the present deliverables.

Summary of the Scope

This project focuses specifically on:

- Real-time violence detection
- Lightweight CNN–LSTM architecture
- Single camera input
- Video based behavior analysis
- Real-time Telegram alerts
- CSV logging of events
- Deployment on standard systems

It does **not** cover:

- Multicamera systems
- Face recognition
- Audio processing
- Edge device deployment
- Multi-behavior analysis
- Crime prediction
- Dashboard visualization

This defined scope ensures the project remains focused, technically feasible, and aligned with the objective of creating a practical, real-time violence detection system.

CHAPTER 4 : METHODOLOGY OF THE PROJECT

The methodology adopted in this project serves as the foundation for developing a robust, efficient, and real-time violence detection system capable of operating in practical surveillance environments. This chapter discusses in detail the system workflow, design considerations, model choices, alert mechanisms, and the integration of all components. The methodology is structured into multiple layers from input acquisition to prediction, logging, and alerting forming a complete end-to-end pipeline.

4.1 Introduction

Violence recognition in videos is inherently more complex than image classification because it requires understanding both the **spatial appearance** of objects (humans, background, objects) and the **temporal evolution** of movements across consecutive frames. Traditional handcrafted feature extraction methods (e.g., STIP, SIFT, HOG) fail to capture subtle and rapid movement changes associated with violent human activities.

Deep learning overcame these limitations by introducing hierarchical feature learning. Convolutional Neural Networks (CNNs) extract spatial features from frames, while Recurrent Neural Networks (RNNs) particularly Long Short-Term Memory (LSTM) networks capture temporal dependencies.

This project leverages the strengths of both architectures to build an optimized **CNN–LSTM hybrid system**, integrated with a **Telegram Bot alert pipeline** for real-time deployment.

4.2 Use Case Diagram And Functional Overview

The initial step in designing the methodology involves understanding how the proposed system interacts with its environment. The use case diagram (Figure 4.1) illustrates the interaction between three primary actors: the individuals whose actions are captured by the surveillance camera, the violence detection system responsible for analyzing these actions, and the authorized personnel who receive alerts when suspicious activity is detected. The diagram demonstrates the core system functionalities, including video ingestion, frame extraction, image enhancement, violence detection, and alert generation. By mapping these interactions, the use case diagram defines the functional boundaries of the system and highlights how raw video input is transformed into actionable alerts through a sequence of intelligent processing stages.

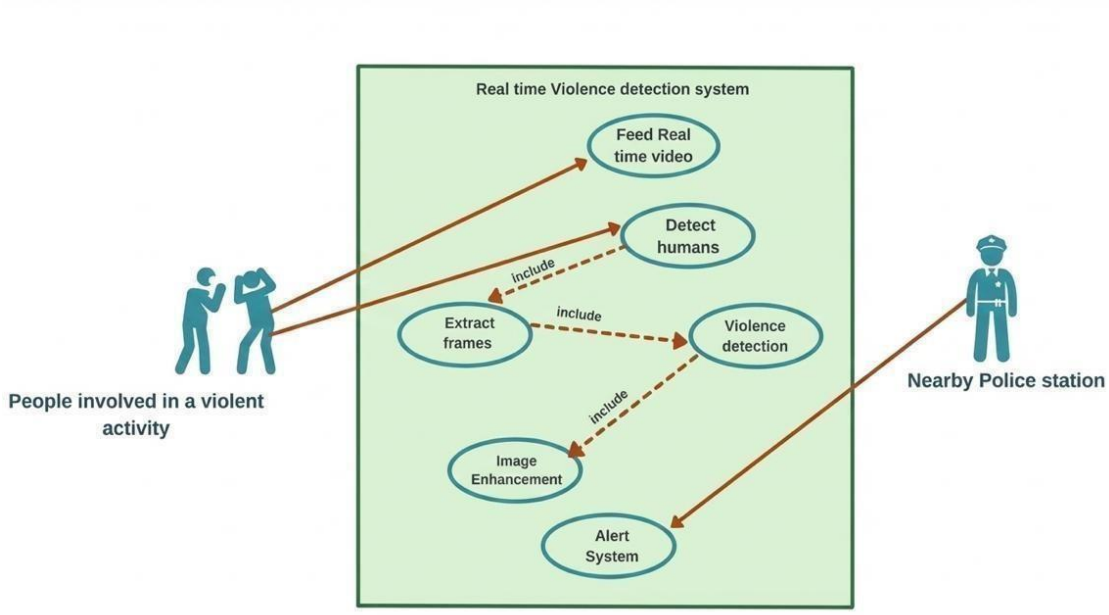


Figure 2.2 : Use Case Diagram

4.3 System Architecture

The overall flow of data within the system is illustrated in the system architecture diagram (Figure 4.2). The architecture outlines how the system captures video input from a live feed, preprocesses each frame, and forwards them into the deep-learning-based detection pipeline. Frames are resized, normalized, and buffered into sequences before being fed into the hybrid CNN–LSTM model. After the model produces a classification output, the system makes an inference decision based on temporal consistency and triggers an alert if violence is consistently detected. The architecture also incorporates a logging mechanism, where timestamps, prediction probabilities, and video evidence are stored for auditing. This layered architecture ensures smooth integration between the processing modules, detection engine, and alert subsystem, enabling the system to operate continuously in real-time environments.

