: N° d'ordre
: N° de série

PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA
Ministry of Higher Education and Scientific Researc

UNIVERSITY OF ECHAHID HAMMA LAKHDAR - EL OUED
FACULTY OF EXACT SCIENCES
Computer Science Departmen

Thesis

submitted in partial fulfilment of the requirements for the degree of

ACADEMIC MASTER

Field: **Mathematics and Computer Science**
Option:**Computer Science**
Specialty:**Distributed Systems and Artificial Intelligence**

Presented by : **Saci Wissam**
Semama Sara

Title

Edge AI Solution for Securing and Monitoring of Retail Property

Examination Commitee

M.	.....	MCA	Chair
M.	.....	MAA	Examinator
M.	Khelaifa Abdennacer	MAA	Supervisor
M.	Allal Imed	MAA	Co-Supervisor

Année Universitaire 2022/2023

Acknowledgement

All praise is due to God, who has blessed us with success,
provided for us, and guided us to complete this endeavor.
We understand that no research work can shine without the support of those around us.
With this in mind,
we extend our heartfelt thanks to our dedicated mentors,
Mr.Khelaifa Abdennacer and Mr.Imed Allal and Mr.Abdelghani KABOT
for their unwavering commitment and perseverance by our side.
Our gratitude also extends to our professors,
who have generously shared their academic wisdom
with us throughout our educational journey.
We're delighted to express our appreciation to all who provided encouragement,
guidance, or contributed to the preparation of this research. Lastly,
we extend our sincere thanks to the members of the jury for accepting the responsibility of
discussing and evaluating our work.
With warm thanks

Abstract

The massive development in Artificial Intelligence (AI) and the growing demand for effective security systems have led to the development of smart solutions suitable for securing and monitoring retail properties. This master's thesis presents the design, development, and evaluation of a deep learning-based model that has a specific focus on real-time detection of abnormal behavior in retail environments. The proposed solution takes advantage of the computing capabilities of end devices to process video streams locally, thus reducing reliance on cloud-based infrastructures and enabling faster response times. To achieve this goal, we will prepare a dataset for our project with data from UCF-Crime, which presents a variety of retail scenarios that include many of the anomalous behaviors encountered in real-world retail environments. The proposed model uses the convolutional neural network based on DenseNet-121 as the feature extractor , which is integrated with a detection system developed to capture spatial and temporal signals from video data.

Keywords: Edge AI, Abnormal behavior, DenseNet-121, UCF-Crime.

Résumé

Le développement massif de l'intelligence artificielle (IA) et la demande croissante de systèmes de sécurité efficaces ont conduit au développement de solutions intelligentes adaptées à la sécurisation et à la surveillance des propriétés commerciales. Ce mémoire de master aborde la conception, le développement et l'évaluation d'un modèle basé sur l'apprentissage profond qui se concentre sur la détection en temps réel des comportements anormaux dans les environnements de vente au détail. La solution proposée tire parti des capacités informatiques des terminaux pour traiter les flux vidéo localement, réduisant ainsi la dépendance aux infrastructures basées sur le cloud et permettant des temps de réponse plus rapides. Pour atteindre cet objectif, nous allons préparer un ensemble de données pour notre projet avec des données d'UCF-Crime, qui présente une variété de scénarios de vente au détail qui incluent de nombreux comportements anormaux rencontrés dans les environnements de vente au détail du monde réel. Le modèle proposé utilise le réseau neuronal convolutif basé sur DenseNet-121 comme extracteur de caractéristiques, qui est intégré à un système de détection développé pour capturer des signaux spatiaux et temporels à partir de données vidéo.

Mots clés : IA périphérique, Comportement anormal, DenseNet-121, UCF-Crime.

الملخص

أدى التطور الهائل في الذكاء الاصطناعي (AI) والطلب المتزايد على أنظمة الأمان الفعالة إلى تطوير حلول ذكية مناسبة لتأمين ومراقبة عقارات التجزئة. تقدم مذكرة الماستر هذه تصميم وتطوير وتقييم نموذج يعتمد على التعلم العميق الذي يركز بشكل خاص على الكشف في الوقت الفعلي عن السلوك غير الطبيعي في بيئات البيع بالتجزئة .

يستفيد الحل المقترح من قدرات الحوسبة للأجهزة الطرفية لمعالجة تدفقات الفيديو محليًا ، وبالتالي تقليل الاعتماد على البنى التحتية القائمة على السحابة وتمكين أوقات استجابة أسرع.

لتحقيق هذا الهدف ،سنقوم بعملية تجميع مجموعة بيانات خاصة بمشروعنا وذلك بالاستعانة ببيانات من موقع UCF-Crime ، حيث تقدم مجموعة متنوعة من سيناريوهات البيع بالتجزئة التي تشمل العديد من السلوكيات الشاذة التي تتم مواجهتها في بيئات البيع بالتجزئة الواقعية.

يستخدم النموذج المقترح الشبكة العصبية التلافيفية القائمة على DenseNet-A121 كمستخرج للميزات ، والذي يتكامل مع نظام الكشف المطور لالتقاط الإشارات المكانية والزمانية من بيانات الفيديو.

الكلمات المفتاحية: Edge AI ، السلوك غير الطبيعي ، DenseNet-121، UCF-Crime.

Table of Contents

List of Figures

List of Tables

List of Acronyms

Acronyms / Abbreviations:

IoT..... Internet of Things
DL ..... Deep Learning
ML ..... Machine Learning
TL ..... Transfer Learning
AWS..... Amazon Web Services
SDKs..... software development kits
SaaS..... Software-as-a-service
PaaS..... Platform-as-a-service
IaaS..... Infrastructure-as-a-service
CNN..... convolutional neural network
IFA..... Inter-fused Autoencoder
LSTM..... Long short-term memory
ABD..... Abnormal Behavior Detection
UCF..... University of Central Florida
DenseNet..... Dense Convolutional Network
PIL..... Python Imaging Library
GUI..... Graphical User Interface
OpenCV..... Open Source Computer Vi- sion
ABD..... Abnormal Behavior Detection
Colab..... Colaboratory
HTML..... Hypertext Markup Language
LaTeX..... Lamport's TEX
Google LLC.... Google Limited Liability Company
TF..... TensorFlow
Python API.... Application Programming Interface
OS X..... Operating System X
iOS..... iPhone Operating System
EDGE..... Enhanced Data rates for GSM Evolution
IDE..... Integrated Development Environment
AUC..... Area Under the Curve
val-loss..... Validation loss
val-auc..... Validation AUC
MIST..... ulti-instance self-training

WSAL..... weakly supervised anomaly localization
MILR..... Multiple Instance Learning with ResNet
RTFM..... robust temporal feature magnitude
GODS ..... Generalized One-class Discriminative Subspaces
Macro Avg ..... The macro-averaged

General Introduction

Surveillance cameras have seen a significant uptick in their use across both public and private spaces in recent years. This surge in deployment aims to bolster the security of these areas. Traditionally, companies have employed human monitors to oversee camera feeds. However, human error often leads to the oversight of abnormal events in the footage. This raises questions about the efficiency and effectiveness of human-based monitoring, as it can sometimes prove to be an inefficient allocation of time and resources. The inherent limitations of human monitoring have prompted a need for alternative solutions. Researchers have responded by delving into surveillance data and devising methods for automatic detection of abnormal events. Human monitors might miss critical anomalies, but with automated systems, any unusual occurrence captured by surveillance cameras can be promptly identified. This highlights the pressing issue of enhancing surveillance efficiency and reliability, which we address in our proposed solution.

In this work, we present an innovative application known as ”Abnormal Behavior Detection” (ABD) specifically designed to detect abnormal events within surveillance camera feeds. Our approach builds upon DenseNet-121 as the foundational tool for extracting crucial features from each frame in the input stream. These extracted features then feed into our anomaly detection framework, facilitating the accurate identification of abnormal events in surveillance footage. Furthermore, we not only detect anomalies but also classify them into appropriate categories, demonstrating the robustness of our method in categorizing various types of anomalies. The content of this paper is organized into several chapters to provide a comprehensive understanding of our work:

Chapter 1: IoT and Edge Computing: In this chapter, we delve into the concepts of IoT (Internet of Things) and edge computing, which play a significant role in the deployment and functioning of surveillance systems.

Chapter 2: State of the Art AI Solutions for Detecting Abnormal Behaviors: Here, we review existing AI solutions and methodologies employed in the field of abnormal behavior detection within surveillance systems, providing a foundation for our proposed solution.

Chapter 3: State of the Art AI Solutions for Detecting Abnormal Behaviors: This chapter continues the exploration of AI solutions, delving deeper into the state-of-the-art technologies and methods for detecting abnormal behaviors.

Chapter 4: Proposed Solution: In the final chapter, we present our proposed solution in detail, emphasizing the use of DenseNet-121 and our anomaly detection framework to enhance the efficiency and accuracy of surveillance camera monitoring, ultimately contributing to increased security in various environments.

Chapter 1

IOT and edge computing

Introduction

In this chapter, we will learn about the critical aspect of security in the retail industry and the pivotal role played by Edge and Cloud Computing in ensuring its effectiveness , and also The combination of Edge and Cloud Computing ensures agility, real-time monitoring, scalability, and deep analytics, enabling retailers to proactively protect assets, customer information, and create a secure retail environment. So, let's explore concepts edge computing and IoT through some real world use cases.

1.1 The Internet of Things (IoT)

1.1.1 Definition of Internet of Things (IoT)

Today's , we live in time the Internet of Things (IoT),a modern concept that has greatly expedited the progress of contemporary wireless communications. The central concept behind this concept revolves around the existence of numerous devices (sensors, actuators, mobile phones, etc.) in our surroundings, all capable of intercommunication and cooperation to fulfill common objectives via a distinctive addressing system. Edge computing, a relatively recent paradigm, seeks to position computational capabilities in close proximity to IoT sensors, smartphones, and interconnected technologies[7].

1.1.2 Internet of Things Characteristics :

1. **Connectivity:**

   This doesn't need too much further explanation. With everything going on in IoT devices and hardware, with sensors and other electronics and connected hardware and control systems there needs to be a connection between various levels.

2. **Things:**

   Anything that can be tagged or connected as such as it's designed to be connected. From sensors and household appliances to tagged livestock. Devices can contain sensors or sensing materials can be attached to devices and items.

3. **Data:**

   Data is the glue of the Internet of Things, the first step towards action and intelligence.

4. **Communication**

   Devices get connected so they can communicate data and this data can be analyzed. Communication can occur over short distances or over a long range to very long range.

5. **Intelligence:**

   The aspect of intelligence as in the sensing capabilities in IoT devices and the intelligence gathered from big data analytics (also artificial intelligence).

6. **Action:** he consequence of intelligence. This can be manual action, action based upon debates regarding phenomena (for instance in smart factory decisions) and automation, often the most important piece.

7. **Ecosystem:**

   The context of the Internet of Things in relation to other technologies, communities, objectives, and the broader framework in which it operates. Considerations include the Internet of Everything aspect, the platform aspect, and the imperative for robust collaborations and partnerships[8].

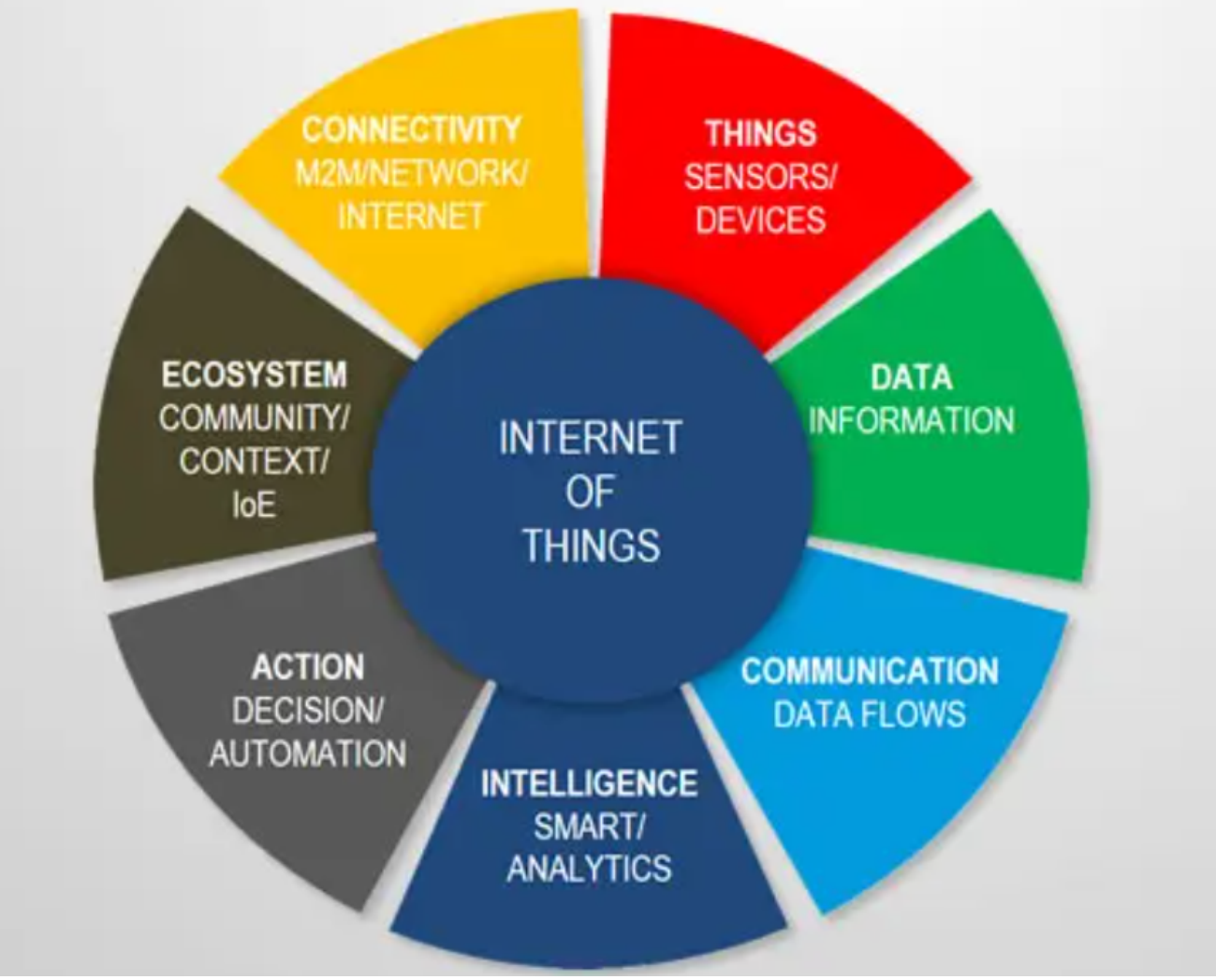

Figure 1.1: Internet of Things Characteristics

1.1.3 Architectures for the Internet of Things :

Basic IoT Architectures In fact, there isn't a single universally accepted IoT architecture. Various researchers and groups have put forth different architectural proposals. Among these, the most fundamental is the protocol-based three-layer architecture Figure 1.2 ,These three layers within the architecture are as follows:

1. **Layer Perception:** This is the physical layer with sensors collecting data/information about the environment.
2. **Layer Network:** This is the layer where the connections to other things, devices and services are realized, and also the sensor data are processed and transmitted.
3. **Layer Application:** This layer is responsible for defining the applications in which IoT can be deployed, and delivering application specific services to the users.

