PURDUE UNIVERSITY®
FORT WAYNE

ECE 66100: Computer Vision Project 3 Report

Mann Raval
(ravam01@pfw.edu)

Course Coordinator:
Dr. Bin Chen

ECE 66100

Problem Statement

Object Detection using the famous object detection model called YOLO. The YOLO is abbreviation of the phrase “You Only Look Once”. Using the Convolutional Neural Network, Canada Goose was to be detected.

Design Methodology

The data set was to be created by the user. Approximately 1000 images of Canada goose were collected and about 9 images were used to validate the model. For testing purposes, The image is given to us in the problem statement were used along with a video. The data set was created using a tool called “simple image” in Python and using open CV library video of Canada goose was taken and some frames of the videos where extracted as images which is used in creating the data set. Using that data set, training of the data was done using YOLOv8 object detection model.

Procedure

The procedure to detect an object using YOLOv8 is as follows:

Data Preparation:

1. Creating a Dataset:
   Gather a group of photos that depict the items you wish to detect. The dataset should be varied, with photographs with varying illumination, backdrops, and object orientations. Ascertain that the photographs are of high quality and in an appropriate format, such as JPEG or PNG.
2. Labelling of images:
   Annotate the photos with bounding boxes using an annotation tool. Each bounding box should include an object to be detected. Give each bounding box a class label that indicates the sort of object it represents.
3. Splitting a Dataset:
   Divide the labelled dataset into three parts: training, validation, and testing. The training set should be the most extensive, as it will be used to train the YOLOv8 model. The validation set is used to assess the model's performance during training and make necessary adjustments to the hyperparameters. The test set is used to evaluate the model's ultimate performance using previously unseen data.

Model Training:

1. Download the model:
   The YOLOv8 model weights can be obtained from the official repository (https://github.com/ultralytics/ultralytics). A pre-trained model can also be used for a similar purpose, such as object detection on the COCO dataset.
2. Setup Configuration:
   Modify the training script included with the YOLOv8 framework to meet your specific requirements. Configure batch size, learning rate, optimizer, and training epochs. Indicate the locations of your training, validation, and test datasets. Change the class names to correspond with the labels you used for picture annotation.

3. Model Training Methodology: Run the training script on the prepared dataset to train the YOLOv8 model. Keep track of the training process by looking at loss and accuracy data. When the model achieves satisfactory performance or after a sufficient number of epochs, stop training.

Inference:

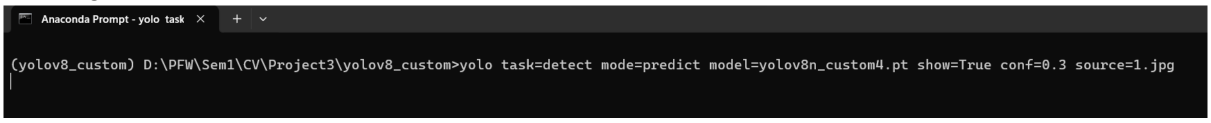
1. Loading Models:
   Load the trained YOLOv8 model weights from the training checkpoint file.
2. Input of an image or video:
   Provide the image or video that will be used for object detection. Images can be loaded in a variety of formats, including JPEG, PNG, and TIFF. Individual frames from videos can be retrieved and processed progressively.
3. Detecting Objects:
   For object detection, provide the image or video frame to the YOLOv8 model. For each object detected, the model will process the data and generate predictions.
4. Visualization:
   In the picture or video frame, draw bounding boxes around the discovered items. Show the class labels and confidence scores for each discovered object.

Implementation

Implementation of this project was done in Python using the tools of Google colab and YOLOv8. For training of the data the data set was created using Python And open CV and a tool called simple image. The training of their data was done on Google colab. After several iterations the weight were downloaded and run on Anaconda prompt using YOLO. To run the training file just upload the data set and and the Python file on Google Drive mount the Google Drive on the Python file and run the data. After running the code on Google colab download the weight file and run YOLO using command line interface on Anaconda prompt.

