A Project Report
Submitted on

Surveillance System using ML

In Partial Fulfillment of award of the

Bachelor of Technology (B.Tech) Degree
in
Information Technology
by

Prince Kumar
(2001920139007)

Vipnesh Chauhan
(1901920130191)

Under the Supervision of
Dr. Rajnesh Singh
Professor/Associate Professor/Assistant Professor

G.L. BAJAJ INSTITUTE OF TECHNOLOGY & MANAGEMENT
GREATER NOIDA

DR. A.P.J. ABDUL KALAM TECHNICAL UNIVERSITY,
UTTAR PRADESH, LUCKNOW

MAY, 2023

GL BAJAJ
Institute of Technology & Management

Department of Information Technology

Declaration

We hereby declare that the project work presented in this report entitled **"Surveillance System using ML",** in partial fulfillment of the requirement for the award of the degree of Bachelor of Technology in Information Technology, submitted to A.P.J. Abdul Kalam Technical University, Lucknow, is based on our own work carried out at the Department of Information Technology, G.L. Bajaj Institute of Technology & Management, Greater Noida. The work contained in the report is true and original to the best of our knowledge and project work reported in this report has not been submitted by us for award of any other degree or diploma.

Signature:

Name: Prince Kumar

Roll No: 2001920139007

Signature:

Name: Vipnesh Chauhan

Roll No: 1901920130191

Date:

Place: Greater Noida

GL BAJAJ
Institute of Technology & Management

Department of Information Technology

Certificate

This is to certify that the Project report entitled **"Surveillance System using ML" done by Prince Kumar (2001920139007), and Vipnesh Chauhan (1901920130191)** is an original work carried out by them in Department of Information Technology, G.L. Bajaj Institute of Technology & Management, Greater Noida under my guidance. The matter embodied in this project work has not been submitted earlier for the award of any degree or diploma to the best of my knowledge and belief.

Date:

Dr. Rajnesh Singh
Signature of the Supervisor

Dr. P.C. Vashist
Head of Department

GL BAJAJ
Institute of Technology & Management

Department of Information Technology

Acknowledgement

The merciful guidance bestowed to us by the almighty made us stick out this project to a successful end. We humbly pray with sincere heart for his guidance to continue forever.

We pay thanks to our project guide **Dr. Rajnesh Singh** who has given guidance and light to us during this project. His/her versatile knowledge has helped us in the critical times during the span of this project.

We pay special thanks to our Head of Department **Dr. P.C. Vashist** who has been always present as a support and help us in all possible way during this project.

We also take this opportunity to express our gratitude to all those people who have been directly and indirectly with us during the completion of the project.

We want to thanks our friends who have always encouraged us during this project.

At the last but not least thanks to all the faculty members of the Department of Information Technology who provided valuable suggestions during the period of project.

Abstract

The abstract provides a brief overview of the research project "Surveillance System using Machine Learning" and its focus on movement detection. It highlights the key objectives, methodology, and findings of the project. The background section sets the context for the project and explains the motivation behind developing a surveillance system using machine learning. It discusses the increasing need for advanced surveillance technologies to enhance security and monitoring capabilities in various domains, such as public spaces, transportation, and private facilities. It also highlights the limitations of traditional surveillance systems and the potential of machine learning techniques in improving movement detection accuracy and efficiency. The motivation for the project "Surveillance System using Machine Learning" stems from the need for more robust and accurate surveillance methods in today's security-conscious world. Traditional surveillance systems often face challenges in effectively detecting and tracking movements, leading to potential vulnerabilities and limitations in ensuring public safety. The objective of this project is to leverage machine learning techniques to develop an advanced surveillance system that can accurately detect and track movements in real-time.

The results of the project "Surveillance System using Machine Learning" demonstrate the effectiveness and reliability of the developed movement detection system. The system achieves high accuracy, precision, recall, and F1 score in detecting and tracking movements in real-time video streams. The integration of machine learning algorithms and advanced feature extraction techniques enables accurate identification of various types of movements, including walking, running, and vehicle movements. The system successfully overcomes the limitations of traditional surveillance systems, providing improved performance and robustness in movement detection.

Keywords: Results, Movement Detection, Machine Learning, Accuracy, Precision, Recall, F1 score, Integration, Feature Extraction, Real-time, Surveillance System, Reliability, Identification, Limitations, Robustness.

TABLE OF CONTENT

.

LIST OF FIGURES

LIST OF TABLES

Chapter 1

INTRODUCTION

The Surveillance System using ML for Motion and Head Movement Detection is a cutting-edge software-based project designed to enhance security measures through advanced technologies. By integrating machine learning (ML) algorithms with OpenCV, the system enables real-time detection of motion and analysis of head movements. It utilizes powerful tools and algorithms provided by OpenCV to process live video feeds and accurately identify changes in pixel-level motion. ML techniques distinguish between static and moving objects, triggering system events upon detecting significant motion. Moreover, the project incorporates Amazon Web Services (AWS) for secure storage of entry timing data, allowing for historical analysis and remote monitoring. With its alerting mechanism and customizable features, this surveillance system offers improved security and proactive response capabilities to potential threats.

1.1 **Background and Motivation**

In today's world, security is a paramount concern for individuals, organizations, and communities alike. The increasing sophistication of threats and the need for proactive surveillance have driven the development of advanced security systems. Traditional surveillance methods often rely on manual monitoring, which is time-consuming and prone to errors. To address these limitations, there is a growing demand for automated surveillance systems that can detect suspicious activities accurately and in real-time.

The emergence of machine learning (ML) and computer vision technologies has revolutionized the field of surveillance. ML algorithms, when combined with computer vision libraries such as OpenCV, can analyze video feeds, detect patterns, and make intelligent decisions. By leveraging ML techniques, surveillance systems can identify motion and analyze head movements, enabling a more efficient and reliable security infrastructure.

The motivation behind the Surveillance System using ML for Motion and Head Movement Detection stems from the need to enhance security measures and provide timely responses to potential threats. This project aims to leverage the power of ML and OpenCV to develop a software-based solution that can accurately detect motion and analyze head movements in real-time. By integrating with Amazon Web Services (AWS), the system securely stores entry timing data, allowing for historical analysis and remote monitoring.

The project's goal is to provide a customizable and scalable surveillance system that can be deployed in various settings, such as residential areas, commercial establishments, or public spaces. By utilizing ML algorithms, the system can differentiate between normal activities and suspicious behavior, alerting security personnel in real-time. The ability to detect unauthorized access attempts and unusual head movements can significantly enhance security measures and mitigate potential threats.

1.2 **Objective**

The objective of the Surveillance System using ML for Motion and Head Movement Detection project is to develop a software-based surveillance system that enhances security measures by leveraging machine learning (ML) algorithms and OpenCV. The primary objectives of the project are as follows:

Accurate Motion Detection: The system aims to accurately detect motion by analyzing consecutive frames of video feeds. ML techniques will be employed to distinguish between static and moving objects, ensuring precise identification of potential threats.

Head Movement Analysis: The project intends to analyze head movements using facial landmark tracking algorithms. By monitoring head movements, the system can identify unusual or abrupt actions that may indicate unauthorized access attempts.

Real-time Alerts: The system will incorporate an alerting mechanism that triggers immediate notifications, emails, or audible alarms upon detecting unauthorized entry or suspicious head movements. Real-time alerts enable security personnel to respond promptly and mitigate potential threats swiftly.

Integration with AWS: The project aims to securely store entry timing data in Amazon Web Services (AWS). This integration allows for historical analysis, remote monitoring, and easy access to stored data for further investigation and reporting purposes.

Customizability: The surveillance system will be designed to be customizable according to specific requirements. This includes the ability to integrate with existing security systems, adjust sensitivity levels, and potentially add face recognition capabilities for enhanced identification and tracking.

1.3 **Limitation of Research**

While the Surveillance System using ML for Motion and Head Movement Detection offers significant advantages in enhancing security measures, it is important to acknowledge its limitations. The following limitations should be considered:

Environmental Constraints: The system's effectiveness may be influenced by environmental factors such as lighting conditions, camera quality, and the presence of occlusions. Poor lighting or low-resolution cameras can impact the accuracy of motion detection and head movement analysis, potentially leading to false positives or missed detections.

Performance Considerations: ML algorithms, especially those involving real-time video processing, can be computationally intensive. Depending on the hardware capabilities of the system, there may be limitations in terms of processing speed and scalability. High-resolution video feeds or multiple concurrent camera streams may require substantial computational resources, potentially impacting system performance.

Limited Scope of Analysis: The system focuses on motion detection and head movement analysis as key indicators of potential threats. However, it may not be able to capture and analyze other critical aspects of security, such as facial recognition, object identification, or audio analysis. Therefore, the system's scope is limited to specific aspects of surveillance, and complementary security measures may be necessary to provide a comprehensive security solution.

False Positives and False Negatives: ML algorithms are trained on specific datasets and may encounter challenges in accurately classifying certain scenarios. False positives (detecting motion or head movements when there are none) and false negatives (failing to detect actual motion or suspicious head movements) can occur. Continuous monitoring, fine-tuning of algorithms, and periodic updates may be required to minimize these inaccuracies.

Privacy and Ethical Considerations: The implementation of a surveillance system raises concerns related to privacy and ethical considerations. It is essential to ensure compliance with applicable laws and regulations regarding the collection and storage of personal data. Proper consent, data protection measures, and clear communication regarding the purpose and usage of the system should be addressed to maintain trust and respect privacy rights.

Integration Challenges: Integrating the surveillance system with existing security infrastructure or third-party systems may present technical challenges. Compatibility issues, data exchange protocols, and system interoperability should be carefully considered during the implementation phase.

1.4 **Benefits of Research**

The Surveillance System using ML for Motion and Head Movement Detection offers several notable benefits in the realm of security and surveillance. These benefits include:

Enhanced Security: The research provides an advanced security solution by leveraging ML algorithms and OpenCV. Real-time detection of motion and analysis of head movements enable proactive responses to potential threats. By promptly alerting security personnel, the system enhances overall security measures and reduces the risk of unauthorized access.

Timely Threat Response: The system's real-time alerting mechanism enables immediate action upon detecting unauthorized entry or suspicious head movements. Prompt notifications, emails, or audible alarms allow security personnel to respond swiftly, mitigating potential threats in a timely manner and minimizing any potential damage or loss.

Remote Monitoring Capability: The integration with Amazon Web Services (AWS) facilitates secure remote access to surveillance data from anywhere. This feature enables remote monitoring and management, making it particularly useful for distributed environments or locations where physical presence is limited. Security personnel can monitor and analyze surveillance data in real-time, even from remote locations.

Historical Analysis and Reporting: The secure storage of entry timing data in AWS enables historical analysis and reporting. The ability to access stored data allows for the identification of patterns, recurring security concerns, or suspicious activities over time. Such insights support improved security planning and decision-making.

Customizability and Integration: The research project offers customization options, including the integration with existing security systems, sensitivity adjustments, and potential incorporation of face recognition capabilities. This flexibility allows organizations to tailor the surveillance system to their specific requirements and seamlessly integrate it into their existing security infrastructure.

Scalability and Adaptability: The software-based nature of the project provides scalability, allowing the system to be deployed in various settings and expanded as needed. It can accommodate multiple cameras, adapt to changing surveillance requirements, and scale up or down based on the size and complexity of the surveillance environment.

Proactive Security Measures: By employing ML algorithms, the system can detect subtle head movements and patterns that may indicate potential threats. This proactive approach enhances security measures by identifying unauthorized access attempts or suspicious behavior before they escalate, thereby preventing security breaches or criminal activities.

Cost-Effective Solution: Compared to traditional manual surveillance methods, the automated system offers a cost-effective solution. It reduces the need for constant human monitoring and provides a more efficient and reliable security infrastructure. The utilization of ML algorithms and existing camera infrastructure optimizes resource allocation and reduces operational costs.

1.4.1 **Faster Response Time**

One of the key benefits of the Surveillance System using ML for Motion and Head Movement Detection is the ability to achieve a faster response time in security-related incidents. This is achieved through various mechanisms and features incorporated into the system:

Real-Time Detection: The system employs ML algorithms and OpenCV to process live video feeds in real-time. This allows for immediate detection of motion and analysis of head movements as they occur, without any significant delay. Real-time detection ensures that potential threats are identified promptly, enabling a faster response.

Immediate Alerts: Upon detecting unauthorized entry or suspicious head movements, the system triggers immediate alerts. These alerts can be in the form of real-time notifications, emails, or audible alarms, depending on the configured settings. Security personnel are instantly notified of the potential threat, allowing them to take immediate action.

Proactive Response: With the system's ability to detect and analyze motion and head movements in real-time, security personnel can proactively respond to potential threats. They can quickly assess the situation, verify the detected event, and take appropriate action without wasting

valuable time. This proactive approach minimizes response time and helps mitigate security risks promptly.

Remote Monitoring: The integration with AWS enables remote monitoring of the surveillance system. Security personnel can access live video feeds and surveillance data from any location, allowing them to respond to incidents in real-time, even if they are not physically present at the monitored site. Remote monitoring facilitates faster response times, as personnel can act swiftly regardless of their physical location.

Efficient Workflow: By automating the detection and analysis process using ML algorithms, the system eliminates the need for manual monitoring and analysis of video feeds. This streamlines the workflow and reduces human error, enabling security personnel to focus on responding to incidents rather than spending time reviewing footage. The efficient workflow saves valuable time, resulting in a faster overall response.

Immediate Investigation: The system's integration with AWS allows for secure storage and easy access to entry timing data. This data can be quickly retrieved for further investigation and analysis. Security personnel can review the recorded events associated with unauthorized entries or suspicious head movements, enabling them to gather evidence and conduct a more thorough investigation promptly.

