**POND FISH DETECTION MODEL**

**Team details:**

Team name: STATIS

Team members: Thanush A A(team leader), Dilli B

**Summary:**

We propose an approach, named CLAHE-yolov8s, for pond fish detection in the context of the DePondFi'23 challenge. As part of this challenge, two models were developed: MSR-yolov8s and CLAHE-yolov8s. Through evaluation on the validation dataset, the performance of CLAHE-yolov8s was found to be slightly higher than that of MSR-yolov8s. Morever, the CLAHE-yolov8s model exhibited slightly faster image processing speed compared to the MSR-yolov8s model.

Consequently, both models have been made available in the project folder, enabling users to choose the model that aligns with their preferences.

Additionally, the project encompasses scripts that facilitate the generation of bounding box coordinates in the yolov5 annotations format, as well as the utilization of the models for fish detection for images and videos, respectively. Detailed explanations of the proposed models, the required dependencies, and instructions for obtaining bounding box coordinates, and guidelines for performing detections on image and video files are presented in the subsequent sections.

**FISH DETECTION MODELS:**

The models proposed in this project are:

(i) MSR-YOLOv8s

(ii) CLAHE-YOLOv8s

The models were trained using the training dataset provided in the DePondFi challenge.

**1. MSR-YOLOv8s:**

Multi Scale Retinex (MSR) is an image enhancement algorithm based on the Retinex theory, which explains human colour perception. The theory is based on the following assumptions:

i. The colour perceived by the eyes is a result of the interaction of light and matter

ii. Each colour region is composed of three primary colours of a given wavelength: red, green, and blue

iii. The three primary colours determine the colour of each unit area

According to retinex theory, the colour of an object is not affected by the intensity of the reflected light, but on the constancy of colour perception.

The MSR-YOLOv8s model processes the input images using MSR algorithm before feeding it to the yolov8s architecture for fish detection.

**2. CLAHE-YOLOv8s:**

Contrast Limit Adaptive Histogram Equalization (CLAHE) is an image enhancement algorithm that is a variation of the adaptive image enhancement (AHE) algorithm. The CLAHE algorithm consists of three parts: The tile generator, Histogram equalizer, and the bilinear interpolator. The given input image is split into smaller units known as tiles. These tiles undergo histogram equalization, which is a five-step computation process. The resultant tiles are then joined using bilinear interpolation. The algorithm can generate output images with increased contrast.

The CLAHE-YOLOv8s model processes the input images using CLAHE algorithm before feeding it to the yolov8s architecture for fish detection.

A comparison of the performance of the models are provided in the table below. The mean average precision values were based on the validation dataset.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Model | mAp50 | F1 score | Image preprocessing[[1]](#footnote-1)  (enhancement)time | Inference time | Total time |
| MSR-YOLOv8s | 0.964 | 0.9288 | 0.0218 | 1.256s | 1.277s |
| CLAHE-YOLOv8s | 0.970 | 0.929 | 0.0185 | 1.165s | 1.183s |

**DEPENDENCIES REQUIRED:**

Python 3.9.7 or higher

Pip 23.0.1 or higher

**Python modules required (can be installed using pip):**

matplotlib>=3.2.2

opencv-python>=4.6.0

Pillow>=7.1.2

PyYAML>=5.3.1

requests>=2.23.0

scipy>=1.4.1

torch>=1.7.0

torchvision>=0.8.1

tqdm>=4.64.0

numpy>=1.21.6

ultralytics (latest version)

natsort>=8.3.1

xlwt>=1.3.0

**HOW TO GENERATE THE BOUNDING BOX COORDINATES:**

1. Open the command line interface of your Operating System in the root directory of the project

2. Type “python image\_text\_annotation.py”

3. A prompt will appear on the image enhancement algorithm to be applied (MSR/CLAHE). Type the algorithm that you want to use

4. Type the directory containing the image(s) using ‘/’ to separate the folders instead of ‘\’. Make sure that the location specified in the prompt ends with ‘/’ to avoid errors

