Objective

- The primary objective of this project was to develop and train an object detection model using the Faster R-CNN architecture to accurately detect objects in images, specifically focusing on faces. This involved not only detecting the facial keypoints but also generating a bounding box that encompasses all detected keypoints. Customizing the model to fit a specific dataset and ensuring its performance through validation and fine-tuning were key components of the project.

Approaches:

1. **Model Selection and Definition**:

- The project began with selecting an appropriate object detection model. The Faster R-CNN model with a ResNet50 backbone was chosen due to its robustness in detecting objects. The model was customized to suit the specific number of classes in the dataset by replacing the default box predictor with a new one configured for the target classes. The Keypoint R-CNN was incorporated to detect facial keypoints. The model was modified to ensure a single bounding box contained all the detected keypoints for the face.

2. **Dataset Preparation**:

- Custom datasets were created and loaded using the `CustomCocoDataset` class, which extended the `CocoDetection` class to handle annotations and apply transformations. This included defining proper bounding box coordinates and labels for object detection. The dataset was processed to ensure all keypoints were contained within a single bounding box. This was crucial for accurate facial detection.

3. **Keypoint Detection and Bounding Box Calculation**:

- A specific algorithm was used to calculate the bounding box for the face based on the detected keypoints. This involved identifying the coordinates of key facial points (e.g., eyes, nose, mouth corners) and computing the minimal bounding rectangle that encloses these points.The keypoint detection algorithm ensured that only visible points were considered, improving the relevance and accuracy of the bounding box.

4. **Training the Model**:

- The model was trained using a training dataset and evaluated on a validation dataset. During this phase, the model's parameters were adjusted, and an optimizer was used to minimize the loss.Challenges in this phase included ensuring the model fit well and achieving optimal performance metrics. The training loop was carefully managed to handle potential issues with invalid labels or keypoints.

5. **Handling Annotations and Data Sequence**:

- Proper formatting and handling of annotations were crucial. The annotations were carefully processed to ensure they were in the correct format for the model. This included consolidating keypoints into a single bounding box per face.Additionally, the sequence of data processing, including data loading and transformations, was managed to ensure effective model training.

6. **Visualization and Evaluation**:

- After training, the model's predictions were visualized to assess its performance. This involved drawing bounding boxes around detected objects and evaluating the confidence levels of these predictions.The results were analyzed to determine the model's effectiveness and identify areas for improvement.

Challenges

- Low Confidence Level in Validation: The model's performance on the validation set was initially poor, requiring iterative adjustments to improve its confidence and accuracy.

- Defining a Proper Model: Selecting and configuring the model to match the dataset's requirements was a complex task, requiring careful adjustments to the Faster R-CNN architecture and the inclusion of Keypoint R-CNN features.

- Fitting the Model: Ensuring the model was properly fitted to the dataset involved extensive training and validation, which was challenging and required multiple iterations.

- Proper Annotation: Handling and formatting annotations accurately were crucial for the model to learn effectively. Any discrepancies in annotations could lead to suboptimal performance.

