Project Progress Report: TeleICU Patient Monitoring System

Team: Tensor Stars

July 1, 2024

1 Introduction

This project aims to develop a comprehensive system for motion detection and object detection in ICU videos. The primary objectives are to detect the presence of various objects and identify motion patterns, particularly focusing on scenarios where the patient is either alone or accompanied by family members. The system integrates YOLOv8s for object detection and an LSTM model for motion detection, leveraging advancements in computer vision and machine learning to enhance patient safety and streamline monitoring processes.

2 Object Detection

2.1 Initialising Requirements

The first step involved identifying the necessary computational resources to ensure smooth processing and model training. A high-performance computer equipped with a powerful GPU was selected to handle the extensive data processing and model training tasks. Essential software requirements included Python, OpenCV, YOLOv8, and other libraries for image and video processing. The setup also included robust storage solutions to manage the large datasets effectively.

2.2 Finding and Preprocessing Appropriate Images

A diverse set of YouTube videos featuring doctors, ICU patients, staff, medical equipment, family members, and ECG monitors were identified

and downloaded. The preprocessing phase involved editing these videos to remove inappropriate content such as logos, watermarks, and irrelevant scenes. This step was crucial to ensure that the data used for training the model was clean and relevant, enhancing the model's performance.

Figure 1: Example of preprocessing steps for cleaning and preparing video frames.

2.3 Converting Videos to Images

The downloaded videos were converted into individual frames using a combination of online tools and custom scripts. Each video was split into frames, and the frames were subsequently filtered based on their relevance and clarity. Frames that were too blurry or irrelevant were discarded, resulting in a curated collection of images ready for annotation and model training.

2.4 Annotating Labels on Images

Using Roboflow, each image was meticulously annotated by creating bounding boxes around the objects of interest. These objects were then classified

according to their respective categories, such as doctors, patients, medical equipment, etc. Accurate annotation is critical as it directly impacts the performance of the object detection model. Each annotated image contributes to the training process, helping the model learn to identify and classify objects correctly.

2.5 Splitting Data

The annotated dataset was split into three subsets: training, validation, and testing. Typically, 70% of the data was allocated for training, 20% for validation, and 10% for testing. This split ensures that the model is trained on a diverse set of images and can be effectively validated and tested. The training set is used to train the model, the validation set is used to tune the hyperparameters and prevent overfitting, and the testing set is used to evaluate the final model performance.

2.6 Creating a data.yaml File

A data.yaml file was created to store the directory locations of the training, validation, and testing images and labels. This file also included the number of classes and their respective labels. The data.yaml file is essential for the training process as it directs the model to the correct data sources, ensuring an organized structure for data access.

2.7 Training the Model

A YOLOv8s model was trained on the custom dataset. The training process involved multiple epochs, with the model's performance being evaluated after each epoch. Hyperparameters such as learning rate, batch size, and number of epochs were fine-tuned to achieve optimal performance. The model achieved an accuracy of 80%, indicating its effectiveness in detecting and classifying objects in the ICU environment. The training phase also involved regular validation to ensure that the model generalizes well to unseen data.

2.8 Predicting Videos Frame by Frame

The trained YOLOv8s model was used to predict objects in videos on a frame-by-frame basis. Each frame of the video was processed, and the model outputted the detected objects along with their bounding boxes and class labels. The results were then compiled into a video file in .mp4

format, showcasing the model's predictions in real-time. This frame-by-frame prediction approach ensures that all relevant objects in each frame are accurately detected and classified.

3 Motion Detection

3.1 Bounding Boxes and Motion Detection

To detect motion, bounding boxes were utilized for the classes in the dataset. Motion detection was implemented by comparing differences between alternate frames. If the difference between consecutive frames exceeded a predefined threshold value, motion was detected. This method helps in identifying significant movements, such as a patient moving or a family member entering the ICU.

3.2 Checking Proximity

A model was created to detect motion when the patient was alone or when family members were nearby. This involved setting up rules to determine the proximity of detected objects. For instance, if the bounding box of a family member overlapped with that of the patient, it indicated that the family member was close to the patient. This proximity check helps in identifying critical scenarios where the patient's condition might require immediate attention.

