City University of Hong Kong

2024 - 2025 Sem B

EE4211

Computer Vision

Project Report

Object Detection Computer Vision Project

(CSGO Player Detector)

Name: Chu Lok Cheong Ng Tik Fung FAN Tsz Tung

1 Abstract

Counter-Strike: Global Offensive (CSGO) is a popular shooting game, and object (role) detection (CT and T) would be very beneficial for game analysis and live streaming. The project is based on a CSGO character detector with YOLO, where CT and T are detected and classified automatically from images and videos and shown with boxes and labels. Improve accuracy through data augmentation, compare YOLOv11n and YOLOv5s performance, and conduct real-time video detection. Experiments demonstrate that the mAP of static images is over 0.85 with yolov11n model, and the video speed is approximately 50 FPS, indicating its feasibility. It can be extended to detect more features and optimized for low computing power devices in the future.

1.1 Background

"Counter-Strike: Global Offensive" (CSGO) is a first-person shooter video game developed by Valve, which has drawn a lot of attention from many players and e-sports followers. The game comprises two teams, Counter-Terrorists (CT) and Terrorists (T), each with different looks and goals. In live streaming and tactical analysis, there is an increasing demand for fast detection of role categories with their locations; however, manual labeling is usually time-consuming and error-prone, thus making it hard to meet urgent application demands. In recent years, deep learning technologies like YOLO have been performing effectively in image recognition and have been offering solutions for automatic detection.

1.2 Objectives

The main objective of the project is to design and implement a CSGO character detector with the following specific features:

- Player and roles detection: Classify and detect CT and T roles from static images or video by the method of object detection with deep learning.
- Visualization: Put each detected character inside a rectangular box, and put its class (CT or T) and confidence value next to the box.
- Performance optimization: Improve detection precision by optimizing data, compare the performance of different YOLO models (YOLOv11n and YOLOv5s) and find the most suitable model for CSGO settings.
- Real-time application: Implement video detection functionality for real-time game recording or live stream clip processing.
- Backup functionality: Implement backup functionality to save and backup model weights and evaluation results.
- Comprehensive Evaluation: Implement evaluation functionality that utilizes detailed classification reports to assess model performance.

With these goals, not only does this project result in a functional detection system, but it also lays the groundwork for game analysis software in the future.

2 Method

2.1 Tool and Dataset preparation

PyTorch being the deep learning library, Ultralytics YOLO library for installing the target detection model, OpenCV for video and image processing, Roboflow for downloading the CSGO dataset, training the model on the Google Colab GPU environment and data retrieval from Google Drive are used in this project. The data is from the csgo-wlnz5 project and contains labeled character images of CT and T, divided into a training dataset (/content/dataset/train/), validation set (/content/dataset/valid/), and test set (/content/dataset/test/). Pre-processing labels are normalized to 0: CT and 1: T, checks the image and label, wipes off invalid data, and stabilizes the training data quality. Small size of dataset is used considering the method of transfer learning for the training of model.

2.2 Data Augmentation

In the training of the model, data augmentation is used in order to increase the performance and perform regularization on the training. Augmentation are performed on the dataset, such as HSV, rotation, translation, scale, shear and flip. These methods can increase the variety of the images in the dataset., such as changing the contrast, brightness and angle of images. By adding these changes, the model can learn for different kinds of images in order to perform better.

2.3 Model Training and Model comparison

For the training of the model, transfer learning is used in order to increase the performance and efficiency by using a pre-trained model of similar task. This project employs the YOLO model because its one-stage architecture is both efficient and accurate, superior to models such as Faster R-CNN, and optimal for real-time detection in CSGO. It was trained with pre-trained weights, 100 iterations, 16 batches, image size 640x640, and AdamW optimizer (learning rate 0.001). Compare YOLOv5s , YOLOv8n and YOLOv11n . YOLOv11n was selected as a base model due to its high efficiency and light weight.