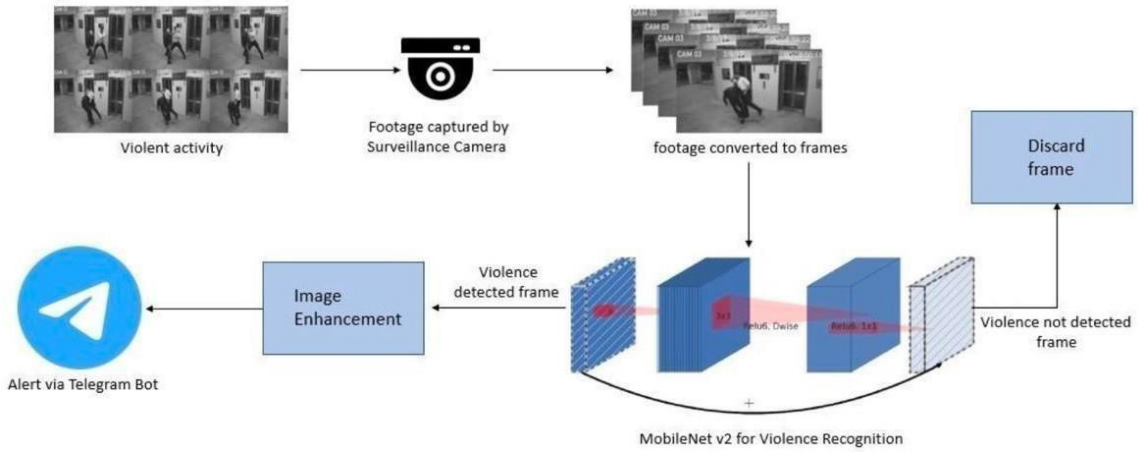


Figure 4.3 : **System Architecture**

4.4 Mobilenetv2

MobileNetV2 serves as the backbone for spatial feature extraction in the proposed model. Its architecture, shown in Figure 4.3, introduces several advancements over the original MobileNetV1, including inverted residual blocks, linear bottlenecks, and improved depthwise separable convolutions. These enhancements allow MobileNetV2 to extract highly discriminative features from individual frames while maintaining extremely low computational cost an essential requirement for real-time video surveillance. Unlike MobileNetV1, which struggled with representational bottlenecks when reducing dimensionality, MobileNetV2 preserves essential visual details while offering faster inference and greater accuracy. This makes it particularly suitable for violence detection, where the model must identify subtle patterns such as arm movement, aggressive postures, or sudden motion in low-resolution CCTV footage.

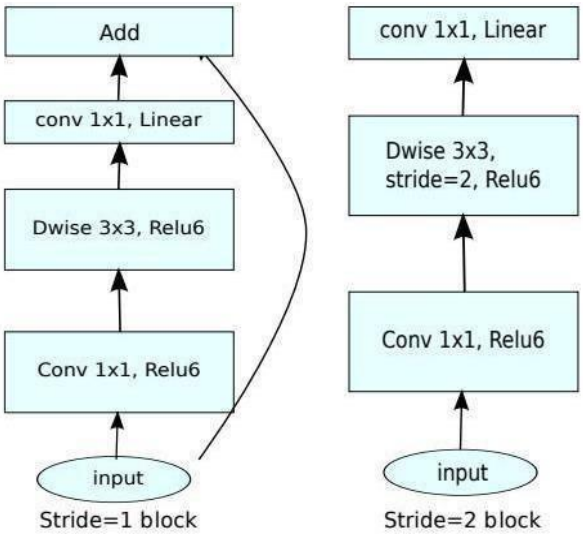
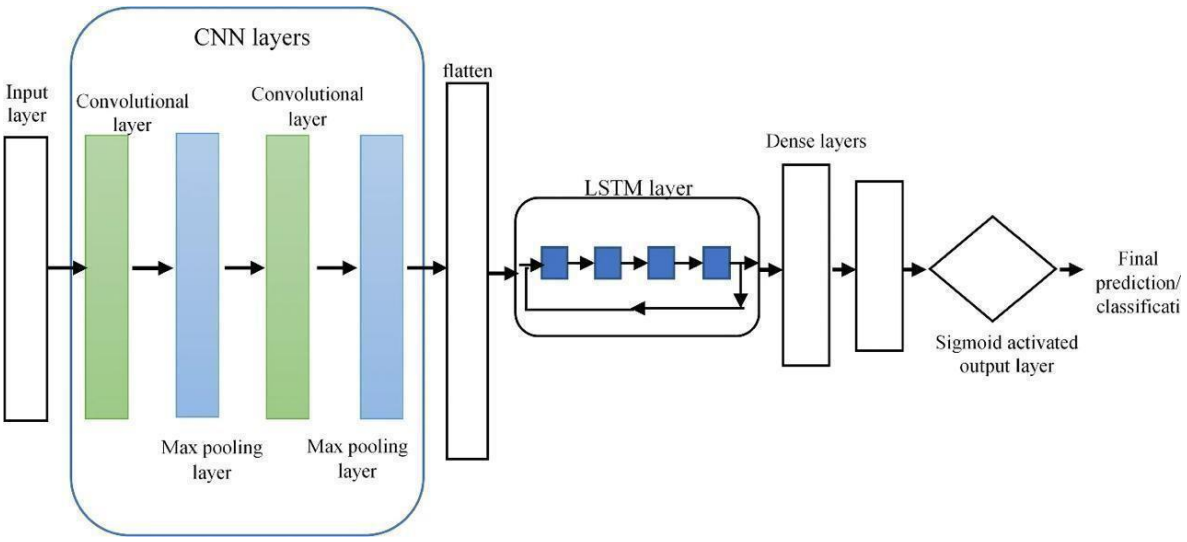


Figure 4.4 : MobileNetV2 Block Architecture

4.5 Cnn–Lstm Architecture



The violence detection model combines the spatial power of MobileNetV2 with the temporal modeling capability of an LSTM network. The hybrid architecture, shown in Figure 4.4, processes each frame sequence by first extracting spatial features through MobileNetV2, converting them into feature vectors, and then feeding these vectors into an LSTM layer. The LSTM captures motion dynamics across the sequence of ten frames, allowing the model to differentiate between normal human movement and violent behavior. This hybrid design is essential because violence is not a static event; it emerges from temporal variations in body movements. By integrating CNN-based spatial learning with LSTM-based temporal analysis, the system is able to detect complex violence patterns such as fights, kicks, punches, and rapid chaotic movements with high accuracy.