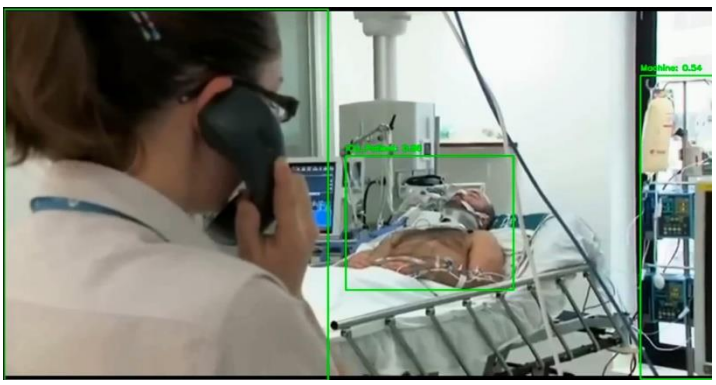

Figure 2: Example of checking proximity between patient and family member.

3.3 Increasing Model Accuracy

To improve the accuracy and reliability of the motion detection model, an LSTM (Long Short-Term Memory) network was integrated. LSTMs are effective in handling sequential data and can capture temporal dependen-

cies, making them ideal for motion detection tasks. The LSTM model was trained to smooth out the predictions and reduce false positives, ensuring more accurate motion detection.

3.4 Output

The motion detection model outputs the top 10 frames where motion is most likely to be detected. These frames are selected based on criteria such as the ICU patient being alone or with family and the frame difference being above the threshold value. This output helps in focusing on critical moments in the video, enabling prompt and effective response in real-world scenarios.

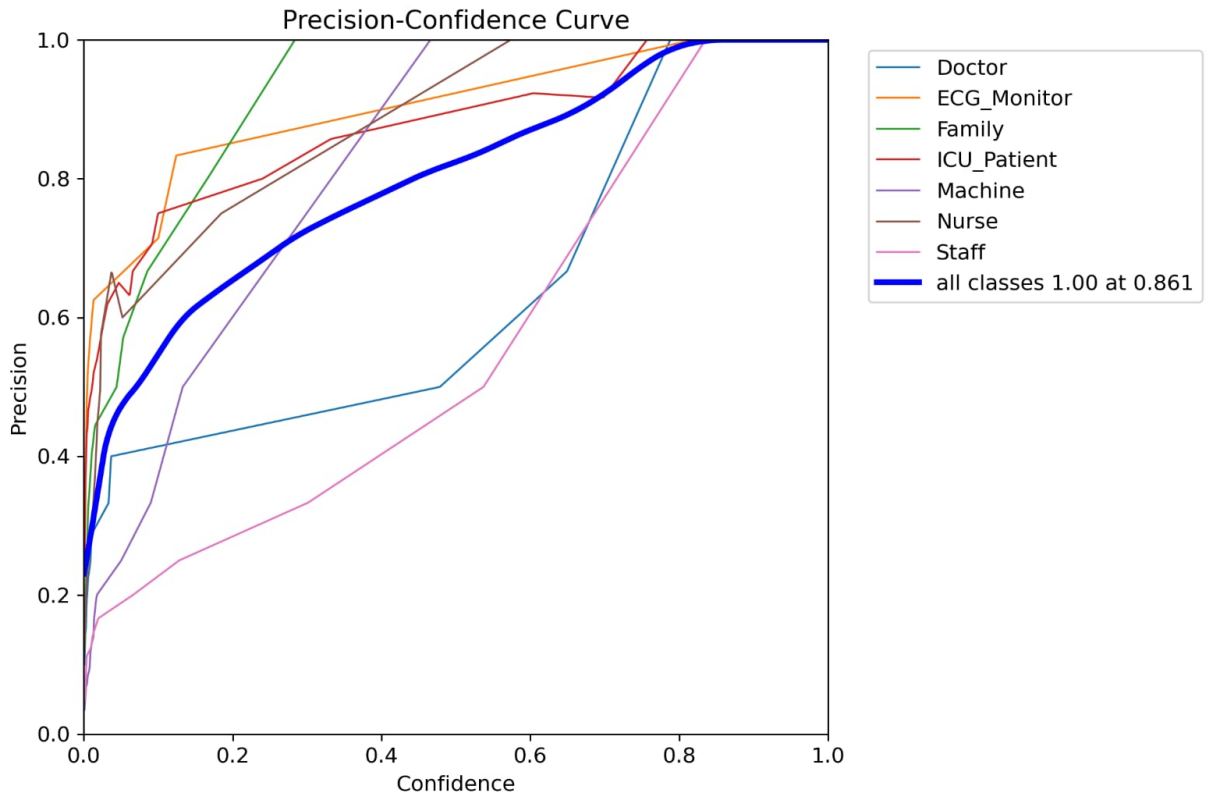

Figure 3: Example of output frames showing detected motion.

