SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
FACULTY OF ENGINEERING AND TECHNOLOGY
DEPARTMENT OF DATA SCIENCE AND BUSINESS SYSTEMS

18CSP107L- MINOR PROJECT / 18CSP461L-Project
Evaluation 1

Weapon Detection using Deep Learning Architecture

Siddhartha Srivastava (RA2011042010122)
Mayank Sandilya (RA2011042010094)
Mentor - Dr. Vijayalakshmi V
Designation: Assistant Professor

TABLE OF CONTENTS

- Introduction – Problem statement
- Objectives
- Literature Survey
- Architecture diagram
- UML Diagrams-ER diagrams/Use case diagram/Activity diagram (whichever is applicable)
- Algorithms used

PROBLEM STATEMENT

The problem of weapon detection is becoming increasingly critical in various security and surveillance applications. Incidents involving firearms and other dangerous weapons pose a significant threat to public safety and necessitate effective and efficient detection methods. Traditional methods, such as manual inspection or metal detectors, are often time-consuming, labor-intensive, and have limitations in terms of real-time monitoring. To address these challenges, the proposed project aims to develop a robust and accurate weapon detection system using deep learning architecture.

SRM INSTITUTE OF SCIENCE & TECHNOLOGY
Deemed to be University u/s 3 of UGC Act, 1956

