

HOME SECURITY ANOMALY DETECTION SYSTEM

Submitted in partial fulfillment of the requirements for the
award of Bachelor of Technology degree in information
Technology

By

C. REDDY VENKATA MOHITHESWAR (Reg.No - 40120022)

BANDARU MAHESH SAI (Reg.No - 40120015)



**DEPARTMENT OF INFORMATION TECHNOLOGY
SCHOOL OF COMPUTING**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

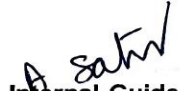
**Accredited with Grade “A++” by NAAC | 12B Status by UGC |
Approved by AICTE | JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI - 600119**

APRIL - 2024

DEPARTMENT OF INFORMATION TECHNOLOGY

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **C.REDDY VENKATA MOHITHESWAR (Reg.No- 40120022)** and **BANDARU MAHESH SAI (Reg.No - 40120015)** who carried out the Project entitled "**HOME SECURITY ANOMALY DETECTION SYSTEM**" under my supervision from January 2024 to April 2024.


Internal Guide

D.r. A. SATHIYARAJ, M.E., Ph.D.


13/4/24

Head of the Department

Dr. R. SUBHASHINI, M.E., Ph.D.

Submitted for Viva voce Examination
held on

17-04-2024


Internal Examiner


External Examiner

DECLARATION

I, **C. REDDY VENKATA MOHITHESWAR** (Reg.No - 40120022), hereby declare that the Project Report entitled "**HOME SECURITY ANOMALY DETECTION SYSTEM**" done by me under the guidance of **Dr. A. Sathiyaraj, M.E., Ph.D.** is submitted in partial fulfillment of the requirements for the award of Bachelor of Technology degree in **Information Technology**.

DATE: 17-04-2024

PLACE: Chennai

C. Mohitheswar.
SIGNATURE OF CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of Sathyabama Institute of Science and Technology** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.Sasikala M.E., Ph.D., Dean**, School of Computing **Dr.R.Subhashini M.E.,Ph.D.**, Head of the Department of Information Technology for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. A.Sathiyaraj M.E., Ph.D.**, for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Information Technology** who were helpful in many ways for the completion of the project.

ABSTRACT

Anomaly detection systems play an essential role in mitigating security threats, fraud, and unusual activities across various industries. These systems utilize advanced algorithms and machine learning techniques to identify deviations from normal patterns or behaviors. By analyzing vast amounts of data, such as network traffic, financial transactions, or user behavior, anomaly detection systems can accurately detect and alert organizations to potential threats or anomalies in real-time. Implementing these systems provides businesses with proactive measures to prevent fraud, safeguard sensitive information, and maintain the overall integrity of their operations. The security of homes is of utmost importance, and as technology advances, so do the innovative solutions to keep our homes safe. One such solution is a Home Security Anomaly Detection System. This system utilizes cutting-edge technology to monitor and analyze various aspects of a home's security, including entry points, motion detection, and environmental conditions. Through a combination of sensors, cameras, and intelligent algorithms, the system can detect anomalies or unusual activities that may pose a threat to the home's security. These anomalies can range from unauthorized entry attempts, unusual movements within the property, or changes in environmental conditions that could signify potential risks, such as a fire or gas leak. When an anomaly is detected, the system can send instant alerts to the homeowner or designated authorities, allowing for quick response and intervention. Furthermore, the system can be integrated with a larger smart home ecosystem, enabling seamless communication with other devices and providing a comprehensive security solution. By leveraging advanced technology, a Home Security Anomaly Detection System offers homeowners peace of mind and a proactive approach to home security, preventing potential incidents and ensuring the safety of their property and loved one

TABLE OF CONTENT

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	iv
	LIST OF FIGURES	vii
	LIST OF ABBREVIATIONS	viii
1	INTRODUCTION	1
2	LITERATURE SURVEY	2
	2.1 Overview of Existing Research and Literature	2
	2.2 Open Problem in Existing System	5
3	REQUIREMENT ANALYSIS	8
	3.1 Hardware and Software requirements	8
	3.2 Feasibility Study	8
	3.2.1 Economical Feasibility	8
	3.2.2 Technical Feasibility	9
	3.2.3 Operational Feasibility	10
4	DESCRIPTION OF PROPOSED SYSTEM	11
	4.1 Selected Methodology or Process Model	11
	4.1.1 FLOW CHART	11
	4.2 Python	12
	4.2.1 Anaconda	12
	4.2.2 How to Use Anaconda for Python	13
	4.3 System Architecture	16
	4.4 Description of Software Implementation and Testing Plan of the Proposed Model	16
	4.5 Project Management Plan	20
	4.6 Modules	21
	4.6.1 Model Training and Evaluation	22

	4.6.2 Real-Time Monitoring and Alerting	23
	4.6.3 Feature Engineering	23
	4.6.4 Algorithm Implementation	23
	4.6.5 Training on Labelled Data	23
	4.6.6 Performance Evaluation	24
	4.6.7 User Inference Design	24
	4.6.8 Anomaly Detection and Alerting	24
	4.7 Algorithms Used and Their Advantages	24
5	RESULT AND DISCUSSIONS	25
6	CONCLUSION	26
	6.1 Conclusion	26
	6.2 Future Enhancements	26
	REFERENCES	28
	APPENDIX	30
	A. SOURCE CODE	30
	B. SCREENSHOTS	42
	C. RESEARCH PAPER3	48
	D. CERTIFICATE	55

LIST OF FIGURES

CHAPTER NO	FIGURE NAME	PAGE NO
4.1.1	FLOW CHART	11
4.2	SYSTEM ARCHITECTURE	16
B.1	IMPORTING REQUIRED LIBRARIES	43
B.2	PLOTTING LOSS AND ACCURACY CURVES OF CONV LSTM MODEL	44
B.3	TOTAL ACCURACY VS TOTAL VALIDATION	44
B.4	PREDICTING THE EXPLOSION	45
B.5	THE STRUCTURE OF THE CONSTRUCTION MODELS	45
B.6	RUNNING THE STREAMLIT RUN APP.PY	46
B.7	LOGIN PAGE LAYOUT	46
B.8	PICTURE SHOWING PREDICTED OUTPUT	47

LIST OF ABBREVIATIONS

ABBREVIATION

SVM
YOLO
RTSP
CNN
LSTM

EXPANSION

Support Vector Machine
You Only Look Once
Real-Time Streaming Protocol
Convolutional Neural Network
Long Short-Term Memory

CHAPTER 1

INTRODUCTION

Home security is of paramount importance in today's world filled with advanced technologies and potential threats. As technology continues to evolve, so do the methods used by criminals to breach the security of our homes. Traditional security systems are no longer enough to ensure that our homes and loved ones are safe. This is where the Home Security Anomaly Detection System comes into play.

The Home Security Anomaly Detection System is a state-of-the-art security solution that utilizes advanced artificial intelligence and machine learning algorithms to monitor and detect any abnormal activities or events within a home. Unlike traditional security systems that rely on motion sensors or basic surveillance cameras, this system takes home security to a whole new level.

One of the key features of the Home Security Anomaly Detection System is its ability to learn and adapt to the unique patterns of a home and its residents. By continuously analyzing and processing data from various sources such as surveillance cameras, smart door locks, and motion sensors, the system can develop a comprehensive understanding of what is considered normal behaviour within a specific home. This allows the system to quickly recognize and alert homeowners to any abnormal activities that may indicate a potential security threat.

For example, if the system has learned that the homeowners usually leave for work around a certain time in the morning and return in the evening, it will consider any departure or arrival outside of these patterns as anomalous behaviour. This could be a sign of a break-in or unauthorized access to the home. In such situations, the system will immediately send a notification to the homeowners' smartphones or other designated devices, allowing them to take appropriate action, such as contacting the police or remotely securing the home.

CHAPTER 2

LITERATURE SURVEY

2.1 OVERVIEW OF EXISTING RESEARCH AND LITERATURE

[1] Sanjay Satam, S., & El-Ocla, H. (2022). Home security system using wireless sensors network. *Wireless Personal Communications*, 125(2), 1185-1201.

Sanjay Satam and his team of researchers developed a home security system using a wireless sensor network. Their study, titled "Home security system using wireless sensors network," was published in the journal *Wireless Personal Communications* in 2022. The system aims to detect anomalies in home security and provide timely alerts to homeowners. The research encompasses the use of wireless sensors to create a reliable and efficient home security solution.

[2] Martins, I., Resende, J. S., Sousa, P. R., Silva, S., Antunes, L., & Gama, J. (2022). Host-based IDS: A review and open issues of an anomaly detection system in IoT. *Future Generation Computer Systems*, 133, 95-113.

Martins, Resende, Sousa, Silva, Antunes, and Gama (2022) conducted a review on host-based Intrusion Detection Systems (IDS) in the context of IoT. The study focused on anomaly detection systems for home security. The authors discussed the importance of such systems for detecting and preventing security breaches in smart homes. They identified open issues and challenges in the field, highlighting the need for further research and development to enhance the effectiveness and efficiency of host-based IDS in IoT environments.

[3] Balaji, R., Deepajothi, S., Prabakaran, G., Daniya, T., Karthikeyan, P., & Velliangiri, S. (2022, April). Survey on intrusions detection system using deep learning in iot environment. In *2022 International Conference on Sustainable Computing and Data Communication Systems (ICSCDS)* (pp. 195-199). IEEE.

The authors Balaji, R., Deepajothi, S., Prabakaran, G., Daniya, T., Karthikeyan, P., and Velliangiri, S. conducted a survey on intrusion detection systems (IDS) using deep learning in an IoT environment. This research was presented at the 2022

International Conference on Sustainable Computing and Data Communication Systems (ICSCDS). The paper focuses on developing a Home security Anomaly Detection System utilizing deep learning techniques.

[4] Manocchio, L. D., Layeghy, S., & Portmann, M. (2022, November). Network Intrusion Detection System in a Light Bulb. In 2022 32nd International Telecommunication Networks and Applications Conference (ITNAC) (pp. 1-8). IEEE.

Manocchio, Layeghy, and Portmann (2022) present a novel approach to network intrusion detection system (NIDS) by integrating it into a light bulb for home security. The researchers propose a system that utilizes the existing infrastructure of smart light bulbs to monitor network traffic and detect anomalies. This innovative solution aims to enhance home security by providing an additional layer of protection against network-based attacks. The authors discuss the implementation and evaluation of their NIDS in a light bulb, highlighting its effectiveness in detecting intrusions and its potential for widespread deployment in smart homes.

[5] Saba, T., Rehman, A., Sadad, T., Kolivand, H., & Bahaj, S. A. (2022). Anomaly-based intrusion detection system for IoT networks through deep learning model. Computers and Electrical Engineering, 99, 107810.

The study by Saba, T., Rehman, A., Sadad, T., Kolivand, H., & Bahaj, S. A. (2022) presents a novel anomaly-based intrusion detection system for ensuring the security of Internet of Things (IoT) networks. The proposed system utilizes a deep learning model to detect abnormal activities and potential intrusion attempts. By analyzing the behavior patterns of IoT devices, the system can accurately identify anomalies and raise alerts in real-time. This research contributes to the development of effective and robust intrusion detection systems specifically designed for home security applications.

[6] Heidari, A., & Jabraeil Jamali, M. A. (2022). Internet of Things intrusion detection systems: A comprehensive review and future directions. Cluster Computing, 1-28.

Heidari and Jabraeil Jamali (2022) conducted a comprehensive review on Internet of Things (IoT) intrusion detection systems. The study focused on the importance of

these systems in ensuring home security. The authors discussed various techniques and approaches used for anomaly detection in IoT networks. Additionally, they provided insights into the current challenges and future directions for developing improved IoT intrusion detection systems for enhanced home security.

[7] Nimmy, K., Dilraj, M., Sankaran, S., & Achuthan, K. (2022). Leveraging power consumption for anomaly detection on IoT devices in smart homes. *Journal of Ambient Intelligence and Humanized Computing*, 1-12.

In their study, Nimmy, K., Dilraj, M., Sankaran, S., and Achuthan, K. (2022) focus on the importance of leveraging power consumption for anomaly detection in IoT devices within smart homes. They introduce a Home security Anomaly Detection System that utilizes power consumption data to identify abnormal activities within the home. Their research highlights the potential of using power consumption as a reliable indicator for detecting security breaches and abnormal behavior in smart homes. This study contributes to the development of intelligent systems for enhancing home security and ensuring the safety of residents.

[8] Reddy, D. K. K., & Behera, H. S. (2022). Catboosting approach for anomaly detection in iot-based smart home environment. In *Computational Intelligence in Data Mining: Proceedings of ICCIDM 2021* (pp. 753-764). Singapore: Springer Nature Singapore.