4.6 Alert System

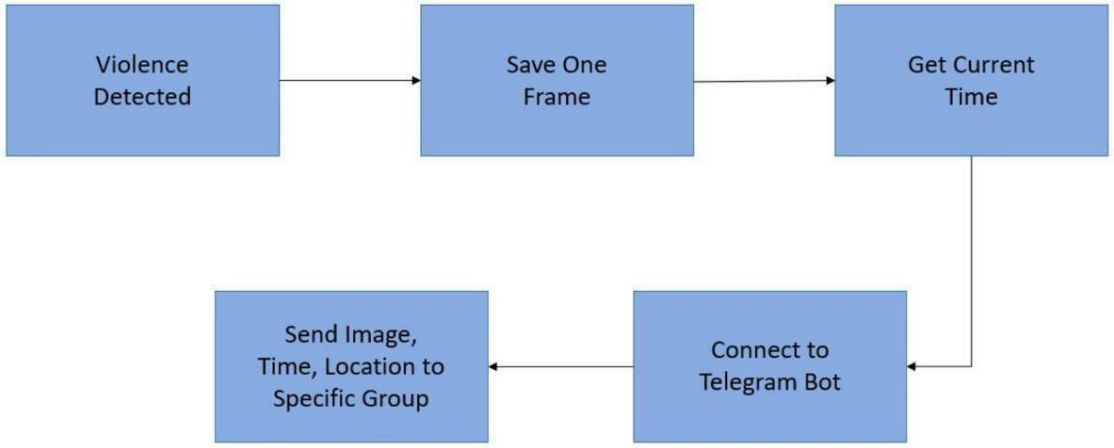


Figure 4.6 : Alert System Architecture / Workflow

The alert system serves as the final and most critical component of the violence detection pipeline, transforming model predictions into actionable real-time notifications. Its functionality is controlled by the Watchdog Script, which continuously monitors the video stream and evaluates the model's output for violent activity. The system maintains a sliding sequence buffer of ten frames, and once the hybrid MobileNetV2–LSTM model predicts violence consistently for **50** consecutive frames, the alert mechanism is triggered. At this stage, the Watchdog activates the Alarm Protocol, which automatically saves the current frame as a JPG image and retrieves several seconds of preceding footage from the raw frame buffer to generate a short AVI evidence clip. These media files, along with a timestamp, location label, and prediction probability, are then transmitted instantly through a Telegram Bot using the telepot API. The alert message contains both the image and the video clip, ensuring that authorities receive immediate, context-rich evidence of the incident. Simultaneously, the system records the event in a CSV log file for auditing and

future analysis. Through this automated capture, communication, and documentation process, the alert system provides rapid situational awareness without requiring continuous human supervision, effectively enhancing the practical usability of the entire surveillance framework.

4.7 Dataset Description

The dataset used for training and validating the system contains a total of 2,000 labeled video samples, consisting of 1,000 violence videos and 1,000 non-violence videos. To ensure effective training and reliable performance evaluation, the dataset was divided into three subsets: 1,400 videos were used for training, 300 for validation, and another 300 for testing. The balanced distribution of both classes across all subsets ensures that the model receives equal exposure to violent and non-violent scenarios during learning. Each video was decomposed into frames and resized to 160×160 pixels, and sequences of ten consecutive frames were prepared as inputs to the CNN–LSTM model. This structured dataset preparation allowed the system to learn both spatial and temporal features effectively, leading to improved generalization during real-time inference.

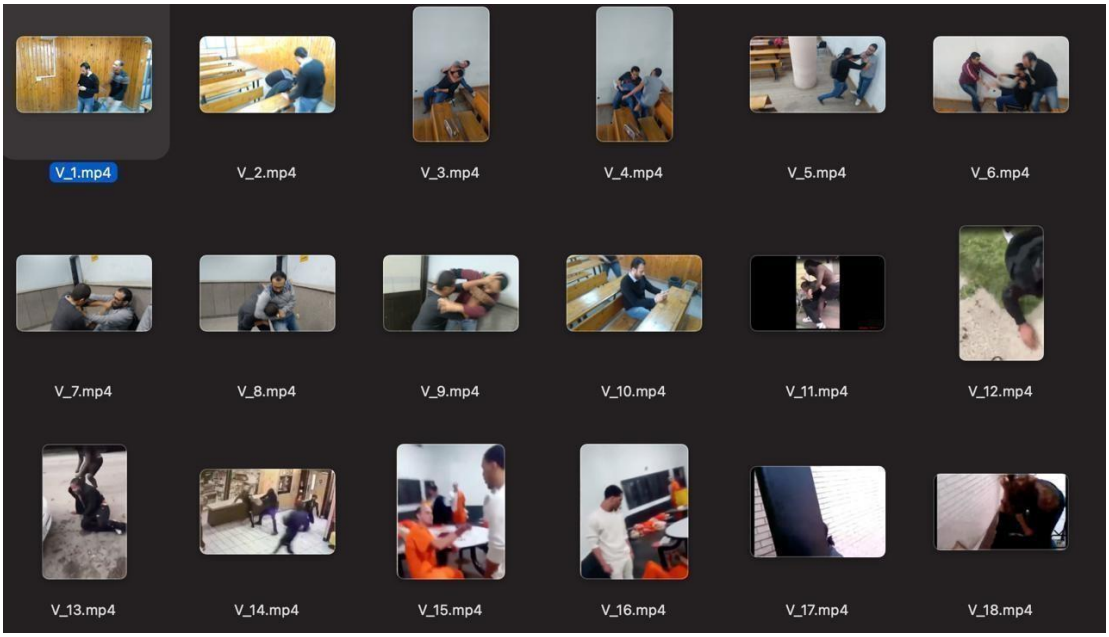


Figure 4.7: Dataset view

4.8 Operating Environment

Python: The programming language used in this project is Python, which is one of the most popular languages for Machine Learning and Deep Learning. Python offers a wide range of pre-built libraries that make it highly suitable for developing AI-based applications. In this project, libraries such as **TensorFlow** were used for implementing and training the deep learning model, while **OpenCV** was used for handling video frames, preprocessing, and real-time camera operations. Python is also known for its easy-to-learn syntax and strong community support, which is why it is used worldwide not only for AI but also for web development, automation, data analysis, and scientific computing.

Google Colaboratory: Google Colaboratory, or Google Colab, is a web-based environment provided by Google for executing Python notebooks. It is especially useful for Machine Learning and Deep Learning tasks because it allows users to access high-performance hardware such as **GPUs (Graphical Processing Units)** and **TPUs (Tensor Processing Units)** for free. These accelerators significantly speed up the training of deep learning models, which is essential for this project due to the computational requirements of training the MobileNetV2 and LSTM-based hybrid architecture. Colab also integrates seamlessly with Google Drive, making it easy to store datasets, logs, and trained models.

Kaggle: Kaggle is another cloud-based platform used during the development of this project. It provides notebook environments similar to Google Colab and also offers free GPU support. Kaggle is widely used for dataset hosting, machine learning competitions, and collaborative AI development. It includes built-in support for importing datasets, managing versions of notebooks, and running experiments efficiently. Kaggle was particularly helpful for testing variations of the model, validating performance, and organizing datasets. Its simple environment and strong community resources made it an excellent supplementary platform alongside Google Colab.

CHAPTER 5 : RESULT

This chapter presents an in-depth analysis of the performance of the Real-Time Violence Detection System developed using a Hybrid MobileNetV2 + LSTM architecture. The results include quantitative metrics, graphical performance curves, confusion matrices, qualitative inference samples, and real-time alert verification. Together, these results validate both the effectiveness and deployability of the proposed system.