   The 3-layer architecture addresses the fundamental IoT requirements but lacks

the intricate features necessary for conducting advanced IoT research, which can be explored through the incorporation of additional layers. To illustrate, a 5-layer architecture encompasses the following strata: perception, transport, processing, application, and business. The perception and application layers fulfill the previously mentioned functions, while the transport, processing, and business layers operate as follows:

4. **Transport Layer :** This layer facilitates bidirectional transmission of sensor data between the perception layer and the processing layer.

5. **Processing or Middleware Layer:**This layer is responsible for storing, organizing, and processing the substantial volumes of data received from the transport layer.

6. **Business Layer:** The functions of this layer encompass overseeing the entire IoT system, including aspects such as user privacy, applications, profitability, and business processes, among others. [1] □ The 3-layer and 5-layer architectures are illustrated in Figure 1.2 .

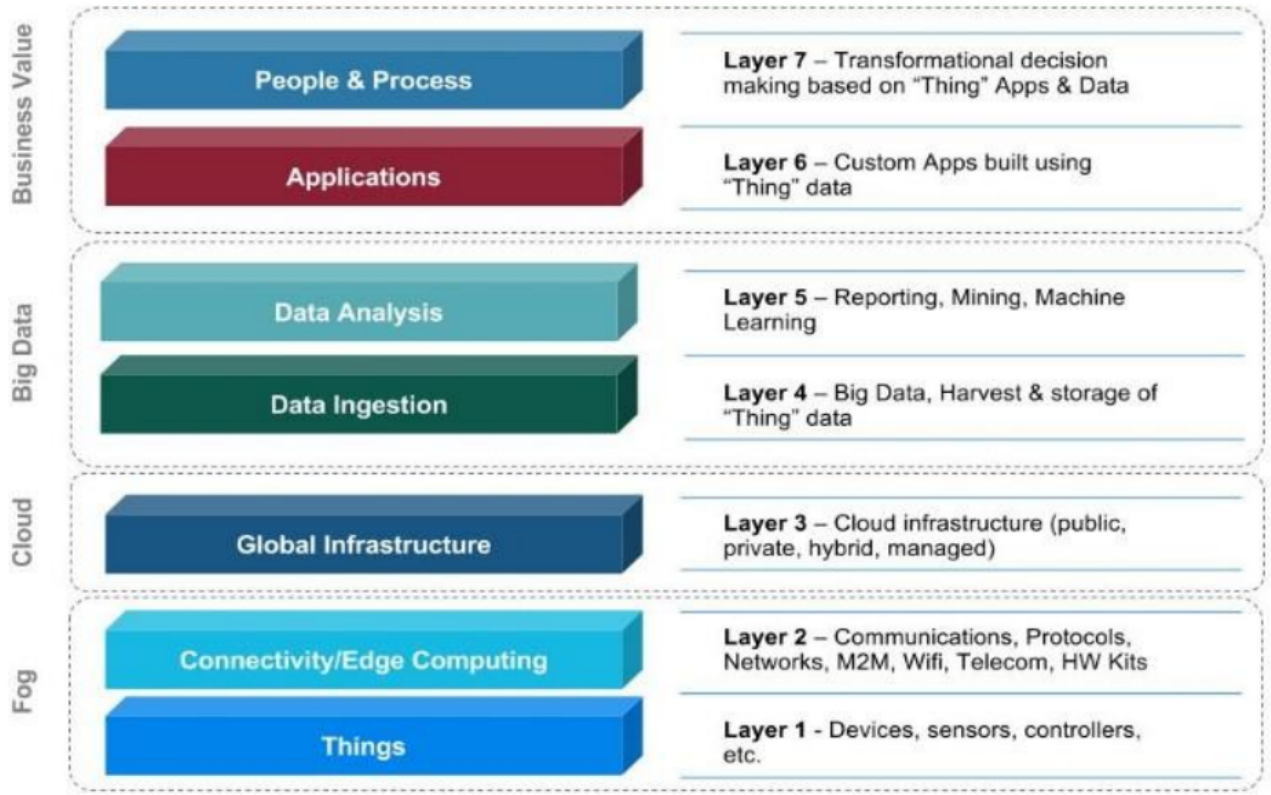

Figure 1.2: IoT represented by a seven layer architecture.

1.1.4 IoT Platforms

An IoT platform refers to an integrated software service system designed to facilitate the inclusion of devices into an IoT system, streamline data flow, support communication, manage devices, and enable the functionality of applications. In particular, IoT platforms assist with the following:

1. Connect hardware.
2. Handle various communication protocols.
3. Integrate with other web services.
4. Offer security and authentication for things, devices, services, and users [1].

Comparison of the IoT Platform

1. **Amazon Web Services (AWS):** As the leader in public cloud computing, Amazon Web Services (AWS) has dedicated substantial efforts to cloud computing, artificial intelligence (AI), and the Internet of Things (IoT) for an extended period. AWS offers IoT services through its AWS IoT platform, equipped with an extensive array of tools.
2. **Google :** Google stands as one of the leading IoT platform providers globally, simplifying the process for developers to construct connected devices. This search engine giant offers Cloud IoT Core as its primary IoT solution, ensuring the development of secure and cutting-edge solutions.
3. **IBM :** IBM has solidified its status as a leader in The Forrester Wave: Industrial IoT Software Platforms, Q3 2018. This recognition substantiates IBM's place as one of the premier providers of IoT services.
4. **Microsoft :** Microsoft also holds a leadership position in Forrester's Industrial IoT Software Platforms, as indicated in the image above. With its suite of IoT services, this technology giant is committed to catalyzing digital transformation across various sectors, including manufacturing, transportation, and retail

[23].

1.1.5 Examples of IoT Applications

IoT has a wide range of applications, spanning from smart homes and wearables to industrial automation and pollution control. It plays a role in energy management,

enhances our daily comforts, and simplifies routine tasks . A partial list of IoT applications is the following: smart homes, smart wearables, smart cities, smart offices, smart agriculture, smart healthcare, smart robotics, smart industrial automation, smart automotive/transportation, smart energy management, smart retail and logistics, smart business.

Smart homes: A smart home is characterized by devices that can communicate with each other and interact with their surroundings. In a smart home, residents have the ability to tailor and oversee the home environment, enhancing security and optimizing energy usage. A diverse range of technologies can be employed to establish smart homes, including smart home lighting, air quality sensing, learning thermostats, smart refrigerator controls, and more (Figure 1.3) [1].

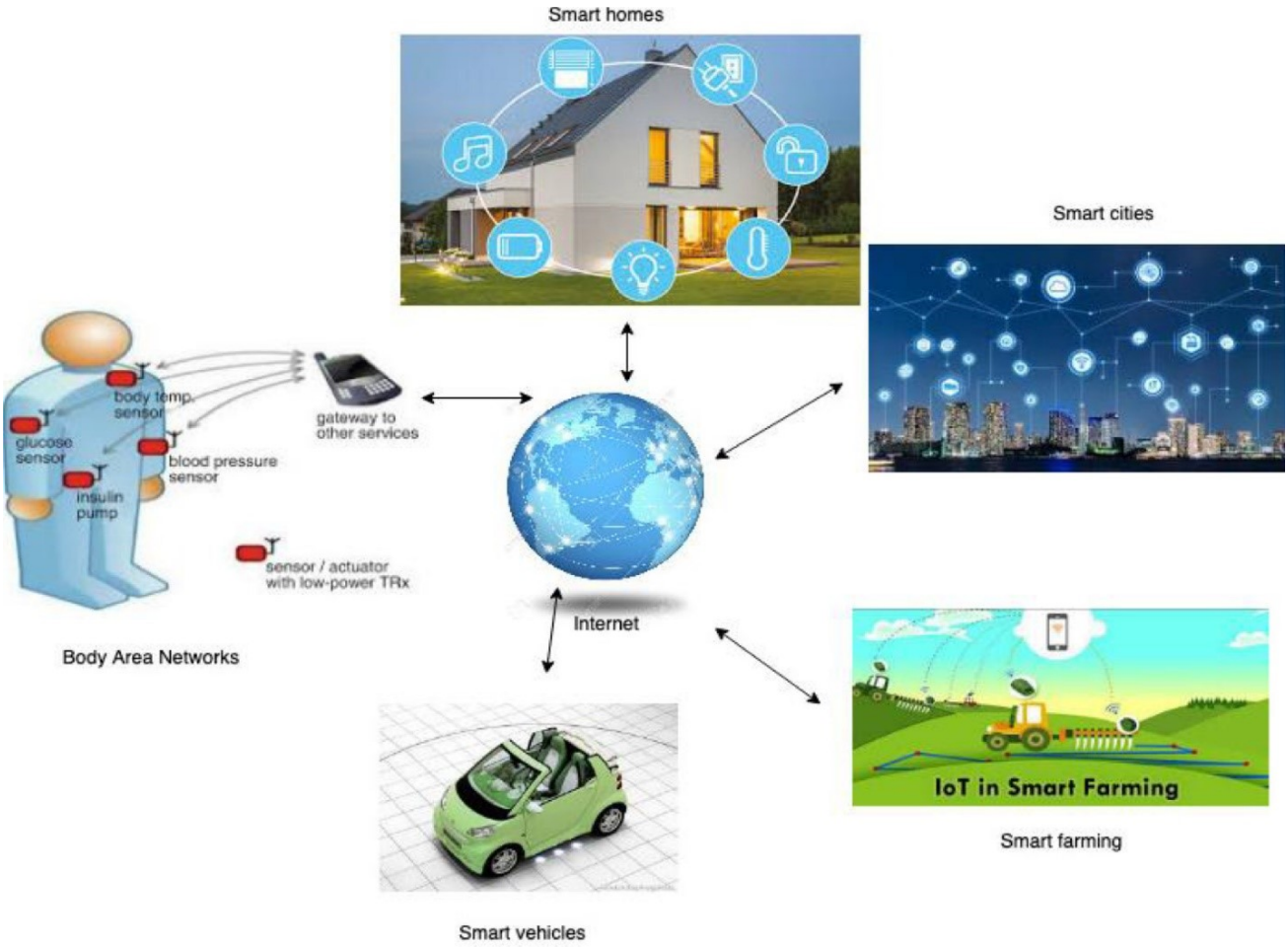

Figure 1.3: Examples of IoT Applications

1.2 Cloud Computing

Cloud computing offers a range of computing services, encompassing servers, databases, networking, software, and data analytics, all delivered over the internet. This approach ensures swift deployment, adaptable resource allocation, and the benefits of

economies of scale. Furthermore, there is a noticeable shift from the centralized paradigm of cloud computing to the decentralized paradigm of fog computing. Fog computing conducts data analytics on edge devices, enabling real-time processing, enhancing data privacy, and reducing operational expenses. The confluence of portable devices, artificial intelligence (AI), and cloud computing lays a strong foundation for the advancement of IoT in the enhancement of security and monitoring within the retail property domain.

1.2.1 Cloud computing models

Cloud computing deployment models are classified into four types models:

1. **Public cloud:** A public cloud infrastructure offers services to the general public over the Internet. This infrastructure is owned, operated, and managed by one or more cloud service providers.
2. **Private cloud:** A private cloud infrastructure delivers services exclusively for a specific purpose within a single organization, catering to various consumers within that organization. This infrastructure is owned, operated, and managed by the organization itself.
3. **Community cloud:** A community cloud infrastructure is employed to offer services for particular use cases shared by specific consumers from different organizations who have common interests or concerns. For example, multiple schools may utilize the same platform provided by a third-party organization to meet their unique needs.
4. **Hybrid cloud:** A hybrid cloud is formed by integrating one or more distinct cloud service models, with each model retaining its individual characteristics. For instance, it might involve combining a public cloud service like Google Cloud with a private cloud service such as Amazon Web Services (AWS). These separate cloud models are interconnected through policies that enable the portability of data and applications, creating a unified and flexible computing environment [10].

1.2.2 Cloud computing services over the Internet

Cloud computing provides services over the Internet. These services are categorized into three categories depending on the provided benefits:

1. **Software-as-a-service (SaaS):** in this model, vendors provide end-users with a software or an application, mainly via a browser, to do and store their work online

.

2. **Platform-as-a-service (PaaS):** vendors provide end-users with platforms that allow the end-user to deploy and run new applications without the need to construct or maintain the entire infrastructure .

3. **Infrastructure-as-a-service (IaaS):** also known as hardware-as-a-service, vendors provide the consumer with hardware, data centers, network components, and storage [10].

1.2.3 The cloud-computing paradigm

The cloud computing paradigm is employed to address IoT challenges, primarily centered around storage and networking. Cloud applications generalize machine learning models by training them with data gathered from IoT devices situated in various environments. While the cloud computing paradigm provides a partial solution to IoT challenges, it offers several advantages for large-scale deployments in IoT networks (LSD-IoT). These advantages include:

1. **Storage:** Cloud computing offers cost-effective, secure storage solutions and computational services for IoT data. It grants customers the flexibility to access their data from anywhere and at any time while ensuring they pay only for the storage they actually use.