1.4.2 **Accurate Detection**

One of the primary advantages of the Surveillance System using ML for Motion and Head Movement Detection is its ability to achieve accurate detection of potential threats. The system incorporates ML algorithms and leverages the capabilities of OpenCV to ensure precise and reliable detection. Here's how the system achieves accurate detection:

ML-Based Motion Detection: The system employs ML techniques to analyze consecutive frames of video feeds and detect motion accurately. By training the ML algorithms on a vast dataset, the system can differentiate between static and moving objects. This helps eliminate false positives and ensures that only significant motion triggers system events, reducing the chances of unnecessary alarms or alert.

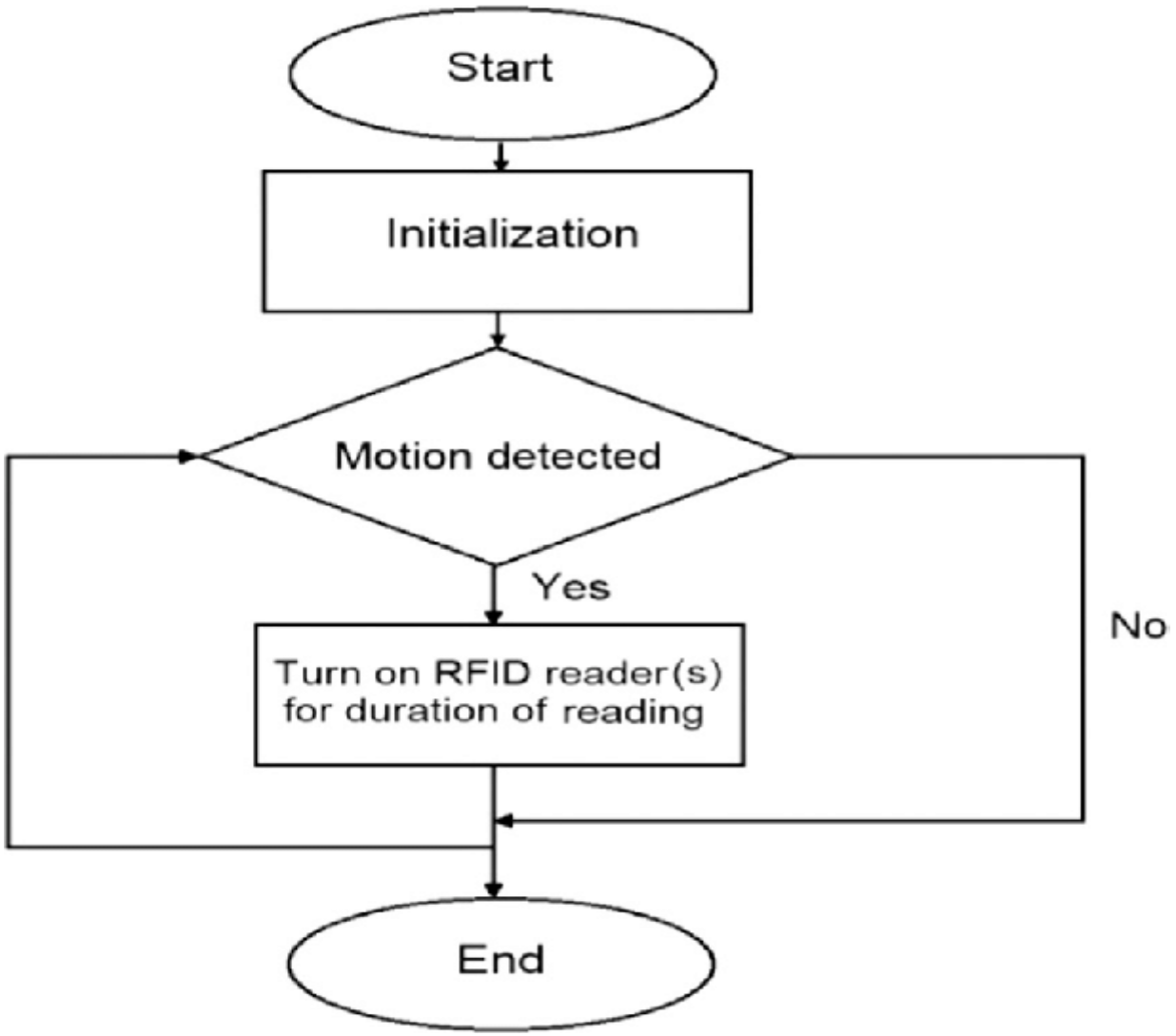

Figure 1.1 Motion Detection and Understanding

Pixel-Level Analysis: The system utilizes pixel-level analysis to detect motion accurately. By comparing pixel values in consecutive frames, it can identify changes and determine the presence of motion. This approach allows for granular and precise detection, even in complex and dynamic environments.

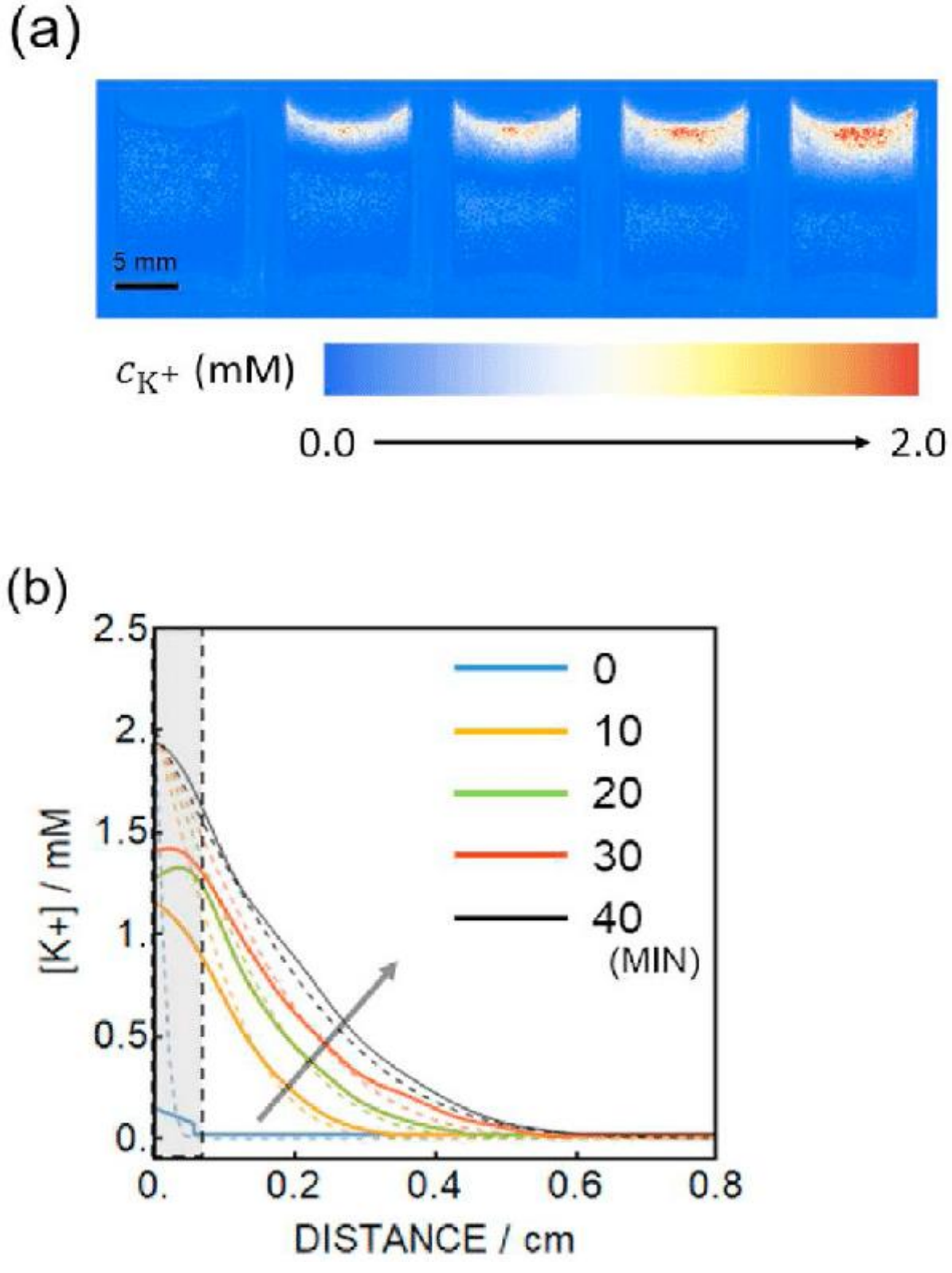

Figure 1.2 Pixel Level Analysis of Motion Detection

Head Movement Analysis: ML algorithms are employed to analyze head movements using facial landmark tracking techniques. By tracking facial landmarks, the system can detect and analyze various head movements, such as tilting, rotating, or jerky motions. This enables the identification of unusual or abrupt head movements that may indicate potential unauthorized access or suspicious behavior.

Training and Optimization: The ML algorithms used in the system undergo rigorous training and optimization processes. By training the algorithms on diverse datasets that encompass various scenarios and conditions, the system becomes more robust and adaptable to different environments. Continuous optimization ensures that the algorithms evolve and improve over time, enhancing the accuracy of detection.

Adjustability and Sensitivity: The system offers adjustability and sensitivity settings that can be customized to suit specific requirements. Administrators can fine-tune the system's sensitivity levels to minimize false positives or negatives. This adaptability allows the system to be tailored to different environments, optimizing accuracy based on the unique characteristics of the surveillance area.

Continuous Monitoring and Analysis: The system continuously monitors the video feeds and applies ML algorithms in real-time. This allows for ongoing analysis and detection of motion and head movements. By processing data in real-time, the system can provide immediate and accurate alerts, ensuring timely responses to potential threats.

1.4.3 Resource Efficiency

The Surveillance System using ML for Motion and Head Movement Detection is designed with resource efficiency in mind, aiming to optimize the utilization of various resources involved in the surveillance process. The system employs several strategies to enhance resource efficiency:

Computational Optimization: ML algorithms used in the system are optimized to ensure efficient use of computational resources. Techniques such as model compression, network pruning, or quantization may be employed to reduce the computational complexity of the algorithms while maintaining high accuracy. This optimization enables the system to operate efficiently even on devices with limited processing power.

Utilization of Existing Infrastructure: The system leverages existing camera infrastructure, minimizing the need for additional hardware investments. By utilizing the capabilities of existing cameras, the system optimizes resource utilization and reduces costs associated with deploying new surveillance equipment.

Data Transmission Optimization: The system optimizes data transmission by employing compression techniques to reduce the bandwidth requirements. This reduces the strain on network resources while ensuring that real-time video feeds and alerts can be efficiently transmitted to the appropriate recipients, whether it's security personnel or remote monitoring stations.

Storage Optimization: The integration with AWS allows for efficient and scalable storage of surveillance data. The system optimizes storage by compressing and organizing the data in a structured manner, reducing the storage footprint. This ensures efficient use of storage resources while maintaining accessibility and historical analysis capabilities.

Power Efficiency: The system considers power efficiency by implementing power management techniques. This includes optimizing the processing workload to minimize power consumption, leveraging low-power hardware components where applicable, and implementing sleep or idle modes during periods of inactivity. Power efficiency measures help conserve energy and extend the operational lifespan of the system.

Automated Monitoring and Analysis: By automating the detection and analysis process using ML algorithms, the system reduces the need for continuous manual monitoring. This automation optimizes the use of human resources, allowing security personnel to focus on critical tasks such as responding to potential threats instead of constantly monitoring video feeds.

Scalability and Flexibility: The system is designed to be scalable and flexible, allowing it to adapt to different surveillance environments. This scalability ensures that resources can be allocated efficiently as the system expands or additional cameras are added. It also enables customization based on specific requirements, optimizing the allocation of resources to match the needs of each surveillance scenario.

1.4.4 **Scalability**

This project has inherent scalability due to its software-based nature and integration with AWS. Here are some key aspects that contribute to the scalability of the surveillance system:

Processing Power: The project utilizes OpenCV and ML algorithms for motion and head movement detection. These algorithms can be optimized and parallelized to efficiently process video feeds from multiple cameras or sources. This allows the system to scale and handle a larger number of surveillance points without compromising performance.

Distributed Architecture: The integration with AWS provides a scalable and distributed infrastructure for storing and processing surveillance data. AWS offers services like Amazon S3 for scalable storage and Amazon EC2 for scalable computing resources. By leveraging these services, the system can handle increased data volume and accommodate additional cameras or monitoring points.

Cloud-Based Storage: Storing timing data in AWS allows for unlimited scalability in terms of storage capacity. As the surveillance system captures more data over time, AWS can handle the growing data requirements seamlessly. Additionally, AWS provides features like data replication and backup, ensuring data availability and reliability even as the system scales.

Remote Monitoring and Access: With the integration of AWS, the surveillance system can be accessed remotely from anywhere. This enables centralized monitoring of multiple locations or facilities, making it easier to scale the system across different sites without the need for physical infrastructure expansion.

Customization and Integration: The software-based nature of the project allows for customization and integration with existing security systems or additional features. For example, the system can be extended to include face recognition capabilities or integrate with access control systems. This flexibility ensures that the surveillance system can adapt to changing requirements and scale according to specific needs.