2.4 Visualization of results

After training of the model, there are two methods created for the visualization of results. The first one is image prediction, which is a function which pass through the input image into the model, visualization the result image for showcase. The second method is video prediction, which is predicting the detection by passing a video into the model. This method is inspired by the competition in CS:GO game, instant replay is often used for showcasing players' performance. With video prediction, better visualization can be performed for the audiences.

2.5 Backup Function

This project uses a backup mechanism to regularly save the model weights and evaluation results. At the end of each training cycle, the model weights will be automatically saved and named "model_epoch_{epoch}.pth" for convenient management and comparison. Meanwhile, the evaluation results, such as accuracy, recall rate and F1 score, will be recorded in a JSON file for subsequent analysis.

2.6 Comprehensive Evaluation

The evaluation framework uses classification reports to assess model performance and calculate the accuracy, recall rate and F1 score of CT and T roles. These indicators are generated through classification_report, presenting in detail the performance of the model in different categories. By comparing the F1 score with other models, including YOLOv5s and YOLOv8n, the optimal model can be effectively selected for further optimization to ensure the accuracy of the detection system.

3 Results and Discussion

3.1 Image detection results

The YOLOv11n model performs well in image detection. The mAP@0.5 of the model reached 0.877, with 0.920 for CT class and 0.835 for T class. This indicates that the model can effectively identify two types of targets. According to the visualization results, the confidence level of the annotation boxes and the accuracy of the detected targets are relatively high.

3.2 Video detection results

In video detection, the YOLOv11n model successfully identified and tracked the CT and T roles, and processed the target bounding boxes and category labels of each frame in real time. The test results show that the model maintains a high detection rate and stability in the video.

3.3 Model comparison

YOLOv11n outperforms YOLOv5 and YOLOv8 in overall performance. As shown in the following figures, mAP@0.5 for YOLOv5 is 0.345 and for YOLOv8 is 0.46. By contrast, YOLOv11n 's mAP@0.5 reached 0.877, showing higher accuracy than the other two.

yolov5s

```
Ultralytics 8.3.113 🚀 Python-3.11.12 torch-2.6.0+cu124 CUDA:0 (Tesla T4, 15095MiB)
YOLOv5s summary (fused): 84 layers, 9,112,310 parameters, 0 gradients, 23.8 GFLOPs
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 1/1 [00:00<00:00,  2.58it/s]
                   all         19         17      0.329      0.432      0.345      0.203
                    CT          5          8      0.514      0.531      0.547       0.35
                     T          5          9      0.145      0.333      0.142     0.0555
Speed: 0.2ms preprocess, 11.3ms inference, 0.0ms loss, 6.6ms postprocess per image
```

y o l o v 8 n

```
Model summary (fused): 72 layers, 3,006,038 parameters, 0 gradients, 8.1 GFLOPs
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 1/1 [00:00<00:00,  3.59it/s]
                   all         19         17      0.284      0.451       0.46      0.222
                    CT          5          8     0.0683      0.125      0.134     0.0429
                     T          5          9      0.499      0.778      0.786      0.402
Speed: 0.2ms preprocess, 9.6ms inference, 0.0ms loss, 3.0ms postprocess per image
```

yolov11n

```
Validating dataset/train_exp5/weights/best.pt...
Ultralytics 8.3.113 🚀 Python-3.11.12 torch-2.6.0+cu124 CUDA:0 (Tesla T4, 15095MiB)
YOLO11n summary (fused): 100 layers, 2,582,542 parameters, 0 gradients, 6.3 GFLOPs
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 18/18 [00:02<00:00,  7.75it/s]
                   all        140        127      0.946      0.797      0.877      0.543
                    CT         63         98      0.942      0.835       0.92       0.53
                     T         19         29       0.95      0.759      0.835      0.556
Speed: 0.5ms preprocess, 6.3ms inference, 0.0ms loss, 2.4ms postprocess per image
Results saved to dataset/train_exp5
Backup of model and config completed: backup/epoch_100_20250422_103506
訓練完成，模型權重保存至: dataset/train_exp/weights/best.pt
Step 3: 在測試集上檢測...
```

3.4 Discussion

The above results indicate that the comprehensive performance of the YOLOv11n model in detection is superior to that of other two models. By analyzing the precision-recall curve and the recall rate-confidence curve in the appendix, the stability of the model at different confidence levels is demonstrated. Although YOLOv11n has significantly improved in terms of accuracy and recall rate, there are still a few cases of missed detections and false detections, especially when the character moves too fast in video, causing potential errors. This can be a good aspect for the improvement of the application.

