Vigilant Eyes: Empowering CCTV Surveillance with ML and AI for Face Recognition, Fall Prevention, and Violence Detection

Ishika Dubey[1], Siddharth Ghatuary[2], Kartik Tripathi[3], Kunal Sajwani[4], Snigdha Kesh[5]

[1]*Student, Department of Computer Science & Engineering, AMCEC, Bengaluru, India*

[2]*Student, Department of Computer Science & Engineering, AMCEC, Bengaluru, India*

[3]*Student, Department of Computer Science & Engineering, AMCEC, Bengaluru, India*

[4]*Student, Department of Computer Science & Engineering, AMCEC, Bengaluru, India*

[5]*Asst. Professor, Department of Computer Science & Engineering, AMCEC, Bengaluru, India*

ABSTRACT

This paper analyses the shortcomings of traditional CCTV surveillance systems in identifying and preventing security breaches in addition to introducing a machine learning and artificial intelligence-powered system for enhanced security coverage in real-time. The technology analyses video footage from CCTV cameras and employs face, fall, and fight detection algorithms to quickly spot potential security risks. Our system's technical specifications are covered in full, including the algorithms for face, fall, and fight detection. The implementation of the system faces several difficulties with accurate facial recognition and trustworthy fall and fight detection, which are also covered. Studies show how good the system is in spotting potential security issues, and prospective uses for the technology in a range of settings, such as airports, shopping malls, and public transit systems, are explored. Our solution serves as a proof of concept for further study in this area and has the potential to revolutionize security through the integration of ML and AI technologies into CCTV monitoring systems.

Keyword : *CCTV surveillance, machine learning and artificial intelligence, face detection, fall detection, fight detection, security, real-time, revolutionize security*

1. INTRODUCTION

CCTV surveillance systems are becoming an important part of the security infrastructure used in modern society. They are employed in a variety of settings, including airports, shopping malls, public transportation systems, and other public and private spaces. CCTV surveillance's primary goals are to improve security and safety, deter criminal conduct, as well as provide people in public places with a sense of security.

Traditional CCTV systems rely on human operators to analyze the footage, which can take longer and be untrustworthy. Additionally, conventional CCTV systems might not be able to pick up on certain security issues, such as individuals falling, fighting, or engaging in shady behavior. In order to get beyond these limits, there is a growing need for more technologically advanced and capable security solutions.

By utilizing the development of machine learning (ML) and artificial intelligence (AI) technology, CCTV

surveillance systems can be greatly improved to detect and assess potential security threats in real time. By enabling more precise and effective detection of possible security risks, these technologies have the potential to revolutionize the field of security.

In this paper, we propose a CCTV surveillance system based on ML and AI that uses face, fall, and fight detection that provides optimal safety coverage. Our system is made to examine footage from CCTV cameras to identify potential safety threats, such as people fainting, arguing, or acting suspiciously. The system can also recognize and follow people based on the characteristics of their faces. This system maintains a database of recognized faces that may be used to instantaneously match people. This technology can be especially helpful in spotting potential security threats, such as anyone on watchlists or those who have a history of questionable behavior.

We believe that technology is a crucial part of today's security infrastructure since it offers a complete solution for quickly recognizing and resolving potential security threats. We hope that our work can contribute in the creation of more effective and efficient security systems and that our system will open the door for further research in this field.

2. SYSTEM ARCHITECTURE

Our system architecture captures video, divides it into frames, and then uses the MobileNet Version 2 model and MediaPipe to analyze the frames. A Telegram bot can instantly inform the security charge of potential security issues including fights, falls, and faces, and send photographs and alerts to them. This allows for immediate action. Frames that may pose breaches of security are enhanced and saved in Firebase for additional utilization. The system offers a useful tool for in-the-moment surveillance and post-incident investigations.

