Video Object Detection and Tracking Pipeline

1. Introduction

The goal of this project is to develop a robust video object detection and tracking pipeline to identify and track individuals (children and therapists) in videos. This system aims to help analyse behaviours, emotions, and engagement levels, particularly in the context of children with Autism Spectrum Disorder and their interactions with therapists. The pipeline is built using a custom-trained YOLOv8 object detection model and a tracking algorithm to assign unique IDs to individuals, handle re-entries, and maintain accurate tracking through occlusions or partial visibility.

2. Approach and Implementation

The project is broken down into several key steps, each with a specific purpose and outcome. Below is a detailed breakdown of each step and its implementation:

2.1. Requirements Setup

- A `requirements.txt` file was created to specify the necessary Python libraries, including `pytube` for video downloads, `YOLOv8` for object detection, and additional dependencies for tracking, data processing, and video manipulation.
- The environment was set up on a GPU-enabled machine to allow for faster training and inference.

2.2. Downloading Videos

- A script named `download_videos.py` was developed using the `pytube` library to download videos from YouTube. This script automates the process of fetching videos required for training and testing the object detection model.
- The videos are saved locally for further processing.

2.3. Auto-Annotating Videos

- The `auto_annotate.py` script was created using the AutoDistill framework to generate initial annotations for the downloaded videos. The script identifies all persons in each video frame and annotates them with bounding boxes labeled as "person."
- This automated annotation serves as a preliminary step, reducing the manual effort required to label large datasets.

2.4. Manual Annotation Correction

- The initial annotations generated by the auto-annotation script were refined manually to differentiate between "child" and "adult" classes. Tools like Roboflow and LabelImg were used to change labels accordingly.

- Accurate annotations are crucial for training an effective object detection model that can distinguish between different types of individuals.

2.5. Training the Object Detection Model

- The `train.py` script was developed to train a custom YOLOv8 model. The model was trained on the manually corrected dataset to detect two classes: "child" and "adult."
- The model was trained for 25 epochs, and the weights were saved for future use. The YOLOv8n model was selected for its balance between speed and accuracy, suitable for real-time video processing.

2.6. Object Tracking in Videos

- The `track.py` script was implemented to track objects in input videos. The script uses the trained YOLOv8 model to detect individuals in each frame and applies a tracking algorithm to assign unique IDs, handle re-entries, and maintain tracking accuracy through occlusions or partial visibility.
- The tracking script outputs a video with bounding boxes around detected individuals, labeled with their corresponding class ("child" or "adult") and unique IDs.

3. Current Performance and Results

- The current pipeline successfully detects and tracks children and therapists in the provided videos. The YOLOv8 model performs well in terms of detection accuracy, while the tracking algorithm provides acceptable performance in assigning unique IDs and handling occlusions and re-entries.
- The output video demonstrates clear labeling and consistent tracking of individuals throughout the video duration.

4. Limitations and Areas for Improvement

While the current pipeline is functional, there are several areas where it can be further optimized and improved:

4.1. Tracking Algorithm Enhancement

- **Issue**: The current tracking implementation is basic and relies on a default tracking algorithm. It may struggle with complex scenarios involving multiple people, re-entries, or prolonged occlusions.
- **Improvement**: Incorporate more advanced tracking algorithms such as **Deep SORT**, **BoT-SORT**, or **ByteTrack**. These algorithms use deep learning models to track objects more accurately by considering both appearance features and motion information. This will enhance the ability to maintain unique IDs, even in challenging conditions.

4.2. Fine-Tuning Tracker Parameters

- **Issue**: The current parameters used in the tracker might not be optimized for our specific dataset and video characteristics, leading to suboptimal tracking performance.
- **Improvement**: Perform fine-tuning of the tracker parameters (e.g., confidence thresholds, maximum age, minimum hits) to better suit the specific dynamics of our videos. Use grid search or other hyperparameter optimization techniques to find the optimal settings.

4.3. Multi-Camera and Multi-Person Tracking

- **Issue**: The current setup only processes single-camera video feeds, and the tracking might not be robust in multi-camera setups where individuals move between different camera views.
- **Improvement**: Develop or integrate multi-camera tracking capabilities to maintain consistent tracking across different camera angles. Use cross-camera tracking algorithms that match individuals based on appearance and temporal information.

4.4. Improving Occlusion Handling

- **Issue**: The current system has limited ability to handle complete occlusions or situations where individuals leave and re-enter the frame.
- **Improvement**: Implement advanced techniques like Kalman Filters, Recurrent Neural Networks (RNNs), or Graph-based algorithms to predict and track individuals more effectively through occlusions and exits/entries.

4.5. Real-time Processing Optimization

- **Issue**: While YOLOv8n is optimized for speed, the current pipeline may still face delays in real-time video processing due to heavy computational loads from the detection and tracking steps.
- **Improvement**: Optimize the code for parallel processing and make use of GPU acceleration wherever possible. Implement techniques like frame skipping (processing every nth frame) to reduce computational overhead.

4.6. Enhanced Data Augmentation

- **Issue**: The current model training relies on a relatively straightforward data augmentation strategy, which may not cover all possible variations in the dataset.
- **Improvement**: Use more sophisticated data augmentation techniques such as random rotations, color jittering, and synthetic occlusion generation to create a more robust training dataset. This can help improve model performance under varied lighting conditions, angles, and occlusions.

5. Conclusion

The video object detection and tracking pipeline is a solid foundation for analyzing behaviors and engagement levels in children with Autism Spectrum Disorder and their interactions with

therapists. The current approach successfully trains a custom YOLOv8 model and utilizes a basic tracking algorithm to achieve the desired objectives.

6. Future Directions

To further enhance the system's accuracy, robustness, and real-time processing capabilities, the following steps are recommended:

- Integrate advanced tracking algorithms like Deep SORT, BoT-SORT, or ByteTrack.
- Fine-tune tracker parameters to optimize performance.
- Explore multi-camera tracking approaches.
- Improve occlusion handling using advanced prediction techniques.
- Optimize real-time processing with GPU acceleration and parallelization.
- Enhance data augmentation to create a more diverse and robust training dataset.

By implementing these improvements, the pipeline can become a more powerful tool for behaviour analysis, ultimately contributing to more effective treatment plans and interventions for children with Autism Spectrum Disorder.