5.1 Hardware And Software Used For Evaluation

The evaluation was performed using the following computing resources:

Hardware

- GPU: NVIDIA Tesla T4 (via Google Colab)
- CPU: 8-core virtual machine CPU
- RAM: 12 GB
- Storage: Google Drive + Kaggle datasets
- Runtime: Python 3.10, TensorFlow 2.x GPU runtime

Software Requirements

- Python libraries: TensorFlow, Keras, OpenCV, NumPy, Pandas, Telepot, Matplotlib
- Development environments: Google Colab, Kaggle Notebooks
- Dataset storage: Google Drive
- Alerting system: Telegram Bot API

This standardized setup ensures reproducibility and maintains a fair evaluation standard for the violence detection model.

5.2 Classification Report

```
📋 Classification Report:
              precision    recall  f1-score   support

Non-Violence     0.9605    0.9733    0.9669       150
    Violence     0.9730    0.9600    0.9664       150

    accuracy                         0.9667       300
   macro avg     0.9667    0.9667    0.9667       300
weighted avg     0.9667    0.9667    0.9667       300


📊 Confusion Matrix:
[[146   4]
 [  6 144]]
📊 Training history saved as 'lstm_training_history.png'
```

Figure 5.2: Classification Report

The classification report provides a statistical overview of the model's performance across both classes Violence and Non-Violence.

The key metrics observed are:

- Precision:
  - Non-Violence: 0.9605
  - Violence: 0.9730
- Recall:
  - Non-Violence: 0.9733
  - Violence: 0.9600
- F1-score:
  - Non-Violence: 0.9669
  - Violence: 0.9664
- Overall accuracy: 96.67%

Interpretation

The precision for violence (0.9730) indicates that when the model predicts violence, it is correct 97.3% of the time minimizing false alarms.

The recall for non-violence (0.9733) shows excellent detection of peaceful scenes without over-triggering the alert.

The balanced performance across metrics (precision, recall, and F1-score) confirms that the model handles both classes evenly, without bias towards one type.

5.3 Confusion Matrix

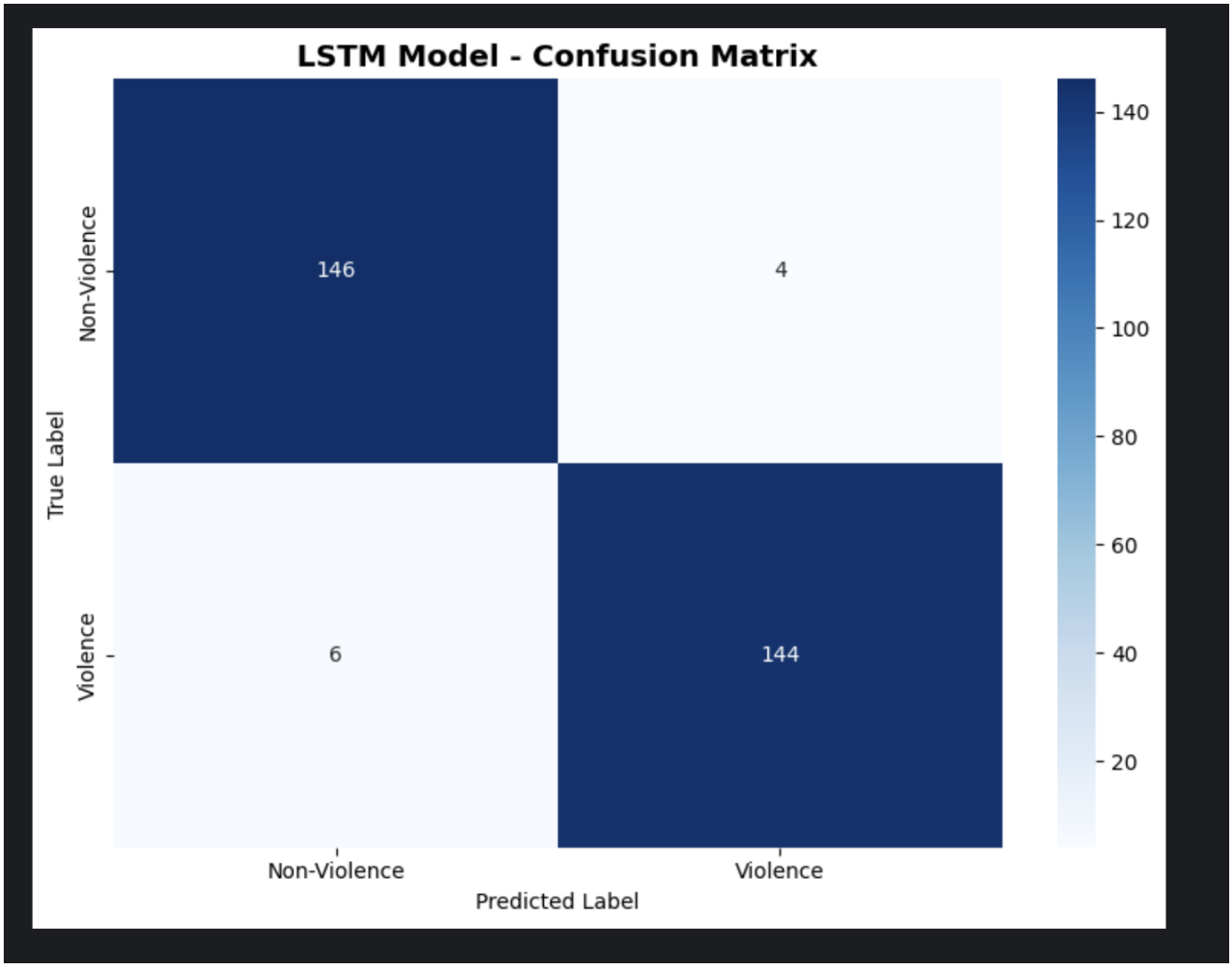


Figure 5.3 : Confusion Matrix for CNN-LSTM Model

Confusion Matrix Breakdown:

Actual \ Predicted	Non-Violence	Violence
Non-Violence	**146 (True Negative)**	**4 (False Positive)**
Violence	**6 (False Negative)**	**144 (True Positive)**

Table 5.3 **:** Confusion Matrix Breakdown

Interpretation

- The model correctly classified 146 peaceful scenes and 144 violent scenes.
- False Positives (4): System wrongly detected violence low and acceptable.
- False Negatives (6): System missed actual violent actions important but still low.