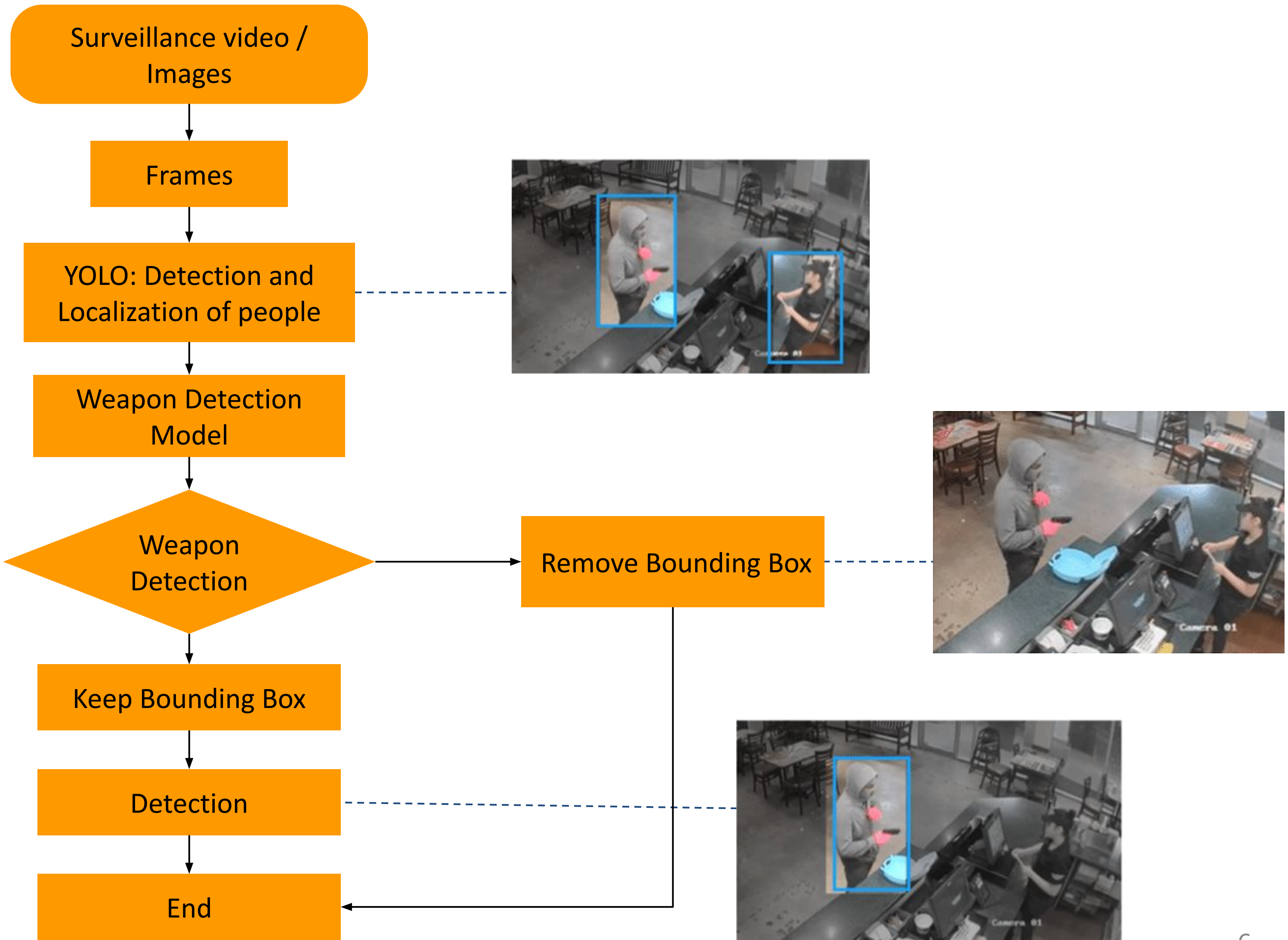
OBJECTIVE

- Develop a Robust Deep Learning Model: Create a deep learning model that can accurately identify and classify various types of weapons, including firearms, knives, and blunt objects, with high precision.
- Real-Time Detection: Implement the model to achieve real-time weapon detection, ensuring rapid response to potential threats in security and surveillance scenarios.
- Minimize False Positives: Fine-tune the model to reduce false positives by implementing advanced post-processing techniques and mitigating the misidentification of harmless objects as weapons.
- Continuous Learning and Improvement: Implement mechanisms for continuous model retraining and improvement to keep the weapon detection system up-to-date with emerging threats and changing environments.