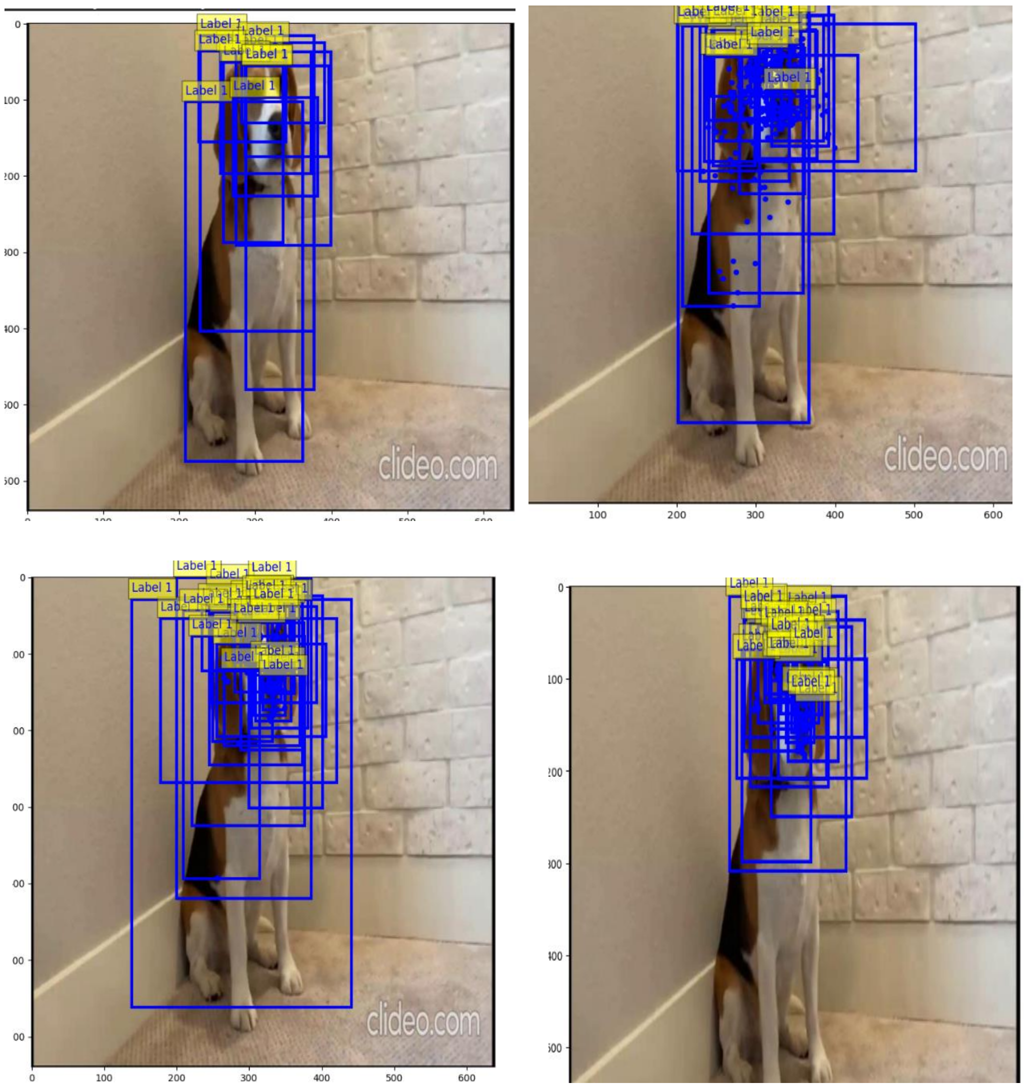
- Sequence of Data: Managing the sequence of data processing, including transformations and loading, was important to ensure the model received the correct input and training.

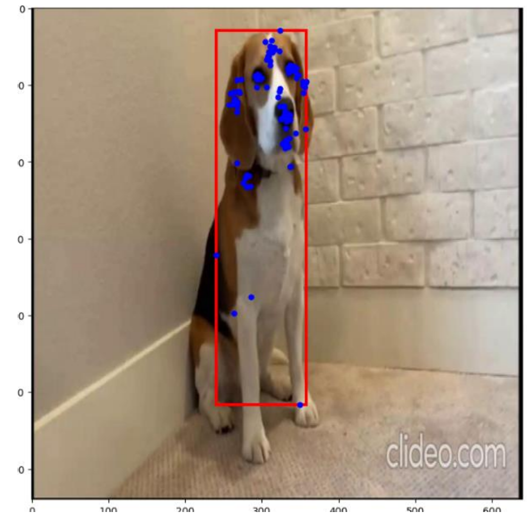
- Bounding Box Calculation: Calculating the bounding box to contain all keypoints accurately was challenging. Initial attempts did not reference the keypoints correctly, resulting in inaccurate bounding boxes. This required iterative adjustments and refinements to the algorithm.

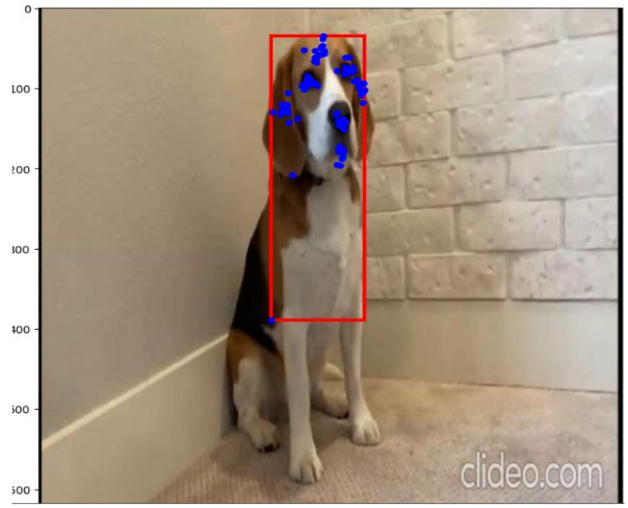
- Referring to Codes: Integrating and adapting external code required careful consideration to ensure compatibility and correctness in the context of the project's goals.

Screenshot:

Trails







Optimized results :