2. **Scalability:** Cloud computing possesses the capability to accommodate the increasing number of IoT devices by efficiently managing and troubleshooting them. Additionally, it allows for the optimization of resources such as bandwidth and storage to align with the specific requirements of IoT applications.

3. **Performance:** Cloud computing has the potential to enhance performance by handling the processing of IoT data in the cloud, thereby alleviating the workload on IoT devices. Moreover, cloud computing often boasts superior processing capabilities compared to IoT devices, further contributing to improved performance [10].

1.3 Edge AI technology

Edge AI technology encompasses both Edge Computing and Artificial Intelligence (AI) technology. It leverages edge computing to analyze and respond to data, such as security camera feeds, in real time, eliminating the necessity to transmit data to the

cloud for processing. This approach enhances efficiency and reduces latency in data processing .

1.3.1 Definition of EDGE Compiting

Edge computing represents a novel distributed IT architecture in which data storage, services, and computational applications are either partially or fully decentralized, bringing them closer to the end-user. Given the substantial data generation by IoT devices, which could potentially reach a staggering 500 zettabytes, issues related to bandwidth, storage, and data processing have arisen. Edge computing technology addresses these challenges by facilitating real-time applications, reducing latency, and improving response times. It effectively extends the capabilities of cloud computing to the network's edge. edge and cloud computing complement each other, with edge computing ensuring the continuity of services, while the cloud manages the network. This approach enables the distribution of storage, control, and communication in proximity to the end-user, enhancing overall system performance [10] .

1.3.2 Edge Intelligence

Conventional edge-computing devices typically possess limited intelligence capabilities. Their primary function is to handle local data processing tasks, such as feature extraction and data transmission to cloud servers. Nevertheless, when edge devices are enhanced with machine-learning capabilities, they gain increased intelligence, enabling them to analyze data and make decisions autonomously without relying on cloud connectivity. This advancement leads to a reduction in latency and time-to-action.

The growing interest in shifting intelligence from the cloud to edge devices is a compelling area of research for several reasons [10] :

1. **Security:** When IoT data is transmitted to cloud servers for analysis, it can introduce security and privacy concerns, especially when using public or private infrastructure. Processing data near the end-user helps protect user privacy.

2. **Performance:** In time-sensitive IoT applications like vehicular communications, even minor latency delays are noticeable. Therefore, edge processing is highly desirable to ensure optimal performance.

3. **Bandwidth :** The growing number of devices continuously generating data, such as cameras streaming videos and photos, can consume significant Internet band-

width. Processing data at the edge helps alleviate this bandwidth consumption.

4. **Data integrity :** Transmitting data to edge devices doesn't necessitate compression or alterations to the data format. Additionally, data is less exposed to noise during the transmission process, preserving data integrity.

Conclusion

In this chapter we had defined the basic elements and concepts involved in this work .Trying to clarify and offer a background, by discussing the different technologies involved. Next chapter we will provide State of the art for the different AI techniques used in the context of securing and monitoring of retail property .

Chapter 2

State of the art AI solition for detecting abnormail bihaviors

Introduction

In this chapter, we take an idea of these elements, Artificial intelligence, machine learning, and computer vision , Transfer Learning, And her role addressing abnormal behavior in the retail industry. Then we will take an idea of similar works in this field.

2.1 Artificial intelligence (AI)

is a field within computer science that focuses on the development of intelligent machines capable of emulating human actions and responses. As stated by Andrew Ng in 2015, AI seeks to create machines that can function and react like humans. The primary objective of AI is to mimic the cognitive functions of the human brain, which include decision-making, problem-solving, and learning from the environment. Within the realm of AI, there exist numerous subfields, as depicted in Figure 2.1. These subfields encompass various aspects of artificial intelligence, contributing to the development of intelligent systems and technologies [11] .

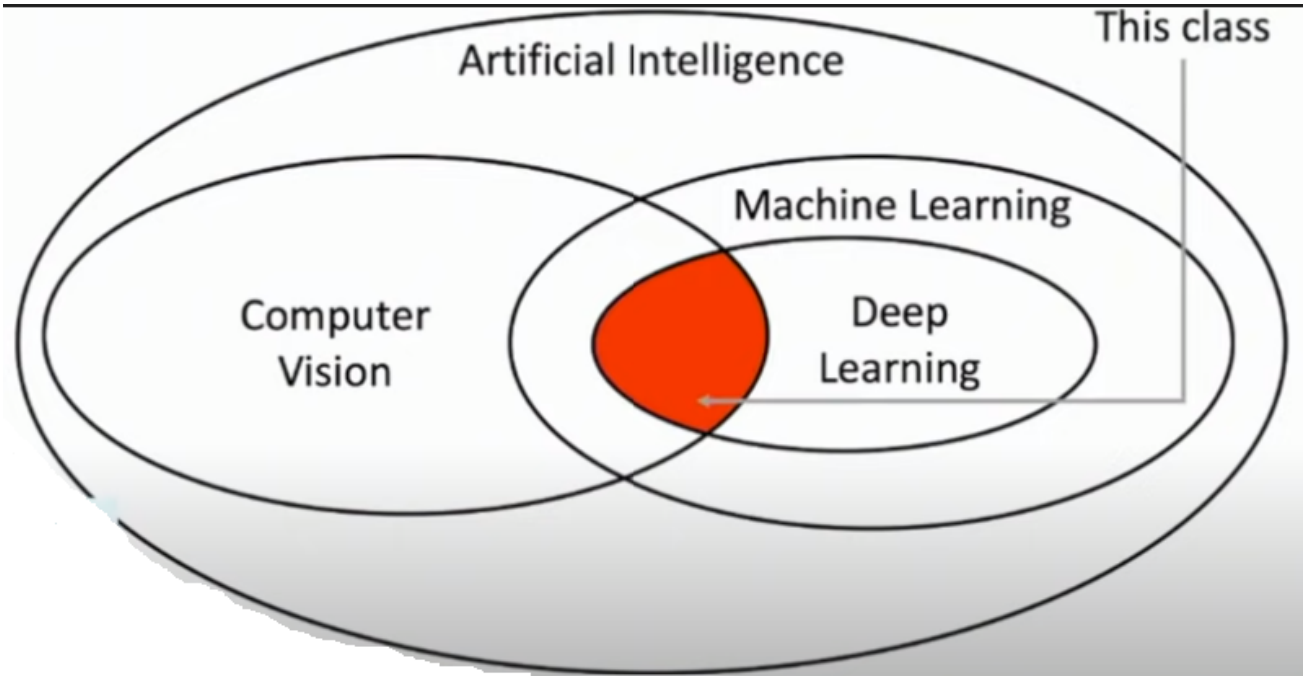

Figure 2.1: Artificial intelligence

2.2 Machine Learning

Machine learning is the field of study that focuses on enabling computers to learn and emulate human-like behavior. It involves the development of algorithms and models that can autonomously improve their learning capabilities over time by ingesting and processing data and information, often in the form of observations and real-world interactions. As expressed by Arthur Samuel in 2016, machine learning aims to make computers learn and adapt in ways similar to humans.

Machine learning represents one of the most significant application areas within the domain of artificial intelligence. Machines are designed to learn from vast amounts of data and undergo training. Once a machine learning model is trained on this data, it can make predictions on data samples it has never encountered before. Machine learning relies on the availability of data and doesn't depend on rule-based programming. These algorithms, by generalizing from examples, can perform essential tasks autonomously [13] .

2.2.1 Basic Process Flow of Machine Learning

Below Figure 2.2 illustrates the fundamental process followed in most machine learning scenarios. This basic process flow forms the foundation of machine learning, enabling the creation of models that can make predictions and decisions based on data [14] . The key steps in this process are as follows:

1. Data Collection: The process begins with the collection of data, which serves as the foundation for training and evaluation.

2. Data Preprocessing: Data collected may be noisy or contain missing values. Data preprocessing involves cleaning, transforming, and organizing the data to make it suitable for model training.

3. Feature Selection/Engineering: Selecting relevant features or creating new ones can enhance model performance. Feature engineering involves choosing the right input variables to improve predictions.

4. Model Training : The selected algorithm is trained on the training data, allowing it to learn patterns and relationships within the data.

5. Model Selection: Various machine learning algorithms are available, and the choice of the appropriate one depends on the specific problem. Common tasks include regression (predicting continuous values) and classification (categorizing data into classes).

6. Model Evaluation : Once trained, the model is evaluated using a separate set of data, often called the test set. The model's performance is assessed based on its ability to make accurate predictions on unseen examples.

7. Hyperparameter Tuning : Fine-tuning model hyperparameters can further improve its performance. This step involves adjusting settings that are not learned from the data but impact the model's behavior.

8. Model Deployment: After satisfactory performance is achieved, the trained model can be deployed in real-world applications to make predictions or automate tasks.

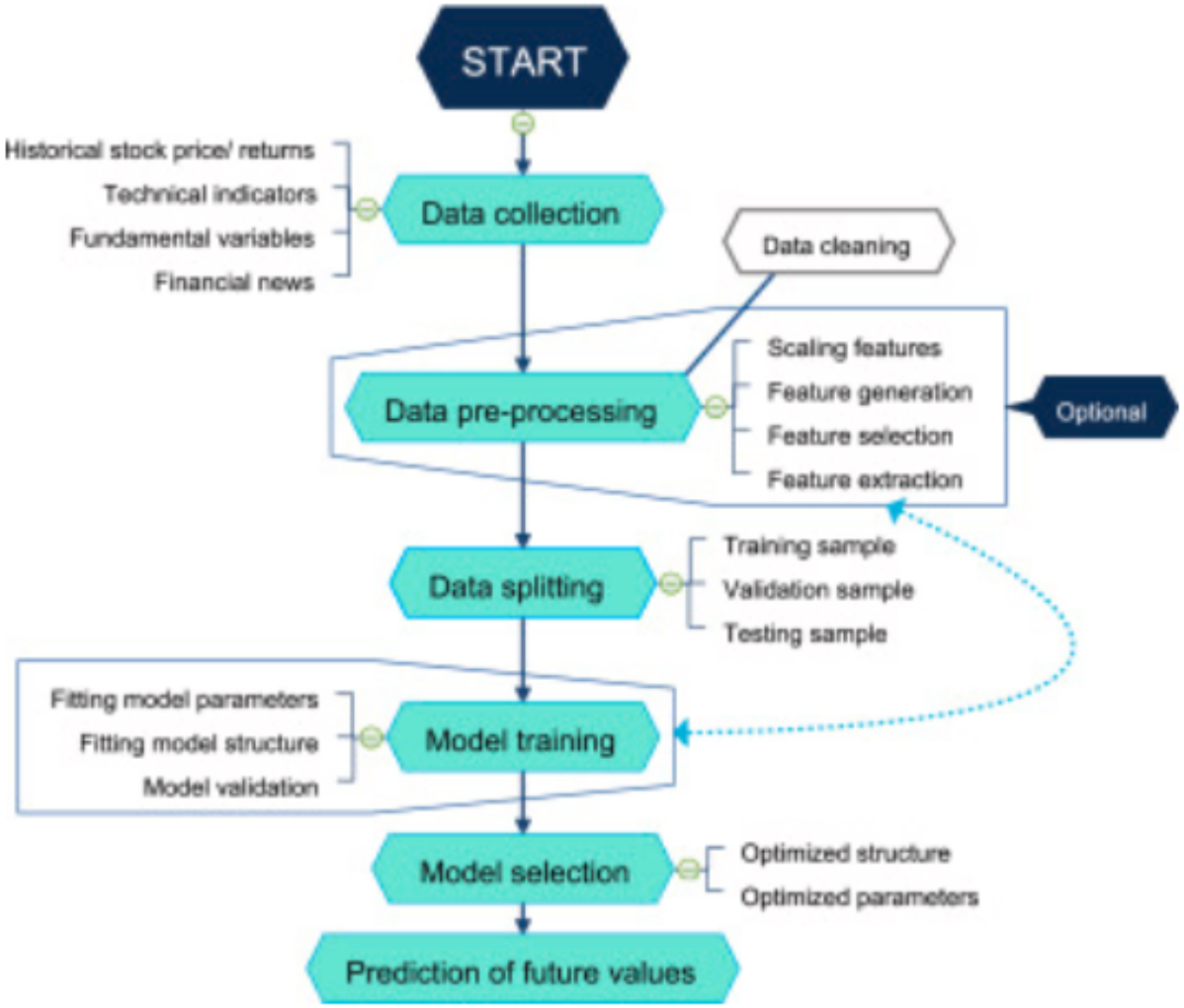

Figure 2.2: Process Flow of Machine Learning

2.2.2 Types of Machine Learning Techniques:

Machine learning encompasses various techniques, each tailored for specific learning scenarios. The primary types of machine learning techniques are as follows:

Supervised Learning :

1. In supervised learning, the algorithm is provided with labeled outputs for each input training example. This enables the model to learn the patterns and relationships between inputs and their corresponding outputs.

2. Examples of supervised learning algorithms include K-Nearest Neighbors, Linear Regression, Support Vector Machines, Logistic Regression, and many more [3].

Unsupervised Learning:

Unsupervised learning does not involve input-to-output mappings in the dataset. Instead, the model analyzes hidden patterns and groupings within the data to identify similarities and differences among data points [3] .

Reinforcement Learning:

Reinforcement learning is based on the interaction between an agent and its environment. The program learns to make a sequence of decisions, aiming to maximize rewards while minimizing penalties. This learning mechanism is often used in scenarios where an agent learns by trial and error[3].
Below figure 2.3 shows the three different types of Machine learning techniques that are widely used .