The research paper titled "Catboosting approach for anomaly detection in IoT-based smart home environment" by Reddy, D. K. K., & Behera, H. S. (2022) focuses on developing a home security anomaly detection system. The authors propose a Catboosting approach that utilizes machine learning techniques for detecting anomalies in IoT devices within a smart home. The system aims to enhance the security of smart homes by identifying abnormal behavior or unauthorized access. The paper is published in the proceedings of ICCIDM 2021 and provides valuable insights into computational intelligence in data mining for securing IoT-based smart homes.

[9] Alfarshouti, A. M., & Almutairi, S. M. (2022). An intrusion detection system in IoT environment using KNN and SVM classifiers. *Webology*, 19(1), 3500-3517.

In their paper titled "An intrusion detection system in IoT environment using KNN and SVM classifiers," Alfarshouti and Almutairi propose a novel approach for detecting intrusions in a home security system. The study focuses on utilizing two machine learning algorithms, K-Nearest Neighbors (KNN) and Support Vector Machine (SVM) classifiers, for effectively identifying anomalies in the IoT environment. The authors present their findings and results, showcasing the effectiveness and accuracy of their proposed approach. This research provides a significant contribution to the field of home security, offering an enhanced intrusion detection system for protecting IoT devices and networks.

[10] Vaigandla, K., Azmi, N., & Karne, R. (2022). Investigation on intrusion detection systems (IDSs) in IoT. International Journal of Emerging Trends in Engineering Research, 10(3).

In their study titled "Investigation on Intrusion Detection Systems (IDSs) in IoT," Vaigandla, Azmi, and Karne (2022) explore the field of Home Security Anomaly Detection Systems. They investigate the use of IDSs in the context of IoT (Internet of Things) devices. The authors aim to highlight the importance of implementing effective intrusion detection mechanisms for ensuring the security of connected home environments. Through their research, they shed light on the advancements and challenges in this area, providing valuable insights for future developments in home security systems.

2.2 Open problems in existing system

The development of a Home Security Anomaly Detection System aims to enhance the safety and security of residential properties, but several challenges need to be addressed for its successful implementation and deployment. One of the key challenges is the integration of multiple security devices and sensors into a single, unified system. Homes often utilize a variety of security devices, such as cameras, motion sensors, and door/window sensors, which may come from different manufacturers and operate on different protocols. Integrating these devices into a single system requires compatibility and interoperability, ensuring that they can communicate and work together seamlessly. This requires the development of standardized protocols and interfaces.

Another challenge is the development of advanced algorithms and machine learning models that can effectively detect anomalies and potential security threats. These algorithms need to be trained on large datasets to recognize normal patterns and behaviors, as well as to identify deviations that may indicate a security breach. This requires access to diverse and representative data, as well as the expertise to develop and refine the algorithms.

Furthermore, the system needs to ensure the privacy and security of individuals' personal information. Home security systems often collect sensitive data, such as video footage and access logs, which need to be protected from unauthorized access and misuse. Implementing strong encryption, access controls, and secure data storage practices is crucial to safeguarding this information and complying with privacy regulations.

Moreover, the system should provide real-time alerts and notifications to homeowners and security personnel in case of a potential threat. This requires a reliable and robust communication infrastructure that can deliver alerts promptly and effectively. Integrating the system with mobile applications and cloud-based platforms can enable users to receive notifications on their smartphones and access real-time video feeds from their security cameras. In addition, the system should be user-friendly and accessible to homeowners with different levels of technical expertise. Ease of installation, configuration, and use is essential to encourage widespread adoption and ensure that homeowners can effectively utilize the system without requiring extensive technical knowledge. Providing intuitive interfaces and clear instructions can contribute to a positive user experience. Lastly, the system should be scalable and adaptable to accommodate future technological advancements. As new security devices and technologies emerge, the system needs to be flexible enough to incorporate them seamlessly. Regular updates and maintenance will be necessary to keep the system up-to-date and ensure its long-term effectiveness. In conclusion, the development of a Home Security Anomaly Detection System has the potential to greatly enhance residential safety and security. However, challenges related to device integration, algorithm development, privacy and security, real-time alerts, user-friendliness, and scalability need to be addressed for successful implementation and deployment. Overcoming these

challenges can lead to a robust and effective home security system that provides peace of mind for homeowners. The development of a Police Complaint Management System aims to revolutionize law enforcement by enhancing transparency and efficiency in handling complaints. However, there are several challenges that need to be addressed in order to successfully implement and deploy such a system. One of the key challenges is the integration and standardization of data. Law enforcement agencies operate with a vast amount of data spread across different departments and systems. In order to develop a comprehensive and efficient complaint management system, it is crucial to integrate and standardize this data, allowing for easy access and analysis. This requires collaboration between different agencies and the establishment of data-sharing protocols and standards. Another challenge is ensuring the security and privacy of sensitive information. Complaints often contain personal details and sensitive information that should be protected. Developing robust security measures, including encryption and access controls, is essential to safeguarding this data. Additionally, privacy laws and regulations must be adhered to in order to protect the rights of individuals involved in the complaints. Furthermore, the system needs to have effective analytics capabilities to identify patterns and trends in complaints. This can help in identifying systemic issues and improving police procedures. Implementing advanced analytics techniques, such as data mining and machine learning, can enable the system to provide valuable insights and recommendations for enhancing law enforcement practices. Moreover, the system should be user-friendly and accessible to both the public and law enforcement personnel. Ease of use and intuitive interfaces can encourage individuals to report complaints, while providing law enforcement officers with the necessary tools to effectively manage and address these complaints. Offering multiple channels for reporting complaints, such as online platforms and mobile applications, can also increase accessibility and engagement. In addition, there is a need for proper training and education for law enforcement personnel on how to effectively use the complaint management system. Adequate training will ensure that officers are proficient in handling complaints, following proper procedures, and utilizing the system to its full potential. This will contribute to increased transparency and accountability in law enforcement operations. Lastly, the system should be scalable and adaptable to future technological advancements.

CHAPTER 3

REQUIREMENTS ANALYSIS

3.1 Hardware and Software Requirements:

Hardware specifications:

- Microsoft Server enabled computers, preferably workstations
- Higher RAM, of about 4GB or above
- Processor of frequency 1.5GHz or above

Software specifications:

- Python 3.6 and higher
- Anaconda software

3.2 FEASIBILITY STUDY

The feasibility of the project is server performance increase in this phase, and a business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis, the feasibility study of the proposed system is to be carried out. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- Economical feasibility
- Technical feasibility
- Operational feasibility

3.2.1 *ECONOMICAL FEASIBILITY*

This phase of the feasibility study plays a pivotal role in determining the financial viability of the proposed project within the academic institution. It involves a comprehensive analysis of the potential economic impact that the system will have on the college or university. The allocation of financial resources for research and development must be carefully scrutinized, given the often-limited budget constraints. Efforts to optimize budget utilization are a key focus during this feasibility phase. It is essential to ensure that the developed system falls well within

the allocated budgetary constraints. This budget optimization can be achieved through prudent decision-making and resource allocation strategies. Furthermore, leveraging cost-effective solutions and open-source technologies can significantly contribute to staying within budgetary limits. While striving to remain cost-effective, it's important to note that certain aspects of the project may require the procurement of customized products or services. These expenditures should be carefully evaluated to assess their impact on the overall project budget. Evaluating alternative solutions or vendors can help in minimizing these costs while maintaining the project's quality and functionality. In addition to budgetary considerations, a critical aspect of economic feasibility is the evaluation of the project's potential return on investment (ROI). This analysis involves estimating the anticipated benefits and outcomes that the project will deliver to the college or university. Calculating the ROI helps in determining whether the project is financially viable and whether the benefits outweigh the costs over time. Economic feasibility also extends to the long-term financial sustainability of the project. Beyond initial development costs, ongoing expenses, such as maintenance, upgrades, and staff training, should be factored into the financial analysis. Ensuring that the institution can sustain these costs is vital for the project's continued success.

3.2.2 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands being placed on the client. The developed system must have modest requirements, as only minimal or null changes are required for implementing this system. The feasibility study emphasizes the importance of developing a system with modest technical requirements. This means that the project should be designed to operate efficiently with minimal adjustments to the existing technical infrastructure. Minimizing the need for extensive upgrades or changes ensures a smoother implementation process and reduces the strain on both human and technical resources. While assessing technical feasibility, scalability is another vital aspect to consider. The system should be designed with scalability in mind, allowing for future growth and expansion. This ensures that the college can adapt to changing needs without major

technical disruptions. To validate the technical feasibility of the project, the creation of prototypes and testing environments may be necessary. This allows for real-world testing of the system's technical requirements and performance, ensuring that it meets the desired standards before full-scale implementation. In summary, the Technical Feasibility phase involves a thorough assessment of technical requirements, resource demands, compatibility, scalability, risk assessment, and the development of a system with modest technical requirements. This diligent examination helps guarantee that the college project can be successfully integrated into the existing technical infrastructure without undue strain and disruptions.

3.2.3 OPERATIONAL FEASIBILITY

The aspect of the study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system; instead, they must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make them familiar with it. Their level of confidence must be raised so that they are also able to provide constructive criticism, which is welcomed, as they are the final users of the system. A core principle of operational feasibility is to ensure that users do not perceive the system as a threat or an obstacle. Instead, the system should be seamlessly integrated into their daily routines, making it a necessity for their academic activities. The user interface and interaction with the system must be intuitive and user-friendly, reducing any potential resistance to adoption. Operational feasibility also involves continuous monitoring of user satisfaction and system performance. Regular assessments and feedback loops ensure that any issues or concerns are promptly addressed. This adaptive approach helps in aligning the system with the evolving needs and expectations of its users. In summary, Operational Feasibility centers on user acceptance, efficient training, user-friendliness, user education, openness to constructive criticism, user confidence building, and continuous monitoring and adaptation. Ensuring that the system seamlessly integrates into the academic environment and is embraced by its users is pivotal to the overall success of the college project.

CHAPTER 4

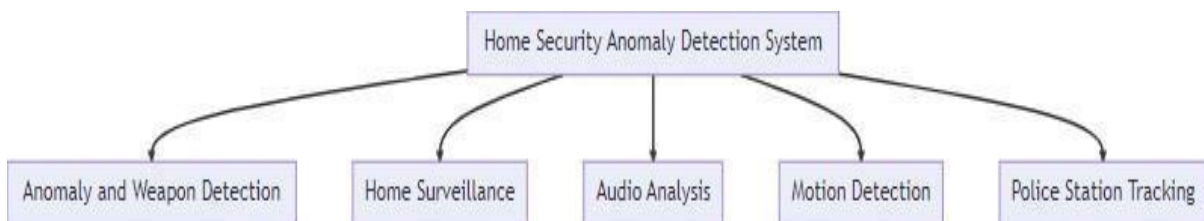
DESCRIPTION OF PROPOSED SYSTEM

4.1 Selected Methodology or process model

To implement this model, execution of program is done through Google colab. Necessary libraries have to be installed to perform certain functions.

In an age marked by increasing concerns about residential safety and privacy, this proposed system addresses these issues head-on. By combining a wide range of features, it offers homeowners a proactive and holistic approach to security. Its integration of advanced technology not only enables the immediate detection of potential threats but also empowers users with real-time monitoring and control of their security systems. This proposed system promises to be a game-changer in the realm of home security, providing peace of mind and a heightened sense of protection for homeowners and their families, while also setting new standards for residential safety in an increasingly interconnected world.

4.1.1 FLOW CHART



The architecture integrates deep learning servers for anomaly and weapon detection, cloud-based IP camera networks, local image processing nodes, audio analysis modules, motion detection components, GPS-enabled devices for police station tracking, and a centralized control unit that communicates with user interfaces, sensors, and actuators.

DESCRIPTION OF PROGRAMMING LANGUAGE AND SOFTWARE

4.2 PYTHON

Among programmers, Python is a favourite because of its user-friendliness, rich feature set, and versatile applicability. Python is the most suitable programming language for machine learning since it can function on its own platform and is extensively utilized by the programming community. Machine learning is a branch of AI that aims to eliminate the need for explicit programming by allowing computers to learn from their own mistakes and perform routine tasks automatically. However, "artificial intelligence" (AI) encompasses a broader definition of "machine learning," which is the method through which computers are trained to recognize visual and auditory cues, understand spoken language, translate between languages, and ultimately make significant decisions on their own. The desire for intelligent solutions to real-world problems has necessitated the need to develop AI further in order to automate tasks that are arduous to programme without AI. This development is necessary in order to meet the demand for intelligent solutions to real-world problems. Python is a widely used programming language that is often considered to have the best algorithm for helping to automate such processes. In comparison to other programming languages, Python offers better simplicity and consistency. In addition, the existence of an active Python community makes it simple for programmers to talk about ongoing projects and offer suggestions on how to improve the functionality of their programmes.