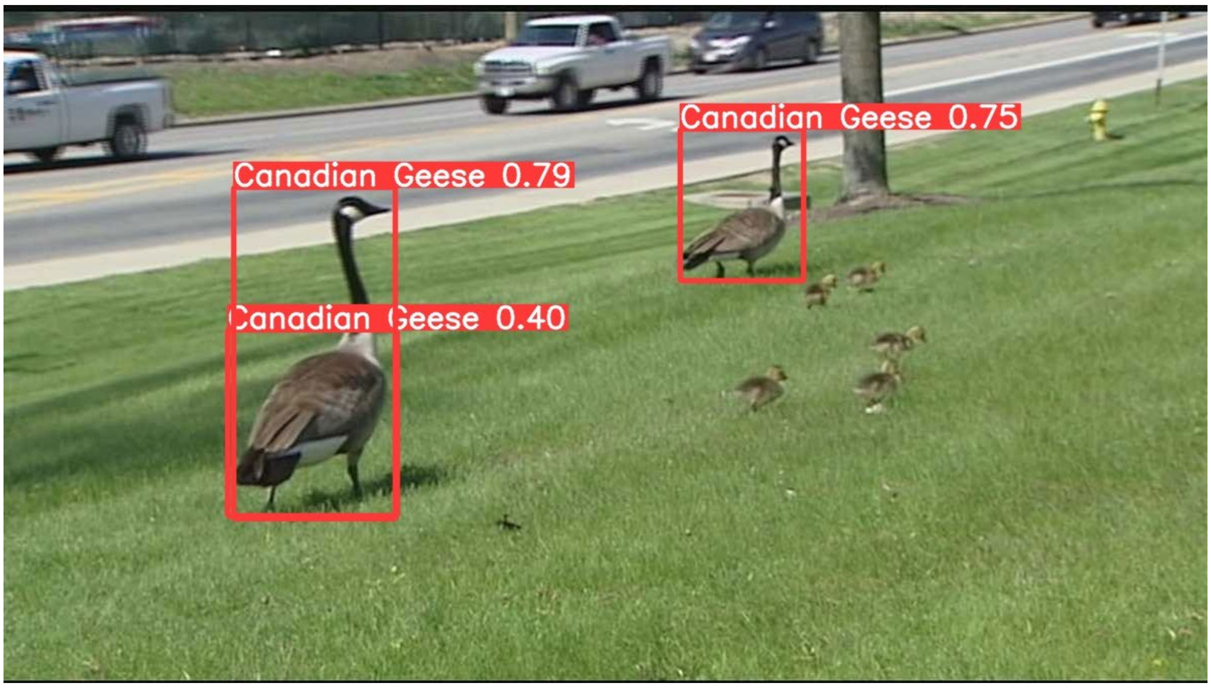
Results

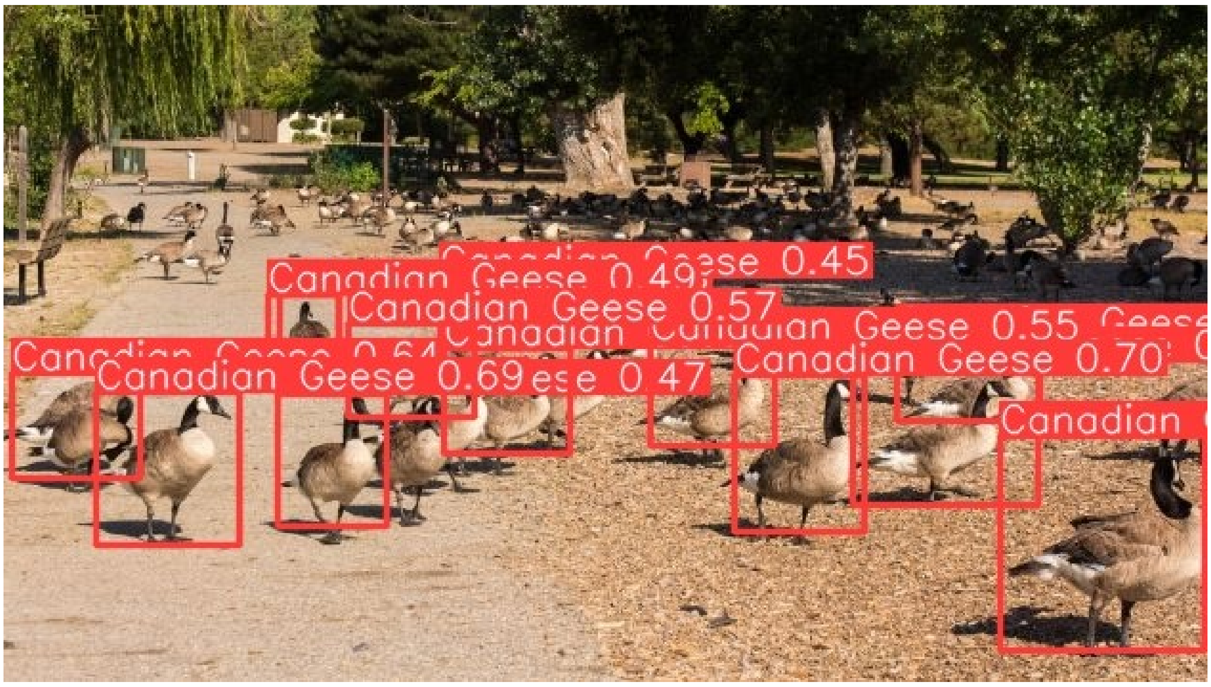
To get the results the following command is used in the Command Line Interface of the Anaconda Prompt after installing the YOLO model in Anaconda. Following command must be run to get the results:



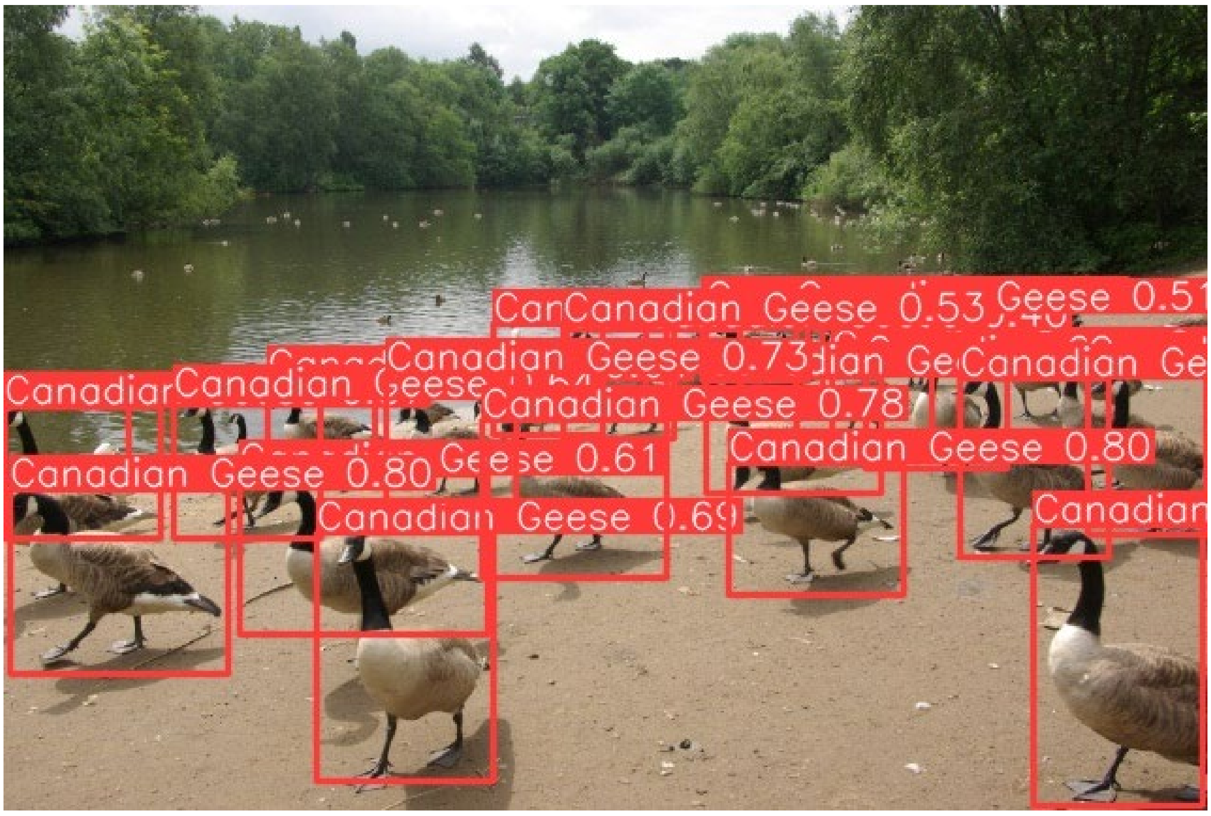
```
(yolov8_custom) D:\PFW\Sem1\CV\Project3\yolov8_custom>yolo task=detect mode=predict model=yolov8n_custom4.pt show=True conf=0.3 source=1.jpg
```