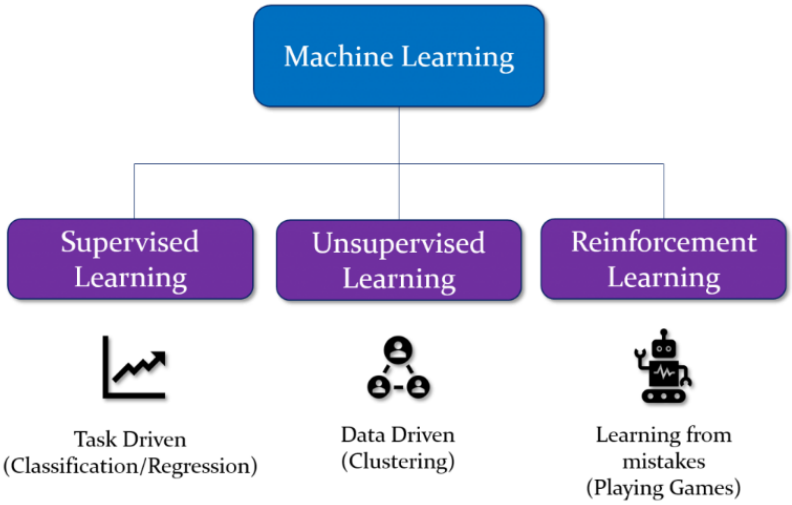

Figure 2.3: types of machine learning techniques

2.3 Deep Learning

Deep learning is a subset of machine learning that draws inspiration from the biological structure and functioning of the human brain, which consists of billions of neurons. Deep learning is often referred to as ”Deep Neural Networks” because it employs neural networks with many layers of interconnected neurons. These deep neural networks are capable of recognizing patterns in raw input data and making decisions in a manner similar to human cognitive processes. Key components of deep learning include [14] :

Input Layer :

1. Neurons in the input layer correspond to each feature in the dataset.
2. The values of each instance in the dataset serve as inputs to this layer.

Hidden Layer :

1. The values from the input layer, when multiplied by weights, become the inputs to the neurons in the hidden layer.
2. These input values then pass through an activation function to produce final outputs from the hidden layer.

Output Layer :

1. The activated outputs from the hidden layers serve as inputs to the neurons in the output layer.
2. Similar to the hidden layer, these inputs are multiplied by weights and passed through an activation function to generate final outputs from the output layer.

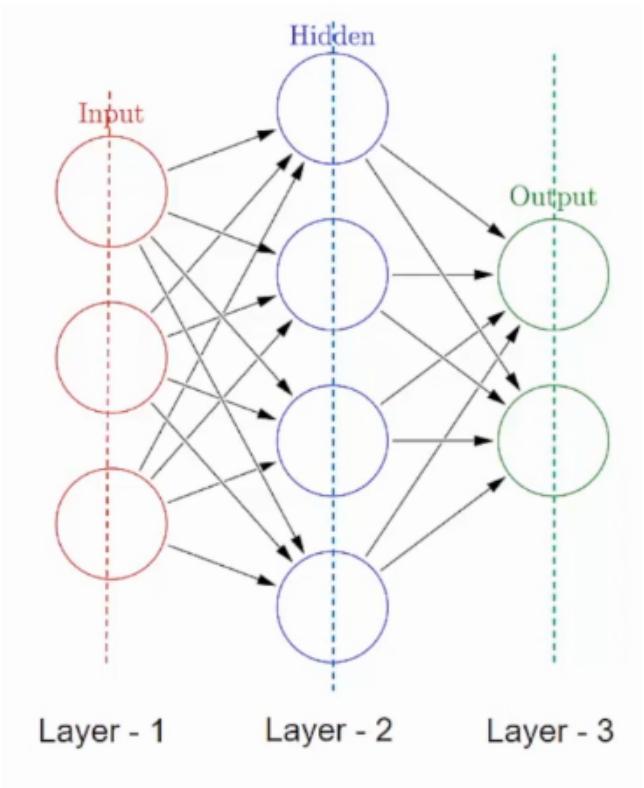

Figure 2.4: Deep neural network layers

Deep neural networks learn patterns from data through a technique called back-propagation.

The combination of these layers, including the input, hidden, and output layers, enables deep learning models to perform complex tasks such as image recognition, natural language processing, and more. Deep learning has gained prominence in recent years due to its ability to handle large datasets and solve intricate problems effectively.

2.3.1 Transfer Learning(TL)

Transfer learning is a machine learning technique where knowledge gained from training a model on one task is leveraged to improve the performance of a model on a different but related task. Instead of starting the learning process from scratch, transfer learning allows models to transfer their understanding, features, or representations acquired from one domain to another. This approach is especially valuable when the target task has limited labeled data, as the pre-trained model can serve as a feature extractor or provide a useful initialization point .Transfer Learning in 6 steps :

1. Select Pre-trained Model: Choose a pre-trained model that was previously trained on a large dataset for a related task.

2. Fine-Tuning: Customize the selected model by adding a new set of output layers, typically adapted to the specific target task. These new layers are randomly initialized.

3. Data Preparation: Gather and preprocess the data for the target task. Ensure that it is properly formatted and labeled.

4. Feature Extraction: Use the pre-trained model as a feature extractor. Pass the data through the pre-trained layers and extract valuable features.

5. Training : Train the model using the extracted features and the labeled data for the target task. Fine-tune the weights of the new layers while keeping the pre-trained layers frozen [18].

6. Evaluation and Fine-Tuning: Assess the model's performance on a validation dataset. Adjust hyperparameters or further fine-tune the model if necessary to achieve opti mal results.

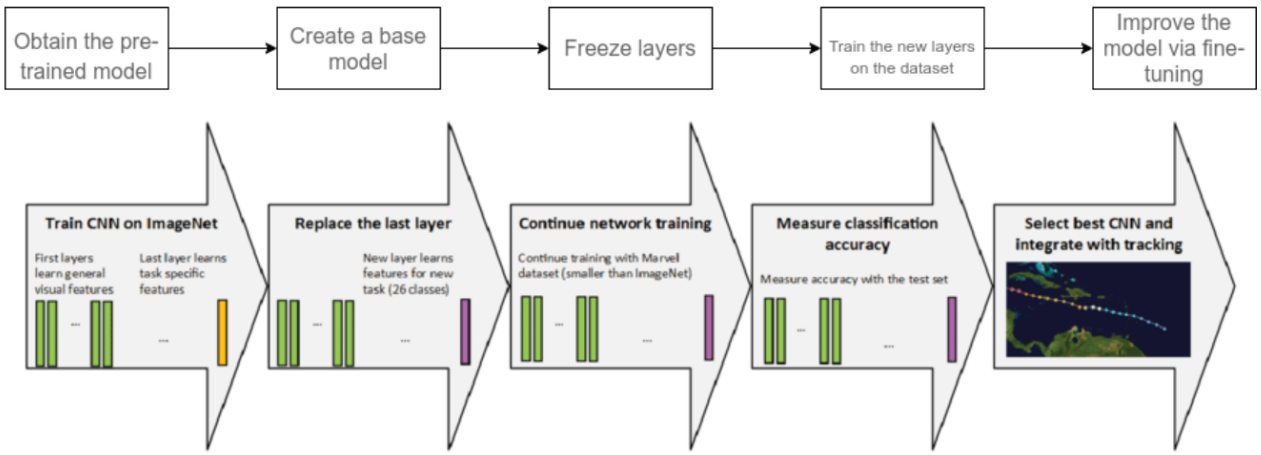

Figure 2.5: Transfer Learning

2.4 Computer vision

Computer vision is a subfield of artificial intelligence (AI) that focuses on teaching computers to interpret and comprehend the visual world. It enables machines to process digital images captured by cameras and videos, utilizing deep learning models to accurately recognize, classify, and respond to objects and patterns in their visual input [19] .

Key aspects of computer vision include:

1. Object Recognition: Identifying and categorizing objects or entities within an image or video stream.

2. Image Classification: Labeling images based on their content, such as distinguishing between different types of animals or objects.

3. Object Tracking: Monitoring and following the movement of objects or subjects across frames in a video.

4. Anomaly Detection: Identifying unusual or unexpected events or behaviors based on visual data.
   Computer vision is instrumental in automating tasks that involve visual data analysis, such as surveillance systems. It enables the automatic and efficient detection of abnormal events or behaviors in both indoor and outdoor settings. This capability reduces the need for manual analysis of extensive surveillance video data, which can be labor-intensive, error-prone, and time-consuming.

2.4.1 convolutional neural network (CNN)

A Convolutional Neural Network (CNN) is a type of deep learning network commonly used in computer vision tasks for recognizing and classifying visual features in images. A typical CNN model consists of several layers, as depicted in Figure 2.6:

1. **the input layer :** The input layer receives raw images as input data. These images are then forwarded to the hidden layers for feature extraction.

2. **The hidden layer :** The hidden layer comprises three essential types of layers: convolutional layers, pooling layers, and fully connected layers:

   - Convolution Layer : This layer is responsible for automatic feature extraction from raw images. It uses a fixed-size filter that slides over the input image to extract relevant features. Convolutional layers capture patterns and features like edges, textures, and shapes.

   - Pooling Layer: The pooling layer is employed to reduce the dimensionality of feature maps while preserving important features. Common pooling techniques include max-pooling and average-pooling.

   - Fully Connected Layers : These layers are crucial for making predictions. They flatten the feature maps from previous layers and connect all neurons to each other.

3. **output layer :** The output layer is the final layer in a CNN model. It takes the flattened feature maps and performs tasks like classification or regression [5] .

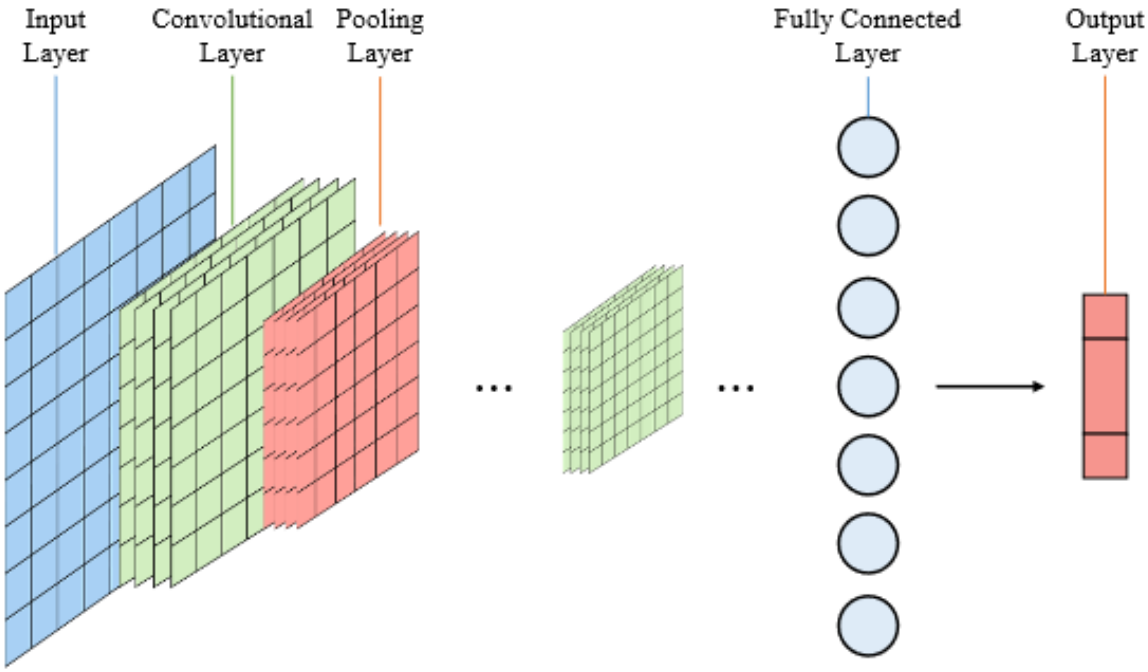

Figure 2.6: Structure of the CNN model

2.5 The most important techniques for Detecting Abnormal Behavior :

N	Method	Year	AUC	dataset
1	GODS	2019	70.46	Dash-Cam-Pose, UCF Crim
2	MILR	2020	76.67	UCF-Crime
3	MIST	2021	82.30	UCF-Crime ShanghaiTech
4	WSAL	2021	85.38	TAD dataset, UCF-Crime
5	RTFM	2021	84.30	UCF-Crime

Table 2.1: Some related work

2.5.1 ” GODS ” by J Wang and all

One-class learning is the classic problem of fitting a model to data for which annotations are available only for a single class. In this paper, they propose a novel objective for one class learning ” GODS ” (Generalized One-class Discriminative Subspaces) for Anomaly Detection. their key idea is to use a pair of orthonormal frames – as subspaces – to “sandwich” the labeled data via optimizing for two objectives jointly:

1. minimize the distance between the origins of the two subspaces.

2. to maximize the margin between the hyperplanes and the data , either subspace demanding the data to be in its positive and negative orthant respectively.

To study the effectiveness of their scheme, they proposed a new dataset Dash-Cam-Pose [24].

2.5.2 ” MILR ” by Shikha Dubey and all

In this study , they proposed method 3D deep Multiple Instance Learning with ResNet (MILR) , a deep neural network to detect abnormal activities in video clips. Where, they used unnatural and normal videos to train their network. This proposed model is trained with the help of deep MIL coupled with a new proposed order loss function. They validated their proposed algorithm on the UCF-Crime dataset with the help of the ROC curve and AUC evaluation methods. All experimental results show that the proposed algorithm gives a significant improvement in reducing false alarm rates and gives better accuracy in the abnormal activity detection task [6].

2.5.3 ” MIST ” by Feng, J and all

Weakly supervised video anomaly detection (WS-VAD) is to distinguish between anomalies and normal events based on discriminatory representations. Most of the extant works are limited in inadequate video representations. In this work, they develop a multi-instance self-training (MIST) framework to efficiently improve task-specific discriminative representations using only video-level annotations. In particular, MIST consists of:

1. A multi-instance pseudo-label generator, which adapts a continuous sparse sampling strategy to produce more reliable pseudo-labels at the snippet level.
2. An encoder with an enhanced self-attention feature that aims to automatically focus on anomalous regions in frames while extracting task-specific representations.

Moreover, they adopted a self-training system to improve both components and finally get a feature-specific encoder [7] .

2.5.4 ” WSAL” by Lv, H. and all

Focused on the location of anomalies in surveillance footage and suggested a weakly trained network for deep temporal context exploration in consecutive segments .

Hui Lv and all proposed a weakly supervised anomaly localization (WSAL) method that focuses on the temporal localization of anomalous parts within anomalous videos.