5. Wait for the enhancement and detection to complete

6. The coordinates would be saved in the results folder in yolov5 annotations format

**HOW TO GENERATE IMAGE WITH BOUNDING BOX OUTPUT:**

1. Open the command line interface of your Operating System in the root directory of the project

2. Type “python image\_detection.py”

3. A prompt will appear on the image enhancement algorithm to be applied (MSR/CLAHE). Type the algorithm that you want to use

4. Type the directory containing the image(s) using ‘/’ to separate the folders instead of ‘\’. Make sure that the location specified in the prompt ends with ‘/’ to avoid errors

5. Wait for the enhancement and detection to complete

6. The directory containing image(s) with bounding boxes would be saved in the ‘runs/detect/’ folder

**HOW TO GENERATE VIDEO WITH BOUNDING BOX OUTPUT:**

1. Open the command line interface of your Operating System in the root directory of the project

2. Type “python video\_detection.py”

3. A prompt will appear on the image enhancement algorithm to be applied (MSR/CLAHE). Type the algorithm that you want to use

4. Type the location of the video file

(e.g.): path/to/video.extension

5. Wait for the enhancement and detection to complete

6. The directory containing image(s) with bounding boxes would be saved in the ‘runs/detect/’ folder

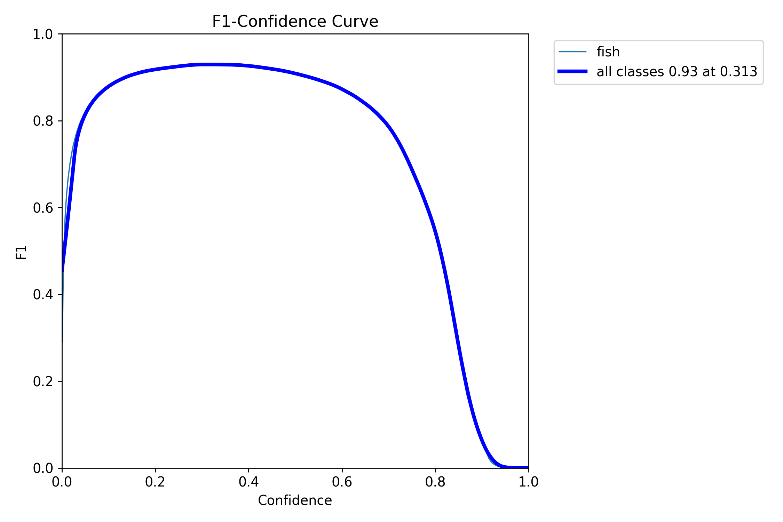
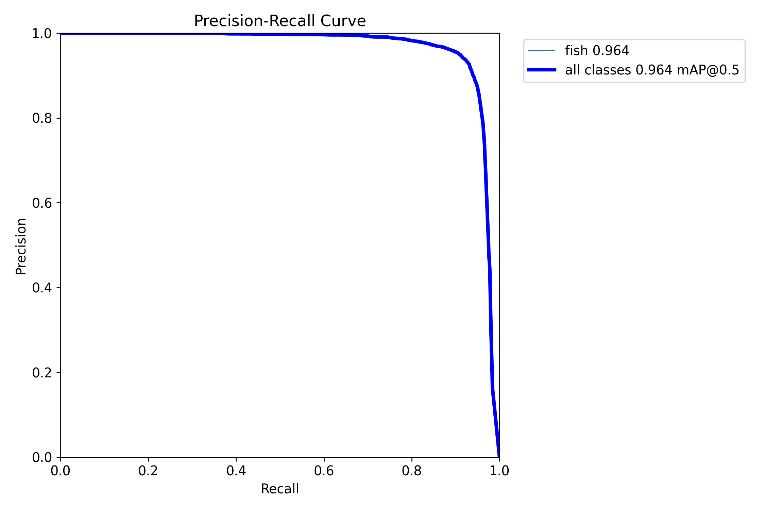
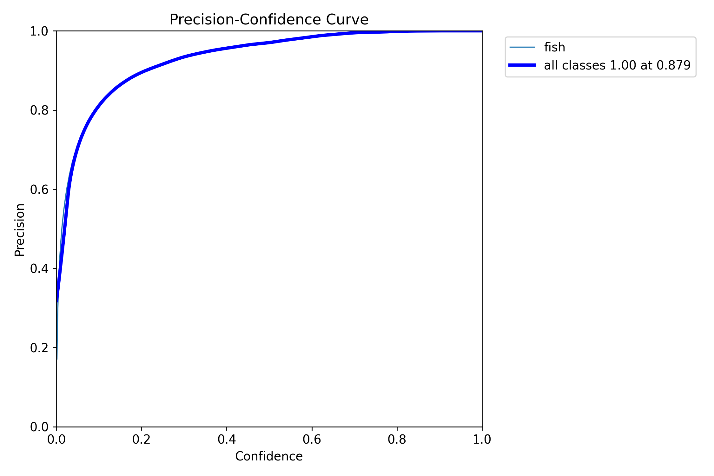
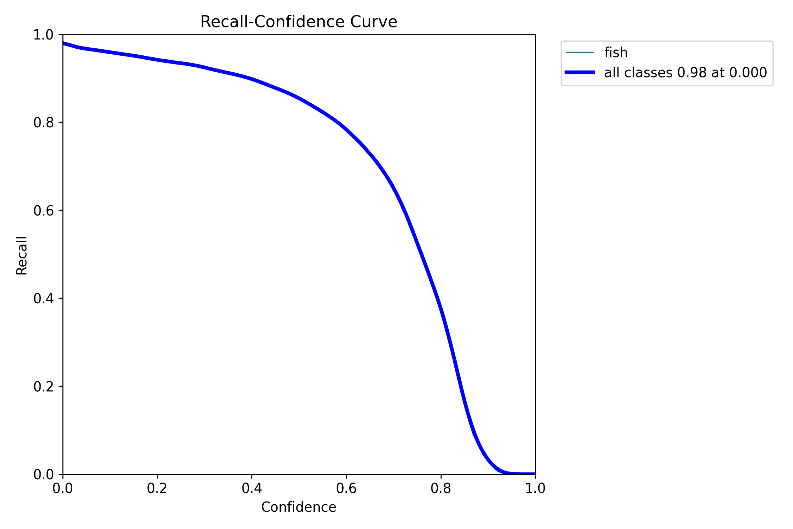
7. Make sure that the video in the detections directory is deleted before running the code again

