Yolo Report

Name: Rawan Mohamed Said **Id:** 4

➢ Dataset Preparation:

- The dataset chosen from Roboflow is digit captcha which includes images of digits not fully clear like those ones that are normally used to verify that you are not a robot.

- It contains in total 1701 image; they are divided into:
  → 1494 images for training
  → 139 images for validation
  → 68 images for testing

- Data Preprocessing Done:
  → Auto Orient: Applied
  → Resize: Stretch to 640x640
  → Grayscale: Applied
  → Auto-Adjust Contrast: Using Histogram Equalization

- Data Augmentation Applied:
  → Outputs per training example: 3
  → Brightness: Between -10% and +10%
  → Noise: Up to 10% of pixels
  → Cutout: 25 boxes with 5% size each
  → Bounding Box: Blur: Up to 8px
  → Bounding Box: Noise: Up to 8.02% of pixels

- Dataset Architecture:
  digit_captcha-14/
  train/images/ → Containing training images.
  train/labels/ → Containing training labels.

valid/images/ → Containing validation images.
valid/labels/ → Containing validation labels.
data.yaml → Containing YAML file having paths to dataset and the classes' names.

➢ Model Training and Challenges Faced:

- The model used is YOLOv8 due to its computational efficiency and balance between performance and speed
- It was trained on the previous data through 10 epochs and I didn't set an optimizer letting YOLO choose the best optimizer.
- I couldn't try another version of the model or manipulate with the number of epochs or batch size as the training took a very long time to finish one epoch.
- One of the challenges I faced is Ultralytics version conflict as a conflict in YOLO versioning caused a warning about the required version of the Ultralytics library. This was fixed by ensuring the correct version was installed with pip install ultralytics==8.0.196.
- Another challenge I faced is dataset path issues as the data.yaml file pointed to incorrect paths, resulting in a FileNotFoundError. This was resolved by ensuring the correct directory structure and updating the data.yaml file.

➢ Model Performance Metrics:

- Mean Average Precision (mAP) at IoU=0.5: 0.987
- Mean Average Precision (mAP) at IoU=0.5:0.95: 0.814