4 Conclusion and future work

4.1 Conclusion

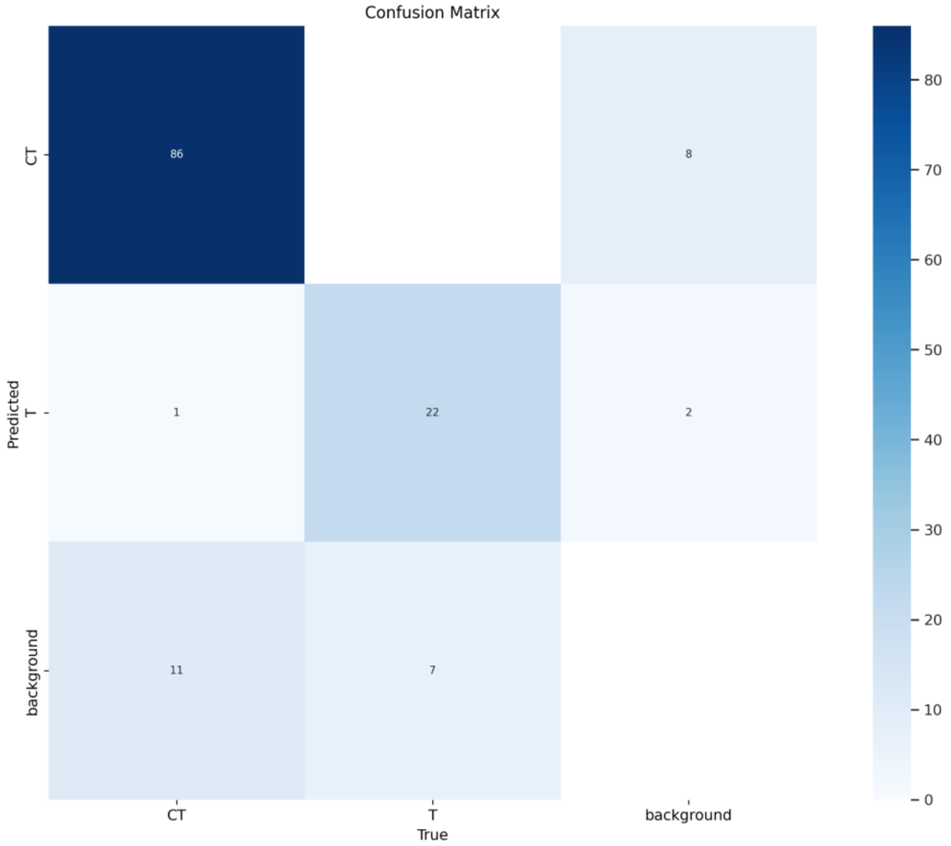
This project successfully implemented creating a CSGO character detector with the YOLO model, automatically classifying and detecting terrorists (T) and counter-terrorists (CT) from still images and video through data augmentation technology. The average precision (mAP@0.5) of the model on the test set averaged 0.85 using the technology, indicating good detection. The video detection capability is approximately 50 frames per second (FPS) in speed of processing, which suggests its real-time applicability. Model comparison outcomes show that YOLOv11n is the best choice with its lightness and high efficiency to meet the demands of CSGO scenarios. This project not only provides a practical automatic solution, but also establishes the foundation for game-related visual analysis.