Inspired by the difference in appearance in the anomalous videos, the evolution of contiguous temporal segments is assessed for the localization of the anomalous segments. To this end, a higher-order contextual encoding paradigm has been proposed to not only extract semantic representations but also to measure dynamic differences so that the temporal context can be used effectively. Additionally, in order to take full advantage of the spatial context information, . An optimization strategy is also proposed to deal with noise interference and the absence of localization guidance in anomaly detection. Furthermore, to facilitate diversity requirements for anomaly detection criteria [17].

2.5.5 ” RTFM ” by Tian and all

In this proposal a new method, called RTFM (Robust Temporal Feature Magnitude), enables high-quality MIL methods to detect weakly supervised video anomalies. RTFM learns a temporal feature size mapping function that:

1. You discover rare abnormal excerpts from abnormal video clips that contain many normal excerpts,
2. It guarantees a large margin between normal and abnormal snippets.

This improves the classification of the following MIL-based abnormalities in two major aspects:

1. This RTFM-enabled model learns more discriminatory features that improve its ability in distinguishing complex anomalies (eg, subtle deviations) from challenging negative examples.
2. It also enables the MIL classifier to achieve significantly improved exploitation of abnormal data.

These two capabilities respectively lead to accurate anomaly discrimination and better sample efficiency than current SOTA MIL methods. They are also the main drivers for the model to achieve SOTA performance on all three big benchmarks [21] .

2.6 Conclution

As we conclude this chapter, we have established a foundational understanding of the key components and methodologies crucial for addressing abnormal behavior in the retail industry. The insights gained here will serve as the basis for the detailed exploration of our proposed AI solution in the following chapter.

Chapter 3

proposed solution

Introduction

In this chapter we will explain the proposed solution ” Abnormal Behavior Detection (ABD) ” , Which is a technique used to identify and classify abnormal behavior in a given dataset into different categories. This solution involves using machine learning algorithms to analyze large amounts of data and detect patterns that indicate abnormal behavior.

3.1 Our general idea

The proposed Abnormal Behavior Detection model is lightweight and implemented on high-end hardware to provide safe and secure monitoring for users. Our idea is to collect a dataset from the UCF crime website, which includes various videos depicting different criminal activities, and we will process this dataset to classify several abnormal behaviors to improve the accuracy and efficiency of detecting anomalies in different systems.

These two diagrams: the general perception of the ABD model (Figure 3.1) General structure for Abnormal Béhavior détection system and (Figure 3.2) representing the detailed visualization of our proposed idea.

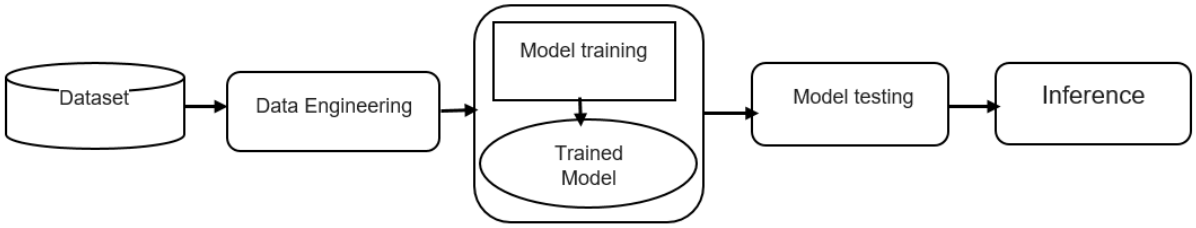

Figure 3.1: the general perception of the ABD model

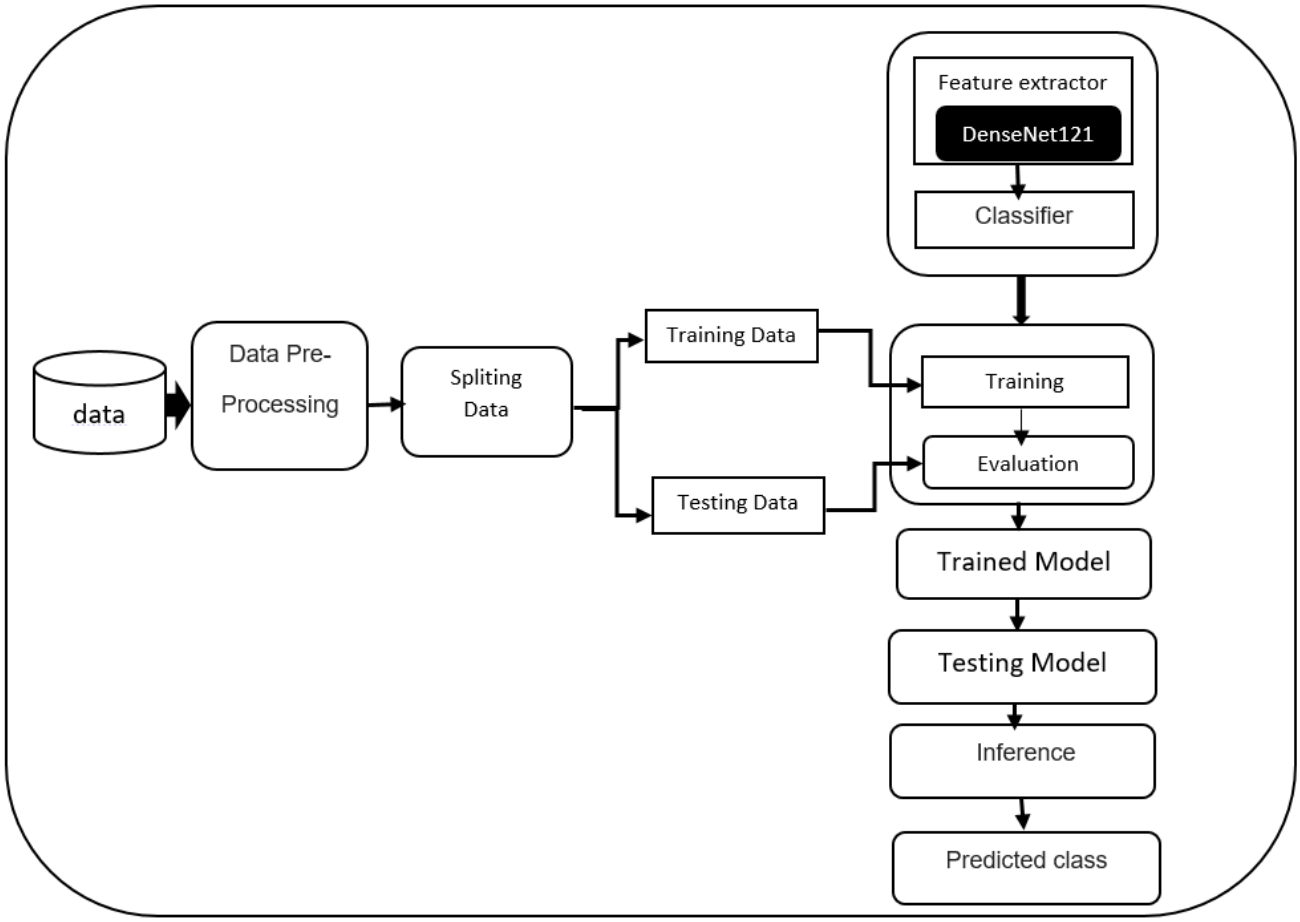

Figure 3.2: General structure for Abnormal Béhavior détection system

3.2 overview for implementing the proposed architecture

3.2.1 Dataset

In our project, we set out on the task of compiling a dataset obtained from the UCF website to assess crimes committed, and carefully curating this data that shows a

variety of behaviors. We systematically divided each of this data (train data + test data) into 5 classes: 'Arson','NormalVideos','Shoplifting','Stealing','Vandalism', by leveraging From this data set, we aim to contribute to the development of effective crime prevention solutions particularly in retail centres. The size of this data is as follows:

1. There are a total of 33253 images belonging to 5 classes in the training dataset.
2. There are a total of 7854 images belonging to 5 classes in the test data set.

3.2.2 Data Preprocessing and Augmentation

Apply data preprocessing techniques, including rescaling and DenseNet-specific preprocessing. We augmented the training dataset with techniques like horizontal flipping and shifting .

3.2.3 Data Splitting

: To accurately evaluate our model's performance, we partition the prepared dataset into two subsets: the training dataset 80% images and the testing dataset 20% images. The training dataset is utilized to train the model, enabling it to learn and understand the underlying patterns and characteristics of abnormal behavior. On the other hand, the testing dataset is reserved for assessing the model's performance on unseen data, simulating real-world scenarios.

3.2.4 Model Architecture Definition and Compilation

By training our model using the DenseNet-121 architecture, extracting relevant features, and employing a CNN classifier, we enable the model to learn the complex relationships between input data and abnormal behavior. The trained model, encapsulated in the ModelS.h5 file, becomes a powerful tool for accurately detecting abnormal behavior in new and unseen instances.We explained these steps as follows :

1. **feature extraction (DenseNet-121):**
   During the training process, our model learns to extract meaningful features from the input data. These features capture important patterns and characteristics that distinguish abnormal behavior. The DenseNet121 architecture, with its deep layers and complex structure, allows the model to automatically learn and represent these features through a process called feature extraction.
   DenseNet-121 : Dense Convolutional Network (DenseNet) is a design that centers around making the deep learning networks go much more profound, and yet

making them more effective to prepare, by utilizing more limited associations between the layers.
The number 121 in densenet-121 is computed as follows:

DenseNet-121: - 5 + [2 x (6 + 12 + 24 + 16)]

5– Convolution and Pooling Layer

3– Transition Layers (6, 12, 24)

1– Classification Layer (16)

2– Dense Block (1 × 1 and 3 × 3conv)[20] .

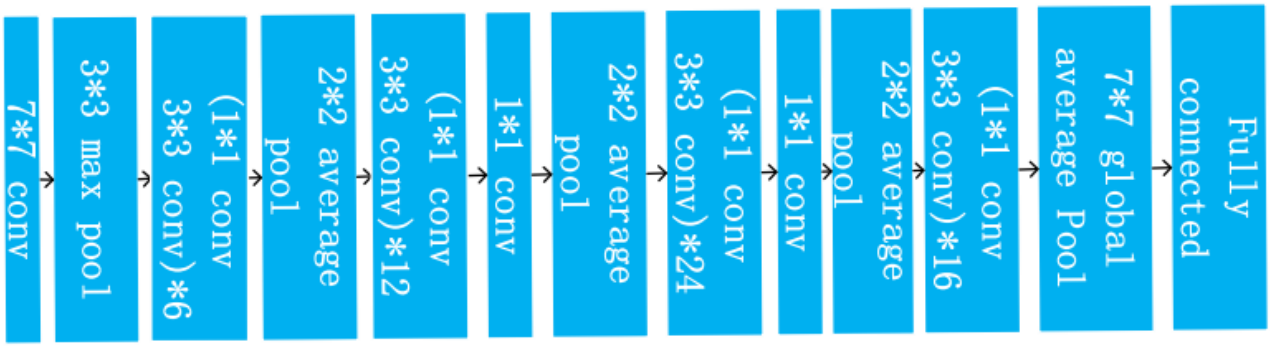

Figure 3.3: The Structure of DenseNet-121

As shown in the Figure 3.5 , DenseNet-121 first goes through convolution and max pooling, then through a series of DenseBlock and Transition Layer, and finally through global average pooling and full connection Layer to output the classification results. The growth rate for DenseNet-121 is k=32. Each “conv” layer shown in the figure corresponds the sequence BN-ReLU-Conv [15].

2. **Classifier (CNN):**

   Alongside the feature extraction layers, we incorporate a classifier, which is another CNN layer, into our model. This classifier takes the learned features as input and performs classification based on them. It maps the extracted features to specific abnormal behavior classes, enabling the model to make accurate predictions. The classifier is trained to identify and distinguish different types of abnormal behavior based on the learned features.

Layer (type)	Output Shape	Param
input-1 (InputLayer)	[(None, 32, 32, 3)]	0
densenet121 (Functional)	(None, 1, 1, 1024)	7037504
global-average-pooling2d	(None, 1024)	0
dense (Dense)	(None, 256)	262400
dropout (Dropout)	(None, 256)	0
dense-1 (Dense)	(None, 1024)	263168
dropout-1 (Dropout)	(None, 1024)	0
dense-2 (Dense)	(None, 512)	524800
dropout-2 (Dropout)	(None, 512)	0
classification (Dense)	(None, 5)	2565
Total params: 8,090,437		
Trainable params: 8,006,789		
Non-trainable params: 83,648		

Table 3.1: Model Architecture

3.2.5 the training process

this step, the deep learning model can be trained using the training data generator (train-generator) , and evaluate its performance on the validation dataset (test-generator) over a specified number of epochs.

3.2.6 Trained Model

Throughout the model training process, the model parameters, configurations, and learned representations are continuously updated and optimized. The resulting trained model, with its learned weights and biases, is saved and stored in a file format called ModelS.h5. This file serves as a snapshot of the trained model, allowing us to later use it for inference and prediction on new input data.

3.2.7 Testing:

After the model is trained, we proceed to the testing phase. Here, we evaluate the performance of the trained model using the Cleaned Data Testing subset. The model is applied to this data, and its predictions are compared against the known labels to measure metrics such as precision, recall. Testing allows us to assess how well the model generalizes to new, unseen instances of abnormal behavior related to crimes.

3.2.8 Inference (New Input Processing and Predict):

Finally, in the inference stage, we process new input data by utilizing the trained model. This processing involves inputting the new data into the model, and based on its learned patterns and features, the model generates predictions or classifications related to abnormal behavior in crimes. The output of the inference stage is the predicted class or label assigned to the new input, indicating whether it is considered abnormal behavior in the context of crimes.

3.3 Concept Validation (Dynamic Input Testing):

1. Initial Program Design: We started by creating a Python program for image classification a cornerstone for our anomaly detection system.
2. Deep Learning Model Integration: Our program gained power from a pre-trained deep learning model, which accurately classified images.
3. User-Friendly Interface: Using tkinter, we designed a graphical interface for smooth user interaction.
4. Efficient Image Handling: The Python Imaging Library (PIL) streamlined image processing across different formats.
5. Model Integration with TensorFlow: TensorFlow smoothly loaded our pre-trained 'modelS.h5', crucial for precise predictions.
6. Video Frame Extraction: OpenCV (cv2) enabled frame extraction from videos, a pivotal initial step.
7. Frame Reading and Prediction: We processed video frames, using 'modelS.h5' for predicting class labels.
8. Visualizing Frames and Labels: tkinter showcased frames and predicted labels, enhancing user experience.