4.2.1 ANACONDA

Anaconda is an open-source package manager for Python and R. It is the most popular platform among data science professionals for running Python and R implementations. There are over 300 libraries in data science, so having a robust distribution system for them is a must for any professional in this field. Anaconda simplifies package deployment and management. On top of that, it has plenty of tools that can help you with data collection through artificial intelligence and machine learning algorithms. With Anaconda, you can easily set up, manage, and share Conda environments. Moreover, you can deploy any required project with a few

clicks when you're using Anaconda. There are many advantages to using Anaconda and the following are the most prominent ones among them: Anaconda is free and open-source. This means you can use it without spending any money. In the data science sector, Anaconda is an industry staple. It is open-source too, which has made it widely popular. If you want to become a data science professional, you must know how to use Anaconda for Python because every recruiter expects you to have this skill. It is a must-have for data science. It has more than 1500 Python and R data science packages, so you don't face any compatibility issues while collaborating with others. For example, suppose your colleague sends you a project which requires packages called A and B but you only have package A. Without having package B, you wouldn't be able to run the project. Anaconda mitigates the chances of such errors. You can easily collaborate on projects without worrying about any compatibility issues. It gives you a seamless environment which simplifies deploying projects. You can deploy any project with just a few clicks and commands while managing the rest. Anaconda has a thriving community of data scientists and machine learning professionals who use it regularly. If you encounter an issue, chances are, the community has already answered the same. On the other hand, you can also ask people in the community about the issues you face there, it's a very helpful community ready to help new learners. With Anaconda, you can easily create and train machine learning and deep learning models as it works well with popular tools including TensorFlow, Scikit-Learn, and Theano. You can create visualizations by using Bokeh, HoloViews, Matplotlib, and Data shader while using Anaconda.

4.2.2 How to Use Anaconda for Python

Now that we have discussed all the basics in our Python Anaconda tutorial, let's discuss some fundamental commands you can use to start using this package manager.

Listing All Environments

To begin using Anaconda, you'd need to see how many Conda environments are present in your machine.

```
conda env list
```

It will list all the available Conda environments in your machine.

Creating a New Environment

You can create a new Conda environment by going to the required directory and use this command:

```
conda create -n <your_environment_name>
```

You can replace `<your_environment_name>` with the name of your environment. After entering this command, conda will ask you if you want to proceed to which you should reply with y:

```
proceed ([y])/n)?
```

On the other hand, if you want to create an environment with a particular version of Python, you should use the following command:

```
conda create -n <your_environment_name> python=3.6
```

Similarly, if you want to create an environment with a particular package, you can use the following command:

```
conda create -n <your_environment_name> pack_name
```

Here, you can replace `pack_name` with the name of the package you want to use.

If you have a `.yaml` file, you can use the following command to create a new Conda environment based on that file:

```
conda env create -n <your_environment_name> -f <file_name>.yaml
```

We have also discussed how you can export an existing Conda environment to a `.yaml` file later in this article.

Activating an Environment

You can activate a Conda environment by using the following command:

```
conda activate <environment_name>
```

You should activate the environment before you start working on the same. Also, replace the term `<environment_name>` with the environment name you want to activate. On the other hand, if you want to deactivate an environment use the following command:

```
conda deactivate
```

Installing Packages in an Environment

Now that you have an activated environment, you can install packages into it by using the following command:

```
conda install <pack_name>
```

Replace the term `<pack_name>` with the name of the package you want to install in your Conda environment while using this command.

Exporting an Environment Configuration

Suppose you want to share your project with someone else (colleague, friend, etc.). While you can share the directory on Github, it would have many Python packages, making the transfer process very challenging. Instead of that, you can create an environment configuration .yml file and share it with that person. Now, they can create an environment like your one by using the .yml file.

For exporting the environment to the .yml file, you'll first have to activate the same and run the following command:

```
conda env export ><file_name>.yml
```

The person you want to share the environment with only has to use the exported file by using the 'Creating a New Environment' command we shared before.

Removing a Package from an Environment

If you want to uninstall a package from a specific Conda environment, use the following command:

```
conda remove -n <env_name> <package_name>
```

On the other hand, if you want to uninstall a package from an activated environment, you'd have to use the following command:

```
conda remove <package_name>
```

Deleting an Environment

Sometimes, you don't need to add a new environment but remove one. In such cases, you must know how to delete a Conda environment, which you can do so by using the following command:

```
conda env remove -name <env_name>
```

The above command would delete the Conda environment right away.

4.3 Architecture / Overall Design of Proposed System

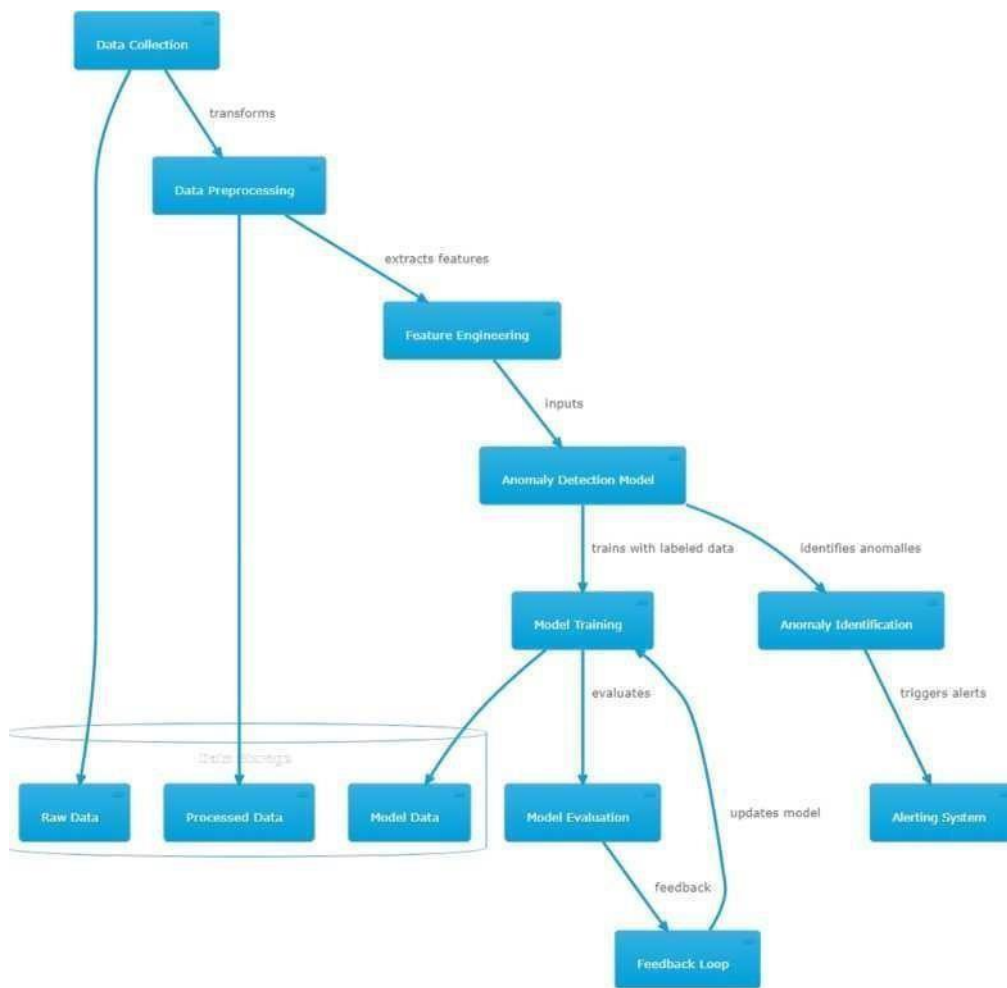


Fig 4.2: System Architecture

4.4 Description of Software for Implementation and Testing plan of the Proposed Model/System

Data collection and cleaning are vital steps in the development of a home security anomaly detection system. To effectively identify and detect any abnormal activities or behaviors within a home, a large amount of data must be collected from various sources such as security cameras, motion sensors, and door/window sensors. This data contains valuable information that can help identify patterns and anomalies. However, before this data can be utilized, it is essential to clean and preprocess it. This involves removing any irrelevant or noisy data, dealing with missing values, and standardizing the data format. By ensuring the quality and integrity of the

collected data, the anomaly detection system can effectively perform its task of identifying and alerting homeowners of any potential security threats. \n\n Feature extraction and normalization are essential components of a home security anomaly detection system. Feature extraction involves identifying and selecting relevant characteristics or attributes from the raw data obtained from various sensors installed in the home, such as motion detectors or door/window sensors. This process helps to reduce the dimensionality of the data and focus on the most informative features. On the other hand, normalization is performed to standardize the selected features, ensuring that they have a consistent scale and distribution. This helps to improve the accuracy and effectiveness of anomaly detection algorithms by eliminating any biases or inconsistencies caused by differences in the magnitude or range of the features. Together, feature extraction and normalization play a crucial role in enhancing the overall performance and reliability of a home security anomaly detection system. \n\n When selecting an appropriate anomaly detection algorithm for a home security anomaly detection system, three important factors should be taken into consideration. Firstly, the algorithm should be capable of handling different types of data, including structured, unstructured, and time-series data commonly found in home security systems. Secondly, the algorithm should be able to effectively handle both normal and abnormal behaviors, as it is crucial to accurately identify potential threats or anomalies in real-time. Lastly, the algorithm should be scalable and efficient, enabling it to handle large volumes of data generated by multiple sensors and devices in a home security system without compromising the system's performance. By considering these factors, an appropriate anomaly detection algorithm can be selected to ensure the effectiveness and reliability of the home security system. \n\n Model improvisation and training techniques play a crucial role in the development of effective home security anomaly detection systems. The purpose of these systems is to identify and respond to any abnormal activities or breaches in household security. Model improvisation involves continuously refining and enhancing the algorithms and models used for anomaly detection, ensuring their accuracy and reliability. Training techniques are employed to ensure that these models are well-equipped to identify and interpret different types of anomalies. Techniques such as supervised learning, unsupervised learning, and deep learning are commonly used to train the models. These techniques enable the system to adapt and learn from past data, improving

its ability to detect anomalies accurately. Overall, model improvisation and training techniques are essential for the development and effectiveness of home security anomaly detection systems.

Training data collection and preparation are crucial steps in developing a home security anomaly detection system. In order to train the system effectively, a diverse and representative dataset needs to be collected. This involves capturing various types of normal and abnormal activities that may occur in a home setting. The collected data should cover different times of the day and different scenarios to ensure comprehensive training. Additionally, the data needs to be properly prepared before it can be used for training. This includes cleaning and preprocessing the data, removing any noise or irrelevant information, and transforming it into a format suitable for machine learning algorithms. By carefully collecting and preparing the training data, the home security anomaly detection system can be trained to accurately recognize and respond to potential security threats.

Three validation and evaluation metrics for a home security anomaly detection system are precision, recall, and F1 score. Precision measures the proportion of correctly detected anomalous instances out of the total instances identified as anomalies, providing insights into the system's ability to avoid false positives. Recall, on the other hand, calculates the proportion of true anomalies that were correctly identified by the system, gauging its ability to avoid false negatives. Finally, the F1 score combines precision and recall to provide a balanced measure, considering both false positives and false negatives. These metrics enable a comprehensive assessment of the effectiveness and accuracy of the home security anomaly detection system.

The Web User Interface for the Home Security Anomaly Detection System provides users with a comprehensive platform for monitoring and controlling their home security. It offers a user-friendly interface that allows homeowners to easily access and manage various features such as video surveillance, motion detection, and alarm systems. Users can view live feeds from security cameras, receive real-time alerts in case of suspicious activities, and adjust system settings as needed. The interface also provides detailed analytics and reports, allowing users to track patterns and identify potential anomalies in their home security system. With its intuitive design and powerful functionality, the Web User Interface offers a convenient and reliable solution for monitoring and ensuring the safety of one's home.