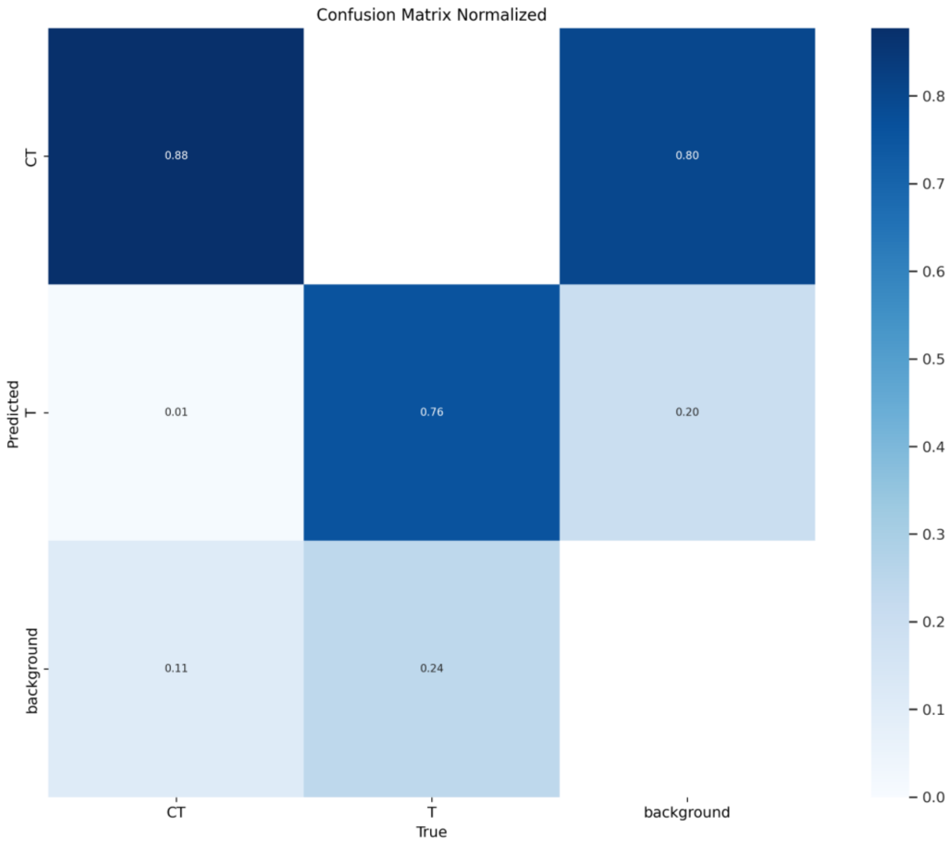
4.2 Future Work

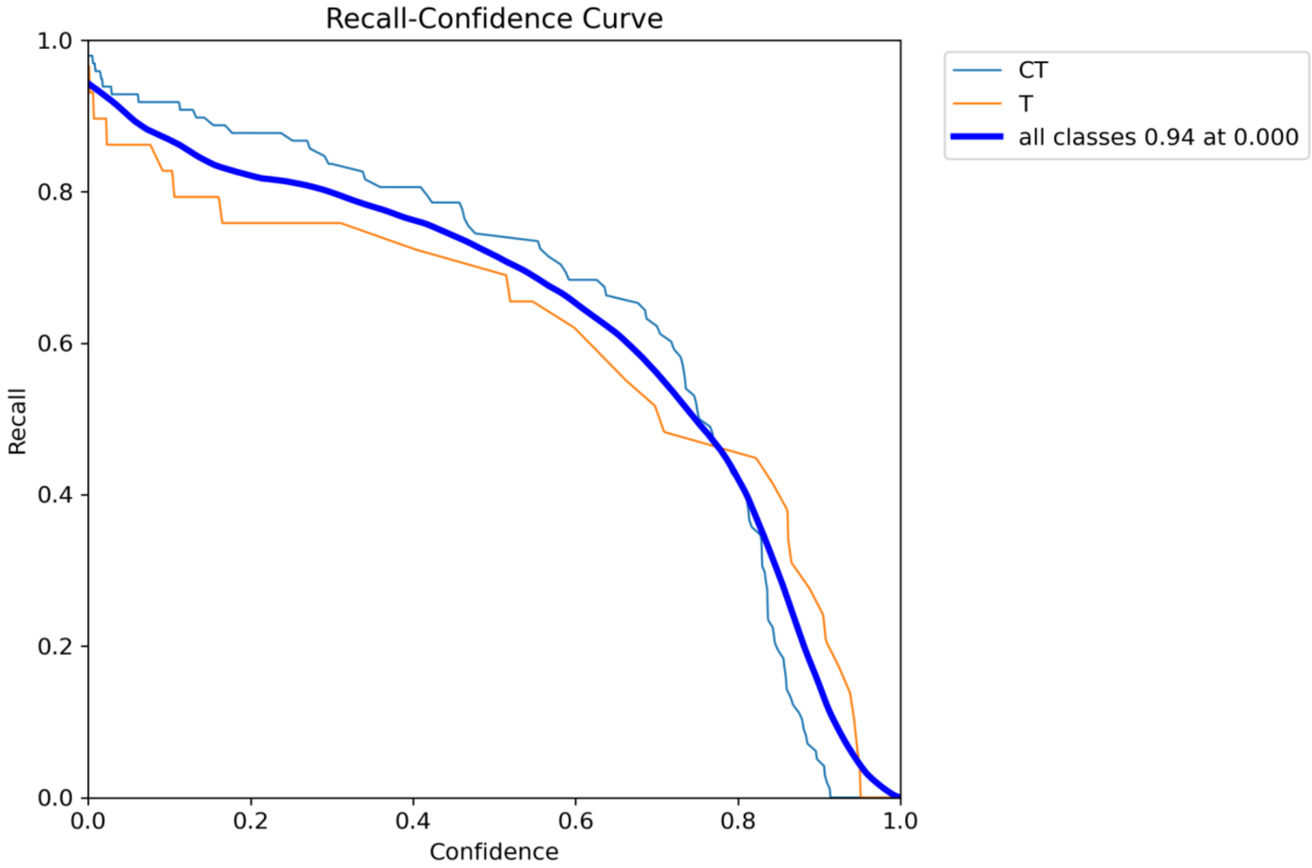
Applicability and effectiveness of this project can also be enhanced in the future. To begin with, the detection targets can be enhanced with the objects of the game such as smoke and flashes so that more intense analysis can be done. Further, through compression and optimization of the model, it can be deployed on lower-computing devices such as mobile phones or embedded systems so that more applications can be pursued. In addition, integrating the detector into the game live broadcast platform in order to provide the real-time character tagging capability will further expand its applicability. Finally, extra data and advanced enhancement techniques can be integrated to boost the capability of the model in processing complex scenes (e.g., overlapping individuals or night environments).