9. User-Triggered Processing: A button, labeled ”Process Frames,” initiated frame processing, involving image extraction, inference, and label display.
10. Comprehensive Anomaly Detection: Through seamless coordination, our workflow executed the anomaly detection program. Users interactively classified video frames and identified anomalies using predictive insights.

    This integrated process enables us to effectively classify video frames and identify anomalies using a comprehensive anomaly detection system

The Figure 3.5 shows these steps :

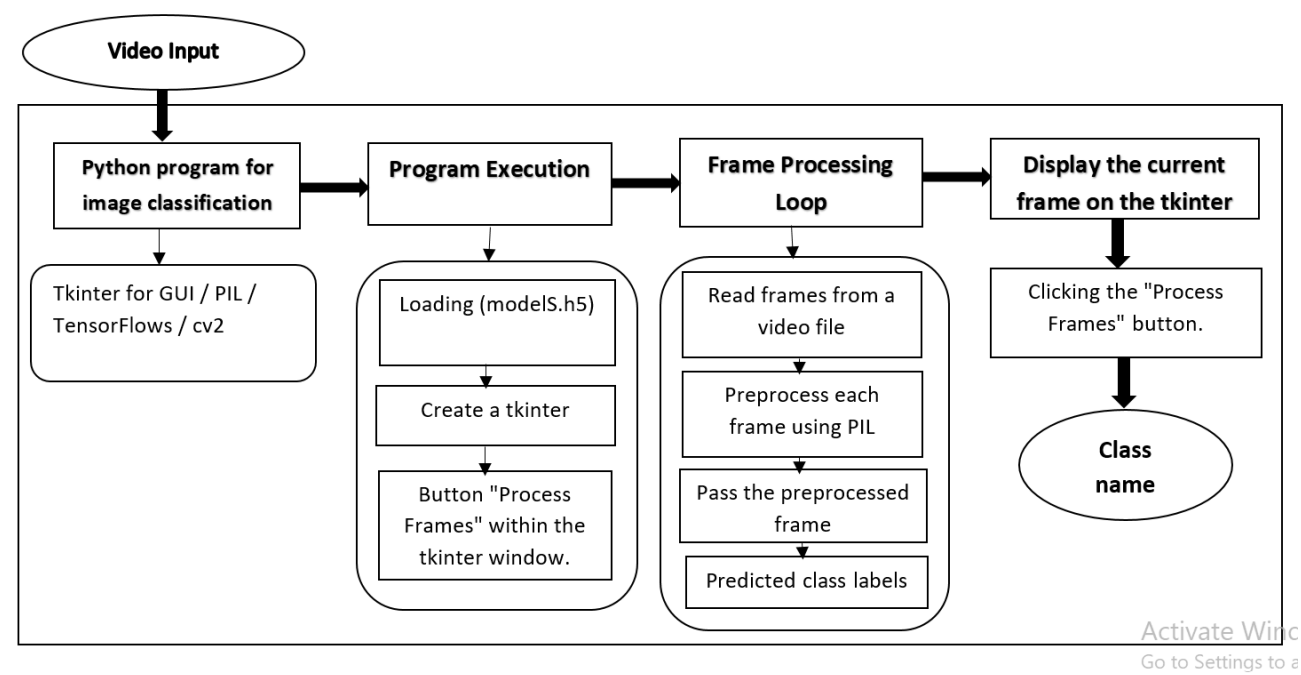

Figure 3.4: Workflow of system Abnormal Behavior Detection ABD

3.4 Conclution

In conclusion, this chapter introduced the methodology and deep learning algorithms utilized in the development of our intelligent system for abnormal behavior multi-classification. We discussed the steps involved in training and deploying the model , By proceeding to the next chapter, we will gain a comprehensive understanding of the implementation details, results, and insights derived from our approach to abnormal behavior multi-classification.

Chapter 4

Implementation And Results

Introduction

In this chapter, we introduce the objectives and needs of our project, which focuses on detecting abnormal behavior. We provide a step-by-step explanation of the processes involved in implementing the system. We also describe the software tools used and present the results collected so far, discussing their implications.

4.1 Development environment

In this section, we will employ the following software and hardware resources:

4.1.1 Environment for implementing the software model :

1. **Google Colab :**
   Colaboratory, often referred to as ”Colab,” is a versatile tool for data analysis and machine learning. It permits the integration of executable Python code with rich text, including charts, images, HTML, LaTeX, and other elements, all within a unified document saved in Google Drive. Colab seamlessly interfaces with robust Google Cloud Platform runtimes, providing access to substantial computing resources. This functionality makes it convenient for users to share their projects and engage in collaborative work with others [12] .

2. **Kaagle :**
   Kaggle, a subsidiary of Google LLC, made its debut in 2010 as an online data science platform. Its primary role is to host data science competitions that address a range of business challenges, serve as a recruitment tool, and support academic research endeavors. Kaggle boasts a thriving community of data scientists with diverse backgrounds who actively engage in these competitions and con-

tribute to discussion forums by sharing their expertise and exchanging insights. Notably, Kaggle competitions distinguish themselves by offering real-time feedback to participants, setting them apart from most academically hosted forecasting competitions [2] .

3. **TensorFlow :**
   TensorFlow, abbreviated as TF, serves as a comprehensive machine learning platform developed and backed by Google. Within TF, you can access a wide array of tools and libraries tailored for data processing and loading, model construction (whether customized or leveraging pre-existing ones), and the execution and deployment of models in diverse environments, including production systems. Fundamentally, a TF program is structured into two primary components: the construction phase, where machine (or deep) learning network models are designed, and the execution phase, which facilitates the training and evaluation of these network models [4] .

4. **Keras :**
   Keras stands as a deep learning framework, constructed atop TensorFlow version 2. Its primary purpose is to furnish developers with a Python API that simplifies the process of creating and experimenting with network models [4] .

5. **OpenCV :**
   OpenCV, short for Open Source Computer Vision, represents a programming language library encompassing various functions primarily designed for computer vision tasks. To put it plainly, it serves as a library dedicated to image processing, enabling a wide range of operations related to images. OpenCV is compatible with multiple platforms, encompassing Windows, OS X, Linux, iOS, and Android [22] .

6. **Python :**
   Python, conceived by the Dutch programmer ” Guido van Rossum ” in the late 1980s, was crafted as an interpreted, object-oriented, high-level, and versatile programming language. Its design principles prioritize attributes like execution efficiency and code readability. Over the past few years, Python has surged in popularity, eventually surpassing C and claiming the top spot as the most widely used programming language according to the latest TIOBE index (February 2022). Numerous Python editors are at your disposal, including Spyder, VSCode, Jupyter Notebook, Sublime Text, PyCharm, Atom, and more [16] .

4.1.2 Hardware Environment

In our project, our initial plan was to use the EDGE device as the primary hardware environment. However, since it was not available, we had to adapt our approach and make use of an 8th generation DALE CORE IS I5 PC instead. With powerful processing capabilities and robust specifications, we were able to run simulations and conduct our experiments effectively. In addition, integrated development environment (IDE). PyCharm provided us with an easy-to-use and efficient platform to implement, test, and improve our model. Despite the change in hardware , we were able to adapt and successfully achieve our project goals.

4.2 Performance of the proposed approach :

4.2.1 training a model

We utilized Google Collaboratory, colloquially known as colab, to train the model. Colab offers a notebook environment conducive to Python notebook creation and grants free access to computing resources, including GPUs. The training process resulted in loss and accuracy metrics, along with performance statistics for the model. The training duration amounted to 3 hours, The steps we took to train the model are as follows :

1. Download data from Google Drive

```
train_dir = "/content/drive/MyDrive/traine"
test_dir = "/content/drive/MyDrive/teste"

SEED = 12
IMG_HEIGHT = 32
IMG_WIDTH = 32
BATCH_SIZE = 16
EPOCHS =5
LR =  0.0003
NUM_CLASSES = 5
CLASS_LABELS = ['Arson','NormalVideos','Shoplifting','Stealing','Vandalism']
```

Figure 4.1: A path of data and name class labells

2. Train data Distribution

Figure 4.2: Train data

3. Test Data Distribution

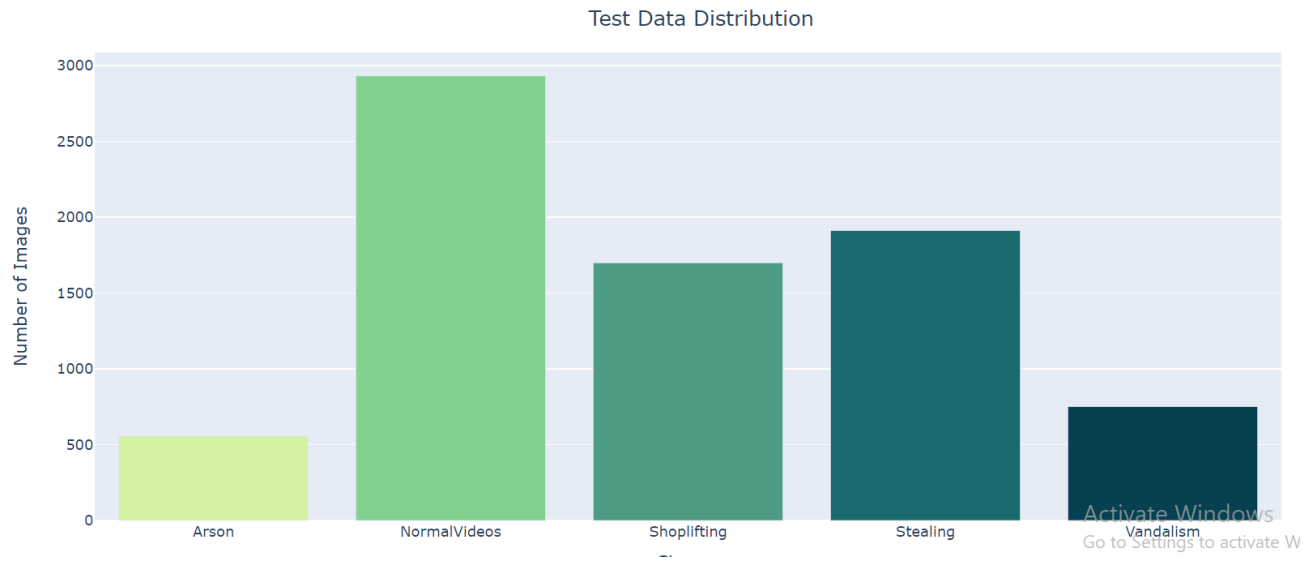

Figure 4.3: test Data

4. Train the model

```
Epoch 1/5
2079/2079 [==============================] - 8322s 4s/step - loss: 1.6866 - auc_1: 0.5814 - val_loss: 1.4184 - val_auc_1: 0.7149
Epoch 2/5
2079/2079 [==============================] - 895s 430ms/step - loss: 1.3756 - auc_1: 0.7301 - val_loss: 1.2247 - val_auc_1: 0.7867
Epoch 3/5
2079/2079 [==============================] - 889s 428ms/step - loss: 1.0615 - auc_1: 0.8551 - val_loss: 1.3063 - val_auc_1: 0.8131
Epoch 4/5
2079/2079 [==============================] - 890s 428ms/step - loss: 0.7678 - auc_1: 0.9262 - val_loss: 1.5777 - val_auc_1: 0.8264
Epoch 5/5
2079/2079 [==============================] - 891s 428ms/step - loss: 0.5566 - auc_1: 0.9605 - val_loss: 1.8204 - val_auc_1: 0.8233
```

Figure 4.4: Training model

these results depict the model's performance at different training stages, with

lower training and validation losses and higher AUC values indicating better model performance. The model seems to improve from epoch to epoch.

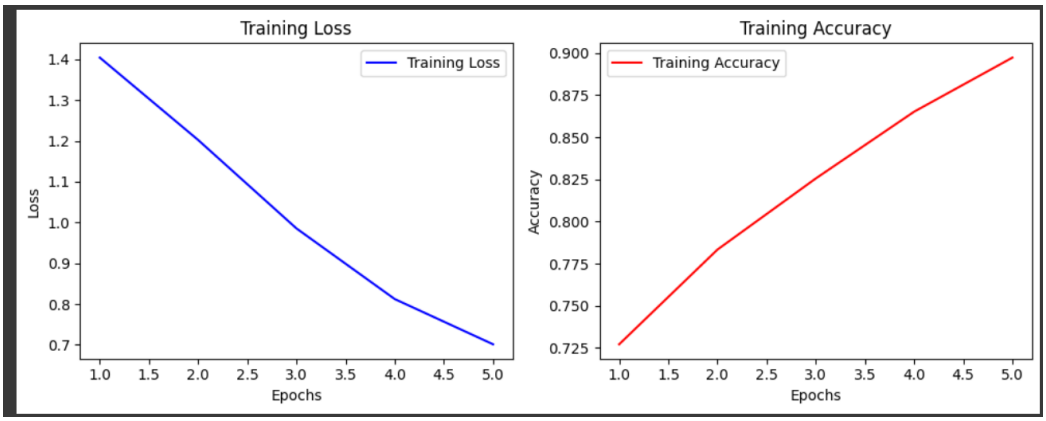

Figure 4.5: Graph training loss and Graph accuracy

These two Graphes for understanding how model learns and performs over time, helping as make informed decisions during the model development process.

Model accuracy

Model accuracy is a machine learning classification model performance metric that is defined as the ratio of true positives and true negatives to all positive and negative observations. In other words, accuracy tells us how often we can expect our machine learning model will correctly predict an outcome out of the total number of times it made predictions. Mathematically, it represents the ratio of the sum of true positive and true negatives out of all the predictions.

Accuracy Score = (TP + TN)/ (TP + FN + TN + FP)

The precision score

is a useful measure of the success of prediction when the classes are very imbalanced. Mathematically, it represents the ratio of true positive to the sum of true positive and false positive.