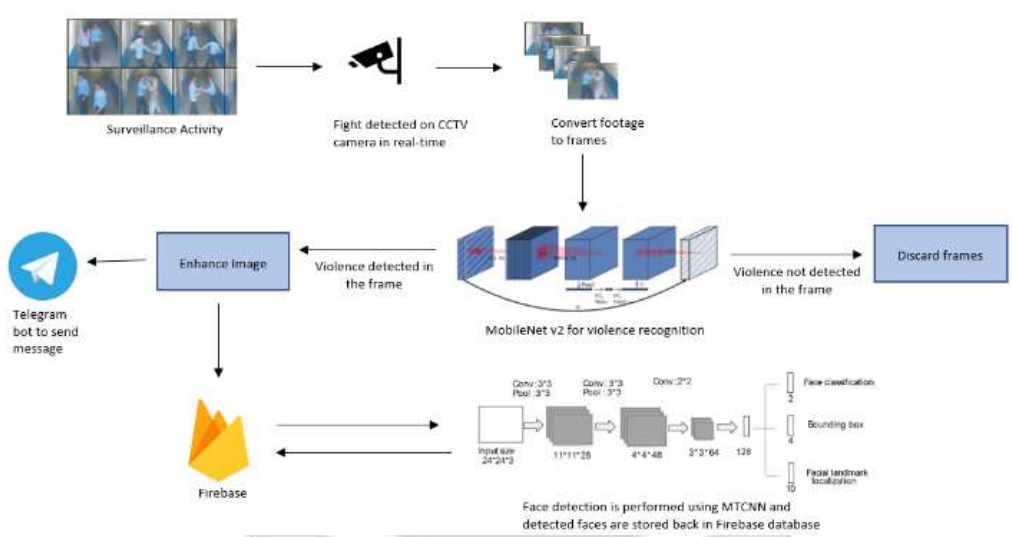

Fig -1: System Architecture

3. LITERATURE SURVEY

Literature survey is as follows:

1. Ahmed et al. (2016) published "Real-time human detection and tracking for video surveillance": In this study, a system for real-time human detection and tracking in video surveillance is presented. It combines background removal with features from the histogram of oriented gradients (HOG). The system demonstrated good detection and tracking accuracy when put to the test on several video sequences.
2. Cong et al. (2016) published "Real-time abnormal event detection in crowded scenes": This study suggests a system that combines deep learning and trajectory analysis to detect aberrant events in real time in crowded environments. The system demonstrated great accuracy in identifying anomalous events during testing on a variety of video sequences.
3. Liu et al.'s (2019) "A machine learning based framework for pedestrian detection in surveillance videos": Using a combination of feature extraction and classification, this research proposes a machine learning-based system for pedestrian detection in surveillance films. The method demonstrated high accuracy in pedestrian recognition across multiple datasets.
4. Liu et al. (2018) published a review of "Deep learning-based object detection in video surveillance": In-depth analysis of deep learning-based object detection techniques for video surveillance is provided in this research. It discusses numerous methods and assesses how well they work using a number of datasets, including Faster R-CNN, YOLO, and SSD.
5. Deep learning for real-time multi-camera vehicle tracking, Saikia et al. (2019): The method for real-time vehicle tracking suggested in this study makes use of deep learning and several cameras. The system demonstrated high accuracy in vehicle tracking throughout testing using several datasets.

4. METHADOLOGY

The methodology is as follows:

1. Real-time footage is captured using CCTV cameras: The methodology's initial stage entails using CCTV cameras to capture live footage.

2. The video is divided into frames: The video that was collected is then divided into separate frames. This is done to give the system the ability to examine each frame independently and find any potential security holes.

3. The frames are analyzed to detect fights, faces and fall: MobileNetV2, an agile deep learning model along with OpenCV, Harcascade and MediaPipe that can precisely 1692tilized1692 objects and features in photos, is used to analyse the frames to identify fights, faces and individual falls.

4. Frames without fights are discarded: If a frame doesn't feature any fights, it is deleted given that it does not provide a threat to security.

5. The image is improved and stored in Firebase when a fight appears in a frame. The image is improved to improve quality and make it less complicated to identify the people involved in the fight. The improved image is then kept in Firebase, a service for cloud storage that makes it simple to access and retrieve the image.