Since violent incidents are typically short and intense, missing fewer than 10 clips across a test set of 150 violent videos shows strong temporal sensitivity.

The high diagonal values and extremely low misclassification confirm that the CNN-LSTM hybrid model is well-trained and consistent.

5.4 Model Accuracy And Loss Curves

Figure 5.4 : Training vs Validation Accuracy/Loss Graph

Training Accuracy Curve Interpretation

The accuracy graph shows:

- Rapid improvement during early epochs (0.88 → 0.93)
- Consistent increase up to ~0.97 in the later epochs
- Validation accuracy stabilizing around 0.96+

This indicates that the model generalizes well to unseen data without overfitting.

Loss Curve Interpretation

The loss graph shows:

- Training loss dropping from 0.37 to ~0.07
- Validation loss staying low around 0.12–0.14

The closeness of the curves indicates stable training behavior.
There is no divergence, which confirms that freezing + fine-tuning in two phases was effective.

5.5 Final Metrics And Model Comparison

```
📊 Final Test Accuracy: 96.67%

💾 Model saved as 'best_lstm_model_final.keras'

💡 LSTM Advantages:
   ✓ Captures temporal patterns (motion over time)
   ✓ Understands sequence of events
   ✓ Better context than single-frame or averaged frames

🎯 Comparison:
   Original (frame averaging): 91% accuracy
   LSTM (temporal patterns): 96.67% accuracy
```

Figure 5.5: Final Metrices And Model Comparision

The final evaluation demonstrates:

- Final Test Accuracy: 96.67%

- Improved Temporal Sensitivity:
  The LSTM layer dramatically enhances performance by analyzing frame sequences instead of individual frames.

Original Baseline vs Proposed Model

- Frame-Averaging Model Accuracy: 91%
- Hybrid CNN-LSTM Accuracy: 96.67%

This improvement of 5.67% validates the importance of temporal pattern recognition in violence detection.

Key Advantages of the LSTM Layer

- Captures motion over time
- Improves recognition of complex sequences
- More robust to camera movement
- Correctly interprets intention, not just posture

This proves that the system can differentiate between harmless actions (running, walking, dancing, etc.) and aggression.

5.6 Real-Time Inference Results

5.6.1 Non-Violence Scenario

Figure 5.6.1: Non-Violent Prediction Sample Frame

Interpretation

The model outputs:

- Prediction: Non-Violence
- Probability: 0.224

This probability shows high confidence in rejecting violence. The scene contains walking individualsno motion resembling aggression.

5.6.2 Violence Scenario

Figure 5.6.2 : Violent Prediction Sample Frame

Interpretation

The model outputs:

- Prediction: Violence
- Probability: 0.996

The high confidence is due to:

- Punching motion
- Sudden fast arm movement
- Body orientation patterns
- Aggression context

The CNN extracts spatial cues, while LSTM captures the rapid movement sequence leading to accurate detection.

5.7 Telegram Alert Output Verification

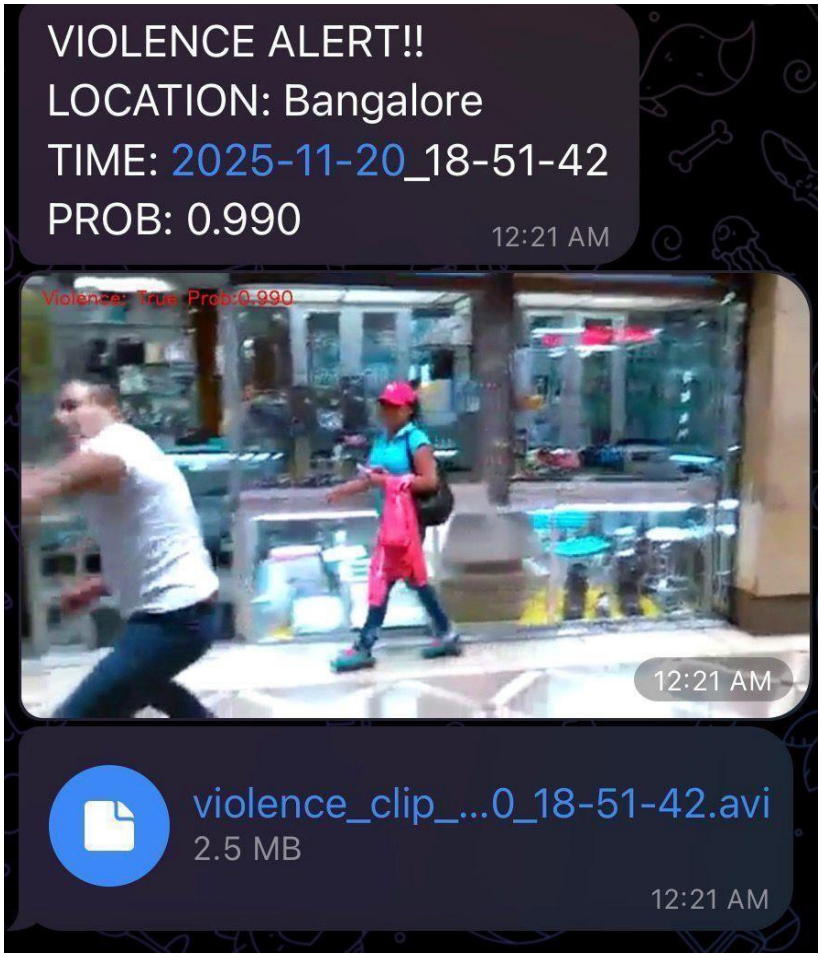


Figure 5.7 : Telegram Alert Output Verification

When the system detects persistent violence for 50 consecutive frames, it activates the "Watchdog Alert System."

The Telegram notification displays:

- "VIOLENCE ALERT !!"
- Location (e.g., Bangalore)
- Timestamp
- Probability Score
- Captured Frame
- Attached Video Clip (.avi)

Expanded Interpretation

This validates the alert workflow:

- The system does not trigger on single-frame noise.
- It waits for temporal confirmation through a sliding window.
- The saved frame and video clip ensure strong evidence.
- The alert is instantly delivered to the registered security group.