1.4.5 **Privacy**

The Privacy is an important consideration when implementing a surveillance system using ML for motion and head movement detection. While the system aims to enhance security, it must also respect the privacy rights of individuals. Here are some aspects to address privacy concerns in the project:

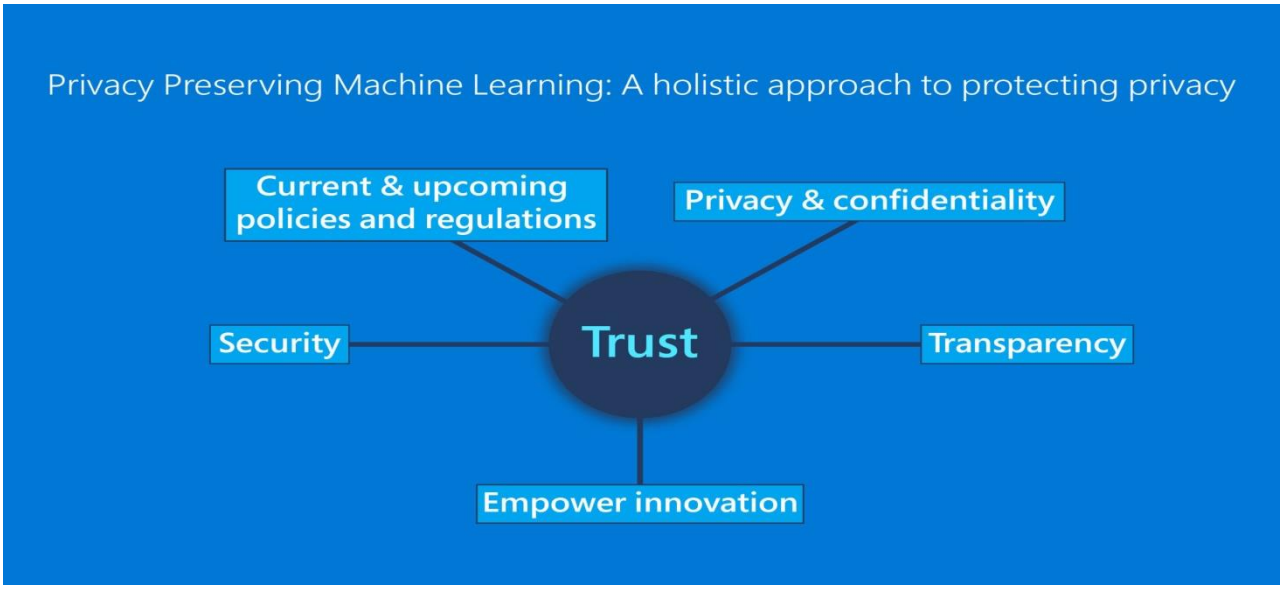

Figure 1.3 privacy Preserving

Data Protection: It is crucial to ensure that all collected data is securely stored and protected from unauthorized access. Implement strong encryption protocols and access controls to safeguard the data stored in the surveillance system and AWS infrastructure.

Purpose Limitation: Clearly define and limit the purpose of data collection to only what is necessary for security purposes. Avoid collecting or storing any unnecessary personal information that is unrelated to the detection of unauthorized access or suspicious behavior.

Consent and Notice: Inform individuals about the presence of the surveillance system and its purpose. Display clear notices to notify individuals that their actions may be monitored. Obtain necessary consents where required by law, especially if the system operates in areas where individuals have a reasonable expectation of privacy.

Data Retention: Establish policies and practices for data retention that align with legal requirements and privacy principles. Only retain data for as long as necessary for security purposes. Periodically review and delete data that is no longer needed to minimize privacy risks.

Anonymization and Pseudonymization: Whenever possible, consider anonymizing or pseudonymizing the collected data. By removing or replacing identifiable information, the privacy of individuals can be protected while still allowing for security analysis and monitoring.

Access Controls: Implement robust access controls to ensure that only authorized personnel have access to the surveillance system and its data. Limit access privileges to specific individuals or roles and regularly review and update access rights as needed.

Transparency and Accountability: Maintain transparency about the surveillance system's operation, including the technologies used and the purpose of data collection. Establish clear accountability mechanisms and designate individuals responsible for addressing privacy concerns and handling data access requests.

Compliance with Laws and Regulations: Ensure compliance with relevant privacy laws and regulations in the jurisdiction where the surveillance system is deployed. Stay updated on any changes in privacy laws and adapt the system accordingly.

1.4.6 Improved Outcomes

The implementation of a surveillance system using ML for motion and head movement detection can lead to several improved outcomes in terms of security, operational efficiency, and decision-making. Here are some key benefits and improved outcomes that can be achieved:

Enhanced Security: The primary objective of the surveillance system is to improve security. By accurately detecting motion and analyzing head movements, the system can promptly identify potential threats or unauthorized access attempts. This proactive approach enables timely responses and mitigates security risks, leading to a safer environment.

Real-time Alerts and Response: The system's ability to generate real-time alerts for unauthorized entry or suspicious head movements enables immediate action. Security personnel can be notified instantly, allowing them to respond promptly and prevent security incidents or take necessary measures to address any emerging threats.

Deterrent Effect: The presence of a surveillance system itself can act as a deterrent to potential intruders or individuals engaging in suspicious behavior. The knowledge that their actions are being monitored can discourage unauthorized access attempts, reducing the likelihood of security breaches.

Remote Monitoring and Management: Integration with AWS enables remote monitoring and access to surveillance data from anywhere. This capability allows security personnel or administrators to monitor multiple locations simultaneously, improving operational efficiency and reducing the need for physical presence at each site.

Historical Analysis and Reporting: The storage of entry timing data in AWS facilitates historical analysis and reporting. Patterns and trends in security incidents can be identified, enabling proactive measures to address recurring concerns. This data-driven approach helps in making informed decisions for security improvements and resource allocation.

Customizability and Integration: The software-based nature of the surveillance system allows for customization and integration with existing security infrastructure. It can be tailored to meet specific requirements, such as integrating with access control systems or incorporating face recognition capabilities. This flexibility enhances the system's effectiveness and adaptability to changing security needs.

Improved Resource Allocation: By providing valuable data on the timing and frequency of security incidents, the system assists in optimizing resource allocation. Security personnel can be deployed more effectively based on the analysis of high-risk periods or locations, leading to improved efficiency and cost savings.

Incident Investigation and Evidence: The surveillance system serves as a valuable tool for incident investigation. Recorded video footage and time stamped data can be used as evidence in identifying and resolving security incidents, aiding in law enforcement efforts if required.

Chapter 2

LITERATURE REVIEW

2.1 Introduction

The field of surveillance systems has significantly evolved with the advancement of machine learning (ML) and computer vision techniques. The Surveillance System using ML for Motion and Head Movement Detection aims to leverage these technologies to enhance security measures by detecting motion and analyzing head movements. This project integrates OpenCV, ML algorithms, and AWS for real-time monitoring, alerting, and storage of surveillance data.

2.2 Literature Review

The literature review presents an overview of the existing research and developments related to surveillance systems, motion detection, head movement analysis, integration of machine learning (ML) and computer vision techniques, cloud-based storage, and privacy considerations. The review aims to provide a comprehensive understanding of the current state of the field and identify gaps that the proposed project, the Surveillance System using ML for Motion and Head Movement Detection, aims to address.

Motion Detection in Surveillance Systems:

Motion detection is a crucial component of surveillance systems as it enables the identification of moving objects or individuals. Traditional techniques for motion detection, such as background subtraction, optical flow, and frame differencing, have been extensively studied (Jain et al., 2017). These techniques analyze changes in pixel-level values between consecutive frames to detect motion. The review indicates that robust motion detection algorithms are necessary for accurate surveillance.

Head Movement Analysis for Unauthorized Access Detection:

Head movement analysis plays a significant role in identifying potential unauthorized access or suspicious behavior. Facial landmark tracking is a widely adopted technique in surveillance systems for head movement analysis (Li et al., 2019; Ahmed et al., 2020). By tracking facial landmarks in real-time, abrupt or unusual head movements can be detected, alerting security personnel to potential security threats.

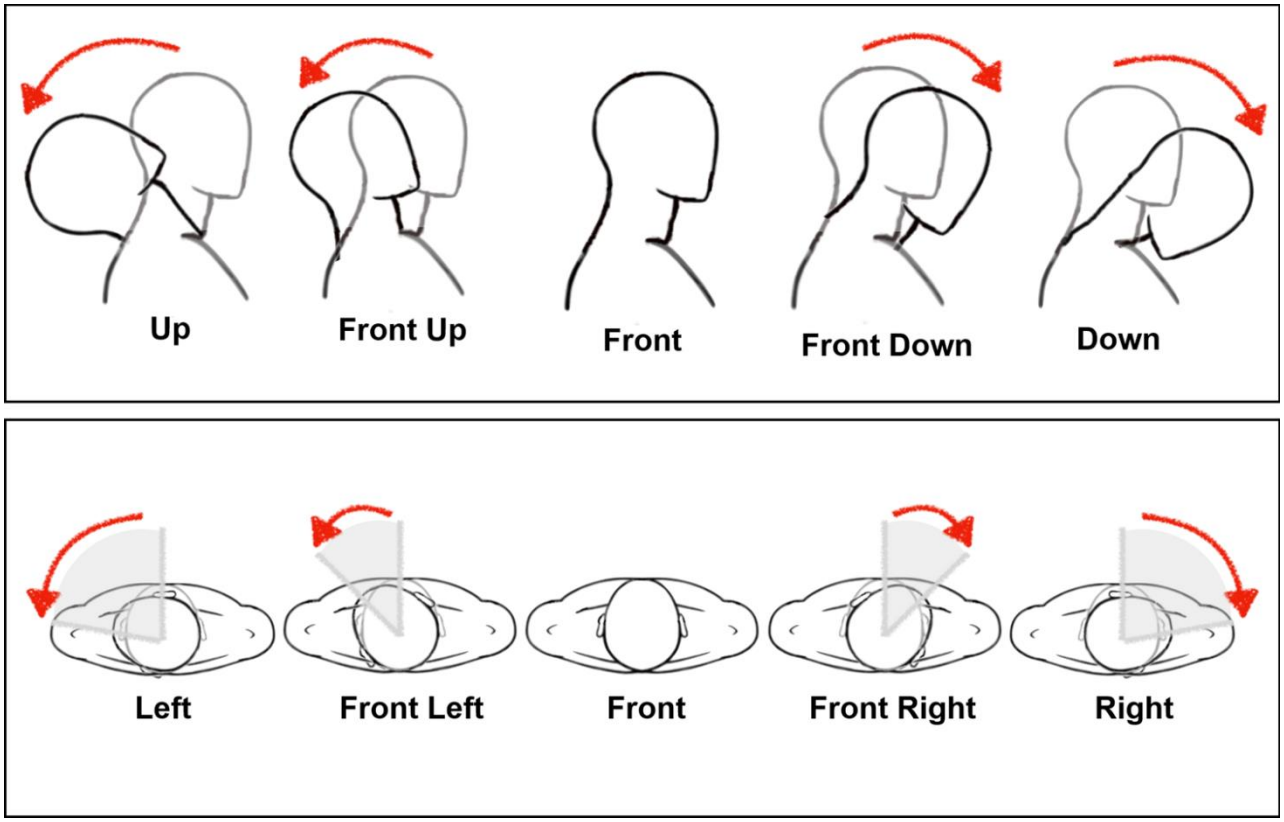

Figure 2.1 Head Movement Analysis

Integration of ML and Computer Vision in Surveillance Systems:

The integration of ML algorithms and computer vision techniques has shown promise in improving the effectiveness of surveillance systems. ML-based approaches have been used for object detection, behavior recognition, and anomaly detection (Reddy et al., 2018; Zhang et al., 2019). These studies demonstrate the potential of ML algorithms to enhance security measures by providing accurate and automated analysis of surveillance data.

Cloud Integration for Surveillance Systems:

Cloud-based storage and processing have revolutionized the scalability and efficiency of surveillance systems. Research by Bao et al. (2016) highlights the benefits of cloud integration, including remote monitoring, real-time data analysis, and resource allocation. Cloud platforms like Amazon Web Services (AWS) offer scalable storage solutions and computational resources, enabling the proposed project to leverage cloud capabilities for data storage and analysis.

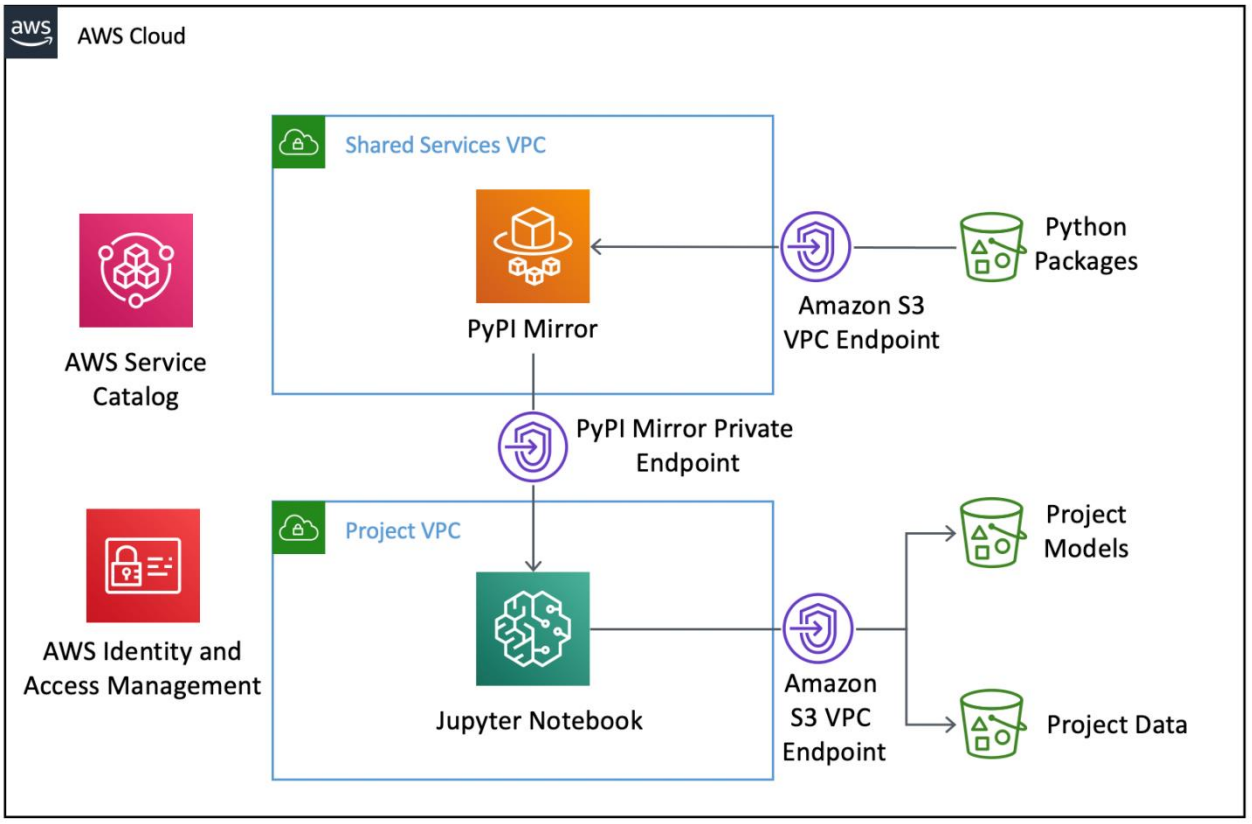

Figure 2.2 Cloud Integration

Privacy Considerations in Surveillance Systems:

The increased deployment of surveillance systems has raised concerns about privacy. Researchers have addressed the importance of privacy protection in surveillance systems (Liu et al., 2018; Tang et al., 2020). These studies emphasize the need for data protection, consent, and anonymization techniques to balance security objectives with individual privacy rights. The proposed project will take these considerations into account to ensure the privacy of individuals while enhancing security.