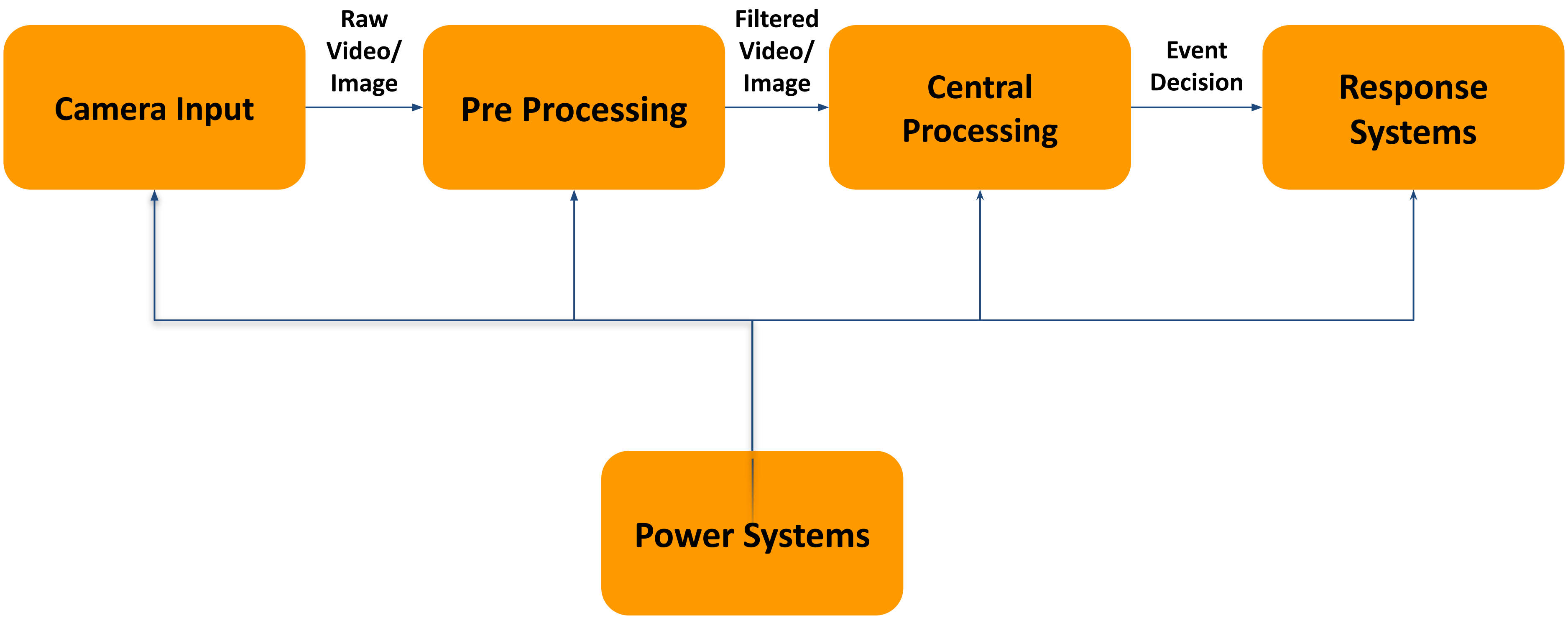
LITERATURE REVIEW

S No.	Name	Author	Year
1	Weapon Detection using Artificial Intelligence and Deep Learning for Security Applications	Harsh Jain, Aditya Vikram	2020
2	A comprehensive study towards high-level approaches for weapon detection using classical machine learning and deep learning methods	Parindar Yadav, Nidhi Gupta	2022
3	Object Detection: Harmful Weapons Detection using YOLO v4	Wan Emilya Izzety Binti Wan Noor Afandi, Naimah Mat Isa	2021
4	Weapon Detection from Surveillance Images using Deep Learning	Anjali Goenka, K. Sitara	2022

Working Model



ARCHITECTURE DIAGRAM



ARCHITECTURE DIAGRAM

- The figure represents the architecture diagram of the project.
- Firstly, the input is taken from the data and algorithm is trained using data sets, which is used as training data.
- The input taken from the camera and the images of different weapon i.e. guns to train the model
- Then produces a frame around the gun or weapon and final decision is made as to alert or give a warning.

PROCEDURE

1. **Data Collection:**

a. Dataset Acquisition: Gather a diverse dataset of images and videos that contain instances of various weapons in different environments. Ensure the dataset includes a wide range of weapon types, lighting conditions, and backgrounds.

b. Data Annotation: Annotate the dataset with bounding boxes around weapons and label them according to weapon types for supervised training.

c. Data Augmentation: Apply data augmentation techniques to increase the dataset's diversity, such as rotation, scaling, and color adjustments.

2. **Data Preprocessing:**

a. Data Cleaning: Remove any noisy or irrelevant data from the dataset.

b. Data Split: Divide the dataset into training, validation, and test sets, ensuring a balanced distribution of weapon categories.

PROCEDURE

3. Model Selection:

- Choose a suitable deep learning architecture for weapon detection, such as Convolutional Neural Networks (CNNs) or more advanced architectures like YOLO (You Only Look Once) or Faster R-CNN.
- Configure the architecture's hyperparameters, including the learning rate, batch size, and optimization algorithm.

4. **Model Training**:

- Train the selected model on the training dataset using the annotated data and appropriate loss functions, like mean squared error or cross-entropy.
- Implement early stopping and model checkpointing to prevent overfitting and save the best-performing model during training. c. Monitor training progress with metrics like accuracy, precision, recall, and F1-score.

PROCEDURE

5. Model Evaluation:

- Evaluate the model's performance on the validation dataset to fine-tune hyperparameters and assess the model's generalization ability.
- Use performance metrics to measure accuracy, false positive rate, and false negative rate.
- Adjust the model based on evaluation results, if necessary.

6. Testing and Validation:

- Assess the final model's performance on the test dataset to ensure its generalization to new, unseen data.
- Calculate performance metrics to validate the model's accuracy and reliability.

7. Real-Time Deployment:

- Implement the trained model in a real-time deployment environment, integrating it with surveillance cameras or systems.
- Optimize the model for real-time processing, taking into account hardware and computational resource constraints.

ALGORITHM USED:

- Objective: Develop a real-time weapon detection system using deep learning to enhance security and public safety.
- Algorithm: Choose a deep learning architecture suitable for object detection (e.g., YOLO or Faster R-CNN) and configure the model.
- Model Training Process:
  - Initialize the model with pre-trained weights for efficient convergence.
  - Define and compute loss functions for object detection, including localization and classification losses.
  - Train the model using an optimization algorithm (e.g., Adam or SGD) on the training dataset.
  - Monitor training progress using metrics such as mean Average Precision (mAP), IoU, and loss.
  - Implement learning rate scheduling, batch normalization, and dropout to improve training stability.
  - Fine-tune model hyperparameters based on validation results.
  - Apply non-maximum suppression (NMS) to refine detection results.
  - Continuously update the model for improved performance and adaptation to evolving threats.
- Output Process Overview:
  - Display real-time alerts when weapons are detected.
  - Provide visual feedback through bounding boxes around detected weapons.
  - Log detection events for post-analysis and incident reporting.

THANK YOU