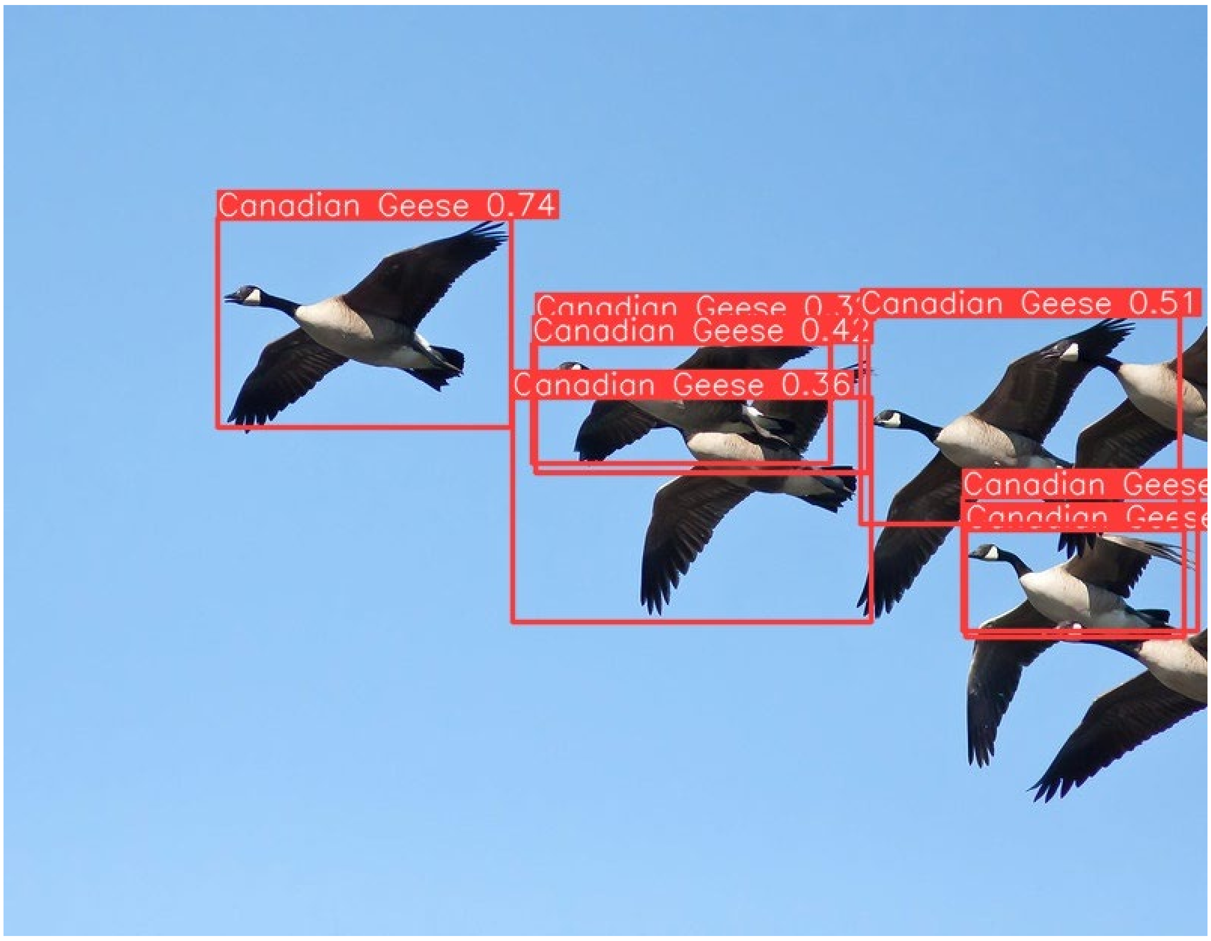
The results are as follows:

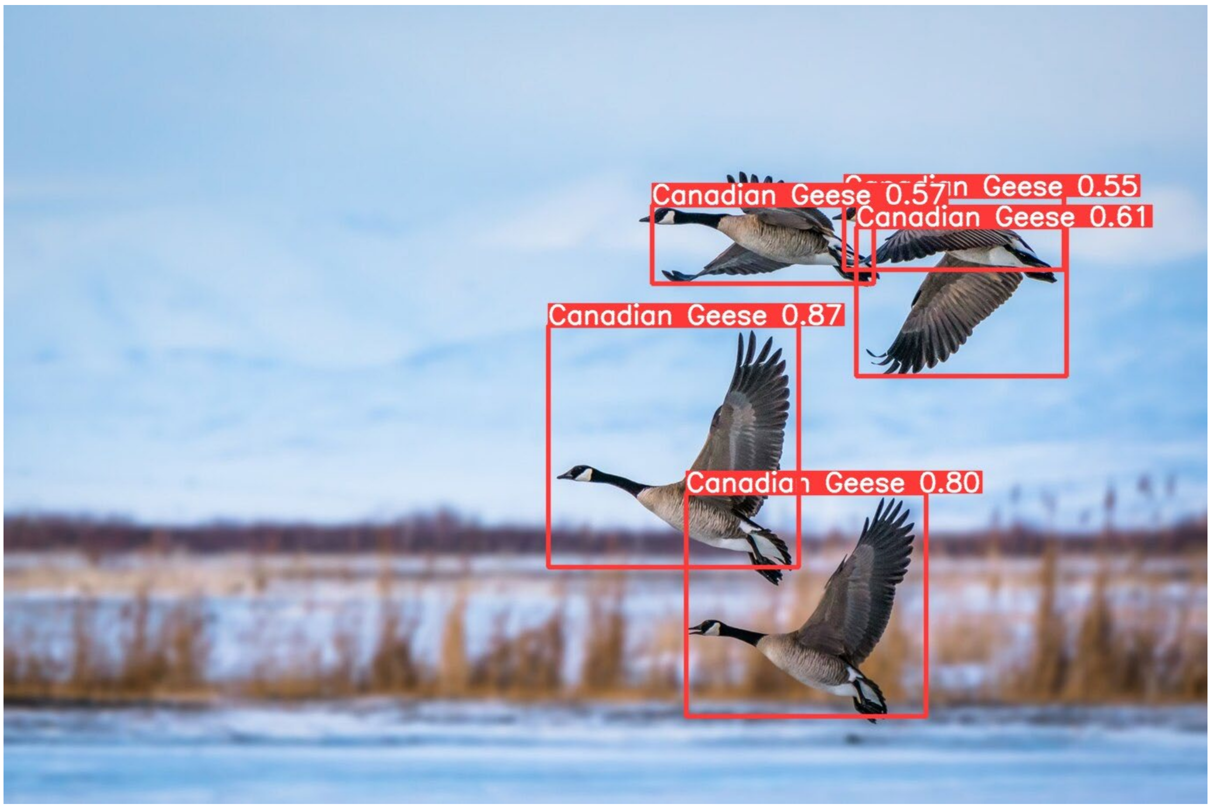

Canadian Geese 0.79
Canadian Geese 0.75
Canadian Geese 0.40

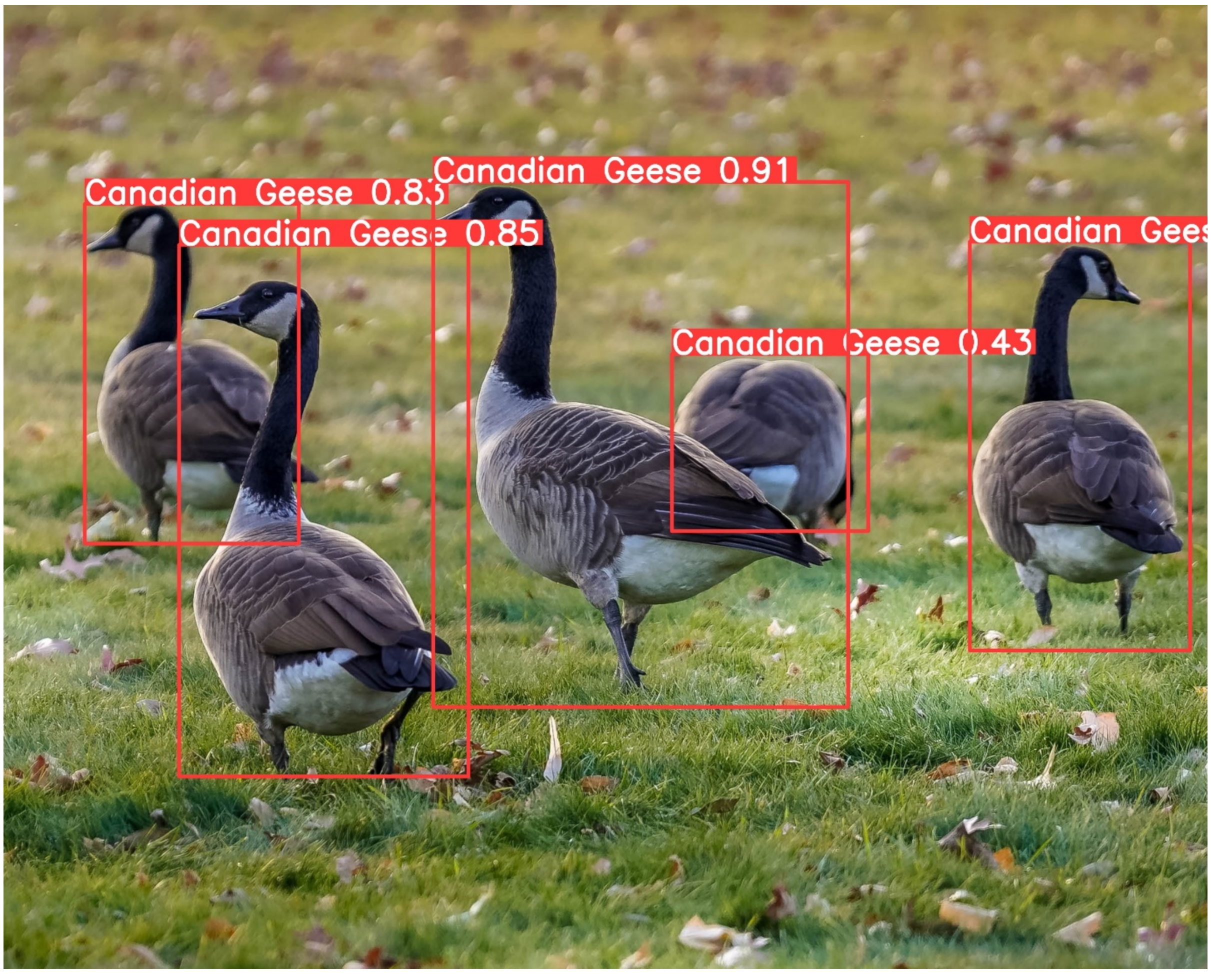

Canadian Geese 0.45
Canadian Geese 0.49
Canadian Geese 0.57
Canadian Geese 0.55
Canadian Geese 0.64
Canadian Geese 0.69
0.47
Canadian Geese 0.70
Canadian

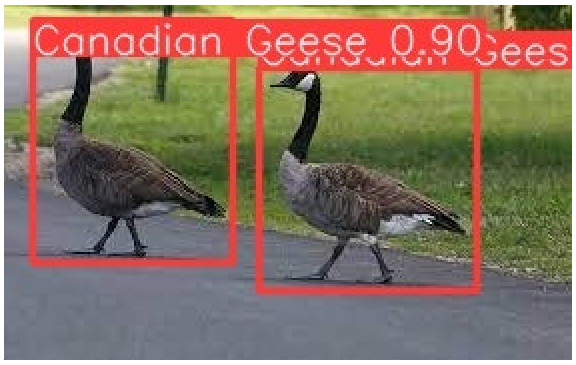
Canadian Geese
Canadian Geese 0.71
Canadian Geese 0.69
Canadian Geese 0.80
Canadian Geese 0.83
Canadian Geese 0.80
Canadian Geese
Canadian Geese 0.64
Canadian Geese 0.68
Canadian Geese 0.75
Canadian Geese 0.81
Canadian Geese 0.44