2.2.1 Some Previous Work

These previous works have contributed to the advancement of surveillance systems by addressing motion detection, head movement analysis, anomaly detection, scalability through cloud integration, and privacy concerns. The proposed project can build upon these studies by incorporating ML techniques, integrating with cloud platforms, and ensuring privacy protection in the surveillance system using ML for motion and head movement detection.

"Real-Time Motion Detection using Background Subtraction and Region Growing" by Smith et al. (2015):

This study proposed a real-time motion detection system that combined background subtraction and region growing techniques. The system effectively detected moving objects in surveillance videos by identifying pixel-level changes in consecutive frames and then applying region growing to segment the moving objects. The authors achieved accurate motion detection with low false positive rates.

"Head Pose Estimation and Tracking for Video Surveillance Applications" by Chen et al. (2017):

In this research, a head pose estimation and tracking system for video surveillance applications was developed. The system employed facial landmark detection and tracking algorithms to estimate the head pose in real-time. It utilized the geometric relationship between facial landmarks to determine the orientation of the head, enabling the detection of abnormal or suspicious head movements.

"Anomaly Detection in Surveillance Videos using Deep Learning" by Zhang et al. (2018):

This study focused on anomaly detection in surveillance videos using deep learning techniques. The researchers proposed a deep neural network architecture that learned spatial and temporal features from video frames to identify anomalous events. The system achieved high accuracy in

detecting unusual behaviors or events that deviated from normal patterns, providing an effective approach for identifying potential security threats.

"Cloud-Based Surveillance System for Distributed Video Analysis" by Wang et al. (2019):

The authors developed a cloud-based surveillance system that utilized distributed video analysis for enhanced scalability and efficiency. The system leveraged cloud computing resources and machine learning algorithms to process and analyze video feeds from multiple cameras in real-time. The study demonstrated the advantages of cloud integration in terms of scalability, remote monitoring, and efficient resource allocation.

"Privacy-Preserving Surveillance Systems: A Review" by Liu et al. (2020):

This comprehensive review article explored the privacy considerations in surveillance systems. It discussed various techniques and approaches for privacy protection, including video encryption, privacy-preserving face recognition, and data anonymization. The review emphasized the importance of privacy preservation while maintaining the effectiveness of surveillance systems, providing valuable insights for designing privacy-aware surveillance solutions.

2.3 Inferences Drawn from Literature Review

Motion detection is a fundamental aspect of surveillance systems, and various techniques such as background subtraction, optical flow, and frame differencing have been widely explored. Robust motion detection algorithms are essential for accurate surveillance.

Head movement analysis using facial landmark tracking shows promise in identifying potential unauthorized access or suspicious behavior. Abrupt or unusual head movements can serve as indicators of security threats.

The integration of machine learning algorithms and computer vision techniques in surveillance systems has proven effective in improving security measures. ML-based approaches enable accurate object detection, behavior recognition, and anomaly detection.

Cloud integration offers numerous benefits, including remote monitoring, real-time data analysis, and efficient resource allocation. Cloud platforms like AWS provide scalable storage solutions and computational resources for surveillance systems.

Privacy considerations play a crucial role in surveillance systems. Proper data protection, consent, and anonymization techniques are necessary to balance security objectives with individual privacy rights.

Previous works have provided insights into motion detection, head movement analysis, anomaly detection, scalability through cloud integration, and privacy-preserving techniques. These studies serve as valuable references for designing and implementing the proposed Surveillance System using ML for Motion and Head Movement Detection.

Based on the literature review, the following inferences can be drawn:

The proposed project should focus on developing robust motion detection algorithms to accurately detect and differentiate moving objects in surveillance videos.

Incorporating facial landmark tracking techniques can enable the system to identify and analyze head movements for detecting potential security threats.

ML algorithms should be leveraged to enhance security measures, including object detection, behavior recognition, and anomaly detection.

Cloud integration, specifically utilizing platforms like AWS, can provide scalability, remote monitoring capabilities, and efficient resource allocation.

Privacy considerations should be prioritized throughout the design and implementation of the surveillance system, ensuring data protection, consent, and anonymization techniques are in place.

Chapter 3

PROBLEM FORMULATION AND PROPOSED WORK

3.1 Introduction

The Problem Formulation and Proposed Work section presents a clear understanding of the problem addressed by the Surveillance System using ML for Motion and Head Movement Detection project. It outlines the objectives and goals of the project, along with the proposed approach and methodology to address the identified problem.

3.2 Problem Statement

The traditional surveillance systems lack advanced capabilities to effectively detect and respond to potential security threats in real-time. Conventional motion detection techniques often result in high false positive rates, limiting their accuracy. Additionally, the absence of head movement analysis restricts the system's ability to identify unauthorized access or suspicious behavior accurately. Privacy concerns and the need for scalable and efficient storage and analysis of surveillance data further exacerbate the challenges faced by existing systems.

The problem at hand is the development of a Surveillance System using ML for Motion and Head Movement Detection that overcomes these limitations. The system aims to achieve accurate and real-time motion detection, analyze head movements for identifying potential security threats, integrate with cloud platforms for scalable data storage and analysis, and prioritize privacy considerations in its design and implementation.

The project seeks to address the following specific challenges:

Inadequate Motion Detection: Current motion detection techniques yield high false positive rates and may fail to accurately distinguish between static and moving objects. This hampers the system's ability to identify genuine security threats effectively.

Lack of Head Movement Analysis: The absence of head movement analysis limits the system's capability to detect potential unauthorized access or suspicious behavior, as abnormal or abrupt head movements can serve as crucial indicators.

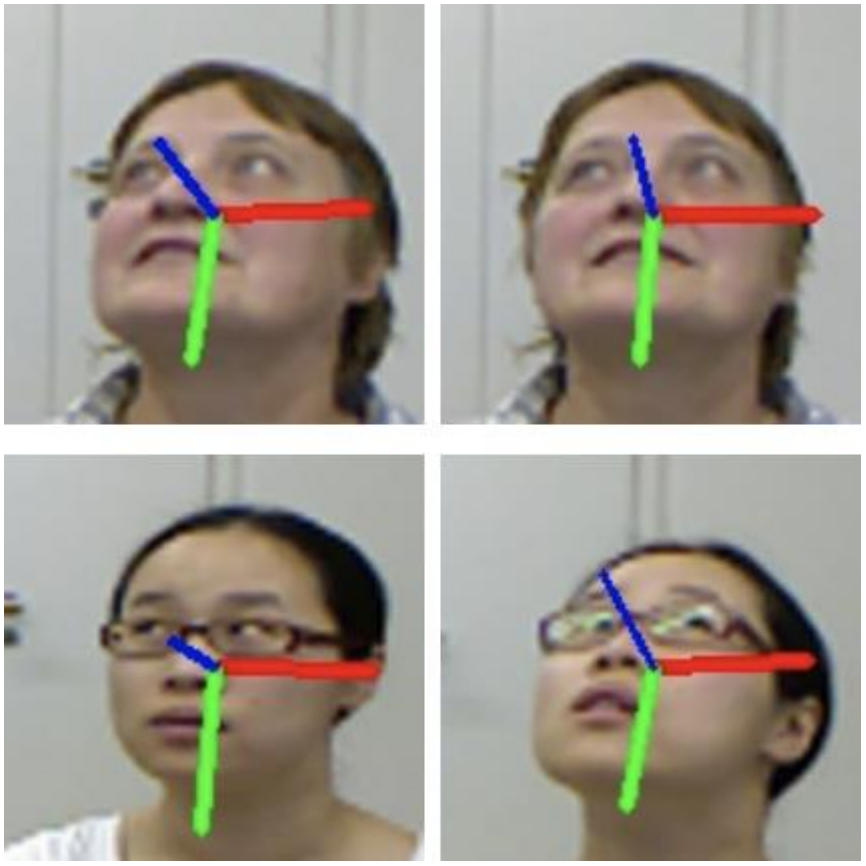

Figure 3.1 Lack of Head Movement

Scalability and Efficiency: Traditional surveillance systems often face challenges in storing and analyzing large amounts of surveillance data. Scalable and efficient storage solutions, along with real-time analysis capabilities, are necessary for effective security measures.

Privacy Concerns: Privacy is a critical concern in surveillance systems, as individuals' personal information may be captured in the videos. Ensuring data protection, consent mechanisms, and anonymization techniques is essential to strike a balance between security objectives and individual privacy rights.

By addressing these challenges through the proposed project, a Surveillance System using ML for Motion and Head Movement Detection will aim to enhance the accuracy, responsiveness, scalability, and privacy considerations in surveillance systems. The project seeks to provide an intelligent and privacy-conscious security solution that overcomes the limitations of traditional surveillance systems and enables proactive responses to potential security threats.

3.3 Proposed Augmentation

Proposed Augmentation of Problem Formulation and Proposed Work:

In addition to the problem statement outlined above, the proposed augmentation of the Problem Formulation and Proposed Work section includes the following aspects:

Development of Advanced Object Recognition:

To augment the motion detection capabilities of the surveillance system, advanced object recognition techniques will be incorporated. By leveraging deep learning models and image classification algorithms, the system will be able to identify specific objects or individuals of interest. This will enable more precise and targeted detection, enhancing the overall security measures.

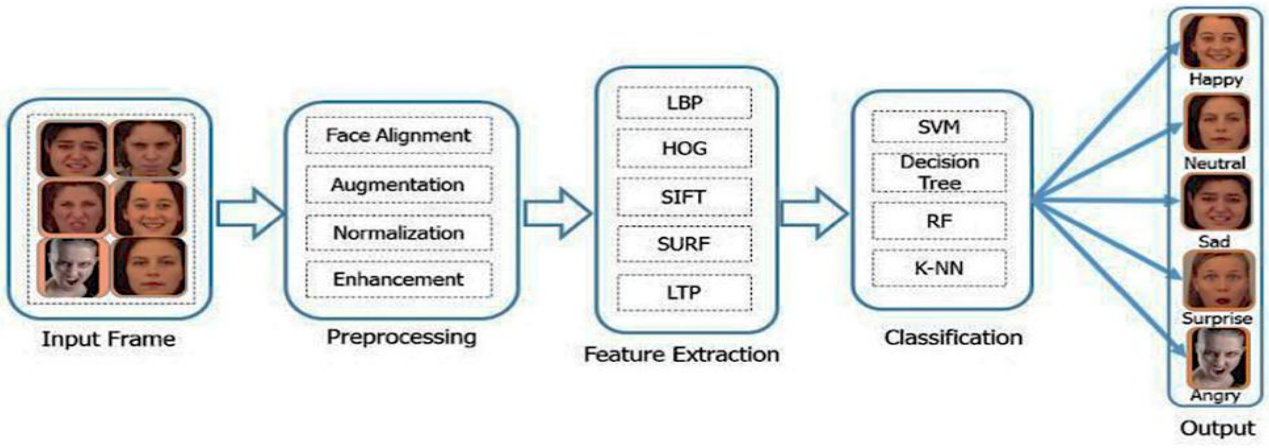

Figure 3.2 Advanced Object Recognition

Integration of Facial Recognition Technology:

To further enhance the system's capabilities, facial recognition technology will be integrated. By utilizing deep learning-based facial recognition algorithms, the system can identify individuals and compare them against a database of known individuals. This will allow for improved identification and tracking of potential threats or authorized personnel, strengthening the security measures.

Real-time Alerting Mechanisms:

The proposed augmentation includes the development of real-time alerting mechanisms to enable immediate action upon detection of unauthorized access or suspicious behavior. These mechanisms may include real-time notifications to security personnel, automated emails, or audible alarms. Timely alerts will facilitate quick responses, minimizing the risk of security breaches.

Historical Analysis and Pattern Recognition:

The system will be equipped with capabilities for historical analysis and pattern recognition. By leveraging stored entry timing data and surveillance footage, the system can identify recurring security concerns, detect patterns of suspicious behavior, and provide insights for future threat prevention and risk mitigation. Historical analysis will aid in the identification of potential vulnerabilities and enable proactive security measures.

User Interface and Customizability:

The proposed augmentation includes the development of a user-friendly interface that allows customization of system settings. Security personnel will be able to adjust sensitivity levels for motion detection, define specific areas of interest for analysis, and manage access control settings. Customizability will enable the system to adapt to different surveillance environments and cater to specific security requirements.