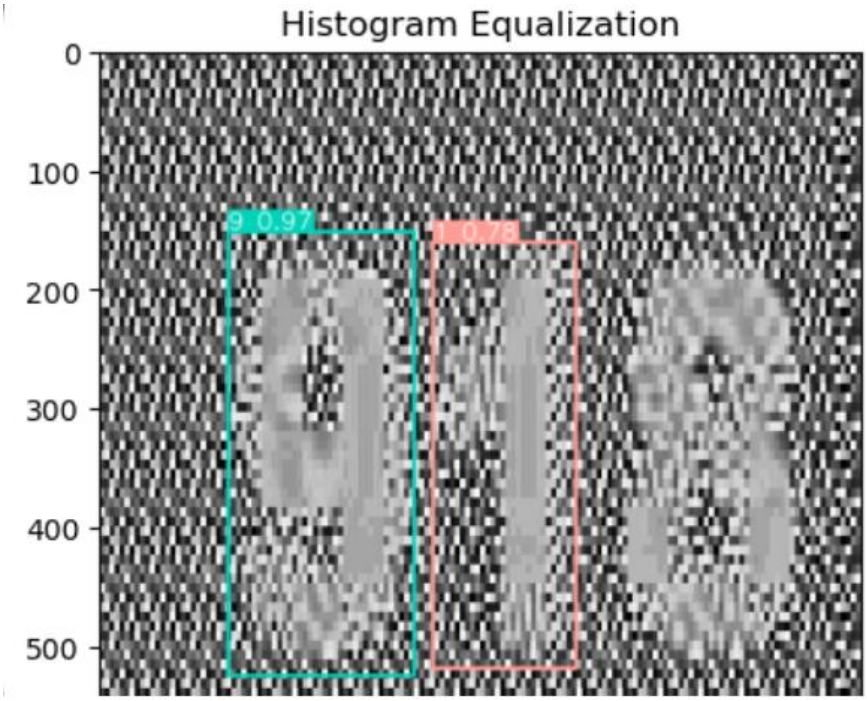
- Precision: 0.982
- Recall: 0.972

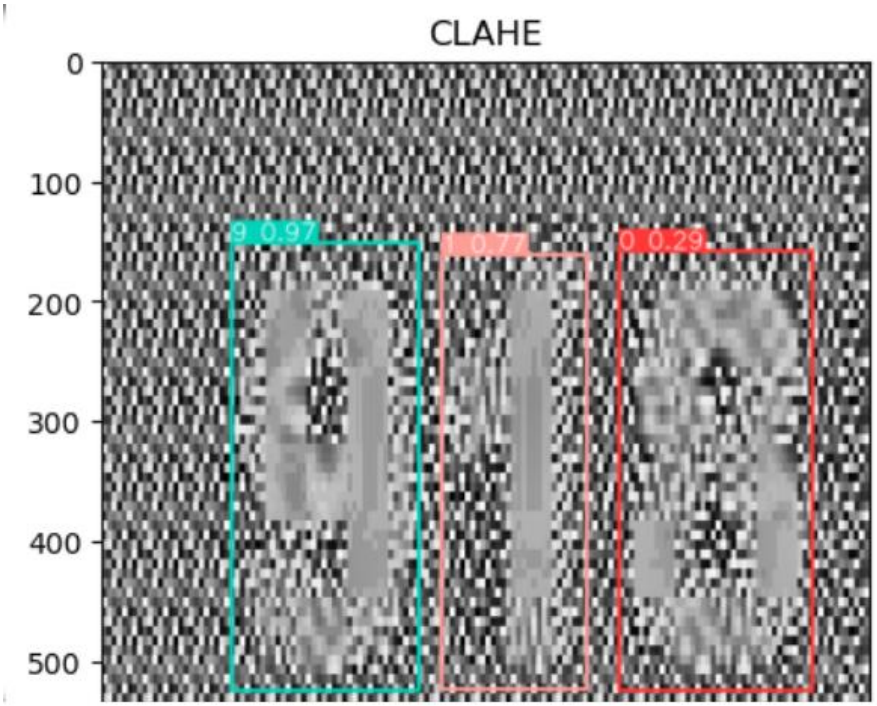
➢ Observations:

- The model demonstrates strong performance with high precision and recall, indicating that it effectively identifies and localizes objects with minimal errors. The high mAP values at both IoU=0.5 and IoU=0.5:0.95 reflect the model's robustness and accuracy in object detection tasks. This means that the model is well-trained and performs reliably across various evaluation metrics.

➢ The Results from testing the model:

- Here I tested the model on an image twice. Once after doing histogram equalization and once after doing CLAHE to show the difference between different image processing techniques.