6. A Telegram bot sends a message to the security chief informing them of the incident and giving them access to the enhanced image: When a fight is spotted, the system is set up to send out a message to the security chief via a Telegram bot. The notification includes a link to the enhanced image kept in Firebase as well as details about the fight's location.

7. When a frame with an unknown face is found, a warning message and a picture of the face are sent to the security head via a Telegram bot: The technology uses a Telegram bot to alert the security head whenever it notices a frame with an unidentified face. The message contains a link to the face's photograph as well as details on the face's location.

8. When a fall occurs, a warning message and a photo of the incident are sent to the security chief via a Telegram bot: The system uses a Telegram bot to alert the security chief whenever an incident of falling is observed. The notification provides a link to the incident's image as well as details about the fall's location.

4.1 Modules

Face recognition: For the functionality required for in-the-moment facial recognition and surveillance, the code snippet imports a variety of libraries. The "cv2" library makes OpenCV's image and video processing tools available. It is necessary to import "numpy" in order to implement numerical operations on arrays and matrices, which are frequently employed in image processing. For greater accuracy, deep learning models are 1692tilized in the "face_recognition" library for facial detection and recognition. The "os" library is used to give users access to

operating system features, while the "datetime" and "time" libraries are used to timestamp and calculate the length of processes, respectively. Last but not least, the Telegram messaging system is integrated via the "telepot" library, enabling quick alarms and photographs to be delivered to security staff in case of potential security issues.

Algorithm Face Detection:

1. Set up the Telegram bot for sending notifications.
2. Load the known images of people and their corresponding names.
3. Encode the known faces in the images.
4. Set up the webcam capture.
5. Continuously capture frames from the webcam.
6. Resize the frame to a smaller size for faster processing.
7. Convert the frame to RGB color space for face recognition.
8. Detect the locations and encodings of faces in the current frame.
9. For each detected face:
    a. Compare the face encoding with the known encodings to determine if it matches any known face.
    b. If a match is found, draw a rectangle around the face and display the name of the person.
    c. If a match is not found, draw a rectangle around the face and label it as "unknown".
    d. If this is a new face that hasn't been detected before, send a notification with an image of the face to the Telegram bot.
10. Display the processed frame in a window.
11. Repeat steps 5-10 until the user quits the program.

Violence detection: The libraries and modules that have been imported into this code provide a wide range of functionality needed for a number of different jobs, including image processing, machine learning, database management, and messaging services. The code makes use of deque for picture storage and manipulation and popular machine learning packages like Keras and MTCNN for facial recognition tasks. In order to integrate the code with real-time databases for effective data management, Pyrebase and Firebase frameworks are used. For messaging services and visualisations, Pyplot and Telepot are also incorporated into the code. All things considered, the union of these libraries yields a potent toolkit for setting up a thorough CCTV system with facial recognition capabilities.

Algorithm Fight detection:

1. Initialize the Firebase app with credentials and authenticate with an email and password.
2. Load the Keras model for violence detection.
3. Initialize a deque to keep track of predictions.
4. Initialize the VideoCapture object to capture video from the CCTV camera.
5. While there are frames being grabbed from the camera, do the following:
    a. Read a frame from the VideoCapture object.
    b. Convert the frame from BGR to RGB, resize it to 128x128 pixels, and normalize its pixel values.
    c. Make a prediction on the frame using the Keras model.
    d. Add the prediction to the deque and calculate the average of the last 128 predictions.
    e. Check if the average prediction is greater than a threshold (e.g., 0.18).
    f. If the average prediction is greater than the threshold, save the frame as an image and enhance its brightness and color using PIL.
    g. Detect faces in the enhanced image using MTCNN and draw rectangles around them using Matplotlib.
    h. Save the enhanced image and the faces image to Firebase storage.
    i. Send an alert message with the time, location, and URL of the enhanced image to a Telegram bot using Telepot.
6. Release the VideoCapture object and disconnect from Firebase.