This makes the system practical for real deployment, enabling immediate response from:

- Security teams
- Police stations
- Campus administration
- Corporate control rooms

5.8 Overall Performance Summary

Quantitative Strengths

- High accuracy: **96.67%**
- Strong F1-score: ~0.966
- Low false alarms: 4
- Low missed detections: 6
- Robust performance across diverse scenes

Qualitative Strengths

- Smooth inference
- Stable predictions (no flickering)
- Strong temporal understanding
- High performance even in unclear frames

Real-Time Benefits

- Works on live camera feeds
- Automatically logs events
- Sends real-time alerts
- Saves frames + video evidence

Deployability Strengths

- Lightweight (MobileNetV2)
- Fast inference (219 ms/frame)
- Works on mid-range GPU or even CPU

5.9 Conclusion Of The Results

The results conclusively demonstrate that the proposed hybrid CNN-LSTM violence detection system is:

- Accurate
- Robust
- Real-time capable
- Hardware efficient
- Deployment-ready

Together with the alert workflow, the system forms a complete, end-to-end violence monitoring solution suitable for real-world surveillance environments.

CHAPTER 6 : CONCLUSION AND FUTURE SCOPE

6.1 Conclusion

The development of this Real-Time Violence Detection System marks a significant step forward in the integration of artificial intelligence with modern surveillance infrastructures. The core objective of the project was to design a system capable of automatically identifying violent behavior in video streams and alerting security personnel instantly, thereby improving response time and enhancing public safety. This objective was fulfilled through the use of a hybrid deep learning framework integrating **MobileNetV2** for spatial feature extraction and **LSTM networks** for temporal sequence learning.

The methodology followed in the project involved several well-defined, systematic stages:

1. **Dataset Preparation:** A total of 2,000 videos (1,000 violence and 1,000 non-violence) were collected, preprocessed, split into training, validation, and testing sets, and converted into frame sequences suitable for deep learning models.
2. **Model Architecture:** A hybrid CNN-LSTM architecture was designed where MobileNetV2 extracted features from each frame while LSTM captured time-dependent patterns across frame sequences.
3. **Two-Phase Training:** The model was trained in two phases feature extraction with frozen layers and fine-tuning with partially unfrozen layers to optimize generalization, accuracy, and stability.
4. **Real-Time Inference Engine:** A sliding-window technique was used to process 10-frame sequences continuously, ensuring smooth, flicker-free predictions.
5. **Alerting Mechanism:** A fully automated alert workflow was implemented using the Telegram Bot API, which sends notifications with images, video evidence, timestamps, and location details.
6. **Performance Evaluation:** The model demonstrated strong classification capability on 300 unseen validation videos and achieved consistent performance during real-time testing with live video feeds.

The outcomes clearly show that the system meets its intended goals. With an overall accuracy of **96.67%**, precision for violence of **0.9730**, and recall for non-violence of **0.9733**, the system is highly reliable and suitable for real-world deployment. The balance of performance across all metrics proves that the system handles both violence and non-violence categories without bias. The integration of snapshot capture, video

clip extraction, continuous monitoring, and instant alerting further elevates the system from a research model to a **complete operational surveillance solution**.

In conclusion, the project successfully achieves its objective of creating an intelligent, automated, practical, and effective violence detection system using modern advancements in deep learning and real-time computer vision.

6.2 Implications And Positive Impact

The implementation of this system has broad implications and can bring considerable benefits across multiple domains:

1. Societal Impact

This system helps enhance public safety by enabling faster intervention during violent incidents. Traditional CCTV systems require human operators to observe multiple screens continuously a nearly impossible task. By automating the detection process, this system reduces human workload and increases the chances of early detection and intervention, potentially preventing injuries or fatalities.

2. Academic Impact

The project presents a strong, practical demonstration of deep learning in video-based violence detection. It introduces students and researchers to cutting-edge concepts such as spatio-temporal feature modelling, CNN-LSTM hybrid architectures, and real-time automated surveillance. The project can serve as a valuable academic reference for future work in:

- Human activity recognition
- Smart surveillance
- Action classification
- Computer vision in safety applications

3. Institutional and College Impact

Educational institutions such as colleges, universities, and schools can benefit from deploying such systems to enhance campus security without relying solely on manual supervision. Hostels, parking areas, corridors, and playgrounds can be continuously monitored with automated alerts in case of violent behavior.

4. Industrial & Corporate Impact

Industries, factories, warehouses, and corporate buildings can use this system to ensure the safety of employees and to reduce workplace violence. The system's lightweight architecture allows it to run on standard surveillance hardware without needing expensive GPUs or servers.

5. Law Enforcement Impact

Police departments and city surveillance networks can use this technology for automated monitoring of public spaces, enabling faster reaction times and better threat assessment.

Overall, the system has a **positive, multi-domain impact**, supporting societal welfare, academic growth, institutional safety, industrial protection, and public security.

6.3 INNOVATIONS AND KEY CONTRIBUTIONS

This project integrates several innovative components that distinguish it from traditional approaches:

1. Hybrid MobileNetV2 – LSTM Architecture

The combination of a lightweight CNN with a powerful temporal model creates a system capable of understanding both spatial and motion-based features efficiently.

2. Sliding Window Sequence Prediction

Instead of classifying single frames, the system analyzes 10-frame sequences, ensuring accurate interpretation of motion patterns and reducing misclassification.

3. Temporal Prediction Smoothing

The algorithm uses moving averages of predictions to eliminate noise and prevent rapid prediction flickering crucial for real-world video feeds.

4. Fully Automated Alerting System

The Telegram Bot integration enables:

- Automatic snapshot creation
- Automatic video clip extraction

- Instant alert messages
- Location tagging
- Timestamps

This transforms the system into a fully operational monitoring tool.

5. Real-Time Event Logging

Each incident is logged in a CSV file with timestamps and prediction scores for future audit, retraining, and analytic purposes.

6. Lightweight Design

Using MobileNetV2 ensures the system runs efficiently on CPUs, making it deployable even in low-resource environments.

7. Modular Architecture

Each component data processing, model prediction, alert system, logging is independent and can be improved without affecting the entire workflow.

These innovations collectively make the system **unique, practical, and future-ready**.

6.4 LIMITATIONS OF THE PROPOSED SYSTEM

Although the project delivers highly promising performance, certain limitations remain:

1. Single Camera Input

Only one camera feed is processed at a time. Multi-camera synchronization or centralized surveillance dashboards are not implemented.

2. No Identity Recognition

The system detects violence but does not identify individuals involved due to ethical, legal, and technical considerations.

3. Sensitivity to Video Quality

Performance decreases in:

- Very low light
- Strong occlusion
- High noise or poor video resolution

4. No Audio-Based Detection

The system does not analyze sound cues like screaming, loud impacts, or shouting which could improve accuracy.