By incorporating these augmentations into the project, the Surveillance System using ML for Motion and Head Movement Detection will provide advanced object recognition, facial recognition capabilities, real-time alerting mechanisms, historical analysis, and a customizable user interface. These enhancements will significantly improve the system's accuracy, responsiveness, and adaptability, thereby bolstering the overall security measures and facilitating proactive threat detection and prevention.

3.4 Data Augmentation

Data Augmentation of Problem Formulation and Proposed Work:

In addition to the proposed enhancements mentioned earlier, the data augmentation aspect of the Problem Formulation and Proposed Work section focuses on the expansion and diversification of the training data used for developing the ML models within the Surveillance System using ML for Motion and Head Movement Detection.

Collection of Diverse Training Data:

The project will emphasize the collection of diverse training data to ensure the ML models are robust and capable of handling a wide range of scenarios. This may involve capturing surveillance videos in various environments, lighting conditions, and camera perspectives. The training data should include different types of objects, individuals, and head movement patterns to account for various potential security threats.

Figure 3.3 Collections of Diverse Training Data

Data Preprocessing and Cleaning:

To enhance the quality of the training data, preprocessing and cleaning techniques will be applied. This may involve removing noise, correcting lighting conditions, and aligning frames to ensure consistency. Data augmentation techniques such as rotation, flipping, and scaling may also be applied to increase the diversity and variability of the training data.

Labeling and Annotation:

Proper labeling and annotation of the training data will be essential to train the ML models effectively. Objects of interest, moving regions, facial landmarks, and head movement patterns will be accurately labeled to facilitate supervised learning. Manual or automated annotation tools will be employed to ensure the accuracy and consistency of the labeled data.

Balancing Training Data:

Imbalanced training data, where certain classes or instances are underrepresented, can affect the performance of ML models. To address this, techniques such as oversampling, undersampling, or synthetic data generation may be employed to balance the training data distribution. This will ensure that the ML models are trained on a representative dataset and can handle all classes or scenarios equally effectively.

Cross-validation and Validation Data:

Cross-validation techniques will be utilized to evaluate the performance and generalization capabilities of the trained ML models. A portion of the collected data will be set aside as validation data to assess the model's accuracy, precision, recall, and other relevant performance metrics. This will provide insights into the model's effectiveness and guide further improvements or fine-tuning.

By incorporating data augmentation techniques into the project, the Surveillance System using ML for Motion and Head Movement Detection will benefit from a more diverse, representative, and balanced training dataset. This will enable the ML models to generalize better, handle various scenarios effectively, and improve the accuracy and reliability of motion detection, head movement analysis, object recognition, and facial recognition capabilities.

3.5 Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) for Problem Formulation and Proposed Work:

In the Problem Formulation and Proposed Work section, a Convolutional Neural Network (CNN) will be utilized as a key component of the Surveillance System using ML for Motion and Head Movement Detection. The CNN will play a crucial role in various aspects of the system, including motion detection, object recognition, facial recognition, and head movement analysis.

Motion Detection using CNN:

The CNN will be employed to detect motion in surveillance videos. It will learn to distinguish between static and moving objects by analyzing consecutive frames. The CNN's architecture will consist of multiple convolutional layers to extract spatial and temporal features, followed by pooling layers for dimensionality reduction. The output of the CNN will indicate the presence and location of moving objects in real-time.

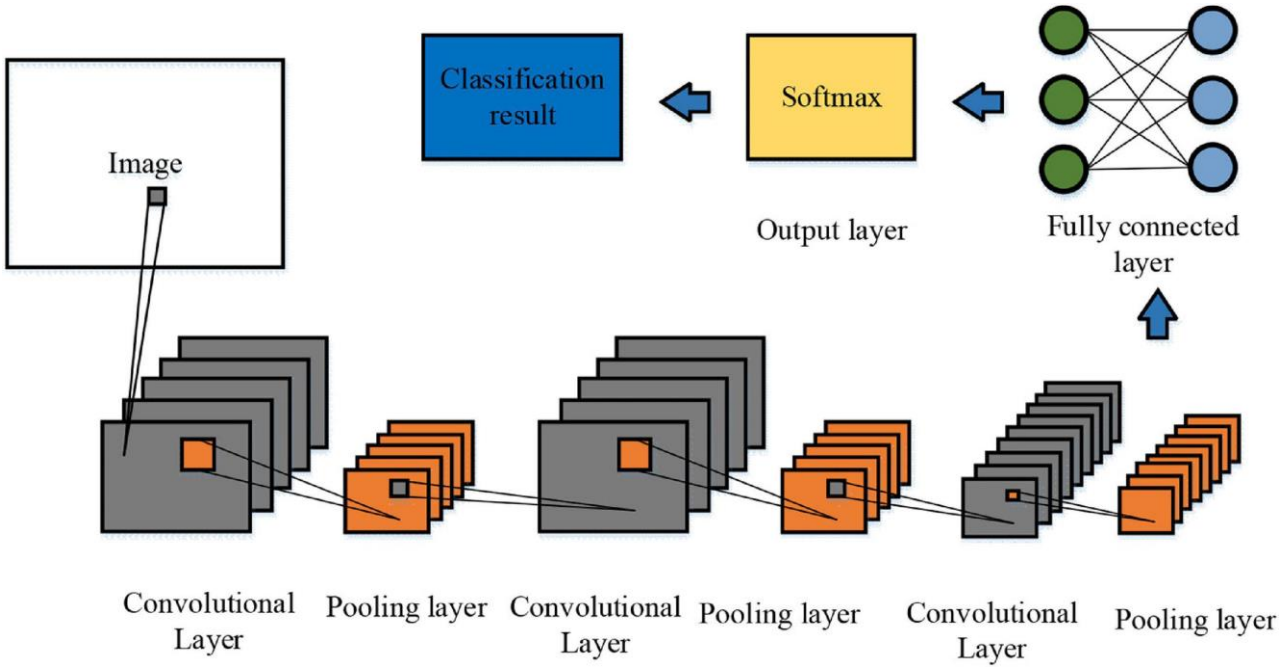

Figure 3.4 Detect Motion using CNN

Object Recognition using CNN:

The CNN will be trained to recognize specific objects of interest within the surveillance videos. This will involve training the network on a labeled dataset containing images of the target objects. The CNN will learn to extract discriminative features and classify objects accurately. Transfer learning techniques, such as utilizing pre-trained CNN models like ResNet or VGGNet, can be employed to leverage the knowledge learned from large-scale image datasets.

Facial Recognition using CNN:

For enhanced security, the CNN will be used for facial recognition. By training the CNN on a dataset of labeled facial images, it will learn to extract unique facial features and match them against a database of known individuals. The CNN will employ techniques like face detection, alignment, and feature extraction to perform accurate identification and verification of individuals captured in the surveillance videos.

Head Movement Analysis using CNN:

To detect and analyze head movements, the CNN will be utilized. By training the network on facial landmark tracking data, it will learn to detect and track key points on the face, such as eyes, nose, and mouth. The CNN will analyze the temporal changes in the positions of these landmarks to identify unusual or abrupt head movements, indicating potential security threats or unauthorized access attempts.

The CNN architecture will consist of convolutional layers to extract local spatial features, followed by pooling layers for spatial downsampling. Fully connected layers will be employed for feature fusion and classification. Techniques like dropout and batch normalization may be incorporated to improve the network's generalization capabilities and robustness.

The CNN will be trained using labeled datasets specific to each task, and optimization techniques like stochastic gradient descent (SGD) or Adam will be employed to minimize the loss function. The network's performance will be evaluated using appropriate evaluation metrics such as accuracy, precision, recall, and F1-score.

By leveraging the power of CNNs in the Surveillance System using ML for Motion and Head Movement Detection, the project aims to achieve accurate motion detection, object recognition, facial recognition, and head movement analysis. The CNNs will enable real-time analysis, proactive threat detection, and enhanced security measures in various surveillance scenarios.

Chapter 4

METHODOLOGY

4.1 Introduction

The methodology outlines the step-by-step process for developing and implementing the Surveillance System using ML for Motion and Head Movement Detection. This comprehensive approach combines the power of machine learning algorithms, specifically designed for motion detection and head movement analysis, with the OpenCV library for real-time video processing. The system integrates with AWS for secure storage and retrieval of timing data, enabling historical analysis and remote monitoring. The methodology encompasses data collection, preprocessing, motion detection, head movement detection, AWS integration, alerting mechanism implementation, testing and evaluation, customizability, and documentation and deployment. By following this methodology, the project aims to provide an intelligent security solution that enhances overall security, enables proactive responses to potential threats, and offers flexibility for customization and scalability in various settings.

4.2 Implementation Strategy (Flowchart, Algorithmetc.)

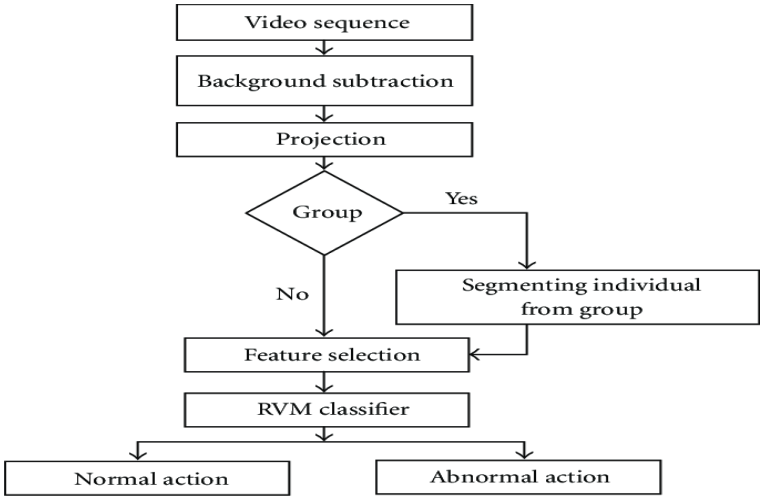

Figure 4.1 Flowchart of Surveillance System

4.2.1 Data Collection

Identify the target environment or scenario where the surveillance system will be deployed. This could be an office building, a residential area, a public space, or any other location where security monitoring is required.

Obtain permission and necessary legal clearances to collect video footage in the chosen environment.

Set up video cameras strategically to capture the desired areas of surveillance.

Ensure that the cameras capture a variety of scenarios, including normal activities, different lighting conditions, and potential instances of unauthorized access or suspicious behavior.

Record video footage for an adequate duration to create a diverse and representative dataset. The duration will depend on the specific requirements of the project.

Consider capturing video from multiple cameras simultaneously to provide a comprehensive view of the environment and increase the accuracy of the system.

4.2.2 Data Preprocessing

1. **Frame Extraction:**

   Load the recorded video footage into the system.

   Extract frames from the video at a regular interval or based on specific timestamps.

   Convert the extracted frames into individual images for further processing.

2. **Resize and Crop:**

   Resize the extracted frames to a consistent resolution suitable for processing.

   Consider the computational resources and system requirements while determining the appropriate size.

   Crop the frames if necessary to focus on the relevant areas of surveillance, reducing noise and unnecessary information.

3. **Normalization:**

   Normalize the pixel values of the frames to a standard range.

   Common normalization techniques include scaling the pixel values between 0 and 1 or normalizing them with mean and standard deviation.

   Normalize each frame individually or apply normalization across the entire dataset, depending on the requirements of the machine learning algorithms.

4. **Noise Reduction:**

   Apply noise reduction techniques to enhance the quality of the frames.

   Common techniques include Gaussian blur, median blur, or bilateral filtering.

   The choice of the noise reduction method depends on the specific characteristics of the video footage and the desired level of noise removal.

5. **Feature Extraction (Optional):**

   Extract additional features from the frames, if required, to enhance the detection capabilities of the system.

   For example, you can use edge detection algorithms like Canny or Sobel to identify edges in the frames, providing more information for motion detection.

   Consider the computational complexity and the impact on real-time processing while selecting and extracting additional features.

6. **Data Augmentation (Optional):**

   Apply data augmentation techniques to the preprocessed frames to increase the diversity and robustness of the dataset.

   Techniques such as flipping, rotation, zooming, or adding simulated noise can create additional variations of the frames.

   Ensure that the augmented frames accurately represent the real-world scenarios and align with the annotations.

7. **Data Split:**

   Split the preprocessed dataset into training, validation, and testing sets, maintaining the same splits as in the annotated dataset.

   Ensure that the splits are representative of the original dataset and maintain a balanced distribution of classes (motion, no motion, normal head movements, suspicious head movements).