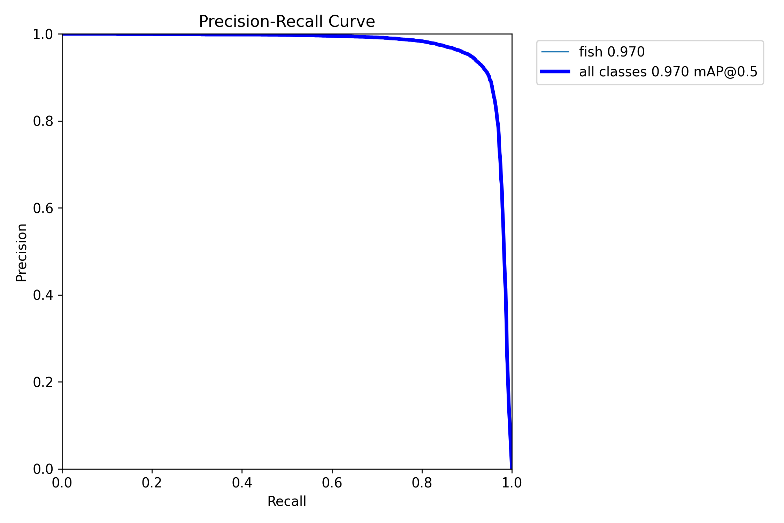
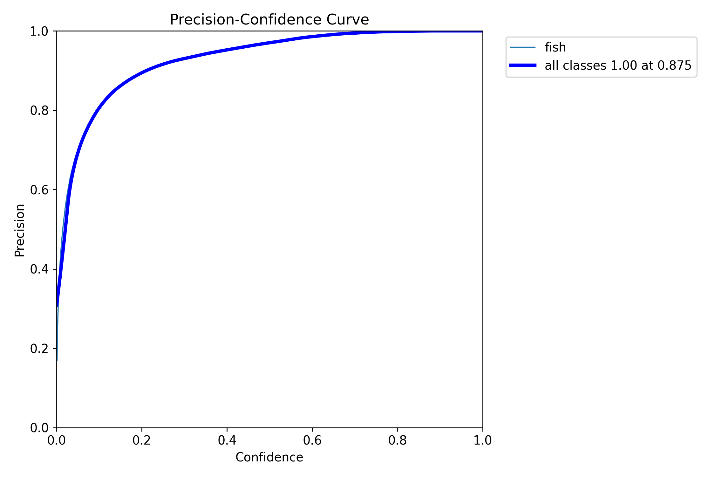
**TEST DATASET RESULTS AND ANNOTATIONS:**

