Name : Nimra Shahzadi
Postion : Research Assistant of AI
Data : 30-06-2024
Project Name : Real-time Event Detect (Fire Detection , Person Count Trcaking).

1. Introduction

- **Project Overview:**
  This project aims to develop a comprehensive real-time detection system capable of identifying fire incidents and counting the number of people within a given environment. The system is designed to process live video feeds, making it suitable for use in various safety-critical settings such as public spaces, buildings, and industrial facilities.
- **Objective:**
  The primary objective is to leverage deep learning models to ensure timely detection of fires and accurate counting of people. By implementing this system, organizations can improve safety measures, enabling quicker response times to emergencies and efficient crowd management.
- **Importance:**
  Real-time detection of fires and counting of people is crucial for preventing accidents, ensuring public safety, and optimizing resource allocation during emergencies. The system's ability to provide real-time alerts and data-driven insights can significantly enhance situational awareness and decision-making processes.

2. Data Collection

- **Define Objectives:**
  The data collection process begins with a clear understanding of the goals: capturing visual information necessary for detecting fires and counting people. This includes determining the variety of scenarios, lighting conditions, and environments that need to be covered to ensure model robustness.
- **Identify Image Sources:**
  Images will be sourced from various channels, such as security camera footage, publicly available datasets, and custom-collected images from specific environments like buildings, streets, or outdoor spaces. This diversity ensures the model can generalize across different settings.
- **Acquire Images:**
  A comprehensive dataset will be compiled, including images with varying numbers of people and different fire conditions. The dataset will represent multiple scenarios, such as crowded areas, isolated individuals, small fires, and large-scale fire incidents.
- **Data Preprocessing:**
  Collected images will undergo preprocessing to enhance their quality. Techniques such as

resizing, cropping, color normalization, and filtering will be applied to improve the model's ability to detect relevant features in different environments and under varying conditions.

3. Model Training

- **Select Model Architecture:**
  Two separate deep learning models will be trained—one for person counting and one for fire detection. The YOLO (You Only Look Once) architecture, known for its speed and accuracy in object detection tasks, will be used for both models, with potential customizations for fire detection to improve performance.
- **Data Splitting:**
  The dataset will be divided into training, validation, and testing sets using an online data labeling tool. This ensures that the model is trained on diverse data and validated on unseen images, providing a realistic measure of its performance.
- **Model Training:**
  The training process involves feeding the models with labeled data. The person-counting model will be trained to recognize and count individuals, while the fire detection model will focus on identifying fire instances within images.
- **Model Evaluation:**
  The performance of each model will be assessed using validation datasets. Key metrics include Mean Average Precision (mAP) for object detection tasks and accuracy, precision, recall, and F1-score for fire detection. These metrics will guide further model improvements.
- **Hyperparameter Tuning:**
  Hyperparameters like learning rate, batch size, and model architecture will be tuned iteratively to optimize the models' performance, ensuring they can accurately and efficiently process real-time data.
- **Testing:**
  The final models will be tested on a separate dataset to evaluate their generalization capability. This step helps ensure that the models perform well on new, unseen images, reflecting real-world scenarios.

4. Model Testing

- **Integration and Frame Detection:**
  - **Integration of Trained Model:**
    The trained models will be integrated into the application's codebase, enabling them to analyze frames from video streams in real-time. The integration will be seamless, ensuring that the models can process data from live camera feeds with minimal latency.

    - **Detection of Frames:**
      The models will analyze each frame, typically at 14 frames per second (fps), to detect people and fire incidents. This rate ensures a balance between processing speed and detection accuracy.
    - **Saving Detected Frames in MongoDB:**
      Detected frames, along with metadata such as frame number and timestamps, will be saved in MongoDB. This NoSQL database is chosen for its flexibility and ability to handle unstructured data like images.
- **Real-time Camera Testing:**
    - **Implementation on Real-time Camera Feed:**
      The integrated models will be deployed to monitor live camera feeds. They will continuously process the incoming video stream, detecting and recording events in real time.
    - **Saving Real-time Timestamped Frames:**
      Frames with detected events will be timestamped and stored in MongoDB. This allows for easy retrieval and analysis of historical data, facilitating post-incident investigations and trend analysis.

5. Flask Web-Based Dashboard

- **Flask Application Setup:**
  The Flask web application will serve as the front-end interface, providing users with access to real-time data and insights. Routes will be created for different pages, including the main dashboard, fire detection dashboard, and person detection dashboard.
- **User Authentication:**
  User authentication is implemented to ensure secure access to the dashboard. The Flask-Login extension will manage user sessions, and forms for login and sign-up will be secured with appropriate validations.
- **MongoDB Integration:**
  MongoDB will be integrated with Flask using PyMongo or Flask-PyMongo. This integration enables seamless data storage and retrieval, allowing the dashboard to display real-time data and historical records.
- **Main Dashboard:**
    - **Person Count Card:**
      A bar graph will display real-time person count data. Users can select a specific year to view monthly data from January to December.
    - **Fire Detection Card:**
      A non-linear graph will show real-time fire detection data, allowing users to monitor incidents over time.
- **Fire Detection Dashboard:**

    - **Bar Graph:**
      This graph will display yearly fire detection records. Users can filter the data by year, and the graph will update dynamically.
    - **Tabular Data:**
      A table will display detailed fire detection records, including timestamps, event types, and associated frames.
- **Person Detection Dashboard:**
    - **Non-Linear Graph:**
      This graph will show yearly person count data, allowing users to select specific years for detailed analysis.
    - **Tabular Data:**
      Detailed records of person counts, including timestamps and counts of people entering or leaving an area, will be displayed in a table.
- **Additional Notes:**
    - HTML templates with Jinja2 will be used for rendering dynamic content.
    - JavaScript libraries will create interactive charts, enhancing user experience.
    - Error handling and validation will ensure robustness, while responsive design will guarantee usability across devices.

6. Adding New Camera

- **Camera Setup:**
  A page will be designed to allow users to add new cameras. The form will include fields for Camera ID and IP address, with the data being stored in MongoDB for easy access and management.

7. Adding New Model

- **Model Registration:**
  Users can add new models via a dedicated page. The form will include fields for Unique ID, Model Name, and Path, with the information being stored in MongoDB to facilitate easy model selection and deployment.

8. View Stream

- **Stream Viewing:**
  The Stream View page will allow users to select a camera and model from the database. Once selected, the system will display real-time images processed by the chosen model, providing immediate insights into the monitored area.