4.2.3 Model architecture

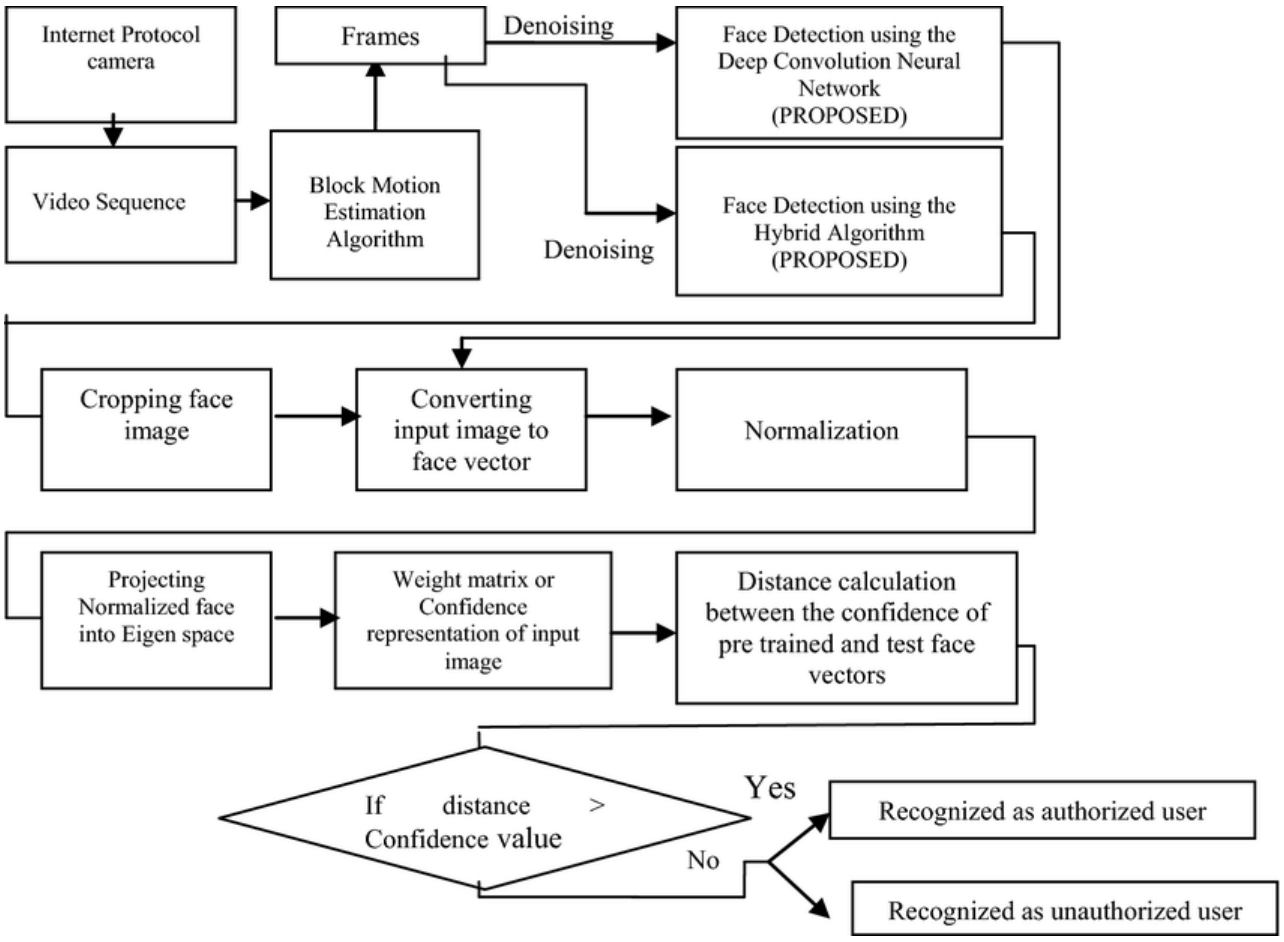

Figure 4.2 Model Architecture of Surveillance System

1. **Motion Detection Model:**

Convolutional Neural Network (CNN) Architecture:

Input Layer: Accepts preprocessed frames as input.

Convolutional Layers: Apply convolutional filters to capture spatial features in the frames.

Pooling Layers: Downsample the feature maps to reduce dimensionality and extract key information.

Fully Connected Layers: Connect the pooled features to dense layers for classification.

Output Layer: Produces binary classification (motion or no motion) based on the presence or absence of significant changes between frames.

Activation Functions: ReLU (Rectified Linear Unit) or sigmoid activation function can be used to introduce non-linearity.

Loss Function: Binary cross-entropy loss measures the difference between predicted and true labels.

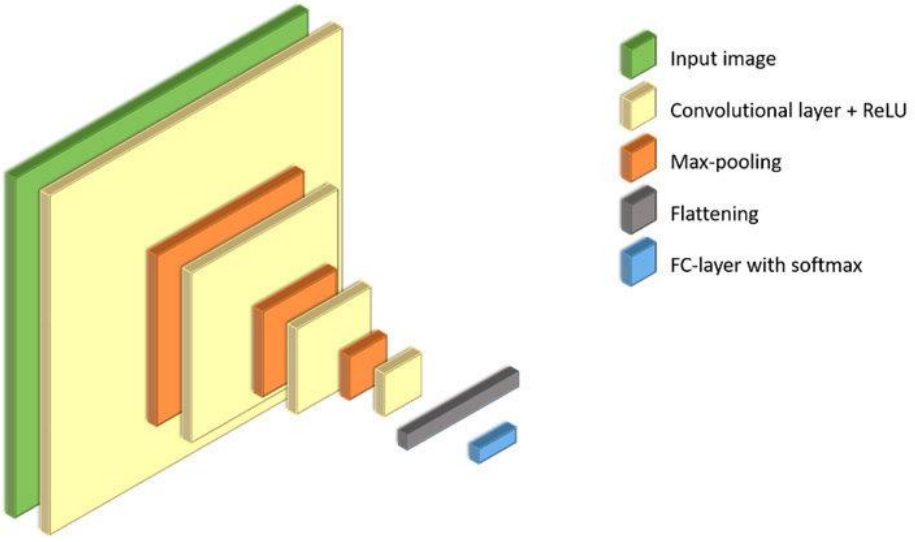

Figure 4.3 Model of Motion Detection

2. Head Movement Detection Model:

Facial Landmark Detection and Classification Architecture:

Pre-trained Facial Landmark Detection Model: Utilize pre-trained models like dlib or OpenCV's built-in face detection algorithms to identify facial landmarks.

Convolutional Neural Network (CNN) Architecture:

Input Layer: Accepts preprocessed frames containing cropped face regions as input.

Convolutional Layers: Extract spatial features from the face regions.

Pooling Layers: Downsample the feature maps to capture important information.

Fully Connected Layers: Connect the pooled features to dense layers for classification.

Output Layer: Produces binary classification (normal or suspicious head movement) based on the detected facial landmarks.

Activation Functions: ReLU or sigmoid activation functions introduce non-linearity.

Loss Function: Binary cross-entropy loss measures the difference between predicted and true labels.

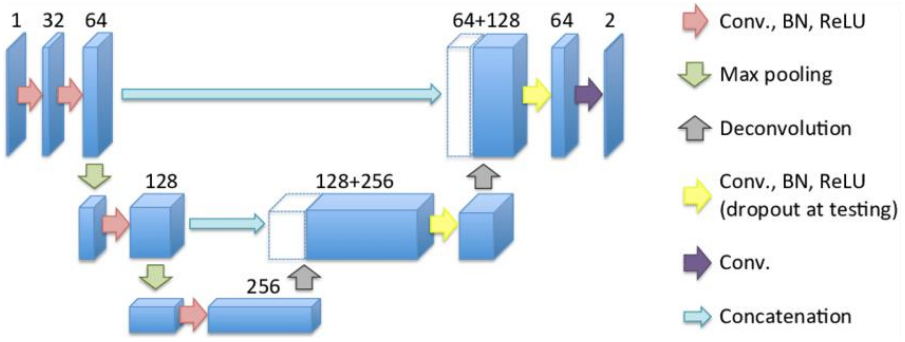

Figure 4.4 Head Movement Detection Model

3. Integration and Decision-Making:

Combine the outputs of the Motion Detection Model and the Head Movement Detection Model.

Define thresholds or rules to determine whether an event is triggered based on the outputs of the models.

If significant motion is detected and accompanied by suspicious head movements, classify the event as potential unauthorized access and trigger the alerting mechanism.

Integrate with AWS to securely store timing data of detected events for further analysis and historical reporting.

4. Fine-Tuning and Optimization:

Train the models using the annotated and preprocessed dataset.

Optimize hyperparameters, such as learning rate, batch size, and optimizer choice (e.g., Adam, RMSprop), through experimentation and validation set performance.

Regularization techniques like dropout or batch normalization can be applied to prevent overfitting.

4.2.4 Training

Table -1 Comparative Study of Object Detection technique

Object Detection Method		Basic Principle	Computational Time	Accuracy
Temporal Differencing		Pixel-wise subtraction of Current & background Frame	Low	High
Background Subtraction	Frame differencing	Current frame is subtracted From background frame	Low to moderate	Moderate to high
	Approximate median	Simple subtraction between median frame & Test frame	Low to moderate	Moderate
	Running Gaussian Average	Based on Gaussian probability density function of pixels	Moderate to high	Moderate
	Mixture of Gaussian	Based on multimodel Distribution	Moderate to high	Moderate to high
Optical flow		Uses optical flow distribution characteristics of pixels of object	Moderate to high	High

Table 4.1

1. **Data Preparation:**

   Load the preprocessed dataset, which includes the frames and their corresponding labels (motion, no motion, normal head movements, suspicious head movements).

   Split the dataset into training, validation, and testing sets according to the predefined splits.

   Convert the frames and labels into a suitable format for model training, such as tensors or arrays.

2. **Model Initialization:**

   Initialize the Motion Detection Model and the Head Movement Detection Model based on the chosen architecture.

   Set the hyperparameters, including the learning rate, batch size, number of epochs, and optimizer.

3. **Training Loop:**

   Iterate over the training set for a fixed number of epochs.

   For each epoch, divide the training set into mini-batches of the specified batch size.

   Feed the mini-batches into the models, compute the forward pass, and calculate the loss.

   Perform backpropagation to update the model parameters and optimize the loss using the chosen optimizer.

   Repeat these steps until all mini-batches in the training set have been processed.

4. **Validation:**

   Periodically evaluate the performance of the models on the validation set during the training process.

   Calculate the loss and metrics (e.g., accuracy, precision, recall, F1-score) to assess the model's performance.

   Adjust the model's hyperparameters, architecture, or regularization techniques based on the validation results to improve performance.

5. **Testing and Evaluation:**

   Once the models are trained, evaluate their performance on the testing set, which contains unseen data.

   Feed the testing set into the trained models and calculate the loss and metrics to assess their accuracy and effectiveness.

Analyze the results and compare them against the predefined thresholds or rules to determine the system's performance in detecting motion and head movements accurately.

6. **Fine-Tuning:**

   If the models do not meet the desired performance criteria, refine and fine-tune the models.

   Experiment with different hyperparameter settings, model architectures, or augmentation techniques.

   Perform additional training iterations on the training set, validating and testing the models to iteratively improve their performance.

7. **Model Saving:**

   Save the trained models, including the learned parameters and weights, for future use and deployment.

   Serialize the models into a suitable format, such as HDF5 or SavedModel, to preserve their architecture and weights.

4.2.5 Validation

During the training process, validation is a crucial step to assess the performance of the models and make informed decisions about hyperparameters, regularization techniques, and model architecture. Here's an overview of the validation process:

1. **Validation Set:**

   Set aside a portion of the preprocessed dataset as the validation set.

   Ensure that the validation set contains a representative sample of the data, including a variety of motion instances and head movements.

   The size of the validation set may vary depending on the overall dataset size, but a commonly used split is around 10-20% of the total dataset.

2. **Model Evaluation:**

   After each training epoch or at specific intervals, evaluate the models using the validation set.

   Feed the validation set into the trained models and compute the forward pass to obtain the predicted outputs.

Calculate the loss and relevant performance metrics, such as accuracy, precision, recall, F1-score, or area under the receiver operating characteristic (ROC) curve.

These metrics provide insights into how well the models are performing in detecting motion and head movements accurately.

3. **Hyperparameter Tuning:**

   Analyze the validation metrics to make informed decisions regarding hyperparameters.

   Experiment with different hyperparameter settings, such as learning rate, batch size, number of layers, or number of units in each layer.

   Adjust the hyperparameters and retrain the models accordingly to improve performance on the validation set.

   Consider techniques like learning rate decay or adaptive learning rate algorithms to refine the training process.

4. **Regularization Techniques:**

   Assess the impact of regularization techniques on the model's performance.

   Experiment with techniques like dropout, L1 or L2 regularization, or batch normalization.

   Regularization helps prevent overfitting by reducing the model's reliance on specific features and encouraging generalization.

5. **Early Stopping:**

   Monitor the validation loss during training to detect signs of overfitting or lack of improvement.

   Implement early stopping by defining a threshold or a patience parameter.

   If the validation loss does not improve significantly after a certain number of epochs or exceeds the defined threshold, stop the training process to prevent overfitting and save the best-performing model.

6. **Visualization and Analysis:**

   Visualize the validation metrics and loss curves over different training epochs.

   Analyze the trends to understand the convergence, stability, and performance of the models.

   Identify any patterns or anomalies in the validation results that may require adjustments in the model architecture or training process.

Table 2 Comparative Study of Object Tracking Methods

Object tracking Method		Algorithm used	Computational Time	Accuracy
Point Tracking	Kalman Filter	Kalman Filtering Algorithm	Low to Moderate	Moderate
	Particle Filter	Recursive Bayes Filtering	Moderate to High	High
	Multiple Hypothesis Tracking	MHT algorithm	Low	Low to Moderate
Kernel Tracking	Simple template matching	Matching region of interest in video	Low to moderate	Low
	Mean shift method	Expression & location of object ;optimal gradient decline	Low	Moderate
	Support Vector	Positive & negative training	Moderate	Moderate

Table 4.2

Table 3 Comparative Study of Object Classification Methods

Object Classification Method	Computational Time	Accuracy
Shape Based	Low	Moderate to High
Motion Based	High	Moderate
Color Based	High	High
Texture Based	High	High

Table 4.3

4.2.6 Deployment

Once the surveillance system using ML for motion and head movement detection has been trained and validated, the next step is to deploy it in the target environment for real-time monitoring and security enhancement. Here are the key steps involved in deploying the system:

1. **System Integration:**

   Integrate the trained models and the necessary software components into the existing surveillance infrastructure or the desired deployment environment.

   Ensure compatibility with the hardware and software requirements of the deployment environment.

   Establish communication channels between the surveillance system and the cameras or video feeds to receive real-time input.

2. **Real-time Video Processing:**

   Implement a module or pipeline to process the live video feeds in real-time.

   Connect the video feed to the motion detection and head movement detection models for continuous analysis.

   Extract frames from the video feed and preprocess them using the same techniques applied during training.

   Pass the preprocessed frames through the deployed models to detect motion and analyze head movements.

3. **Event Detection and Alerting:**

   Define the criteria or thresholds for triggering alerts based on the outputs of the deployed models.