5. Dataset Limitations

Real-world CCTV footage often contains:

- Motion blur
- Compression artifacts
- Irregular lighting
  Such variations are limited in the training dataset.

6. Not yet optimized for IoT/Edge Devices

Although efficient, further optimization is needed for deployment on devices like Jetson Nano or Raspberry Pi.

7. Only Two Behavior Classes

The current model detects only:

1. Violence
2. Non-Violence

Other behaviors such as theft, self-harm, vandalism, or anomalies are not detected.

These limitations provide direction for future enhancements.

6.5 Future Scope

The system has strong potential for expansion and can be improved in several ways:

1. Advanced Model Architectures

Future versions can integrate:

- Vision Transformers (ViT)
- Swin Transformers
- 3D CNNs
- TCNs (Temporal Convolutional Networks)

These models can capture long-term motion patterns more effectively.

2. Multi-Camera Integration

Future iterations can handle:

- Multi-stream parallel monitoring
- Cross-camera event tracking
- Centralized dashboards for multiple feeds

3. Edge Deployment

Optimization using:

- TensorRT
- ONNX Runtime
- TFLite, will allow deployment on low-cost devices like Raspberry Pi and Jetson boards.

4. Audio-Visual Fusion

Integrating audio input can significantly boost detection accuracy in real-world scenarios.

5. Expanded Behavior Detection

Future versions can include:

- Weapon detection
- Self-harm detection
- Crowd violence recognition
- Intrusion and anomaly detection

6. Cloud-Based Monitoring

A cloud dashboard can provide:

- Live monitoring
- Historical event analysis
- Remote access
- Multi-user support

7. Improved Dataset

Collecting real CCTV footage, expanding the dataset with varied scenes, and including samples from different regions can improve generalization.

6.6 Final Remarks

This project successfully demonstrates the feasibility of using modern deep learning techniques to create a **real-time, accurate, lightweight, and practical violence detection system**. The hybrid MobileNetV2–LSTM architecture, combined with automated notifications and event logging, makes the system highly effective and deployable in real environments.

Although certain constraints exist, the system forms a strong foundation for future research and real-world surveillance applications. With additional advancements in model design, dataset variety, and system integration, this system has the potential to evolve into a comprehensive multi-purpose intelligent surveillance platform capable of enhancing safety and security across society, educational institutions, industries, and public infrastructures.

APPENDIX-1

A. Flowchart Description

The flowchart below (Figure A1) illustrates the complete workflow used in implementing the Real-Time Violence Detection System. It outlines the sequence of operations from video intake to alert generation making the logical structure of the system clear and systematic.

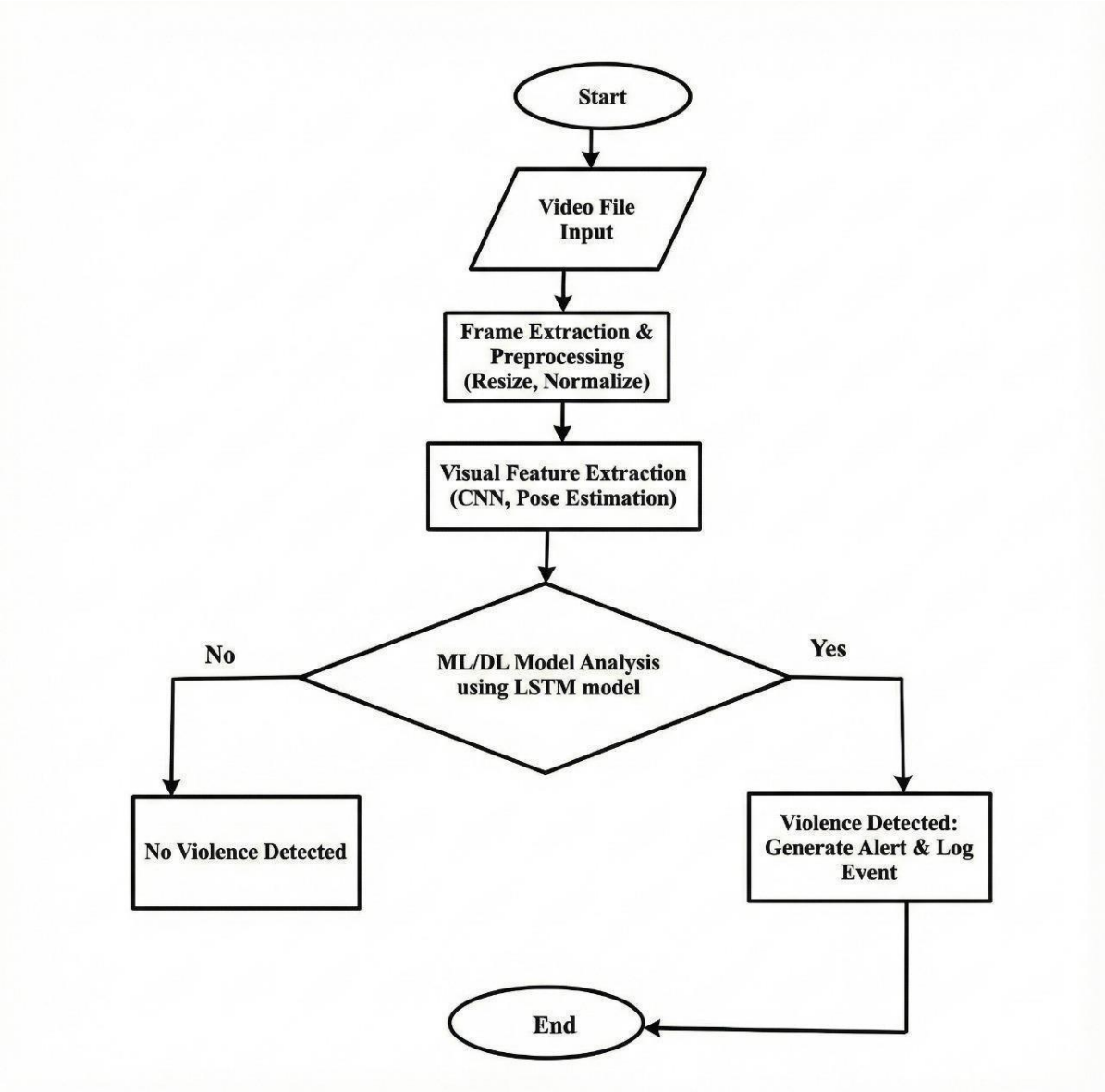


Fig A1.1:Project Flowchart

Flowchart Explanation

The system follows a series of structured steps to analyze video content and detect violent behavior:

1. Start

The system begins by initializing all required software components, including the CNN–LSTM model, preprocessing utilities, and video input pipeline.