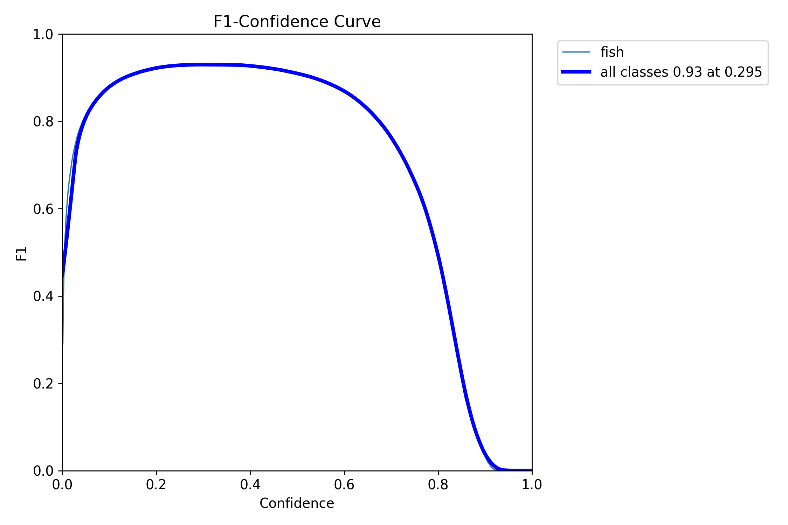
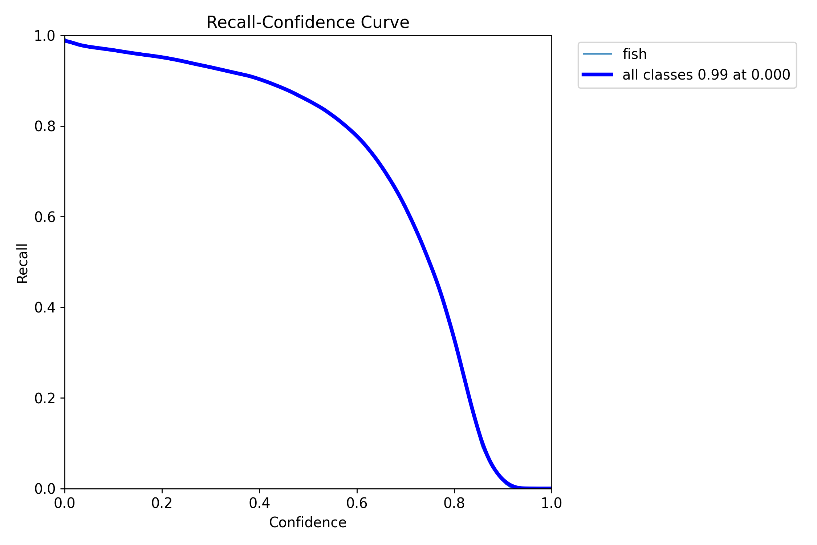
The test dataset results of the individual models containing the bounding box images and annotations are available in the test\_results directory

**TRAINING RESULTS:**

**MSR-yolov8s:**

****

**CLAHE-yolov8s:**

****

1. It refers to the time taken to preprocess an image/frame using the image enhancement algorithms (MSR/CLAHE) before feeding it into the model [↑](#footnote-ref-1)