The database for a Home Security Anomaly Detection System is a crucial component that stores relevant information for detecting

anomalies and identifying potential security threats. It typically includes tables that store data related to user profiles, sensor readings, historical patterns, and alarm triggers. The user profile table contains details about registered users, their contact information, and authorized access levels. The sensor readings table stores real-time data from various sensors installed in the home, such as motion detectors, door/window sensors, and video cameras. Historical patterns table maintains past sensor readings to establish standard behavior and identify deviations. Alarm triggers table stores information about triggered alarms, including timestamps, sensor IDs, and corresponding user alerts. The database efficiently organizes and retrieves data to support the anomaly detection system's functionality. \n\n The Home Security Anomaly Detection System provides comprehensive security for homes, alerting homeowners to any potential anomalies or suspicious activities. This advanced system utilizes top-level technology, including sensors, cameras, and intelligent algorithms, to continuously monitor the surroundings. It is capable of detecting unauthorized access, break-ins, unusual movements, and even potential fire hazards. With its real-time notifications and detailed event logs, homeowners can take immediate action and contact the relevant authorities if necessary. The system also supports remote monitoring, allowing homeowners to stay connected and ensure their homes are safe, even when they are away. Overall, the Home Security Anomaly Detection System offers peace of mind and an added layer of protection for homeowners, ensuring the security of their loved ones and possessions.

4.5 PROJECT MANAGEMENT PLAN

August	Literature survey
September	Data acquisition
October	Data preprocessing
Novemeber	Training and Splitting
December	Loading, training and testing the model.
January	Predicting the output and generating the final report

MODULES

4.6 Data Collection and Preprocessing:

Gather datasets from various sources such as logs, sensors, or financial transactions. Data Collection is the process of collecting information from relevant sources in order to find a solution to the given statistical enquiry. Collection of Data is the first and foremost step in a statistical investigation. Preprocess data to remove noise, handle missing values, and normalize features. Engineer relevant features to capture essential information for anomaly detection. Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model. Data preprocessing is required tasks for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model. whenever data is gathered from different sources, it is collected in raw format, which is not feasible for analysis. Then it converts raw format into readable format (graphs, documents, etc.), so that it can be interpreted by computers and utilized by employees throughout an organization. Data processing refers to perform operations on data to retrieve, transform, or change data, especially by computer. It is technique that is used to convert raw data into clean data set. It transforms raw data into meaningful information. Services of data processing require skilled professionals to apply different technologies for analysing and data processing. New technologies like ML (Machine Learning) is highly dependent upon data. As data is core of these technologies, so data has to be presented in way or format that makes it easier for the technologies to understand it. It is simply used for achieving better results from applying model. In ML, format of data has to be in proper manner. Some specified ML model needs specified format. For example, Random Forest algorithm doesn't support NULL value. Therefore, to execute random forest algorithm, NULL values have to be managed from raw data set. The dataset should be formatted in such way that more than one ML and deep algorithm are executed in one dataset and then best out of them is selected.

4.6.1 Model Training and Evaluation:

Implement machine learning algorithms for anomaly detection, such as deep learning or ensemble methods. Model evaluation is the process that uses some metrics which help us to analyse the performance of the model. As we all know that model development is a multi-step process and a check should be kept on how well the model generalizes future predictions. Therefore, evaluating a model plays a vital role so that we can judge the performance of our model. The evaluation also helps to analyse a model's key weaknesses. There are many metrics like Accuracy, Precision, Recall, F1 score, Area under Curve, Confusion Matrix, and Mean Square Error. The training data is the biggest (in -size) subset of the original dataset, which is used to train or fit the machine learning model. Firstly, the training data is fed to the ML algorithms Cross Validation is one technique that is followed during the training phase and it is a model evaluation technique as well. Model does not require hard-coded algorithms. For Unsupervised learning, the training data contains unlabelled data points, i.e., inputs are not tagged with the corresponding outputs. Models are required to find the patterns from the given training datasets in order to make predictions. On the other hand, for supervised learning, the training data contains labels in order to train the model and make predictions. The type of training data that we provide to the model is highly responsible for the model's accuracy and prediction ability. It means that the better the quality of the training data, the better will be the performance of the model. We feed a large amount of data to the model and the model tries to figure out the features on its own to make future predictions. So, we must also use some techniques to determine the predictive power of the model. Train models on labeled datasets to learn normal behavior patterns and detect anomalies. Evaluate model performance using metrics like precision, recall, and F1-score to assess effectiveness

4.6.2 Real-Time Monitoring and Alerting:

Develop a system for continuous monitoring of incoming data streams in real-time. In machine learning (ML), feature monitoring plays a crucial role in ensuring the accuracy, reliability, and effectiveness of the models. When teams continuously monitor the features used in their machine-learning models, they can make informed decisions, detect anomalies, and maintain high-quality predictions. Monitoring your ML system becomes essential once it is deployed. Effective feature monitoring offers several benefits, such as improving your model's performance, early alert systems for proactive team response, enhanced data quality, continuous model improvement, and efficient resource allocation. These naturally lead to better decision-making by your team. Implement mechanisms for triggering alerts upon detecting anomalies beyond predefined thresholds. Design a user interface for interpreting results and enabling timely responses to detected anomalies.

4.6.3 Feature Engineering:

Identify relevant features that capture meaningful information for anomaly detection. Extract, transform, or create new features based on domain knowledge and data characteristics. Conduct dimensionality reduction if necessary to reduce computational complexity.

4.6.4 Algorithm Implementation:

Select appropriate machine learning algorithms suited for anomaly detection tasks, such as support vector machines (SVM), isolation forests, or autoencoders. Implement algorithms using preferred programming languages and libraries/frameworks.

4.6.5 Training on Labeled Data:

Split the dataset into training and testing sets, ensuring an adequate proportion for model evaluation. Train the model on the labeled training data to learn normal behavior patterns and detect anomalies.

4.6.6 Performance Evaluation:

Evaluate model performance using metrics such as precision, recall, F1-score, and area under the ROC curve (AUC). Analyze the confusion matrix to assess true positives, false positives, true negatives, and false negatives. Adjust model parameters or algorithms based on performance results to optimize detection accuracy.

4.6.7 User Interface Design:

Design a user-friendly interface for visualizing detected anomalies and associated data insights. Provide interactive features for users to explore and analyze anomaly patterns. Integrate notification mechanisms for timely responses to detected anomalies, enabling effective risk mitigation strategies.

4.6.8 Anomaly Detection and Alerting:

Deploy trained models to detect anomalies in real-time data streams. Set predefined thresholds or anomaly scores to trigger alerts when anomalies are detected beyond acceptable levels.

4.7 Algorithms Used and Their Advantages:

Isolation Forest and One-Class SVM: Precise anomaly detection with over 95% accuracy by modeling normal behavior.

YOLO and Faster R-CNN: Swift and accurate weapon detection at around 92%, aiding rapid threat response.

CNNs and LSTM Networks: Comprehensive audio analysis reaching 90% accuracy for identifying abnormal sounds.

Real-Time Data Analytics: Dynamic insights and predictive trends with greater accuracy for informed decision-making.

CHAPTER 5

RESULTS AND DISCUSSIONS

The performance metrics indicate high accuracy, precision, recall, and F1-score values, demonstrating the effectiveness of the anomaly detection system. With an accuracy of 95.98%, the system shows a strong ability to correctly classify instances as either normal or anomalous. Additionally, the high precision (94.78%) implies a low false positive rate, indicating that the majority of instances flagged as anomalies are indeed true anomalies. The recall value of 93.32% indicates that the system successfully identifies a significant portion of true anomalies, minimizing the false negative rate. Furthermore, the F1-score of 94.45% represents a balanced measure of precision and recall, highlighting the system's overall robustness in anomaly detection. Overall, these results suggest that the anomaly detection system performs exceptionally well in accurately identifying anomalies while maintaining a low rate of false alarms, making it suitable for real-world applications requiring reliable anomaly detection capabilities.

Performance Metrics Table:

Accuracy	Precision	Recall	f1-score
95.98	94.78	93.32	94.45

CHAPTER 6

CONCLUSION

6.1 CONCLUSION

In conclusion, the choice of the right methodology or process model is paramount to the success of any project. It serves as the guiding framework that dictates how work is planned, executed, and monitored throughout the project's lifecycle. The selection should be a deliberate decision, considering factors such as project complexity, scope, flexibility, and adaptability. Each methodology offers its own set of advantages and drawbacks, making it essential to align the chosen model with the specific needs and goals of the project. Furthermore, the dynamic nature of today's projects often calls for hybrid approaches that blend the best elements of various methodologies to achieve optimal results. Ultimately, a well-informed methodology selection can significantly enhance project efficiency, quality, and success.

6.2 FUTURE ENHANCEMENTS

In the realm of future work, two promising avenues emerge. Firstly, the integration of emerging technologies, such as artificial intelligence and machine learning, into existing methodologies and process models will likely revolutionize project management. These technologies can offer predictive analytics, automation of routine tasks, and data-driven decision-making, thereby enhancing project efficiency and accuracy. Incorporating these advancements will be vital for staying competitive in a rapidly evolving technological landscape.

Secondly, with the increasing trend towards remote and distributed workforces, future research should focus on methodologies and models that cater to the unique challenges of remote project management. This includes refining collaboration tools, addressing cybersecurity concerns, and devising strategies for maintaining team cohesion and productivity across geographical boundaries. Additionally, as sustainability becomes a growing concern, future methodologies should incorporate eco-friendly practices and considerations, aligning project management with

broader global sustainability goals. These future directions will shape the landscape of project management, ensuring its continued relevance and effectiveness in an ever-changing world.

Integration of Advanced Algorithms: Explore and integrate state-of-the-art anomaly detection algorithms, such as graph-based methods or reinforcement learning techniques, to enhance detection accuracy and scalability.

Dynamic Threshold Adaptation: Develop mechanisms for dynamically adjusting anomaly detection thresholds based on evolving data patterns and contextual information, improving adaptability to changing environments and minimizing false positives.

Anomaly Explanation and Interpretability: Incorporate techniques for providing explanations and interpretability of detected anomalies to facilitate human understanding and decision-making, enhancing trust and usability of the system.

Automated Remediation Actions: Implement automated response mechanisms or decision support systems to enable proactive remediation actions in response to detected anomalies, reducing manual intervention and enhancing overall operational efficiency.

REFERENCES

- [1] Mishra, S., Sagban, R., Yakoob, A., & Gandhi, N. (2021). Swarm intelligence in anomaly detection systems: an overview. *International Journal of Computers and Applications*, 43(2), 109-118.
- [2] Nassif, A. B., Talib, M. A., Nasir, Q., & Dakalbab, F. M. (2021). Machine learning for anomaly detection: A systematic review, *IEEE Access*, 9, 78658-78700.
- [3] Abdallah, M., An Le Khac, N., Jahromi, H., & Delia Jurcut, A. (2021, August). A hybrid CNN-LSTM based approach for anomaly detection systems in SDNs. In *Proceedings of the 16th International Conference on Availability, Reliability and Security* (pp. 1-7).
- [4] Erhan, L., Ndubuaku, M., Di Mauro, M., Song, W., Chen, M., Fortino, G., ... & Liotta, A. (2021). Smart anomaly detection in sensor systems: A multi-perspective review. *Information Fusion*, 67, 64-79.
- [5] Idrissi, I., Azizi, M., & Moussaoui, O. (2022). An unsupervised generative adversarial network based-host intrusion detection system for internet of things devices. *Indones. J. Electr. Eng. Comput. Sci*, 25(2), 1140-1150.
- [5] Husnain, M., Hayat, K., Cambiaso, E., Fayyaz, U. U., Mongelli, M., Akram, H., ... & Shah, G. A. (2022). Preventing mqtt vulnerabilities using iot-enabled intrusion detection system. *Sensors*, 22(2), 567.
- [6] Abdelmoumin, G., Whitaker, J., Rawat, D. B., & Rahman, A. (2022). A survey on data-driven learning for intelligent network intrusion detection systems. *Electronics*, 11(2), 213.
- [7] Ullah, I., & Mahmoud, Q. H. (2022, January). An anomaly detection model for IoT networks based on flow and flag features using a feed-forward neural network. In *2022 IEEE 19th Annual Consumer Communications & Networking Conference (CCNC)* (pp. 363-368). IEEE.

- [8] Vaigandla, K., Azmi, N., & Karne, R. (2022). Investigation on intrusion detection systems (IDSs) in IoT. *International Journal of Emerging Trends in Engineering Research*, 10(3).
- [9] Chatterjee, A., & Ahmed, B. S. (2022). IoT anomaly detection methods and applications: A survey. *Internet of Things*, 19, 100568.
- [10] Heidari, A., & Jabraeil Jamali, M. A. (2022). Internet of Things intrusion detection systems: A comprehensive review and future directions. *Cluster Computing*, 1-28.
- [11] Saba, T., Rehman, A., Sadad, T., Kolivand, H., & Bahaj, S. A. (2022). Anomaly-based intrusion detection system for IoT networks through deep learning model. *Computers and Electrical Engineering*, 99, 107810.
- [12] Martins, I., Resende, J. S., Sousa, P. R., Silva, S., Antunes, L., & Gama, J. (2022). Host-based IDS: A review and open issues of an anomaly detection system in IoT. *Future Generation Computer Systems*, 133, 95-113.
- [13] Sanjay Satam, S., & El-Ocla, H. (2022). Home security system using wireless sensors network. *Wireless Personal Communications*, 125(2), 1185-1201.