4 API Call

4.1 Combining Models

Both the motion and object detection models were combined into a single API. This integration allows for simultaneous object detection and motion detection in one iteration of frames. The API processes the video, detects objects, identifies motion, and outputs a comprehensive result. This combined approach enhances the efficiency and effectiveness of the system.

4.2 API Functionalities

The API provides several functionalities:

- Index: Displays a welcome screen.

- Upload: Allows direct video upload and outputs the results. Users can upload videos through a user-friendly interface, and the API processes the video to provide the output.

- Process: Can be accessed using POSTMAN software with a JSON request. This feature allows users to import the video from a specified location in the JSON file and returns the locations of detected frames and the output video.

These functionalities ensure that the API is versatile and can be easily integrated into various workflows.

4.3 Server Access

The API can be accessed via the local host server at: http://127.0.0.1:9000. This server handles the requests and provides the necessary responses, ensuring seamless integration and usability. The local host setup allows for easy testing and deployment within a controlled environment.

5 Conclusion

This report outlines the steps taken to develop and integrate a motion detection and object detection system for ICU videos. The combination of YOLOv8s and LSTM models has provided a robust solution for the project's objectives. The API integration further enhances the usability, making it easier to deploy and utilize the system in real-world scenarios. Future work will focus on improving the models' accuracy, expanding the dataset, and exploring additional functionalities to enhance the system's capabilities.

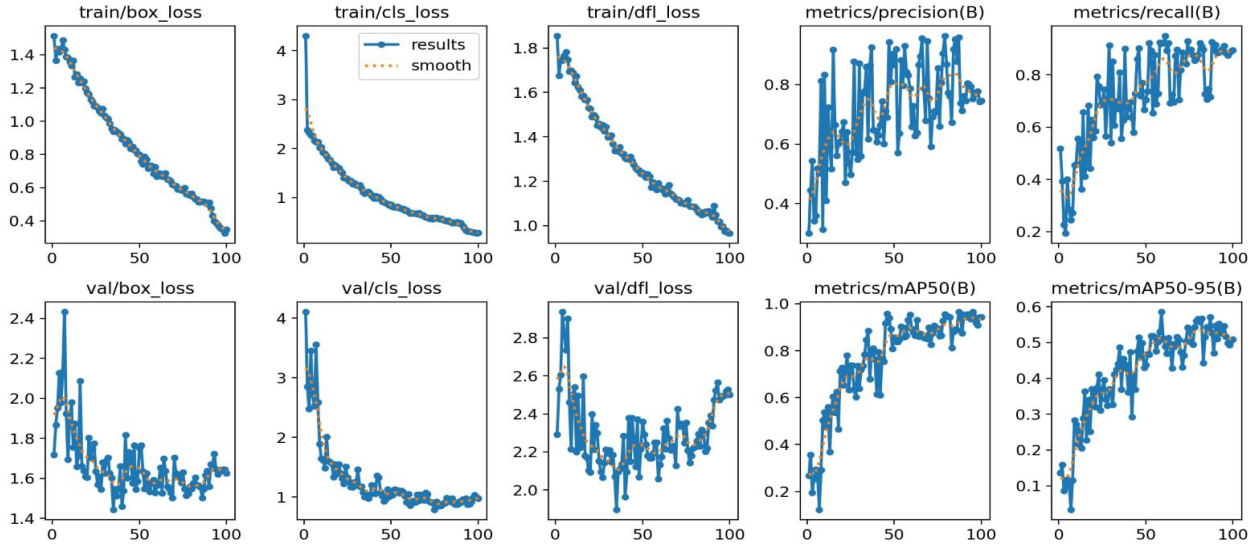

Figure 4: Overview of the developed system showing motion and object detection.