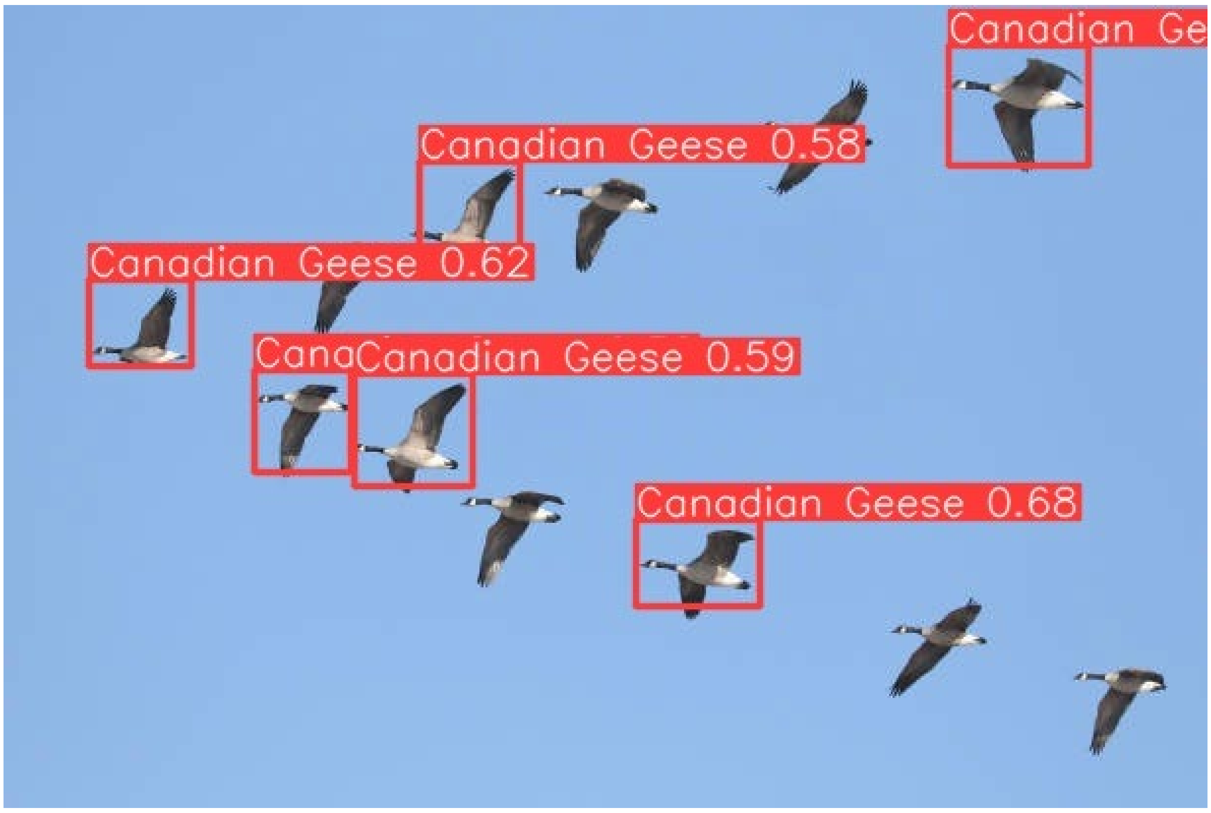

Canadian Geese 0.53
Geese 0.51
Canadian Geese 0.73
Canadian Geese 0.78
Canadian Geese 0.80
Canadian Geese 0.80
Geese 0.61
Canadian Geese 0.69