Precision Score = TP / (FP + TP)

Recall Score

Recall is often used in conjunction with other performance metrics, such as precision and accuracy, to get a complete picture of the model's performance. Mathematically, it represents the ratio of true positive to the sum of true positive and false negative.

Recall Score = TP / (FN + TP)

Model F1 score

Model F1 score represents the model score as a function of precision and recall score. F-score is a machine learning model performance metric that gives equal weight to both the Precision and Recall for measuring its performance in terms of accuracy, making it an alternative to Accuracy metrics (it doesn't require us to know the total number of observations). It's often used as a single value that provides high-level information about the model's output quality . Mathematically, it can be represented as a harmonic mean of precision and recall score.[9]

F1 Score = 2* Precision Score * Recall Score/ (Precision Score + Recall Score/)
This table 4.1 represents these values applied to a model :

	precision	recall	f1-score	support
Arson	1.00	0.50	0.67	1
NormalVideos	1.00	1.00	1.00	1
Shoplifting	0.50	1.00	0.67	1
Stealing	1.00	1.00	1.00	1
Vandalism	1.00	1.00	1.00	1
accuracy			0.83	5
macro avg	0.90	0.90	0.87	5
weighted avg	0.92	0.83	0.83	5

Table 4.1: classification report

Explanation of the table results

Precision: For 'Arson', the precision is 1.00, which means that when the model predicts 'Arson', it's correct 100% of the time, but the recall is 0.50, indicating that it's missing some 'Arson' cases. For 'NormalVideos', 'Shoplifting', 'Stealing', and 'Vandalism', the precision is 1.00, indicating that all predictions for these classes are correct.

Recall : For 'Arson', the recall is 0.50, suggesting that only 50% of the actual 'Arson' cases are correctly predicted. For 'NormalVideos', 'Shoplifting', 'Stealing', and 'Vandalism', the recall is 1.00, meaning that all actual cases for these classes are correctly predicted.

F1-Score : The F1-score is the harmonic mean of precision and recall. It balances the trade-off between precision and recall. The F1-scores for 'Arson', 'NormalVideos', 'Shoplifting', 'Stealing', and 'Vandalism' are 0.67, 1.00, 0.67, 1.00, and 1.00, respectively.

Accuracy : The overall accuracy of the model is 83%, meaning that it correctly predicts the classes for 83 of the samples in your dataset.

Macro Avg : The macro-averaged precision, recall, and F1-score are 0.90, 0.90, and 0.87, respectively. This is the average of these metrics across all classes, treating each class equally.

Weighted Avg : The weighted-averaged precision, recall, and F1-score are 0.92, 0.83, and 0.83, respectively. This is the average of these metrics across all classes, with weights based on the support for each class.

5. Multiclass AUC Curve

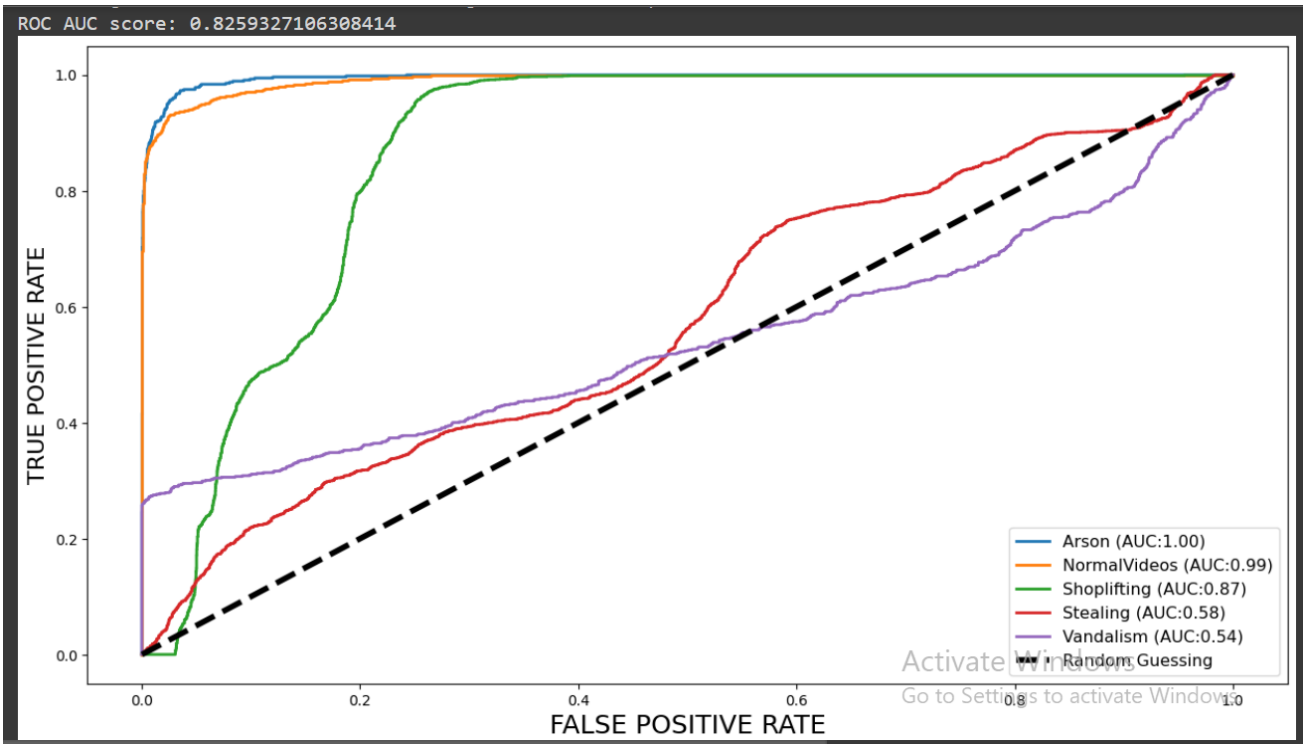

Figure 4.6: AUC Curve

6. Save the trained Model

```
[ ]  model.save('model5.h5')

[ ]  from tensorflow import keras
     model = keras.models.load_model('model5.h5')
```

Figure 4.7: Save the trained Model

4.2.2 Utilize Our Model for Predictive

1. Create the tkinter window

```
window = tk.Tk()
window.title("Abnormal Behavior Detection")
window.geometry("800x800")
```

Figure 4.8: the tkinter window

2. Enter modelS.h5 to the processing function and predict the class of each frame .

```
def process_frames(video_path):
    model = load_model(MODEL_PATH)
    model.summary()

    cam = cv2.VideoCapture(video_path)
    current_frame = 0
```

Figure 4.9: predict the class of each frame

3. Create a button to start processing the frames

```
def start_processing():
    process_frames(VIDEO_PATH)
process_button = tk.Button(window, text="Start Video Processing", command=start_processing)
process_button.pack()
```

Figure 4.10: start processing the frames

4. Create a label to display the predicted class

```
class_label = tk.Label(window, font=("Helvetica", 16))
class_label.pack()
```

Figure 4.11: label to display the predicted class

4.2.3 Results of Programe Abnormal Behavior Detection

1. When the video contains normal images , the prediction label is normal video

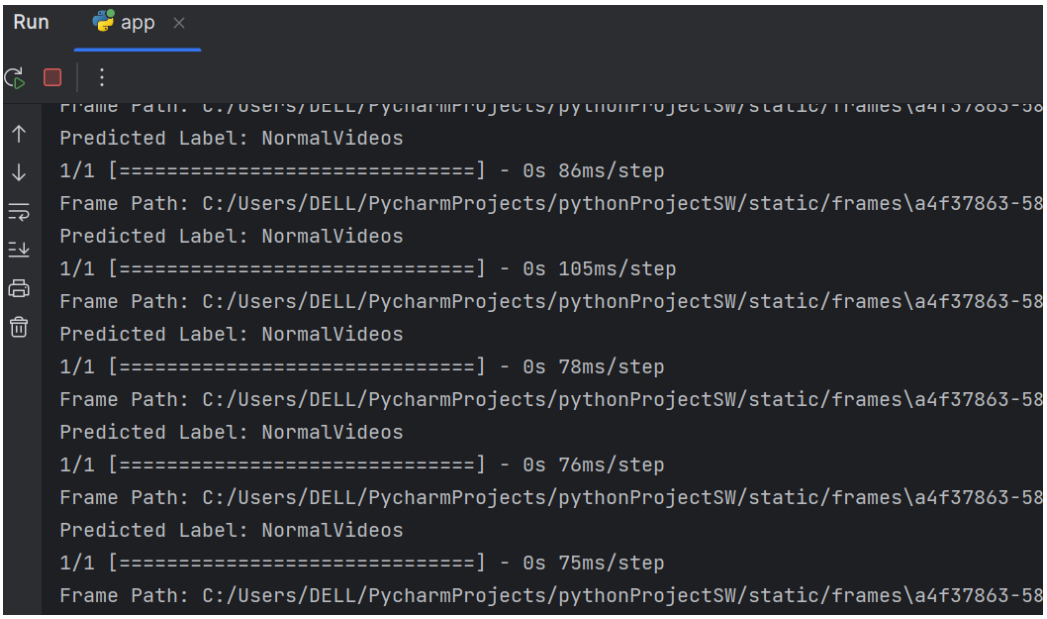

Figure 4.12: Predicted label : NormalVideo

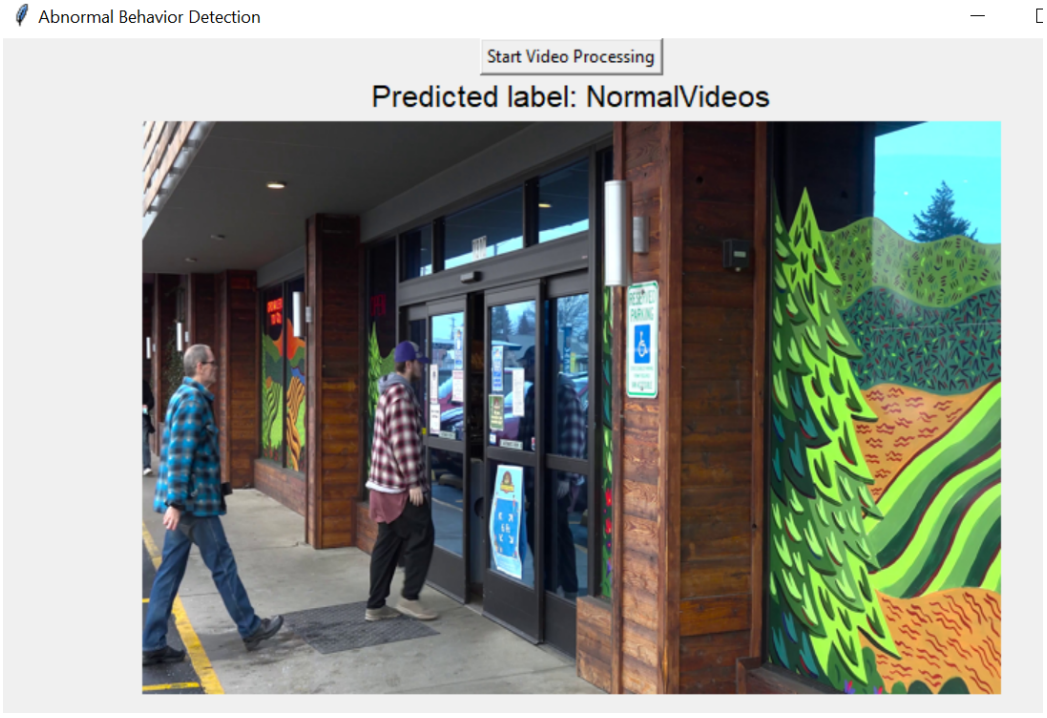

Figure 4.13: Normal picture outside a shop

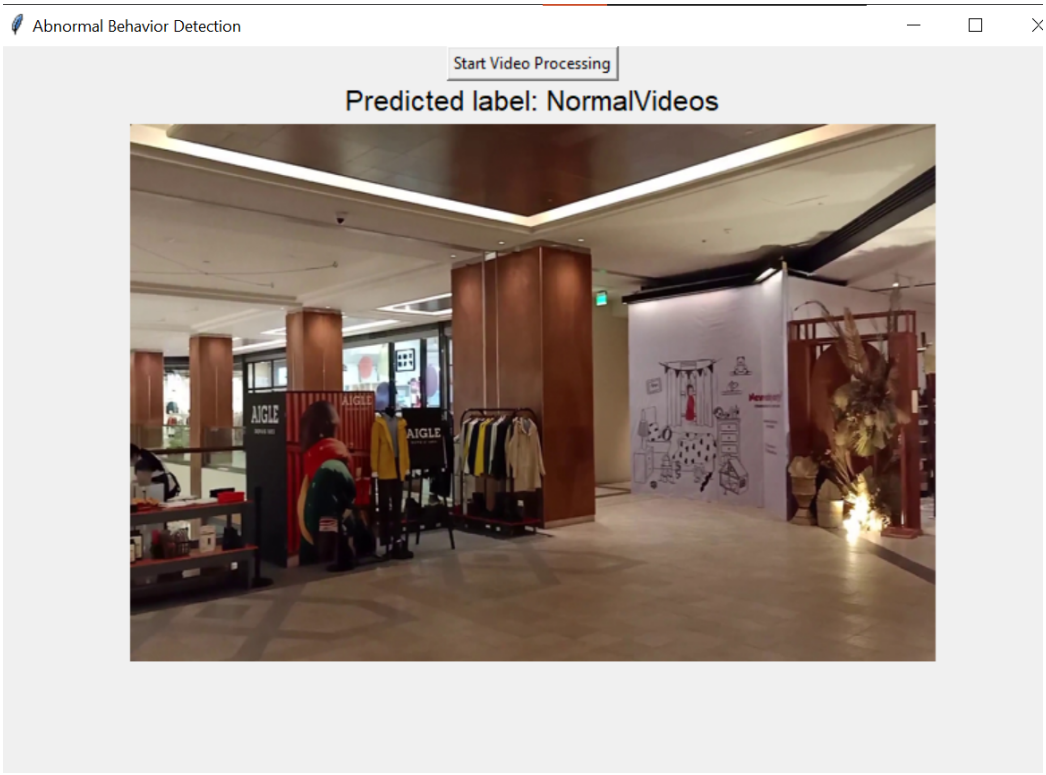

Figure 4.14: Normal picture inside the store

2. When the video contains abnormal behavior the predicted label one of the 4 class : ”’Arson’, ’Shoplifting’, ’Stealing’, ’Vandalism’.