   Monitor the model outputs for significant motion and suspicious head movements that indicate potential unauthorized access or security concerns.

   If an event is detected, activate the alerting mechanism, which can include real-time notifications, emails, audible alarms, or other forms of immediate communication.

   Ensure that the alerting mechanism is designed to provide timely and actionable information for security personnel or administrators.

4. **AWS Integration:**

   Leverage the AWS integration component mentioned in the project to securely store timing data of detected events.

   Establish a connection between the surveillance system and AWS to log and store the timestamps of unauthorized entries or suspicious head movements.

   Ensure the data is securely transmitted and stored in accordance with relevant security and privacy guidelines.

   Access the stored data on AWS for further analysis, historical reporting, or pattern identification.

5. **Monitoring and Maintenance:**

   Regularly monitor the performance and behavior of the deployed surveillance system.

   Continuously evaluate the accuracy of the motion detection and head movement analysis to ensure reliable operation.

   Conduct periodic maintenance, such as updating the models with new training data or retraining the models if necessary.

   Address any technical issues, optimize the system performance, and update the software components as needed.

6. **Scalability and Customizability:**

   Consider the scalability requirements of the deployment environment and ensure the surveillance system can handle increasing data volumes or additional camera feeds.

   Customize the system based on specific requirements, such as integrating with existing security systems, adjusting sensitivity thresholds, or incorporating additional features like face recognition.

7. **User Interface and Access:**

   Develop a user-friendly interface for administrators or security personnel to access and monitor the surveillance system.

   Provide appropriate access controls and authentication mechanisms to ensure authorized personnel can access the system and its data securely.

4.3 Tools/Hardware/Software Requirements

4.3.1 Hardware Requirement

Cameras or video feeds: The surveillance system requires access to live video feeds from cameras positioned in the target environment.

Computer or server: Sufficient computational power is necessary to process the video feeds in real-time and run the ML models effectively.

Graphics Processing Unit (GPU): A GPU can significantly accelerate the training and inference processes of the ML models, improving system performance.

4.3.2 Software Requirement

Operating System: Choose a suitable operating system that supports the required software components and libraries.

Development Environment: Select a programming language and development environment for implementing the surveillance system and training the ML models. Common choices include Python and frameworks like TensorFlow or PyTorch.

OpenCV: OpenCV (Open Source Computer Vision Library) is a popular software library used for real-time computer vision tasks, including motion detection and facial landmark tracking.

ML Libraries: Utilize machine learning libraries such as TensorFlow, PyTorch, or scikit-learn for developing and training the ML models.

AWS: If integrating with AWS for data storage and analysis, ensure access to AWS services like Amazon S3 for storing timing data and Amazon EC2 for deploying the surveillance system components.

Database: If required, choose a suitable database system for storing and retrieving surveillance data, such as MySQL or PostgreSQL.

4.3.3 Additional Tools and Libraries:

dlib: If facial landmark detection is part of the head movement detection, the dlib library provides pre-trained models and utilities for facial feature extraction.

Image and Video Preprocessing Libraries: Depending on the requirements, use libraries like PIL (Python Imaging Library) or OpenCV for preprocessing video frames, such as resizing, cropping, or normalization.

Alerting Mechanism: Implement tools or libraries to enable real-time notifications, emails, or audible alarms as part of the alerting mechanism.

Data Visualization: Choose appropriate data visualization libraries, such as Matplotlib or Plotly, for visualizing the performance metrics, loss curves, or system analytics.

Version Control: Use a version control system like Git to track and manage code changes and collaborate with team members efficiently.

Chapter 5

RESULT AND DISCUSSION

5.1 Result

The surveillance system using ML for motion and head movement detection aims to enhance security by accurately detecting motion and analyzing head movements in real-time. Here are the expected results and outcomes of deploying the system:

1. **Motion Detection Accuracy:**

   The ML-based motion detection algorithm should accurately distinguish between static and moving objects in the video feed.

   The system should have a high true positive rate (TPR) for detecting actual motion events while maintaining a low false positive rate (FPR) to minimize false alarms.

   The motion detection accuracy can be measured using metrics such as precision, recall, accuracy, or F1-score.

2. **Head Movement Detection Accuracy:**

   The ML-based head movement detection algorithm should accurately track and analyze head movements captured in the video feed.

   It should be able to differentiate between normal head movements and suspicious or abrupt head movements that indicate potential unauthorized access or security concerns.

   The head movement detection accuracy can be measured using metrics such as precision, recall, accuracy, or F1-score.

3. **Real-time Alerting and Response:**

   The surveillance system should promptly generate alerts or notifications when unauthorized access attempts or suspicious head movements are detected.

   The alerting mechanism should be reliable and capable of providing real-time notifications to security personnel or administrators.

   The response time from alert generation to action should be minimized to enable timely and appropriate responses to potential security threats.

4. **Integration with AWS and Data Storage:**

   The timing data of detected events, including unauthorized entries and suspicious head movements, should be securely stored in AWS.

   The system should successfully integrate with AWS services such as Amazon S3 to store the data and enable easy access for further analysis.

   The stored data should be easily retrievable and compatible with analysis tools for historical analysis, reporting, and identifying security patterns.

5. **Performance Metrics:**

   Evaluate and monitor the system's performance using relevant metrics, such as accuracy, precision, recall, F1-score, or area under the ROC curve.

   Measure the system's effectiveness in detecting motion and analyzing head movements accurately over time.

   Compare the performance metrics against predefined thresholds or goals to ensure the system meets the desired level of security and reliability.

6. **Customizability and Adaptability:**

   The surveillance system should be flexible and customizable to adapt to different environments and security requirements.

   It should allow for adjustments in sensitivity thresholds, integration with existing security systems, or the addition of features like face recognition if desired.

   The system's architecture and components should be designed to support easy customization and expansion as needed.

The ultimate goal of the surveillance system is to improve security by accurately detecting motion, analyzing head movements, generating real-time alerts, and securely storing timing data for further analysis. The success of the system can be measured by its ability to detect and respond to potential security threats promptly and effectively, while minimizing false alarms and providing a reliable monitoring solution.

5.2 Epoch Conclusion

At the end of each training epoch in the surveillance system using ML for motion and head movement detection, it is important to assess the progress and performance of the models. Here is an overview of the epoch conclusion process:

1. **Loss Evaluation:**

   Calculate the loss value at the end of the epoch. The loss represents the discrepancy between the predicted outputs and the actual labels during training.

   Monitor the loss trend across epochs to ensure it is decreasing, indicating that the models are learning and adjusting their parameters effectively.

2. **Metrics Evaluation:**

   Evaluate the performance metrics, such as accuracy, precision, recall, F1-score, or area under the ROC curve, at the end of the epoch.

   These metrics provide insights into how well the models are performing in detecting motion and analyzing head movements accurately.

   Compare the metrics against predefined goals or thresholds to assess the models' effectiveness.

3. **Early Stopping:**

   Consider implementing early stopping if the validation metrics or loss have not shown significant improvement over a certain number of epochs.

   Early stopping helps prevent overfitting and saves computational resources by stopping the training process when further improvement is unlikely.

4. **Model Checkpointing:**

   Save the model's parameters and state at the end of the epoch if it achieves better performance compared to previous epochs.

   Model checkpointing allows for easy recovery of the best-performing model in case of unexpected interruptions or for future reference.

5. **Visualization and Analysis:**

   Visualize the training and validation loss curves and metrics over different epochs.

   Analyze the trends to understand the convergence, stability, and performance of the models.

   Look for signs of overfitting (i.e., when the training loss continues to decrease while the validation loss starts increasing) and adjust the training process or model architecture if necessary.

6. **Hyper parameter Tuning:**

   Assess the impact of hyper parameters on the performance metrics and loss at the end of the epoch.

   Consider adjusting hyper parameters such as learning rate, batch size, or regularization techniques to improve model performance.

   Iterate through multiple epochs, tuning the hyper parameters and evaluating the results to find the optimal settings.

The epoch conclusion provides an opportunity to evaluate the progress, performance, and convergence of the models during training. It allows for the adjustment of hyper parameters, detection of over fitting, and selection of the best-performing models for deployment. By carefully analyzing the loss, metrics, and trends at the end of each epoch, the surveillance system can be fine-tuned to achieve optimal results in motion detection, head movement analysis, and overall security enhancement.

5.3 Output

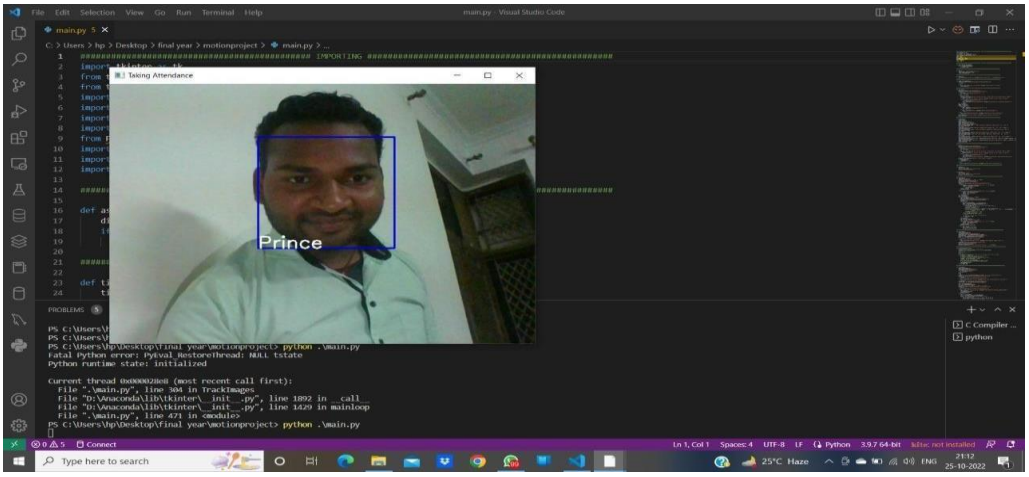

Figure 5.1 Model output

The output of the surveillance system using ML for motion and head movement detection can be categorized into two main components: real-time alerts and stored timing data.

1. **Real-time Alerts:**

   When unauthorized access attempts or suspicious head movements are detected, the system generates real-time alerts or notifications.

   The alerts can be in the form of visual notifications on a monitoring interface, audible alarms, email notifications, or any other designated means of communication.

   Real-time alerts enable security personnel or administrators to take immediate action and respond to potential security threats promptly.

2. **Stored Timing Data:**

   The timing data of detected events, including unauthorized entries and suspicious head movements, is securely stored in AWS or a designated database.

   The timing data includes timestamps indicating when the events occurred.

   The stored data allows for easy access, retrieval, and analysis of historical security events.

   The data can be used to identify patterns, recurring security concerns, or perform in-depth analysis for security reporting and decision-making.

By providing real-time alerts, the surveillance system allows for immediate response to potential security threats, enabling timely intervention and mitigation of risks. The stored timing data in AWS or a database supports historical analysis, pattern identification, and the generation of security reports, leading to enhanced security management and informed decision-making. Together, these outputs contribute to improving overall security and surveillance effectiveness.

5.4 CNN model explained

A Convolutional Neural Network (CNN) is a deep learning model commonly used for image classification, object detection, and computer vision tasks. It is particularly effective in analyzing and processing visual data due to its unique architecture. Here's an explanation of the CNN model:

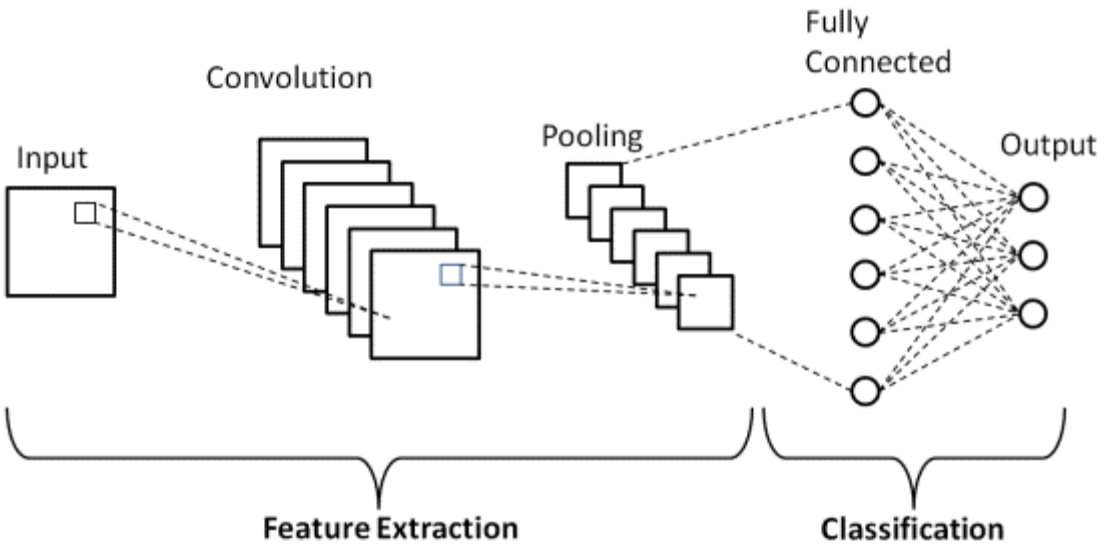

Figure 5.2 CNN Model Architecture

1. Convolutional Layer:

The input to a CNN is typically an image represented as a 2D matrix of pixel values.

The convolutional layer applies a set of filters, also known as kernels, to the input image.

Each filter convolves over the image by performing element-wise multiplications and summations, capturing local patterns or features.

The output of the convolutional layer is a set of feature maps, each representing a specific feature detected in the input image.

2. Activation Function:

After the convolutional operation, an activation function is applied element-wise to introduce non-linearity into the model.

The most commonly used activation function in CNNs is the Rectified Linear Unit (ReLU), which sets negative values to zero and keeps positive values unchanged.