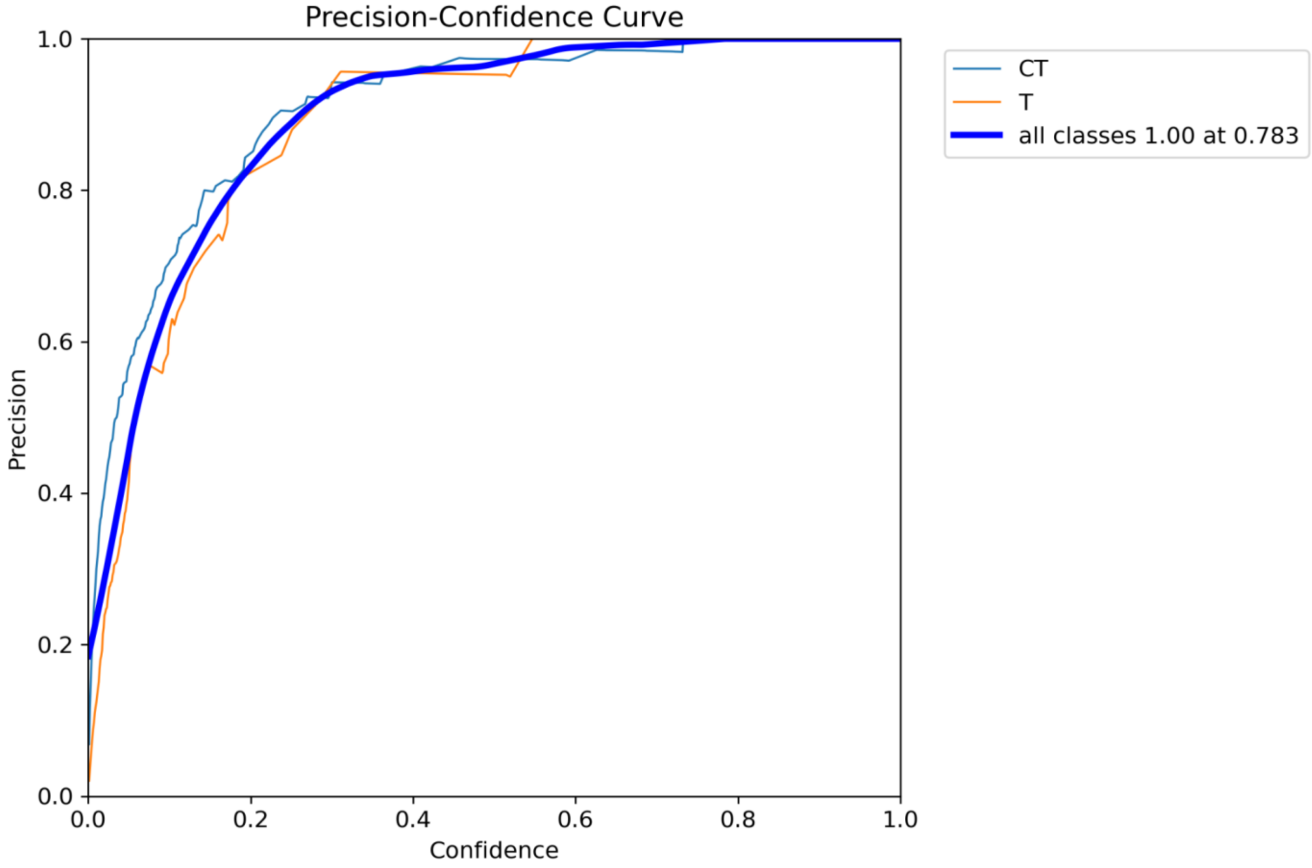
Appendix