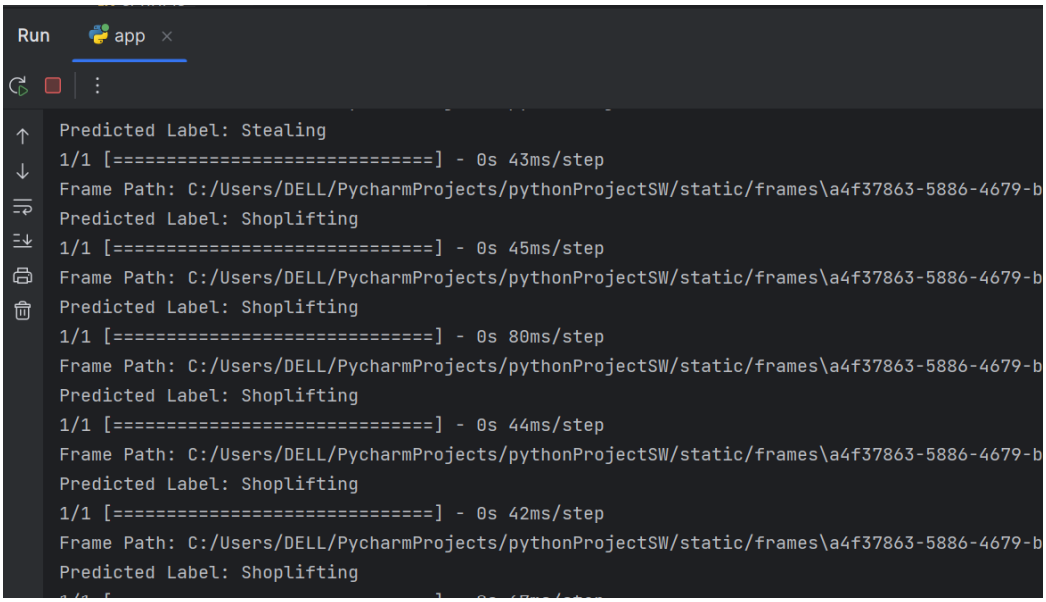

Figure 4.15: Predicted Label : ABNORML

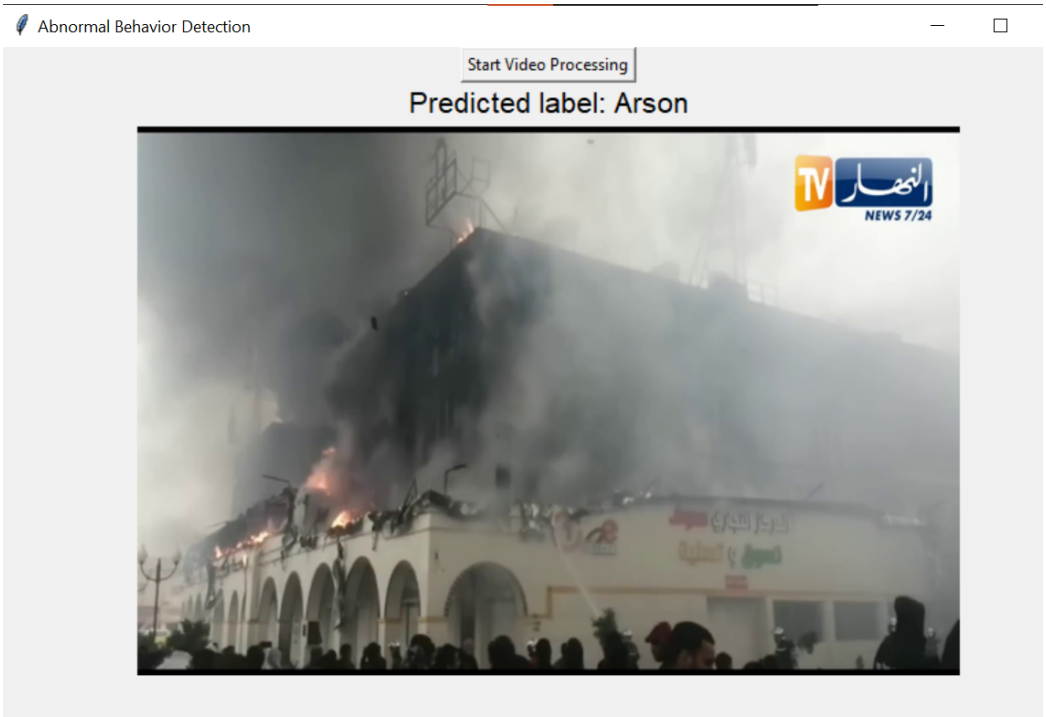

Figure 4.16: Predicted Label : Arson

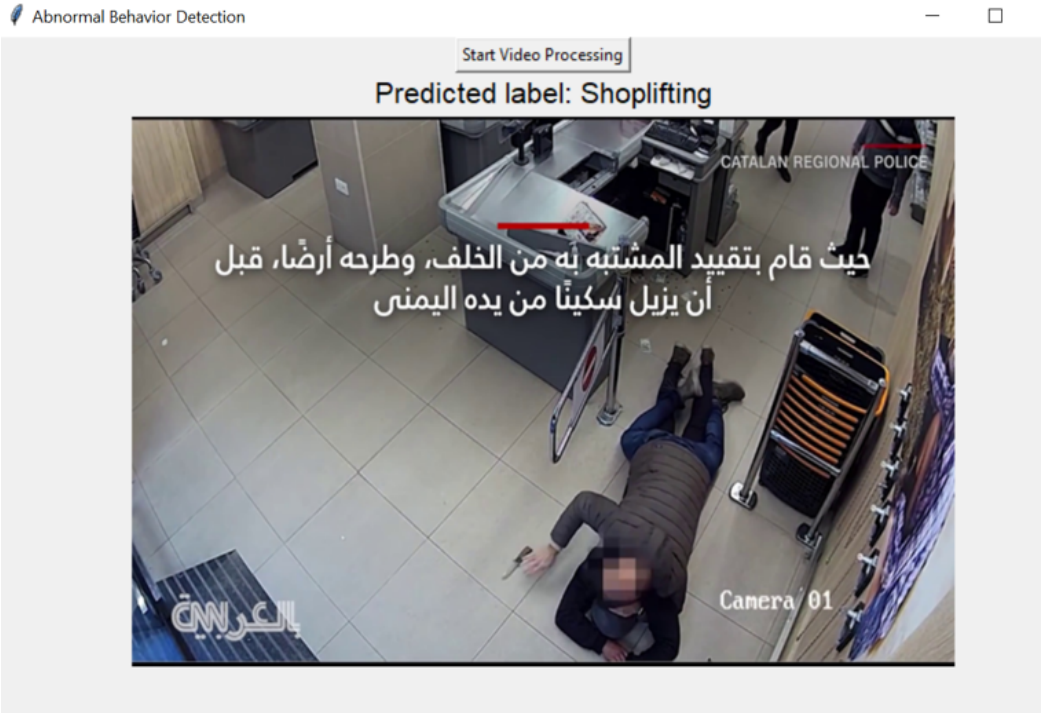

Figure 4.17: Predicted Label : Shoplafting

Figure 4.18: Predicted Label : Stealing

4.3 Evaluating of the Proposed Method's Effectiveness

1. This application offers a platform for real-time video frame prediction and analysis. Including security monitoring, object detection and visual analysis.
2. The application ABD makes use of a pre-trained deep learning model for image classification. This demonstrates a strong foundation in modern AI technologies, enabling accurate and effective predictions.
3. The GUI's ability to display forecast frames and class labels in real time engages users and provides immediate feedback on rating results.
4. Processing video frames can be resource intensive, which can lead to response time or system resource limitations. Improving image pre-processing and model inference can mitigate this.
5. The application ABD lacks mechanisms that allow users to pause, skip frames, or navigate within a video. In the future, we may work on integrating these functions to improve the user experience.
6. Understanding the intended posting context is crucial. For example, running the software on hardware with limited processing power or adapting it for real-time video streams requires careful consideration.

Overall, this application ABD successfully combines the power of deep learning with the interactivity of a graphical user interface, demonstrating the foundation of a ver-

satile image classification application. While the software provides a strong starting point, further development is necessary to address considerations such as model selection, performance optimization, and extended user interactions. This work shows the potential for creating user-friendly AI-powered applications.

4.4 Conclusion

In this chapter, we have looked at several essential aspects of our work. First, we explored the ”hardware environment,” detailing the technology infrastructure that forms the backbone of our application. Next, we provided insights into the ’development environment’, where the fusion of AI and machine learning took shape. We thoroughly evaluated the effectiveness of the proposed method, and discussed the results and implications of our approach for detecting abnormal behaviors in wholesale store surveillance.

General Conclusion

After completing this study and conducting the necessary research described in our dissertation, we extensively reviewed relevant articles and posts to determine the best approach for our model. Our goal of creating a system to identify and prevent abnormal behavior in retail centers has been successfully achieved.

In the initial stages, we comprehensively explained the key concepts critical to this project, including artificial intelligence, the Internet of Things, edge computing, cloud technology, machine learning, and transfer learning. Analysis of projects similar to ours provided us with ideas, enabling us to devise an AI-based solution - the application of abnormal behavior detection in surveillance cameras used by wholesale stores.Next, we collected essential data to implement this app, which is a pivotal step for its effectiveness.

Our core emphasis resided in bridging cutting-edge technology with real-world utility. Regrettably, due to certain limitations, we were unable to seamlessly integrate the application with IoT Edge. Despite this, we adeptly employed the model to forecast outcomes within specific video scenarios. With the culmination of our research, we conducted an appraisal of the application's prospects for refinement, leveraging the available dataset.

We aspire to further enhance and develop this work into an effective preventive tool in the future.

References

[1] Luigi Atzori, Antonio Iera, and Giacomo Morabito. The internet of things: A survey. *Computer networks*, 54(15):2787–2805, 2010.

[2] Casper Solheim Bojer and Jens Peder Meldgaard. Kaggle forecasting competitions: An overlooked learning opportunity. *International Journal of Forecasting*, 37(2):587–603, 2021.

[3] J Chugh. Types of machine learning and top 10 algorithms everyone should know. *URL: https://blogs. oracle. com/ai/types-ofmachine-learning-and-top-10-algorithms-everyone-should-know*, 2018.

[4] Chiara Contoli and Emanuele Lattanzi. A study on the application of tensorflow compression techniques to human activity recognition. *IEEE Access*, 2023.

[5] Sujata Dash, Biswa Ranjan Acharya, Mamta Mittal, Ajith Abraham, and Arpad Kelemen. *Deep learning techniques for biomedical and health informatics*. Springer, 2020.

[6] Shikha Dubey, Abhijeet Boragule, and Moongu Jeon. 3d resnet with ranking loss function for abnormal activity detection in videos. In *2019 International Conference on Control, Automation and Information Sciences (ICCAIS)*, pages 1–6. IEEE, 2019.

[7] Jia-Chang Feng, Fa-Ting Hong, and Wei-Shi Zheng. Mist: Multiple instance self-training framework for video anomaly detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14009–14018, 2021.

[8] A Gazis. What is iot? the internet of things explained. *Academia Letters*, page 2, 2021.

[9] Cyril Goutte and Eric Gaussier. A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. In *European conference on information retrieval*, pages 345–359. Springer, 2005.

[10] Salam Hamdan, Moussa Ayyash, and Sufyan Almajali. Edge-computing architectures for internet of things applications: A survey. *Sensors*, 20(22):6441, 2020.

[11] Kamran Jabbar. Performance optimization of swarm intelligence-based clustering algorithms. Master's thesis, Itä-Suomen yliopisto, 2021.

[12] Patrik Johansson and Amin Kokhaei Pak. Utilising computer simulations to teach derivatives with p5. js and google colaboratory: A comparative study, 2023.

[13] S Kassel. Predicting building code compliance with machine learning models, 2017.

[14] J Kaur Gill. Automatic log analysis using deep learning and ai. *Accès https://www. xenonstack. com/blog/log-analytics-deep-machine-learning*, 2018.

[15] Bin Li. Facial expression recognition by densenet-121. In *Multi-Chaos, Fractal and Multi-Fractional Artificial Intelligence of Different Complex Systems*, pages 263–276. Elsevier, 2022.

[16] Li Li, Jiawei Wang, and Haowei Quan. Scalpel: The python static analysis framework. *arXiv preprint arXiv:2202.11840*, 2022.

[17] Hui Lv, Chuanwei Zhou, Zhen Cui, Chunyan Xu, Yong Li, and Jian Yang. Localizing anomalies from weakly-labeled videos. *IEEE transactions on image processing*, 30:4505–4515, 2021.

[18] Chandeep Sharma and Sayonee Parikh. Transfer learning and its application in computer vision: A review. 2022.

[19] George Stockman and Linda G Shapiro. *Computer vision*. Prentice Hall PTR, 2001.

[20] Akshay Swaminathan, C Varun, S Kalaivani, et al. Multiple plant leaf disease classification using densenet-121 architecture. *Int. J. Electr. Eng. Technol*, 12:38–57, 2021.

[21] Yu Tian, Guansong Pang, Yuanhong Chen, Rajvinder Singh, Johan W Verjans, and Gustavo Carneiro. Weakly-supervised video anomaly detection with robust temporal feature magnitude learning. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4975–4986, 2021.

[22] Mrs Nethravati TL, Rakshita L Patil, S Bhavana, Sairin Ray Choudhury, and S Monisha. Virtual painter using artificial intelligence and opencv.

[23] Derya Ucuz et al. Comparison of the iot platform vendors, microsoft azure, amazon web services, and google cloud, from users' perspectives. In *2020 8th international symposium on digital forensics and security (ISDFS)*, pages 1–4. IEEE, 2020.

[24] Jue Wang and Anoop Cherian. Gods: Generalized one-class discriminative subspaces for anomaly detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8201–8211, 2019.