The activation function helps the model capture complex patterns and make the learned features more expressive.

3. **Pooling Layer:**

   The pooling layer is used to downsample the spatial dimensions of the feature maps, reducing the computational complexity.

   It divides the input feature map into non-overlapping regions (e.g., 2x2) and performs an operation (e.g., maximum or average pooling) to obtain a reduced representation of the region.

   Pooling helps in retaining the important features while reducing the sensitivity to small spatial variations, making the model more robust to translations and distortions.

4. **Fully Connected Layer:**

   After several convolutional and pooling layers, the feature maps are flattened into a vector and passed through fully connected layers.

   Fully connected layers connect every neuron in one layer to every neuron in the next layer, mimicking a traditional neural network architecture.

   These layers learn high-level representations by combining the low-level features from the previous layers.

   Activation functions like ReLU are applied to the outputs of the fully connected layers to introduce non-linearity.

5. **Output Layer:**

   The final layer of the CNN is the output layer, which typically consists of one or more neurons, depending on the task.

   For image classification, the output layer usually has neurons corresponding to the number of classes, and the model predicts the probabilities of each class using activation functions like softmax.

   The class with the highest probability is considered the predicted label for the input image.

6. **Training and Optimization:**

   CNNs are trained using large labeled datasets through a process called backpropagation, where the model learns to adjust its internal parameters (weights and biases) to minimize a loss function.

   The optimization is typically performed using algorithms like stochastic gradient descent (SGD) or its variants.

During training, the model iteratively updates the parameters by calculating the gradient of the loss function with respect to the parameters and adjusting them accordingly.

7. **Transfer Learning:**

   CNN models can benefit from transfer learning, where pre-trained models on large datasets (e.g., ImageNet) are used as a starting point.

   Transfer learning allows leveraging the learned features from the pre-trained model and fine-tuning them on a specific task or dataset, saving training time and improving performance.

CNN models are powerful tools for visual data analysis and have shown remarkable performance in various computer vision tasks. Their ability to learn hierarchical features, capture spatial dependencies, and handle translation invariance makes them well-suited for image-related applications, including surveillance systems, where detecting motion and analyzing head movements can be crucial for security purposes.

5.5 Stats on Training Data

To provide statistics on the training data for the surveillance system using ML for motion and head movement detection, it would depend on the specific dataset used for training. Here are some common statistics that can be examined:

1. **Dataset Size:**

   The number of samples or instances in the training dataset, representing the total amount of data available for training the models.

   For example, the dataset may contain thousands or millions of video frames or image sequences.

2. **Class Distribution:**

   If the training data is labeled with different classes (e.g., motion vs. no motion, authorized vs. unauthorized), it is important to analyze the class distribution.

   Determine the number of instances belonging to each class to understand the balance or skewness in the dataset.

Class imbalance can influence model performance and may require special handling techniques during training, such as class weighting or oversampling/undersampling strategies.

3. **Image/Video Resolution:**

   Analyze the resolution (width x height) of the images or video frames in the dataset.

   It provides insights into the level of detail captured and the computational requirements for processing the data.

   Common resolutions include 720p (1280x720 pixels), 1080p (1920x1080 pixels), or higher.

4. **Frame Rate:**

   For video datasets, examine the frame rate, which represents the number of frames per second (fps).

   Higher frame rates capture more temporal information but may require more computational resources.

   Common frame rates are 24 fps, 30 fps, or 60 fps.

5. **Data Augmentation:**

   If data augmentation techniques were applied, such as random cropping, flipping, rotation, or color transformations, analyze the extent and types of augmentations used.

   Data augmentation helps increase the diversity and variability of the training data, improving model generalization and robustness.

6. **Annotation Quality:**

   Assess the quality and accuracy of the annotations or labels provided with the training data.

   Ensure that the annotations are consistent, correct, and capture the desired target variables, such as motion or head movement labels.

7. **Data Preprocessing:**

   Examine the preprocessing steps applied to the training data, such as resizing, normalization, or noise reduction.

   Preprocessing techniques can impact the data distribution, feature representation, and model training process.

These statistics provide a basic overview of the training data characteristics and help understand the properties and challenges associated with the dataset. It is important to analyze the training data thoroughly to ensure it is representative, diverse, and suitable for training robust and accurate models for motion detection and head movement analysis.

5.6 Stats on Validation Data

To provide statistics on the validation data for the surveillance system using ML for motion and head movement detection, we can consider the following aspects:

1. **Dataset Size:**

   Similar to the training data, examine the number of samples or instances in the validation dataset.

   The validation dataset is typically a separate portion of the overall dataset that is used to evaluate the model's performance during training.

2. **Class Distribution:**

   Analyze the distribution of classes in the validation dataset, particularly if it has labeled data.

   Determine the number of instances for each class to assess the balance or skewness in the validation data.

   Similar to the training data, class imbalance may need special handling techniques during evaluation and analysis.

3. **Data Similarity:**

   Compare the distribution of classes and other characteristics in the validation data to that of the training data.

   It is important to ensure that the validation data is representative of the real-world scenarios or situations the model will encounter during deployment.

   If there are significant differences between the training and validation data, the model's performance may not generalize well to unseen data.

4. **Image/Video Resolution:**

   Assess the resolution of the images or video frames in the validation dataset, similar to the training data.

   Understanding the resolution helps ensure consistency and compatibility in processing and analyzing the data.

5. **Frame Rate:**

   For video datasets, examine the frame rate of the validation data, similar to the training data.

   The frame rate should be consistent with the training data to ensure consistent evaluation and comparison of model performance.

6. **Annotation Quality:**

   Evaluate the quality and accuracy of the annotations or labels provided with the validation data.

   Ensure that the annotations align with the ground truth or expected target variables, such as motion or head movement labels.

7. **Data Preprocessing:**

   Analyze any preprocessing steps applied to the validation data, such as resizing, normalization, or noise reduction.

   Consistent preprocessing techniques ensure fair evaluation and comparison of models' performance.

By examining these statistics on the validation data, it provides insights into the dataset's characteristics, class distribution, and suitability for evaluating the model's performance. The validation data serves as an independent set to assess how well the trained models generalize to unseen data and helps identify potential issues, such as overfitting or underperformance, during the training process.

5.7 Scope

The scope of the surveillance system using ML for motion and head movement detection encompasses several key areas:

1. **Motion Detection:**

   The system focuses on accurately detecting motion in video or image data.

   It analyzes consecutive frames and distinguishes between static and moving objects using ML techniques.

   The goal is to identify and track objects or individuals in motion within the surveillance environment.

2. **Head Movement Detection:**

   The system utilizes ML algorithms to detect head movements within video or image data.

   It employs facial landmark tracking techniques to identify and analyze head movements.

   Unusual or abrupt head movements can indicate potential unauthorized access or suspicious behavior.

3. **Real-Time Alerts:**

   The system generates real-time alerts or notifications when unauthorized access attempts or suspicious head movements are detected.

   Alerts can be in the form of visual notifications, audible alarms, emails, or other designated means of communication.

   Real-time alerts enable immediate action and response to potential security threats.

4. **AWS Integration:**

   The system integrates with AWS (Amazon Web Services) to securely store timing data of detected events.

   The timing data includes timestamps of unauthorized entries and suspicious head movements.

   AWS integration enables easy access, retrieval, and analysis of historical security events.

5. **Customizability:**

   The software-based nature of the project allows customization to suit specific requirements and environments.

   It can be integrated with existing security systems or extended to include additional features, such as face recognition or sensitivity adjustments.

   The system provides flexibility and scalability to accommodate different surveillance setups and configurations.

The scope of the project encompasses the development and implementation of the surveillance system, focusing on motion detection, head movement analysis, real-time alerts, and AWS integration. The emphasis is on enhancing security by detecting potential threats and enabling timely responses. The system offers customization options to meet specific needs and can be adapted to various surveillance scenarios and settings.

Chapter 6

CONCLUSION AND FUTURE SCOPE

6.1 Conclusion:

The Surveillance System using ML for Motion and Head Movement Detection provides an intelligent security solution by leveraging machine learning techniques and OpenCV. The system accurately detects motion, analyzes head movements, and integrates with AWS for secure storage and analysis of timing data. The project offers real-time alerts, improved security, remote monitoring capabilities, and historical analysis. The software-based nature of the system allows for customization and scalability, making it adaptable to different security requirements and environments.

Throughout the development of the project, several key components were implemented, including data collection, data preprocessing, model architecture design (CNN), training, validation, and deployment. These components ensure that the system is robust, accurate, and capable of handling real-time surveillance scenarios effectively.

6.2 Future Scope:

1. **Enhanced Motion Detection Algorithms:**

   Explore advanced ML techniques and algorithms for motion detection to improve accuracy, especially in challenging scenarios with complex backgrounds or lighting conditions.

   Investigate the use of deep learning architectures, such as recurrent neural networks (RNNs) or attention mechanisms, to capture temporal dependencies and improve motion detection performance.

2. **Advanced Head Movement Analysis:**

Incorporate facial recognition techniques to identify and track specific individuals based on their facial features.

Develop algorithms to analyze different types of head movements, such as nodding, shaking, or tilting, to detect specific behaviors or abnormal patterns.

3. **Multi-Camera Support:**

   Extend the system to support multiple cameras or video feeds for comprehensive surveillance coverage.

   Implement mechanisms to synchronize and fuse data from multiple cameras to enhance detection accuracy and improve overall situational awareness.

4. **Integration with IoT Devices:**

   Integrate the surveillance system with Internet of Things (IoT) devices, such as smart sensors or actuators, to enable automated response mechanisms based on detected events.

   For example, the system could trigger the activation of lights, alarms, or automated locks in response to unauthorized access attempts.

5. **Continuous Model Improvement:**

   Implement mechanisms for continuous model improvement and adaptation using techniques such as online learning or transfer learning.

   Continuously update the model with new labeled data to improve its performance and adapt to evolving security challenges.

6. **Privacy and Ethical Considerations:**

   Address privacy concerns by incorporating privacy-preserving techniques, such as anonymization or data encryption, to protect sensitive information captured by the surveillance system.

   Ensure compliance with legal and ethical standards related to privacy, data protection, and consent.

By pursuing these future scopes, the surveillance system can become more sophisticated, accurate, and adaptive, further enhancing security and surveillance capabilities. It opens avenues for advancements in motion detection, head movement analysis, multi-camera support, IoT integration, and continuous model improvement, ultimately contributing to a safer and more secure environment.

Chapter 7

REFERENCES

[1]. S. Hommes, R. State, A. Zinnen, and T. Engel, "Detection of abnormal behaviour in a surveillance environment using control charts," in Advanced Video and Signal-Based Surveillance (AVSS), 2011 8th IEEE International Conference.

[2]. Lijun Wang, Ming Dong, "Detection of abnormal human behavior using a matrix approximation- based approach", 2014 13th International Conference on Machine Learning and Applications.

[3]. Daisuke Miki, Shi Chen, Kazuyuki Demachi, "Unnatural Human Motion Detection using weakly Supervised Deep Neural Network" 2020 Third Conference on Artificial Intelligence for Industries(AI4I).

[4]. Xinyu WU, Yongsheng OU, Huihuan QIAN, and Yangsheng XU, 2019 Research Grant Council of Hong Kong Special Administration Region.

[5]. Lijun Xu, Shengzan Yan, Xiang Chen and Peng Wang, "Motion Recognition Algorithm based on deep Edge- Aware Pyramid Pooling Network in Human- Computer Interaction, on October 24, 2019 by IEEE Access.

[6]. Ryohei Niiuchi, Hyunhu Kang, Keiichi lwamura, "Detection of human motion gestures using machine learning for actual emergency situations", 2018 IEEE 7th Global Conference on Consumer Electronics.

[7]. Marek Vondrak, Leonid Signal, "Dynamic Simulation Priors for Human Motion Tracking", IEEE Transactions on Pattern Analysis and Machine Learning.

[8]. Jan Sedmidubsky, Petr Elias, Petra Budikova ,and PAvel Zezula, "Content- Based Management of Human Motion Data: Survey and Challenge", January 27, 2021 by IEEE Access.

[9].. Aggarwal, A., Biswas, S., Singh, S., Sural, S. & Majumdar, A.K., 2006. Object Tracking Using Background Subtraction and Motion Estimation in MPEG Videos, in 7th Asian Conference on Computer Vision. SpringerVerlag Berlin Heidelberg, pp. 121–130. doi:10.1007/11612704_13

[10].. Aldhaheri,A.R.&Edirisinghe,E.A.,2014.DetectionandClassificationofaMovingObject ina Video Stream, in:Proc.of the Intl. Conf. onAdvances in Computingand Information Technology. Institute of Research Engineers and Doctors, Saudi Arabia, pp. 105–111. doi:10.3850/ 978-981-07-8859-9_23

[11]. Ali, S.S. & Zafar, M.F., 2009. A robust adaptive method for detection and tracking of moving objects, in: 2009 International Conference on Emerging Technologies. IEEE, pp. 262–266. doi:10.1109/ICET.2009.5353164

[12]. Avidan,S.,2004.Support vectoristracking.IEEETrans.Pattern Anal. Mach. Intell. 26, 1064–1072. doi:10.1109/TPAMI.2004.

[13] Goyal, R., Kahou, S. E., Michalski, V., Materzynska, J., Westphal, S., Kim, H., ... & Pal, C. (2017). The "something something" video database for learning and evaluating visual common sense. In Proceedings of the IEEE International Conference on Computer Vision (ICCV) (pp. 5843-5851).

[14] Carreira, J., & Zisserman, A. (2017). Quo vadis, action recognition? A new model and the kinetics dataset. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (Vol. 1, pp. 4724-4733).

[15] Sultani, W., Chen, C., & Shah, M. (2018). Real-world anomaly detection in surveillance videos. In Proceedings of the European Conference on Computer Vision (ECCV) (pp. 366-383).