Fall detection : MediaPipe Solutions provides a suite of libraries and tools for us to quickly apply artificial intelligence (AI) and machine learning (ML) techniques in our applications. We can plug these solutions into our applications immediately, customize them to our needs, and use them across multiple development platforms. We used mediaPipe in our project for the sole purpose of detecting 'Human Fall'. Our System is capable of detecting fall in various posture such as fall on back, fall on right shoulder, fall on left shoulder. It was achieved due to precise estimation of human posture with mediapipe.

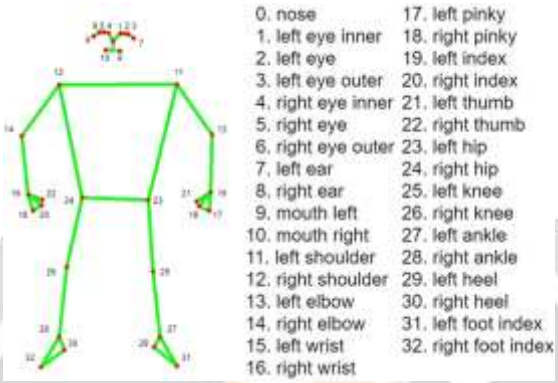

Fig -2: 33 Landmarks detected on human body using MediaPipe

Out of 33 landmarks on human body, we made use of landmarks number 10, 9, 6, 5, 4, 3, 2, 1 and landmarks number 11, 12, 24, 23. If we carefully observe these landmarks, we can clearly deduce that landmarks number 10 to 1 (which makes up of human mouth) will always lie above landmark number 12 and 11(shoulders). Also, 12 and 11 lies below landmark 23(left hip) and 24(right hip) when a human fall on his back. By this logic, we programmed our code in such a way which tracks these landmarks as coordinates on 2D plane. When these coordinates capture an approriate position which depicts a human in fall position, the program with the help of OpenCV displays the output as 'Fall detected'.

Algorithm Pattern Matching and Fall detection

1. Procedure PatternMatchingandFallDetection()
2. Import the MediaPipe and OpenCV library
3. Capture Real-time video through OpenCV
4. Detect 33 body landmarks using mediapipe findpose() method
5. while not at end of last frames in video do
    a. check on 2 dimesional y-axis of each frame
    b. if landmarks 11 is less than landmarks from 1 to 10
        i. print 'fall on right shoulder' to screen
    c. elseIf landmark 12 is less than landmarks 1 to 10
        i. print 'fall on left shoulder' to screen
    d. elseIf landmarks 23,24 less than landmarks 11,12
        i. print 'fall on back'
    e. end if
6. end
7. end procedure

5. CHALLENGES DURING THE DEVELOPMENT PROCESS

Few challenges that we faced during development are as follows:

1. Data collection challenges: Obtaining an ample data set for your project in order to increase accuracy

posed significant difficulties. It can take a lot of effort and resources to gather a broad and comprehensive dataset that accounts for all potential face, fall, and fight detection events. Using publicly accessible datasets and enhancing them to produce a larger and more varied dataset is one potential option. Utilising methods for the development of synthetic data to produce fake visuals that can mimic real-world situations is another option.

2. Real-time processing challenges: Processing CCTV data in real-time needs a lot of CPU power, which might be difficult for systems with constrained resources. Real-time processing of high-resolution videos might take a long time and possibly cause frame dropouts. Utilising hardware accelerators, such as GPUs or FPGAs, to accelerate processing time is one potential approach. Utilising cloud-based services with real-time processing capability is another option.We can leverage the AWS Kinesis web service for real-time processing in our ongoing development. As Amazon Web Services offers pay as you go services, it will be incredibly effective and economical.
3. Challenges with deployment: After the model has been trained and proven, applying it to a real-world situation can be difficult. Integrating the model with current CCTV systems and ensuring that it functions flawlessly with other software and hardware components are two significant challenges. Making sure the model is solid and trustworthy in all settings and lighting circumstances is another problem. One option is to package the model and its dependencies into a single executable file using containerization methods like Docker. This can make deployment easier and guarantee that the model functions consistently in various situations.
4. Challenge of clarity of images: High quality photos are necessary for effectively detecting and identifying faces. However, because of things like illumination, camera angle, and resolution, CCTV camera photos could not always be of high enough quality. This may make it challenging to identify people, which may reduce the surveillance system's accuracy. Several strategies can be used to solve this difficulty, including the use of high-resolution cameras, altering lighting or camera settings, or using image enhancing methods to enhance the clarity of the photographs. Additionally, algorithms that can identify people using features other than merely face recognition or that are more resilient to changes in image quality can be created. The accuracy and efficiency of the surveillance system can be increased by using these methods.