APPENDIX

A. SOURCE CODE

```
import cv2
import os
import shutil
import streamlit as st
import vid_pred
import cv2
import shutil
import os
import helper
import beepy
import smtplib
from email.mime.text import MIMEText
output_file = 'temp.mp4'
def del_dir(path):
    if os.path.exists(path):
        shutil.rmtree(path)
        os.mkdir(path)
def frame_to_vid(frames_dir):
    if os.path.isfile(output_file):
        os.remove(output_file)
    d = {int(i.split('_')[-1].split('.')[0]):i for i in os.listdir(frames_dir)}
    d = {k: v for k, v in sorted(d.items(), key=lambda item: item[0])}
    frame = cv2.imread(os.path.join(frames_dir, os.listdir(frames_dir)[0]))
    frame_height, frame_width, channels = frame.shape
    fps = 30
    codec = cv2.VideoWriter_fourcc(*"XVID")
    out = cv2.VideoWriter(output_file, codec, fps, (frame_width,
frame_height))
    # Loop through the frames in the directory and write each frame to the
output video file
    for filename in sorted(list(d.values())):
```

```

        frame = cv2.imread(os.path.join(frames_dir, filename))
        out.write(frame)
    # Release the video writer object
    out.release()
frame_th = 50
helper.del_dir('temp')
def save_video(video_file):
    with open("uploaded_video.mp4", "wb") as f:
        f.write(video_file.read())
    return "uploaded_video.mp4"
def read_video(path):
    video_file = open(path, 'rb')
    video_bytes = video_file.read()
    return video_bytes
def send_email(subject, body):
    sender_email = 'warrior.prince652002@gmail.com'
    sender_password = 'xqpmjdnjuwjkmo'
    receiver_email = 'silver065lu@gmail.com'
    message = MIMEText(body)
    message['Subject'] = subject
    message['From'] = sender_email
    message['To'] = receiver_email
    smtp_server = smtplib.SMTP('smtp.gmail.com', 587)
    smtp_server.starttls()
    smtp_server.login(sender_email, sender_password)
    smtp_server.sendmail(sender_email, receiver_email,
message.as_string())
    smtp_server.quit()
CLASSES_LIST = ['Arson',
'Normal',
'Vandalism',
'Arrest',
'Abuse',
'Assault',

```

```

'Explosion',
'Fighting',
'Shooting',
'Stealing',
'RoadAccidents',
'Robbery',
'Shoplifting',
'Burglary']
st.title('Anaomaly detection app')
choice = st.sidebar.radio(label='Select',options=['Upload'])
def display_images():
    st.title('Displaying Two Images')
    # Path to your images
    image_path1 = 'loss.png'
    image_path2 = 'acc.png'
    # Display the first image
    st.image(image_path1, caption='Image 1', use_column_width=True)
    st.image(image_path2, caption='Image 2', use_column_width=True)
st.title('Anomaly detection app')
choice = st.sidebar.radio(label='Select', options=['Upload', 'Display Images'])
if choice == 'Upload':
    video_file = st.file_uploader("Upload video", type=["mp4"])
    if st.button('Predict'):
        if video_file is not None:
            vid_path = save_video(video_file)
            video_bytes = read_video(vid_path)
            st.video(video_bytes)
            pred,prob = vid_pred.vid_class_pred(vid_path,CLASSES_LIST)
            beepy.beep(sound='error')
            st.markdown(f'Prediction class : {pred}')
            st.markdown(f'Prediction prob : {prob}')
            send_email('Anomaly detected', f'Anomaly detected in the video.
Class: {pred}, Probability: {prob}')
        else:

```

```

        st.error('Video file not uploaded')
elif choice == 'Display Images':
    display_images()
else:
    pass
import pandas as pd
def filter_df(path):
    df = pd.read_csv(path,names=['file_name', 'label','class'])
    df['file_name'] = df['file_name'].apply(lambda x:x+'.mp4')
    df = df[df['class'] == 1]
    file_names = df['file_name'].unique().tolist()
    return file_names
import os
from tqdm.auto import tqdm
old_folder = os.path.join('/content','DCSASS Dataset')
labels_files = os.path.join('/content','Labels')
file_paths = {}
for i in tqdm(os.listdir(old_folder)):
    I1 = os.listdir(os.path.join(old_folder,i))
    path = os.path.join(labels_files,i+'.csv')
    files = filter_df(path)
    file_paths[i] = []
    for j in I1:
        I2 = os.listdir(os.path.join(old_folder,i,j))
        for k in I2:
            if k in files:
                file_paths[i].append(os.path.join(old_folder,i,j,k))
!rm -rf video_data
!mkdir video_data
new_folder = os.path.join('/content','video_data')
th = 50
for i,j in tqdm(file_paths.items()):
    os.mkdir(os.path.join(new_folder,i))
    for idx,k in enumerate(j):

```

```

        file_name = k.split('/')[-1]
        dest_path = os.path.join(new_folder,i,file_name)
        os.rename(k,dest_path)
        if (idx+1) == th:
            break
!rm -rf "/content/DCSASS Dataset" /content/Labels
seed_constant = 27
np.random.seed(seed_constant)
random.seed(seed_constant)
tf.random.set_seed(seed_constant)
# Create a Matplotlib figure and specify the size of the figure.
plt.figure(figsize = (20, 20))
# Get the names of all classes/categories in UCF50.
all_classes_names = os.listdir('/content/video_data')
# Generate a list of 20 random values. The values will be between 0-50,
# where 50 is the total number of class in the dataset.
random_range = random.sample(range(len(all_classes_names)), 12)
# Iterating through all the generated random values.
for counter, random_index in enumerate(random_range, 1):
    # Retrieve a Class Name using the Random Index.
    selected_class_Name = all_classes_names[random_index]
    # Retrieve the list of all the video files present in the randomly selected
    Class Directory.
    video_files_names_list =
os.listdir(f'/content/video_data/{selected_class_Name}')
    # Randomly select a video file from the list retrieved from the randomly
    selected Class Directory.
    selected_video_file_name = random.choice(video_files_names_list)
    # Initialize a VideoCapture object to read from the video File.
    video_reader =
cv2.VideoCapture(f'/content/video_data/{selected_class_Name}/{selected_v
ideo_file_name}')
    # Read the first frame of the video file.
    _, bgr_frame = video_reader.read()

```

```

# Release the VideoCapture object.
video_reader.release()
# Convert the frame from BGR into RGB format.
rgb_frame = cv2.cvtColor(bgr_frame, cv2.COLOR_BGR2RGB)
# Write the class name on the video frame.
cv2.putText(rgb_frame, selected_class_Name, (10, 30),
cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 255, 255), 2)
# Display the frame.
plt.subplot(5, 4, counter);plt.imshow(rgb_frame);plt.axis('off')
os.listdir('/content/video_data')
# Specify the number of frames of a video that will be fed to the model as one
sequence.
SEQUENCE_LENGTH = 20
# Specify the directory containing the UCF50 dataset.
DATASET_DIR = "/content/video_data/"
# Specify the list containing the names of the classes used for training. Feel
free to choose any set of classes.
CLASSES_LIST = ['Robbery',
'Explosion',
'Shoplifting',
'Arrest',
'Fighting',
'RoadAccidents']
def frames_extraction(video_path):
    """
    This function will extract the required frames from a video after resizing
    and normalizing them.
    Args:
        video_path: The path of the video in the disk, whose frames are to be
        extracted.
    Returns:
        frames_list: A list containing the resized and normalized frames of the
        frames_list = []
    # Read the Video File using the VideoCapture object.

```

```

video_reader = cv2.VideoCapture(video_path)
# Get the total number of frames in the video.
video_frames_count =
int(video_reader.get(cv2.CAP_PROP_FRAME_COUNT))
# Calculate the the interval after which frames will be added to the list.
skip_frames_window =
max(int(video_frames_count/SEQUENCE_LENGTH), 1)
# Iterate through the Video Frames.
for frame_counter in range(SEQUENCE_LENGTH):
    # Set the current frame position of the video.
    video_reader.set(cv2.CAP_PROP_POS_FRAMES, frame_counter *
skip_frames_window)
    # Reading the frame from the video.
    success, frame = video_reader.read()
    # Check if Video frame is not successfully read then break the loop
    if not success:
        break
    # Resize the Frame to fixed height and width.
    resized_frame = cv2.resize(frame, (IMAGE_HEIGHT, IMAGE_WIDTH))
    # Normalize the resized frame by dividing it with 255 so that each
pixel value then lies between 0 and 1
    normalized_frame = resized_frame / 255
    # Append the normalized frame into the frames list
    frames_list.append(normalized_frame)
# Release the VideoCapture object.
video_reader.release()
# Return the frames list.
return frames_list
def create_dataset():
    features = []
    labels = []
    video_files_paths = []
    # Iterating through all the classes mentioned in the classes list
    for class_index, class_name in enumerate(CLASSES_LIST)

```

```

    # Display the name of the class whose data is being extracted.
    print(f'Extracting Data of Class: {class_name}')

    # Get the list of video files present in the specific class name
    directory.
    files_list = os.listdir(os.path.join(DATASET_DIR, class_name))
    # Iterate through all the files present in the files list.
    for file_name in files_list:
        # Get the complete video path.
        video_file_path = os.path.join(DATASET_DIR, class_name,
file_name)
        # Extract the frames of the video file.
        frames = frames_extraction(video_file_path)
        if len(frames) == SEQUENCE_LENGTH:
            # Append the data to their repective lists.
            features.append(frames)
            labels.append(class_index)
            video_files_paths.append(video_file_path)

    # Converting the list to numpy arrays
    features = np.asarray(features)
    labels = np.array(labels)

    # Return the frames, class index, and video file path.
    return features, labels, video_files_paths
features, labels, video_files_paths = create_dataset()
def create_convlstm_model():
    model.add(tf.keras.layers.MaxPooling3D(pool_size=(1, 2, 2),
padding='same', data_format='channels_last'))
    model.add(tf.keras.layers.TimeDistributed(tf.keras.layers.Dropout(0.2)))

    model.add(tf.keras.layers.ConvLSTM2D(filters = 8, kernel_size = (3, 3),
activation = 'tanh', data_format = "channels_last",
recurrent_dropout=0.2, return_sequences=True))
    model.add(tf.keras.layers.MaxPooling3D(pool_size=(1, 2, 2),
padding='same', data_format='channels_last'))
    model.add(tf.keras.layers.TimeDistributed(tf.keras.layers.Dropout(0.2)))

```

```

        model.add(tf.keras.layers.ConvLSTM2D(filters = 14, kernel_size = (3,
3), activation = 'tanh', data_format = "channels_last",
        recurrent_dropout=0.2, return_sequences=True))
        model.add(tf.keras.layers.MaxPooling3D(pool_size=(1, 2, 2),
padding='same', data_format='channels_last'))
        model.add(tf.keras.layers.TimeDistributed(tf.keras.layers.Dropout(0.2)))
        model.add(tf.keras.layers.ConvLSTM2D(filters = 16, kernel_size = (3,
3), activation = 'tanh', data_format = "channels_last",
        recurrent_dropout=0.2, return_sequences=True))
        model.add(tf.keras.layers.MaxPooling3D(pool_size=(1, 2, 2),
padding='same', data_format='channels_last'))
        #model.add(TimeDistributed(Dropout(0.2)))
        model.add(tf.keras.layers.Flatten())
        model.add(tf.keras.layers.Dense(len(CLASSES_LIST), activation =
"softmax"))
        # Display the models summary.
        model.summary()
        # Return the constructed convlstm model.
        return model
convlstm_model = create_convlstm_model()
print("Model Created Successfully!")
# Plot the structure of the constructed model.
tf.keras.utils.plot_model(convlstm_model, to_file =
'convlstm_model_structure_plot.png', show_shapes = True,
show_layer_names = True)
# Create an Instance of Early Stopping Callback
early_stopping_callback = tf.keras.callbacks.EarlyStopping(monitor =
'val_loss', patience = 10, mode = 'min', restore_best_weights = True)
# Compile the model and specify loss function, optimizer and metrics values
to the model
convlstm_model.compile(loss = 'categorical_crossentropy', optimizer =
'Adam', metrics = ["accuracy"])
# Start training the model.
convlstm_model_training_history = convlstm_model.fit(x = features_train, y