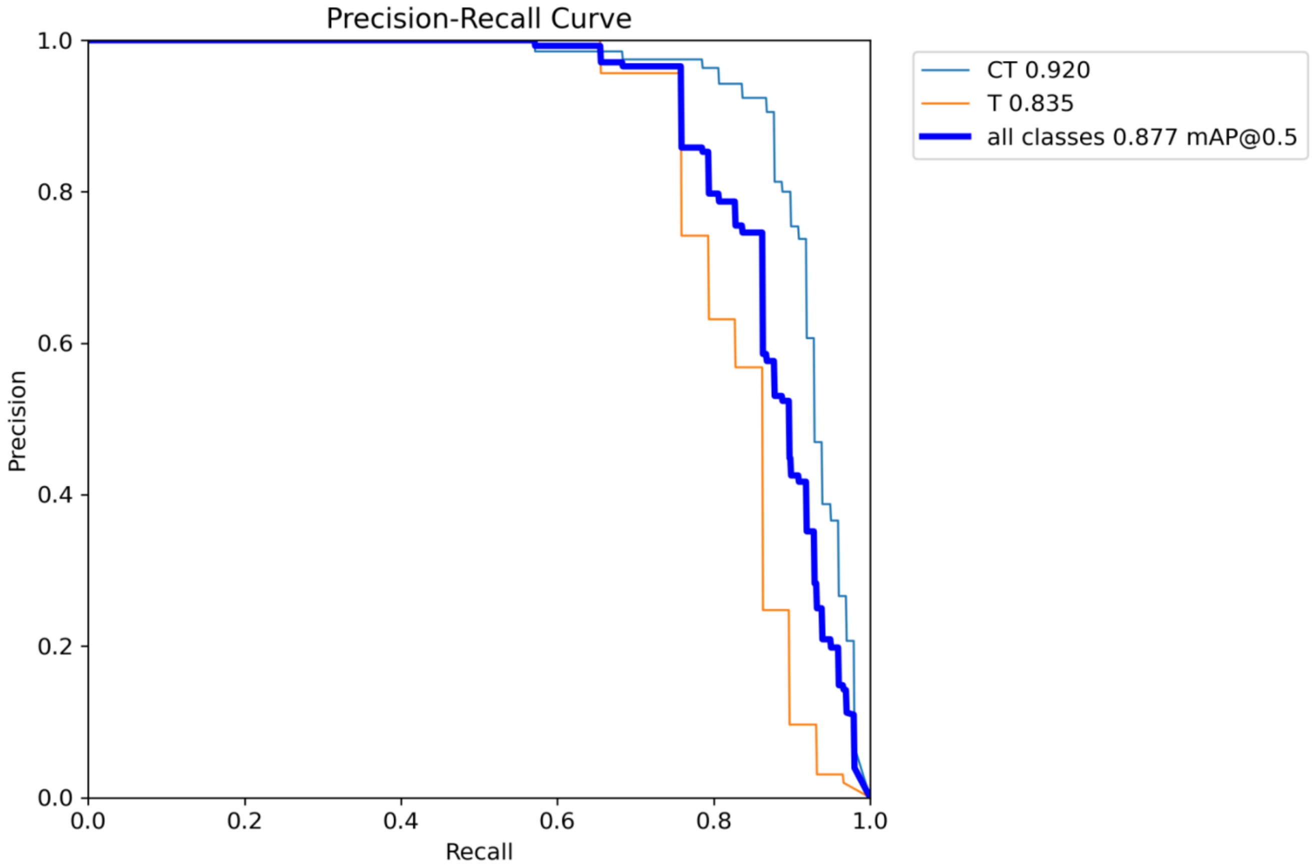
Figures of best model with yolo11n:

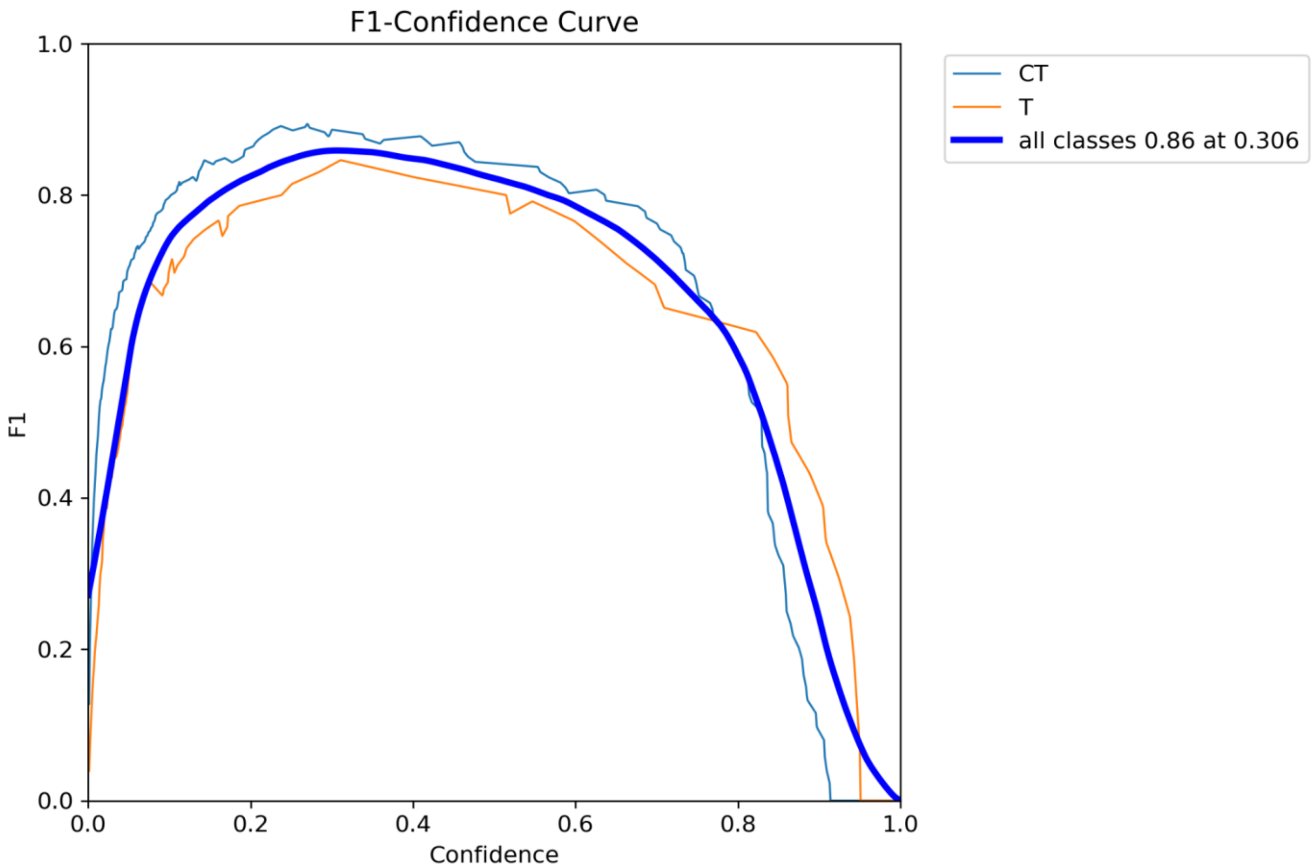





Recall-Confidence Curve
1.0
0.8
0.6
0.4
0.2
0.0
Recall
0.0
0.2
0.4
0.6
0.8
1.0
Confidence
CT
T
all classes 0.94 at 0.000


Precision-Confidence Curve
1.0
0.8
0.6
0.4
0.2
0.0
Precision
0.0
0.2
0.4
0.6
0.8
1.0
Confidence
CT
T
all classes 1.00 at 0.783


Precision-Recall Curve
1.0
0.8
0.6
0.4
0.2
0.0
Precision
0.0
0.2
0.4
0.6
0.8
1.0
Recall
CT 0.920
T 0.835
all classes 0.877 mAP@0.5


F1-Confidence Curve
1.0
0.8
0.6
0.4
0.2
0.0
F1
0.0
0.2
0.4
0.6
0.8
1.0
Confidence
CT
T
all classes 0.86 at 0.306

References

[1] G. Jocher and J. Qiu, "Ultralytics YOLO11, v11.0.0," GitHub Repository, 2024. [Online]. Available: https://github.com/ultralytics/ultralytics

[2] G. Jocher, "Ultralytics YOLOv5, v7.0," GitHub Repository, 2020. [Online]. Available: https://github.com/ultralytics/yolov5, doi: 10.5281/zenodo.3908559

[3] G. Jocher, A. Chaurasia, and J. Qiu, "Ultralytics YOLOv8, v8.0.0," GitHub Repository, 2023. [Online]. Available: https://github.com/ultralytics/ultralytics

[4] P. G., "CSGO TRAIN YOLO V5 Dataset," Roboflow Universe, 2022. [Online]. Available: https://universe.roboflow.com/pg-g-sgpnk/csgo-wlnz5

[5] D. Aubrey, "Counter-Strike: Global Offensive - How To Practice Smokes And Set Up A Custom Server," TheGamer, Feb. 18, 2021. https://www.thegamer.com/custom-csgo-server-practice-smokes/