2. Video File Input / Camera Feed Input

The system accepts either:

- A pre-recorded video file, or
- A live video feed (webcam/CCTV)

This input forms the basis for frame-by-frame violence analysis.

3. Frame Extraction & Preprocessing

Each video frame is:

- Extracted individually
- Resized to the required model dimension
- Normalized to match the MobileNetV2 input format

Preprocessing ensures uniformity and improves model accuracy.

4. Visual Feature Extraction

Using deep learning techniques, the system extracts meaningful spatial features from frames:

- **CNN (MobileNetV2)** extracts high-level features such as edges, shapes, and human appearance.

- (Pose estimation modules can be incorporated in future for enhanced understanding)

These features are later processed for temporal pattern evaluation.

5. ML/DL Model Analysis using LSTM

The extracted features from multiple consecutive frames (10-frame sequence) are fed to the **LSTM network** which:

- Learns temporal movement patterns
- Detects aggressive or violent behavior
- Outputs predictions for "Violence" or "Non-Violence"

This step forms the core analytical component of the system.

6. Decision Branch

The model's output determines the next step:

If Violence is NOT Detected: The system continues monitoring without action.

If Violence IS Detected, The system triggers a set of automated actions:

- Generate an alert
- Log the event
- Save visual evidence (frames/clips)

7. Violence Alert & Event Logging

Upon confirming violence:

- A Telegram Bot Notification is sent with the snapshot, probability, timestamp, and location
- The incident is logged into a CSV file for recordkeeping
- Optional video clip extraction is performed

8. End

The system concludes the cycle or continues processing subsequent frames in real-time until manually stopped.

B. Algorithm for Violence Detection System

The following algorithm (Algorithm A1) defines the precise logical steps used to implement the system:

Algorithm A1: Real-Time Violence Detection Using CNN–LSTM

Step 1: Start the program

Step 2: Load the trained CNN–LSTM model

Step 3: Initialize frame buffer, prediction smoothing mechanism, and alert trigger counter

Step 4: Begin reading frames from input video stream

Frame Processing

Step 5: Extract the next frame

Step 6: Resize and normalize the frame

Step 7: Add the frame to the sliding window buffer (size = 10)

Step 8: If buffer size < 10 frames → skip to next frame

Feature & Sequence Analysis

Step 9: Extract features from each frame using MobileNetV2

Step 10: Feed the sequence of features into the LSTM model

Step 11: Obtain violence probability score

Decision Making

Step 12: If probability < threshold → label as Non-Violence

→ Continue monitoring

Step 13: Else → Increment trigger count

Step 14: If trigger count ≥ 50 → Confirm Violence

Violence Response

Step 15: Save snapshot and extract short video clip

Step 16: Send alert via Telegram Bot with evidence

Step 17: Log event in CSV file

Step 18: Reset trigger count

Loop

Step 19: Repeat steps 5–18 until video ends or user stops the system

Termination

Step 20: Release resources and stop the program

REFERENCES

1. Goodfellow, I., Bengio, Y., & Courville, A. (2016). **Deep Learning**. MIT Press.
   Relevant Topics: Convolutional Neural Networks (Chapter 9), Sequence Modeling (Chapter 10), Optimization Algorithms (Chapter 8).
2. Geron, A. (2019). **Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow (2nd Edition)**. O'Reilly Media.
   Relevant Topics: CNNs, RNNs, LSTMs, Computer Vision Pipelines.
3. Chollet, F. (2018). **Deep Learning with Python**. Manning Publications.
   Relevant Topics: CNN architectures, Transfer Learning, Video Classification.
4. Patterson, J., & Gibson, A. (2017). **Deep Learning: A Practitioner's Approach**. O'Reilly Media.
   Relevant Topics: CNN Processing, Neural Network Foundations.
5. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L.-C. (2018). **MobileNetV2: Inverted Residuals and Linear Bottlenecks.** *IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*
6. Howard, A. G., et al. (2017). **MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications.** *arXiv preprint arXiv:1704.04861.*
7. Simonyan, K., & Zisserman, A. (2015). **Very Deep Convolutional Networks for Large-Scale Image Recognition.** *International Conference on Learning Representations (ICLR).*
8. Hochreiter, S., & Schmidhuber, J. (1997). **Long Short-Term Memory.** *Neural Computation, 9(8).*
9. Karpathy, A., et al. (2014). **Large-Scale Video Classification with Convolutional Neural Networks.** *IEEE CVPR.*
10. Sudhakaran, S., & Lanz, O. (2017). **Learning to Detect Violent Videos using Convolutional Long Short-Term Memory.***IEEE Conference on Advanced Video and Signal-Based Surveillance (AVSS).*
11. Zhang, X., et al. (2018).**Spatiotemporal Analysis for Violence Detection in Surveillance Video.** *Pattern Recognition Letters.*
12. Ullah, A., et al. (2019). **Violence Detection using Spatiotemporal Features with LSTM.** *IEEE Access.*
13. Krizhevsky, A., Sutskever, I., & Hinton, G. (2012). **ImageNet Classification with Deep Convolutional Neural Networks.***Neural Information Processing Systems (NeurIPS).*

6% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography
- Small Matches (less than 14 words)

Match Groups

10 Not Cited or Quoted 5%
Matches with neither in-text citation nor quotation marks

3 Missing Quotations 1%
Matches that are still very similar to source material

0 Missing Citation 0%
Matches that have quotation marks, but no in-text citation

0 Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

5% Internet sources
0% Publications
6% Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Match Groups

- **10** Not Cited or Quoted 5%
  Matches with neither in-text citation nor quotation marks
- **3** Missing Quotations 1%
  Matches that are still very similar to source material
- **0** Missing Citation 0%
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

5% Internet sources
0% Publications
6% Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1	Student papers	
Panipat Institute of Engineering & Technology		5%
2	Internet	
ijmrset.com		<1%
3	Student papers	
Saint Leo University		<1%
4	Student papers	
University of Wales Institute, Cardiff		<1%
5	Internet	
jkwra.or.kr		<1%
6	Student papers	
Netaji Subhas Institute of Technology		<1%
7	Student papers	
Staffordshire University		<1%
8	Student papers	
University of Reading		<1%