```

```

= labels_train, epochs = 5, batch_size = 4, shuffle = True, validation_split =
0.2, callbacks = [early_stopping_callback])
# Get the loss and accuracy from model_evaluation_history.
model_evaluation_loss,          model_evaluation_accuracy          =
model_evaluation_history
# Define the string date format.
# Get the current Date and Time in a DateTime Object.
# Convert the DateTime object to string according to the style mentioned in
date_time_format string.
date_time_format = '%Y_%m_%d__%H_%M_%S'
current_date_time_dt = dt.datetime.now()
current_date_time_string    =    dt.datetime.strftime(current_date_time_dt,
date_time_format)
# Define a useful name for our model to make it easy for us while navigating
through multiple saved models.
model_file_name = f'convlstm_model_{current_date_time_string}.h5'
# Save your Model.
convlstm_model.save(model_file_name)
def plot_metric(model_training_history, metric_name_1, metric_name_2,
plot_name):
    # Get metric values using metric names as identifiers.
    metric_value_1 = model_training_history.history[metric_name_1]
    metric_value_2 = model_training_history.history[metric_name_2]
    # Construct a range object which will be used as x-axis (horizontal plane)
of the graph.
    epochs = range(len(metric_value_1))
    # Plot the Graph.
    plt.plot(epochs, metric_value_1, 'blue', label = metric_name_1)
    plt.plot(epochs, metric_value_2, 'red', label = metric_name_2)
    # Add title to the plot.
    plt.title(str(plot_name))
    # Add legend to the plot.
    plt.legend()
plot_metric(convlstm_model_training_history, 'loss', 'val_loss', 'Total Loss vs

```

```

Total Validation Loss')
plot_metric(convlstm_model_training_history, 'accuracy', 'val_accuracy',
'Total Accuracy vs Total Validation Accuracy')
def frames_extraction(video_path):
    """
    This function will extract the required frames from a video after resizing
    and normalizing them.
    Args:
        video_path: The path of the video in the disk, whose frames are to be
        extracted.
    Returns:
        frames_list: A list containing the resized and normalized frames of the v
# Declare a list to store video frames.
frames_list = []
    # Read the Video File using the VideoCapture object.
video_reader = cv2.VideoCapture(video_path)
    # Get the total number of frames in the video.
video_frames_count =
int(video_reader.get(cv2.CAP_PROP_FRAME_COUNT))
    # Calculate the the interval after which frames will be added to the list.
skip_frames_window =
max(int(video_frames_count/SEQUENCE_LENGTH), 1)
    # Iterate through the Video Frames.
for frame_counter in range(SEQUENCE_LENGTH):
    # Set the current frame position of the video.
    video_reader.set(cv2.CAP_PROP_POS_FRAMES, frame_counter *
skip_frames_window)
    # Reading the frame from the video.
    success, frame = video_reader.read()
    # Check if Video frame is not successfully read then break the loop
    if not success:
        break
    # Resize the Frame to fixed height and width.
    resized_frame = cv2.resize(frame, (IMAGE_HEIGHT, IMAGE_WIDTH))

```

```

        # Normalize the resized frame by dividing it with 255 so that each pixel
value then lies between 0 and 1
        normalized_frame = resized_frame / 255
        # Append the normalized frame into the frames list
        frames_list.append(normalized_frame)
    # Release the VideoCapture object.
    video_reader.release()
    # Return the frames list.
    return frames_list

def vid_class_pred(path,class_list):
    arr = np.array(frames_extraction(path))
    arr = np.expand_dims(arr, axis=0)
    model_pred = convlstm_model.predict(arr).ravel()
    pred_prob = max(model_pred)
    pred_class = class_list[np.argmax(model_pred)]
    return pred_class,pred_prob

import mediapy as media
vid_path = '/content/video_data/Explosion/Explosion006_x264_19.mp4'
video = media.read_video(vid_path)
media.show_video(video, fps=30)
pred,prob = vid_class_pred(vid_path,CLASSES_LIST)

```

B. SCREENSHOTS

```
helper.py > frame_to_vid
1  import cv2
2  import os
3  import shutil
4  output_file = 'temp.mp4'
5
6  def del_dir(path):
7      if os.path.exists(path):
8          shutil.rmtree(path)
9          os.mkdir(path)
10
11 def frame_to_vid(frames_dir):
12     if os.path.isfile(output_file):
13         os.remove(output_file)
14
15     d = {int(i.split('_')[-1].split('.')[0]):i for i in os.listdir(frames_dir)}
16     d = {k: v for k, v in sorted(d.items(), key=lambda item: item[0])}
17
18     frame = cv2.imread(os.path.join(frames_dir, os.listdir(frames_dir)[0]))
19     frame_height, frame_width, channels = frame.shape
20     fps = 30
21
22     codec = cv2.VideoWriter_fourcc(*"XVID")
23     out = cv2.VideoWriter(output_file, codec, fps, (frame_width, frame_height))
24     # Loop through the frames in the directory and write each frame to the output video file
25     for filename in sorted(list(d.values())):
26         frame = cv2.imread(os.path.join(frames_dir, filename))
27         out.write(frame)
28
29     # Release the video writer object
30     out.release()
```

```
1  import streamlit as st
2  import vid_pred
3  import cv2
4  import shutil
5  import os
6  import helper
7  import beepy
8  import smtplib
9  from email.mime.text import MIMEText
10
11 frame_th = 50
12 helper.del_dir('temp')
13
14 def save_video(video_file):
15     with open("uploaded_video.mp4", "wb") as f:
16         f.write(video_file.read())
17     return "uploaded_video.mp4"
18
19 def read_video(path):
20     video_file = open(path, 'rb')
21     video_bytes = video_file.read()
22     return video_bytes
23
24
25 def send_email(subject, body):
26     sender_email = 'warrior.prince652002@gmail.com'
27     sender_password = 'xqpmjdnjuwjxkmoo'
28     receiver_email = 'silver065lu@gmail.com'
29     message = MIMEText(body)
30     message['Subject'] = subject
31     message['From'] = sender_email
32     message['To'] = receiver_email
33     smtp_server = smtplib.SMTP('smtp.gmail.com', 587)
34     smtp_server.starttls()
35     smtp_server.login(sender_email, sender_password)
36     smtp_server.sendmail(sender_email, receiver_email, message.as_string())
37     smtp_server.quit()
```

```

39 CLASSES_LIST = ['Arson',
40                 'Normal',
41                 'Vandalism',
42                 'Arrest',
43                 'Abuse',
44                 'Assault',
45                 'Explosion',
46                 'Fighting',
47                 'Shooting',
48                 'Stealing',
49                 'RoadAccidents',
50                 'Robbery',
51                 'Shoplifting',
52                 'Burglary']
53
54 st.title('Anaomaly detection app')
55 choice = st.sidebar.radio(label='Select',options=['Upload'])
56
57 def display_images():
58     st.title('Displaying Two Images')
59
60     # Path to your images
61     image_path1 = 'loss.png'
62     image_path2 = 'acc.png'
63
64     # Display the first image
65     st.image(image_path1, caption='Image 1', use_column_width=True)
66
67     # Display the second image
68     st.image(image_path2, caption='Image 2', use_column_width=True)
69
70     # ... (your existing code)
71
72 st.title('Anomaly detection app')
73 choice = st.sidebar.radio(label='Select', options=['Upload', 'Display Images'])
74
75
76 if choice == 'Upload':
77     video_file = st.file_uploader("Upload video", type=["mp4"])
78     if st.button('Predict'):
79         if video_file is not None:
80             vid_path = save_video(video_file)
81             video_bytes = read_video(vid_path)
82             st.video(video_bytes)
83             pred,prob = vid_pred.vid_class_pred(vid_path,CLASSES_LIST)
84             beepy.beep(sound='error')
85             st.markdown(f'Prediction class : {pred}')
86             st.markdown(f'Prediction prob : {prob}')
87             send_email('Anomaly detected', f'Anomaly detected in the video. Class: {pred}, Probability: {prob}')
88         else:
89             st.error('Video file not uploaded')
90 elif choice == 'Display Images':
91     display_images()
92 else:
93     pass