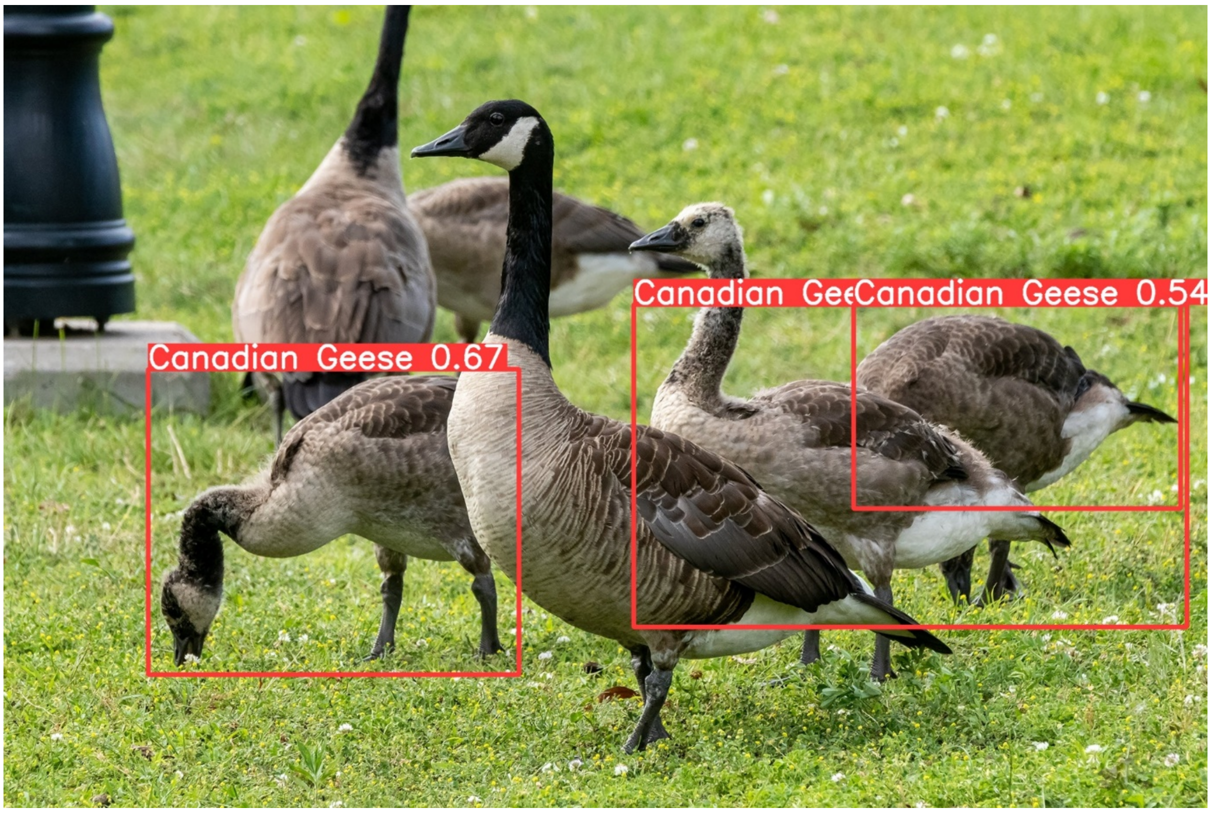

Canadian Geese 0.74
Canadian Geese 0.3
Canadian Geese 0.42
Canadian Geese 0.36
Canadian Geese 0.51
Canadian Geese
Canadian Geese

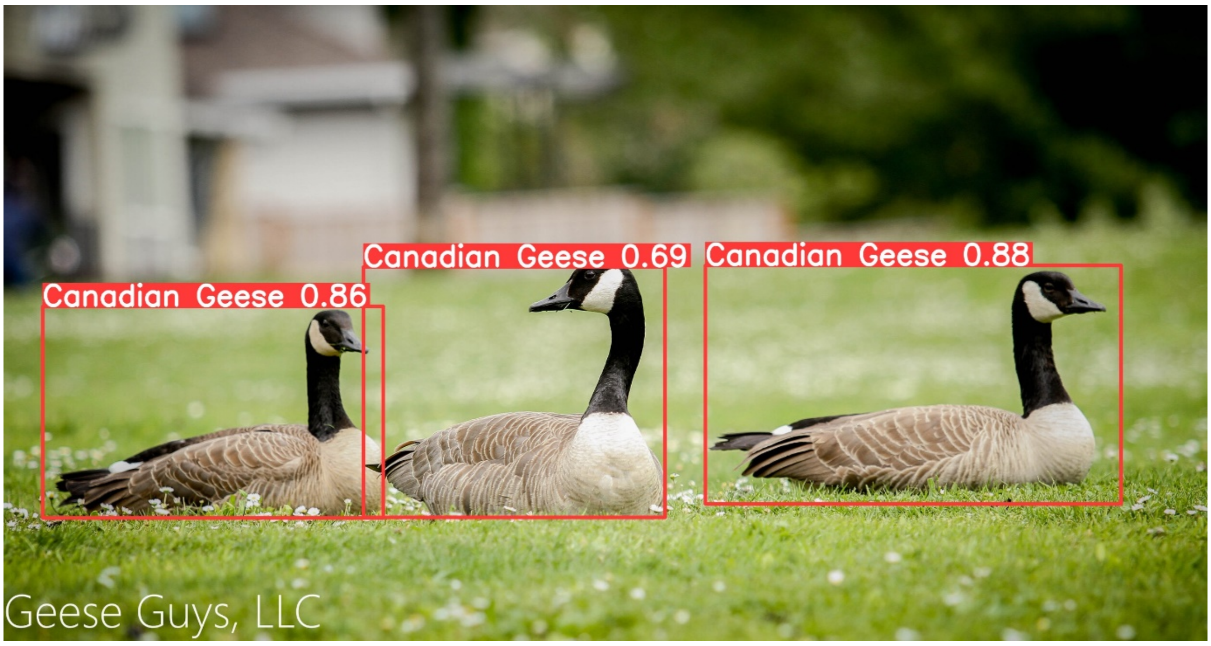

Canadian Geese 0.57
Canadian Geese 0.55
Canadian Geese 0.61
Canadian Geese 0.87
Canadian Geese 0.80


Canadian Geese 0.83
Canadian Geese 0.85
Canadian Geese 0.91
Canadian Geese 0.43
Canadian Gees


Canadian Geese 0.90
Gees


Canadian Ge
Canadian Geese 0.58
Canadian Geese 0.62
Cana
Canadian Geese 0.59
Canadian Geese 0.68


Canadian Gee
Canadian Geese 0.54
Canadian Geese 0.67



The video is attached in the zip file.

Conclusions

- This was a challenging project.
- Prior knowledge of Image Processing would have been an advantage.
- A review of Python was done.
- Implementation of algorithm such as CNN was done.
- The implementation could be done using Python and CNN.
- The model used has a significant impact on the accuracy of the detection.

References

- https://github.com/ultralytics/ultralytics
- https://learnopencv.com/train-yolov8-on-custom-dataset/
- https://blog.roboflow.com/how-to-train-yolov8-on-a-custom-dataset/