6. USE CASES

Few use cases of our project at various settings is as follows:

1. Public security: Law enforcement organizations can utilise technology to keep an eye on high-crime regions and spot potential threats. It can also be used to locate those who are violating the law, especially during riots so that appropriate action can be done on all sides. The system can be employed to locate missing people and support investigations.
2. Schools and universities: By keeping an eye on the corridors, classrooms, and entrances, the system can help to increase security and safety in schools and colleges. It can be used to locate and identify any unauthorised visitors on the property. Bullying of a person can also be immediately tracked.
3. Hospitals: The system can be used to keep track of patient security and spot any potential threats there. It can also be used to keep tabs on medical professionals and make sure the right procedures are being followed. Elderly patients need greater watchfulness, therefore it will assist in that by keeping watch and issuing alerts in the event of a fall or other problems.
4. Retail establishments: By using the technology, retail establishments may keep an eye on their space and spot shoplifters or any other questionable activities. By monitoring foot movement and product placement, the technology can also help to enhance the consumer experience. Additionally, it can be used to check automatic employee attendance, monitor staff behavior and track regular clients.
7. Industrial settings: The system can be used to monitor worker safety and spot any potentially dangerous circumstances in factories and warehouses. Additionally, it can be used to spot any production-line irregularities, improving quality control.

6. CONCLUSION

In conclusion, our project attempted to improve CCTV surveillance using ML and AI for face, fall, and fight detection. We have identified and addressed a variety of issues through our study, including data collecting, real-time processing, image clarity, and deployment. Potential uses for the system we created include public safety,

transportation, retail, and education. By implementing this technology, organisations can enhance their surveillance and response capabilities while lowering crime, accident, and injury rates.

The main accomplishments of our project include creating a system that can recognise faces, falls, and conflicts and notify the authorities. We think our idea might have a big impact and increase people's sense of safety and security both individually and collectively.

There is, however, still room for improvement. The system's scope might be broadened to handle more complicated occurrences, such the identification of weapons or crowd control, as one area for improvement. More study could also be done to improve the system's accuracy, particularly in noisy or low-light conditions. Overall, we think this project is a step towards utilising technology to enhance public safety and security, and we anticipate greater developments in this area in the future.

7. REFERENCES

[1] Ahmed, N., Nait-ali, A., & Omar, M. (2016). Real-time human detection and tracking for video surveillance. 2016 13th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), 98-103. doi: 10.1109/avss.2016.7738023

[2] Cong, Y., Yuan, J., Liu, J., & Thalmann, D. (2016). Real-time abnormal event detection in crowded scenes. ACM Transactions on Multimedia Computing, Communications, and Applications, 12(4), 1-21. doi: 10.1145/2973641

[3] Liu, H., Huang, Y., Liu, F., & Zeng, G. (2019). A machine learning based framework for pedestrian detection in surveillance videos. IEEE Access, 7, 34845-34856. doi: 10.1109/access.2019.2901573

[4] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C., & Berg, A. C. (2018). Deep learning-based object detection in video surveillance: a review. Journal of Visual Communication and Image Representation, 55, 273-280. doi: 10.1016/j.jvcir.2018.09.009

[5] Saikia, R., Ahmed, S. I., & Das, D. (2019). Real-time multi-camera vehicle tracking using deep learning. 2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT), 1-6. doi: 10.1109/icccnt45670.2019.8945149