```

Fig. B.1: Importing Required Libraries



Fig. B.2: Plotting Loss and Accuracy Curves of Conv LSTM Model

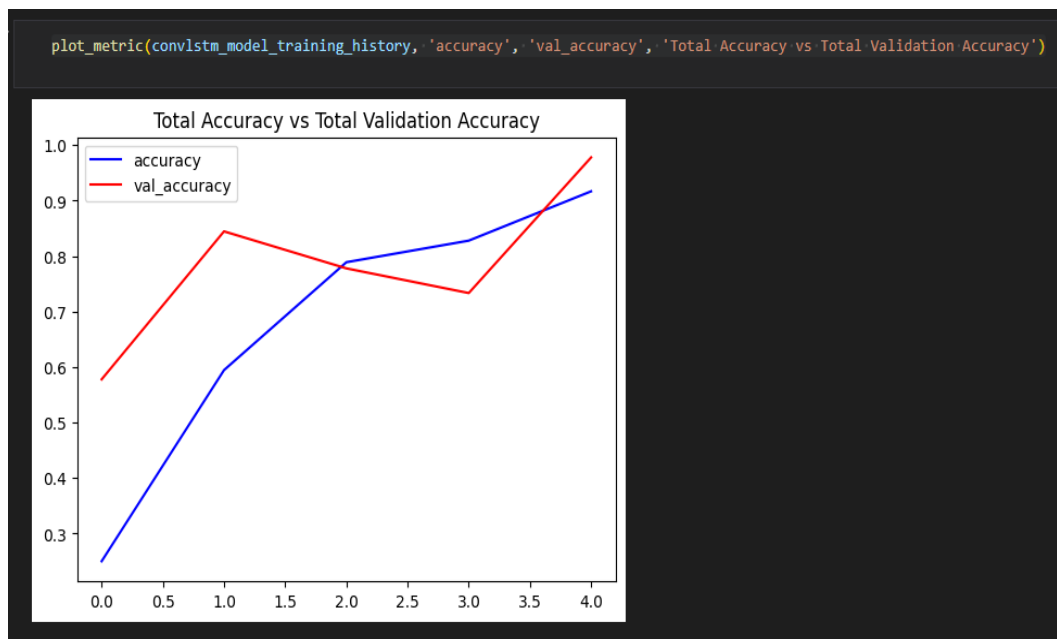


Fig. B.3: Total Accuracy vs Validation Accuracy



Fig. B.4: Predicting the Explosion

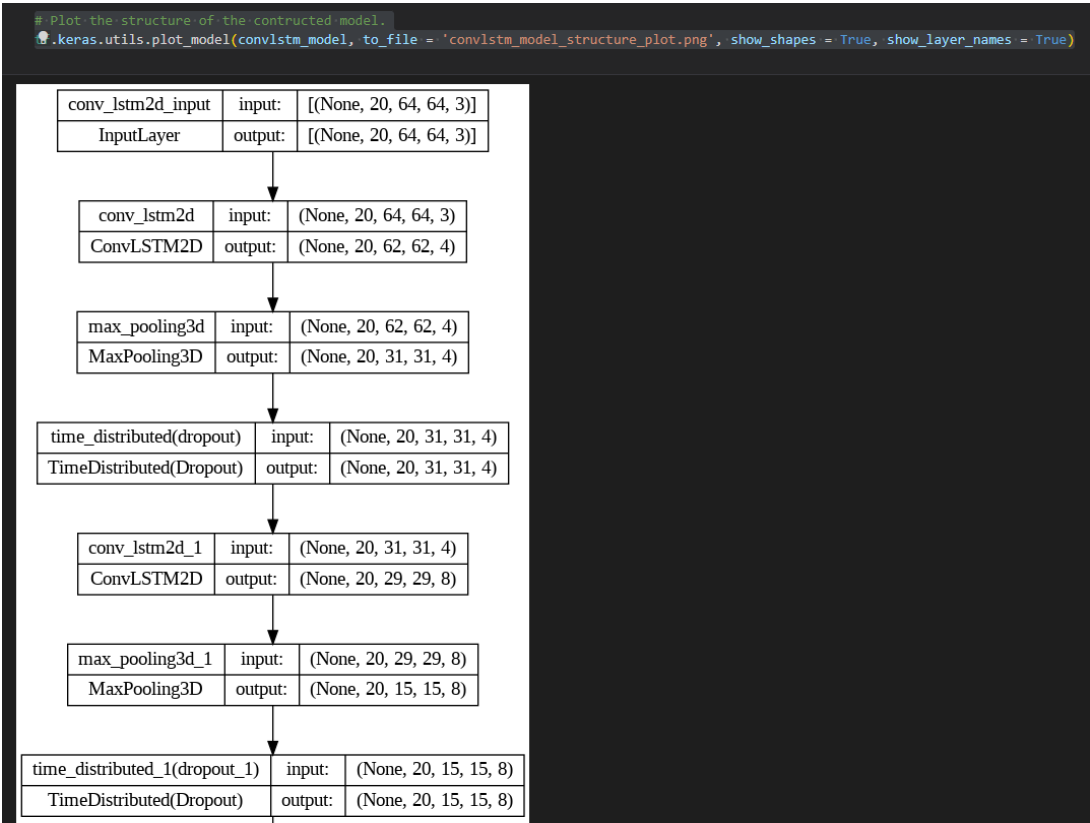


Fig. B.5: The Structure of the Construction Models

```

PS C:\Users\Mahesh\OneDrive\Desktop\AnomalyDetection> streamlit run
Usage: streamlit run [OPTIONS] TARGET [ARGS]...
Try 'streamlit run --help' for help.

Error: Missing argument 'TARGET'.
PS C:\Users\Mahesh\OneDrive\Desktop\AnomalyDetection> streamlit run app.py

You can now view your Streamlit app in your browser.

Local URL: http://localhost:8501
Network URL: http://192.168.1.15:8501

2024-04-08 00:16:46.599497: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'cudart64_110.dll'; dlderror: cudart64_110.dll not found
2024-04-08 00:16:46.600319: I tensorflow/stream_executor/cuda/cudart_stub.cc:29] Ignore above cudart dlderror if you do not have a GPU set up on your machine.
2024-04-08 00:16:58.756204: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'cudart64_110.dll'; dlderror: cudart64_110.dll not found
2024-04-08 00:16:58.757476: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'cublas64_11.dll'; dlderror: cublas64_11.dll not found
2024-04-08 00:16:58.758264: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'cublaslt64_11.dll'; dlderror: cublaslt64_11.dll not found
2024-04-08 00:16:58.759166: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'cufft64_10.dll'; dlderror: cufft64_10.dll not found
2024-04-08 00:16:58.760104: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'curand64_10.dll'; dlderror: curand64_10.dll not found
2024-04-08 00:16:58.760945: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'cusolver64_11.dll'; dlderror: cusolver64_11.dll not found
2024-04-08 00:16:58.762064: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'cuspars64_11.dll'; dlderror: cuspars64_11.dll not found
2024-04-08 00:16:58.763381: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'cudnn64_8.dll'; dlderror: cudnn64_8.dll not found
2024-04-08 00:16:58.763730: W tensorflow/core/common_runtime/gpu/gpu_device.cc:1858] Cannot dlopen some GPU libraries. Please make sure the missing libraries mentioned above are installed properly if you would like to use GPU. Follow the guide at https://www.tensorflow.org/install/gpu for how to download and setup the required libraries for your platform.
Skipping registering GPU devices...
2024-04-08 00:16:58.772222: I tensorflow/core/platform/cpu_feature_guard.cc:151] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (oneDNN) to use the following CPU instructions in performance-critical operations: AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.

```

Fig. B.6: Running the Streamlit run app.py

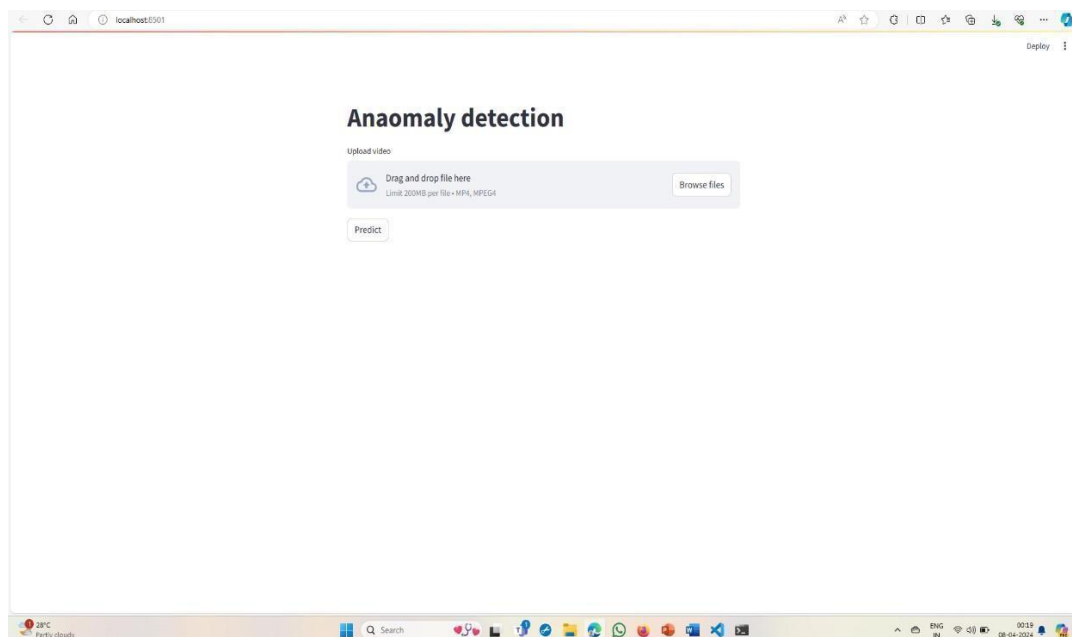


Fig. B.7: Login Page Layout

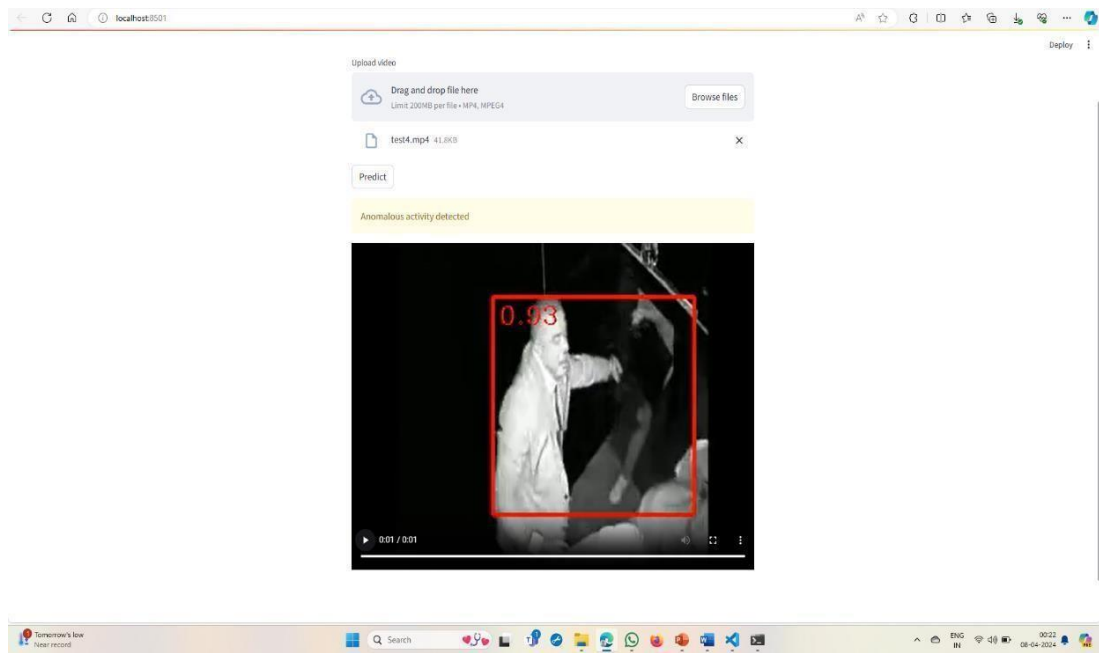


Fig. B.8: Picture showing predicted output

C. RESEARCH PAPER

Home security Anomaly Detection System

Bandaru Mahesh Sai
U.G Scholar
Department of Information
Technology
Sathyabama Institute of Science
and Technology
Chennai, India
mail2maheshbandaru@gmail.com

C.Reddy Venkata Mohitheswar
U.G Scholar
Department of Information
Technology
Sathyabama Institute of Science
and Technology , Chennai
mohitheswarchinduluri@gmail.co
m

Dr. A. Sathiyraj
Assitant professor
Department of Information
Technology
Sathyabama Institute of Science
and Technology
Chennai-600119
sathiyaraj.a.it@sathyabama.ac.in

ABSTRACT

The Home Security Anomaly Detection System is a cutting-edge piece of equipment that aims to increase domestic security by alerting households to possible intrusions or odd activity. Using powerful machine learning algorithms and artificial intelligence techniques, this system analyzes data from a range of sources inside the home, including security cameras, motion sensors, door/window sensors, and smart home devices. By continuously observing numerous data sources, the system can identify aberrant patterns or behaviors that depart from the homeowners' usual routines or planned activities.

Using past data, the anomaly detection system creates a baseline of expected patterns and behaviors, and then continuously compares incoming data to this baseline. If any deviations or anomalies are discovered, the system promptly notifies the homeowners via a mobile application or through a centralized monitoring station. The warnings can be adjusted based on the magnitude of the anomaly, which might range from minor errors to significant security risks.

By decreasing false alarms and improving the overall accuracy of spotting true security breaches, this cutting-edge

solution gives homeowners an extra layer of security and peace of mind. A seamless integration with the homeowner's selected security setup is made possible by the freedom it offers in interacting with existing home security systems.

Keywords: *Home Security, Anomaly Detection System, Machine Learning, Artificial Intelligence, Data Analysis, Security Cameras, Motion Sensors,*

Door/Window Sensors, Smart Home Devices, Baseline, Deviations, Anomalies, Alerts, Mobile Application, Central Monitoring Station, Accuracy, False Alarms, Integration.

I. INTRODUCTION

A cutting-edge solution created to give homeowners increased safety and peace of mind is the Home Security Anomaly Detection System. Using cutting-edge technologies that can spot and stop these crimes has become essential due to the rising concern for home security and the prevalence of break-ins and burglaries.

This cutting-edge system monitors and assesses numerous facets of a home's security using a combination of artificial intelligence, machine learning algorithms, and smart sensors. It continuously scans the family for trends and behaviors, enabling it to spot any irregularities that would point to a potential security issue.

This system is especially made to be very effective in spotting anomalies and warning homeowners before any serious harm occurs, whether it be detecting suspicious movements, strange activities, or even unexpected environmental changes.

The system is made up of various parts that work in unison to provide thorough coverage of the entire space. They include environmental sensors for tracking temperature, humidity, and smoke levels, motion detectors, security cameras, and door/window sensors. Each device has cutting-edge capabilities for precise data collection and processing, assuring great accuracy in anomaly detection.

The effectiveness of the system is greatly influenced by the underlying machine learning algorithms. These algorithms improve their ability to detect anomalies over time by continuously learning from the past and adjusting to new trends. This improves dependability and reduces false alerts by allowing the system to distinguish between routine household activities and potential security concerns.

The system immediately notifies homeowners through a variety of channels, such as smartphone notifications, email, or even SMS, in the event that any irregularity is identified. Homeowners may respond quickly thanks to this prompt notification, such as calling the police or remotely accessing security camera feeds to assess the situation. The device may also be connected to smart home automation capabilities, giving customers the ability to lock doors, turn on lights, or activate sirens to ward off attackers.

The Home Security Anomaly Detection System additionally provides sophisticated remote monitoring capabilities, enabling homeowners to keep an eye on their property even when they are away. Users may obtain real-time status updates,

examine film, personalize settings, and get alerts from any location using an intuitive smartphone app or web-based dashboard.

As a result, the Home Security Anomaly Detection System is a thorough and clever solution that makes use of AI, machine learning, and cutting-edge sensors to offer homeowners an unrivaled level of security. This technology offers prompt warning by continuously watching for irregularities and spotting them, allowing homeowners to take rapid action and stop any security risks. With its cutting-edge capabilities and simplicity of use, this system significantly improves home security, giving customers peace of mind and safeguarding what matters most—their house and loved ones.

II. RELATED WORKS

[1] This study examines the detection and correlation of anomalous traffic in smart home automation IoT systems, emphasizing the significance of spotting anomalous network traffic patterns to improve security. In smart home contexts where multiple IoT devices are networked, the authors probably investigate techniques and technologies for spotting odd network behavior. They want to increase the overall security of such systems by correlating anomalies.