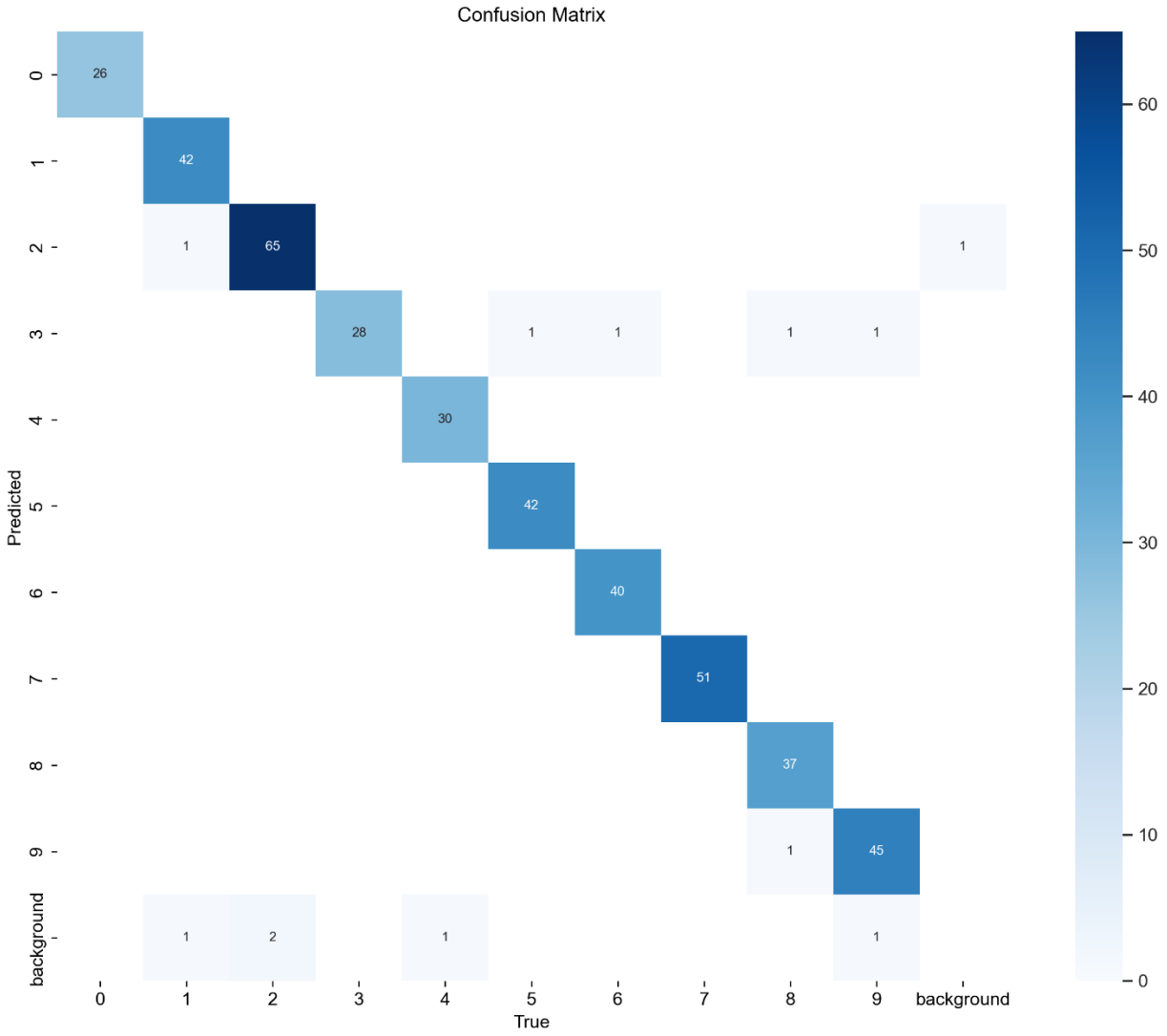




In the first image, the model could successfully detect 2 digits and predict them correctly, while in the second image, the model detected 3 digits, but unfortunately couldn't guess the 3 digits correctly.

From here we can see that using the CLAHE processing technique is better than the histogram equalization technique, but the technique used on the images is the second one which is not the best option.

- The following image shows the confusion matrix after training the model:



➢ Object Detection:

- For that part I used the YOLOv8n trained model

- The method used to track objects is capturing a frame then passing it to the model to the detect the objects in that frame, then saving the resulted frame to a variable to make the final video.

- Here is the resulted video from using that method: https://drive.google.com/file/d/1IwpMmHJ3DlnxcEAjMxN4A6Bh4L2HX1Hz/view?usp=sharing

- About using BYTEtracker, DeepSORT or any other method, I tried running them but for some reason there is an error with opening the video which will be processed; another problem is that the kernel dies whenever I try running that cell.