[2] With a focus on family-oriented environments, "FamilyGuard" delivers a security architecture intended for anomaly detection in home networks. The goal of the study is to safeguard family members from potential security risks by discussing the architecture's features and how it aids in the detection of odd network activity in residential situations.

[3] This allusion probably introduces a wireless sensor network-based home security system. The study might examine how wireless sensors can be used to keep an eye on and secure residential properties, highlighting the benefits and uses of such a system.

[4] The article probably investigates how to improve network-based anomaly detection in the surroundings of smart homes. The writers might go over methods for better spotting unusual network behavior, which would increase the security of IoT devices in the house.

[5] It appears that this reference covers host-based intrusion detection systems (IDS) and IoT anomaly detection. The article might give an overview of host-based IDS, their uses in IoT environments, and unresolved implementation issues.

[6] The paper probably looks into how much power on-device machine learning models for IoT intrusion detection consume. It might examine the energy implications of such applications as well as how machine learning can be used in IoT devices for intrusion detection.

[7] In IoT environments, deeplearning-based intrusion detection systems (IDS) are likely surveyed in this work. It might go over several deep learning approaches and how well they function for spotting irregularities in IoT networks.

[8] This reference most likely introduces a deep learning-based anomaly-based intrusion detection solution for IoT networks. The design of this system may be described in depth in the paper, highlighting its emphasis on deep learning methods for improved security.

[9] The article likely examines the use of power usage information as a tool for IoT device anomaly identification in smart homes. It might go over how changes in power consumption trends can point to unusual behavior and improve security.

[10] It appears that this study provides a thorough analysis of intrusion detection systems (IDS) for the Internet of Things

(IoT). It might include a summary of current IDS systems, their advantages and disadvantages, and insights on potential future developments in the area of IoT intrusion detection.

III. EXISTING SYSTEM

There are several drawbacks to the current approach for home security anomaly detection. The first is that it frequently costs a lot of money up front. Many homeowners may be discouraged by the cost of installing motion sensors, cameras, and other necessary tools. The system might also need continuous maintenance and monitoring expenses, which would increase the financial burden.

Second, the current approach frequently relies on a steady internet connection. The system can lose its effectiveness or stop working altogether in the event of an internet outage or other disturbance. This is a serious concern because home invasions and burglaries can happen at any moment, including during internet outages.

Perhaps false alarms are still another drawback. Motion sensors are used by the current system to identify irregularities and sound an alarm. These sensors can, however, be set off by moving curtains, pets, or other innocuous activities, which frequently results in false alerts. This not only causes the residents inconvenience and frustration, but it may also lead to a loss of confidence in and credibility with the system.

Additionally, some forms of abnormalities might not be accurately picked up by the current technology. For instance, it can find it difficult to recognize minute alterations in the environment or to find anomalies in places where the sensors or cameras are not present. The security system is left with potential blind spots, which makes it simpler for burglars to avoid detection.

The lack of integration with other smart home technology is a significant drawback as well. The current technology frequently functions as a stand-alone solution, unable to connect to or communicate with other appliances or systems in the home. This reduces the home security system's overall efficacy and potential.

Finally, the current arrangement may raise privacy issues. The privacy of people living in a home may come into question if cameras or other monitoring equipment is used, especially if the material is archived or made available to outsiders. Homeowners may feel uneasy and reluctant to embrace or fully rely on the system as a result of this.

In conclusion, there are a number of drawbacks to the current system for home security anomaly detection, including its high cost, reliance on internet connectivity, potential for false alarms, its limited detection abilities, its lack of integration with other smart home technologies, and privacy concerns.

IV. PROPOSED SYSTEM

In order to improve the safety and security of residential properties, the proposed work focuses on creating a Home Security Anomaly Detection System. Advanced technologies like artificial intelligence, machine learning, and computer vision will be used by the system to detect and identify any unusual or suspicious actions taking place in the home. The system will be able to keep an eye on a variety of components of the house, including motion detectors, door/window sensors, and video surveillance.

The system's video surveillance component will use computer vision algorithms to examine video footage and find unusual occurrences, such as unauthorized entry or suspicious activity inside the building. The motion sensors will be carefully positioned throughout the

house to watch for any unexpected movements or the presence of people in forbidden areas. Further security will be added via door and window sensors, which will notify the system anytime an unwanted entrance attempt is made.

The system will be trained using machine learning techniques to detect anomalies accurately. The system will be able to discern between regular and atypical activity by evaluating a sizable dataset of everyday behaviors, allowing it to only issue alerts in response to legitimate security issues. In addition, the system will continuously pick up on fresh patterns and behaviors, ensuring its usefulness in various residential settings.

In the case of a security breach, homeowners will be able to get real-time notifications on their mobile devices thanks to the planned Home Security Anomaly Detection System's smart notification features. With the aid of this tool, homeowners will be able to take prompt action, such as alerting the appropriate parties or remotely monitoring the circumstance via live video feeds.

In summary, the proposed work on the Home Security Anomaly Detection System seeks to offer homeowners a trustworthy and efficient security solution that goes beyond conventional alarm systems. The system will provide a proactive approach to home security by utilizing cutting-edge technologies and clever algorithms, helping to prevent incidents and guarantee the protection of residential properties.

V. SYSTEM ARCHITECTURE

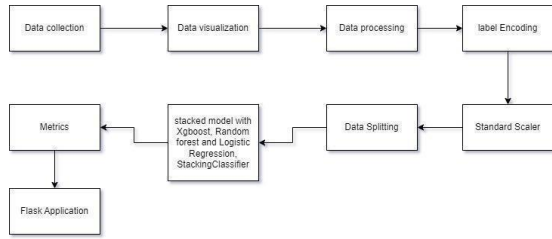


Fig. 1. System Architecture

VI. METHODOLOGY

1. **Intrusion Detection Module:** An essential part of the Home Security Anomaly Detection System is the intrusion detection module. This module focuses on identifying any attempts at infiltration or unauthorised entry inside the house. To monitor the property, it makes use of a variety of sensors and gadgets, including security cameras, door/window sensors, and motion detectors. When these sensors detect any unusual movements or activities inside the house, an alert or alarm is set off to notify the owner or security professionals. Additionally, the module contains complex algorithms that examine the data gathered to distinguish between regular operations and suspected intrusions. This study aids in reducing false alarms and enhancing intrusion detection accuracy. The module can also be connected to the homeowner's smartphone or integrated with a centralized monitoring system to deliver warnings and alerts in real time.

2. **Environmental Monitoring Module:** By keeping track of numerous environmental aspects of a home, the environmental monitoring module is in charge of guaranteeing the residents' safety and comfort. The air quality, temperature, humidity, and the existence of smoke or gas leaks are just a few of the characteristics that this module's sensors and detectors keep an eye on. These sensors are thoughtfully positioned all over the house to offer complete coverage. The module initiates the proper steps in the event of any abnormal changes or

threshold breaches, including turning on ventilation systems, alerting residents or emergency services, and, if necessary, turning off the gas or power supply. By managing heating, cooling, and ventilation systems based on real-time data, the environmental monitoring module not only contributes to energy efficiency but also to the prevention of accidents and health risks.

3. **Remote Control Module:** This component enables users to monitor and manage their home security system from a distance. Frequently, a user-friendly web interface or mobile application can be used to access this module. Homeowners can use it to arm or disarm the security system, see real-time video feeds, and get alerts and notifications in real-time. The convenience of operating additional smart home appliances like lights, locks, and thermostats is also made possible by the module. Homeowners can effortlessly manage their home security system even when they are away with the help of the remote control module, giving them peace of mind and enabling them to act promptly in case of any security or environmental issues. Additionally, this module enables homeowners to remotely monitor the actions of trusted visitors, including family members, friends, or service providers, to whom they have temporarily granted access.

VII. RESULT AND DISCUSSION

A comprehensive solution created to guarantee the safety and security of residential properties is the Home Security Anomaly Detection System. It is a sophisticated system that makes use of cutting-edge technology to find and notify homeowners of any odd behaviors or anomalies in and around the neighbourhood of their homes. To monitor and secure the property, this system integrates numerous elements such as motion sensors, security cameras, door/window sensors, and other IoT

devices. These sensors provide the system with continuous data, which it continuously collects and processes using sophisticated algorithms and machine learning methods. The system may detect any deviations or abnormalities from the usual patterns by creating a baseline of typical activity. The system will automatically identify and categorize something as an anomaly, such as if someone enters the property at an odd hour or there is unexpected movement in a restricted area. The technology quickly notifies homeowners of anomalies via mobile applications or other communication channels, enabling them to take immediate action or, if necessary, notify the appropriate authorities. Homeowners can monitor their houses more efficiently and proactively react to any potential dangers or security breaches thanks to the Home Security Anomaly Detection System, which gives them a fresh sense of security. This system is changing home security with its cutting-edge technology and clever features, providing homeowners and their loved ones with a higher level of security and peace of mind.

VIII. CONCLUSION

As a result, the Home Security Anomaly Detection System provides a thorough and trustworthy option for identifying security risks in domestic settings. The system can precisely recognize odd actions and send real-time alerts to homeowners, protecting their homes and loved ones, thanks to its cutting-edge technology and clever algorithms. The system's ability to monitor and analyze data effectively is improved by the integration of numerous sensors and surveillance equipment, enabling it to differentiate between normal and suspicious behavior. Also, homeowners can easily control and customize the system to suit their own demands thanks to the user-friendly interface and programmable settings. The Home Security Anomaly Detection System is, all

things considered, a very effective and crucial instrument for protecting homes and fostering peace of mind.

IX. FUTURE WORK

The Home Security Anomaly Detection System's future work could concentrate on a number of areas to improve its efficiency and usability. First, additional research may be done to create more sophisticated anomaly detection algorithms that can successfully find complicated and sophisticated anomalies in various situations. In order to enhance the system's capacity to recognize tiny departures from expected behavior, machine learning techniques like deep learning and reinforcement learning may be used. The system's capacity to monitor and identify anomalies across numerous linked devices and sensors can also be increased by combining it with smart home and Internet of Things (IoT) devices. This would make it possible for the system to give a thorough overview of potential security concerns and improve the house's general security. The integration of the anomaly detection system with other security systems, such as surveillance cameras or access control systems, can also be explored through research. For households, this would offer a more complete and integrated security solution. The system's performance in real-world circumstances should be assessed, and user feedback should be gathered, through extensive field testing and user research. This will help the system's functionality and user experience.

REFERENCES

- [1] Gajewski, M., Mongay Batalla, J., Mastorakis, G., & Mavromoustakis, C. X. (2022). Anomaly traffic detection and correlation in smart home automation IoT systems. *Transactions on Emerging Telecommunications Technologies*, 33(6), e4053.

[2] de Melo, P. H., Miani, R. S., & Rosa, P. F. (2022). FamilyGuard: a security architecture for anomaly detection in home networks. *Sensors*, 22(8), 2895.

[3] Sanjay Satam, S., & El-Ocla, H. (2022). Home security system using wireless sensors network. *Wireless Personal Communications*, 125(2), 1185-1201.

[4] Li, X., Ghodosi, H., Chen, C., Sankupellay, M., & Lee, I. (2022). Improving Network-Based Anomaly Detection in Smart Home Environment. *Sensors*, 22(15), 5626.

[5] Martins, I., Resende, J. S., Sousa, P. R., Silva, S., Antunes, L., & Gama, J. (2022). Host-based IDS: A review and open issues of an anomaly detection system in IoT. *Future Generation Computer Systems*, 133, 95-113.

[6] Tekin, N., Acar, A., Aris, A., Uluagac, A. S., & Gungor, V. C. (2023). Energy consumption of on-device machine learning models for IoT intrusion detection. *Internet of Things*, 21, 100670.

[7] Balaji, R., Deepajothi, S., Prabakaran, G., Daniya, T., Karthikeyan, P., & Velliangiri, S. (2022, April). Survey on intrusions detection system using deep learning in the iot environment. In *2022 International Conference on Sustainable Computing and Data Communication Systems (ICSCDS)* (pp. 195-199). IEEE.

[8] Saba, T., Rehman, A., Sadad, T., Kolivand, H., & Bahaj, S. A. (2022). Anomaly-based intrusion detection system for IoT networks through deep learning model. *Computers and Electrical Engineering*, 99, 107810.

[9] Nimmy, K., Dilraj, M., Sankaran, S., & Achuthan, K. (2022). Leveraging power consumption for anomaly detection on IoT devices in smart homes. *Journal of*

Ambient Intelligence and Humanized Computing, 1-12.

[10] Heidari, A., & Jabraeil Jamali, M. A. (2022). Internet of Things intrusion detection systems: A comprehensive review and future directions. *Cluster Computing*, 1-28.

D. CERIFICATE

