An Application for the Human-Computer Interaction Game Based on Yolov5

Jianxi Chen *

Shenzhen College of International Education high school, Shenzhen, China

* Corresponding Author Email: s22723.chen@stu.scie.com.cn

Abstract. The rapid development of computers makes human-computer interaction possible, and hence more people have been attracted to this subject in several different ways. After searching online about the ways to increase the accuracy. Based on the algorithm of yolov5, this article used the characters from a popular PC game called Counter-Strike: Global Offensive (CSGO). By using the 3 characteristics from the game (Human-T, Human-CT, and Head). This article shows the results of the detection of those 3 by using different numbers of epochs and images. In the two detections, 240 images and 150 epochs, and 500 images and 450 epochs are used. The results of the two detections have a large difference. In the second detection, the result is much better than the first one by having a small change in the dataset. Meanwhile, this article will also mention other ways made to increase the accuracy of the detection, such as the way to label images, etc.

Keywords: Yolov5, Machine Learning, CSGO.

1. Introduction

With the gradual development of convolutional neural network techniques, object detection has become a very popular subject in deep learning. Yolov5 is one of the most popular examples of deep learning which use the images and the .txt file of the images to extract the characteristic of the parts of the images labelled [1-4]. The reason why yolov5 is a deep learning project instead of a machine learning project is that machine learning uses algorithms to parse data and make decisions based on what the computer has learned from the dataset, while deep learning will create an artificial neural network which can learn and make decisions by the computer itself, which has been proved in many tasks [5-8]. In this case, deep learning is the basic idea of yolov5 object detection, as computers need to study the images and recognize different environments of the same thing. Based on yolov4, yolov5 has great improvements in its speed and accuracy by changing the parameter. In yolov5, there are 4 models (yolov5x, yolov5l, yolov5m, and yolov5s). Although yolov7 is about to release, yolov5 is still quite popular among people.

In terms of the history and introduction of the Yolo, the Yolo project is based on python and depends on, some other things, such as OpenCV, PyTorch, and so on. yolov1 is first published in the May of 2015. At the same time, yolov1 brought much attention from the public, due to its extremely high speed compared with others in 2015. The best aspect of YOLO is that it is a real-time detection algorithm that can process 45FPS to 155FPS. When time goes to 2016, a new version of YOLO comes out which is the yolov2 which is released by Joseph Redmon and Ali Farhadi. The title of the new version of YOLO is 'YOLO9000' which means it could detect over 9000 categories of objects, which is relatively better, stronger, and faster. Two years after, the version of YOLO rise to YOLOv3. In this version, YOLO is implemented using Kara's or OpenCV deep learning libraries, which means better improvements are made to the YOLO project. In April 2020, yolov4 is published by Alexey Bochkovsky, which is the fourth installment of YOLO. It achieved SOTA performance on the COCO dataset which comprises 80 different object classes. When times go to the later 2020, yolov5 was released by a company called Ultalytics which is published in the GitHub repository by Glenn Jovher, Founder & CEO at Ultralytics, and quickly gained traction soon after its publishing. Yolov5 can also find on the iOS App Store under the app name "iDection" and "Ultralytics LLC". Recently, the new version of YOLO(yolov7) is about to release. This program is written by Joseph Redmon, Ali Farhadi, and Santosh Divvala. At release, this architecture will be much faster than other object detectors and

become state-of-the-art for real-time computer applications.

In this YOLO research field, there are many detections for the recognition of humans or cars which can be found online. However, after searching this field for a long time, the article about the detection of the game characters is rare to find. Therefore, this paper is mainly focused on the detection of the game characters. In this article, there will be three objects from CSGO, Human-T, Human-CT, and Head will be used to train, and detect. In addition, this article will show the exact way how to train and make people's dataset by using make sense.

2. Method

To label the images, there are several ways to do it. Make sense, labelimg, for example, are places where people can label the parts of their images and export the image in .txt forms. But this study will recommend using make sense, as people can use it by searching the official web online [9]. Be careful if the starters want to first use either labeling or make sense, and then use the other one, they need to make sure that the number at the front of the line in the txt file is the name of the parts they have chosen should be the same, otherwise, there will be a bug. As a result, what is recommended is to use only one tool to label the images. Once the images and the related .txt file have finished labelling and exported as a txt file, the folders should be built in the way below. For the images, it goes to the images/train, and the related .txt file will be placed in the labels/train.

In addition, for the model of the dataset, one model from the models (yolov5x, yolov5l, yolov5m, and yolov5s), should be pasted into the folder, and the name of the model should be changed into something_model.yaml. For the parameter, the coco128.yaml should be placed in the parameter folder and change its name in something_parameter. In this case, the name before the _ can be whatever possible, but the name should only be in English.

3. The experiments and results

3.1. Results

The experiments in the training set, the characters in CSGO were used. There are three things needs to be detected, Human-CT, Human-T, and the head. In order to get those images, a video of the game was recorded Then, after the video is done. It is the time to use some tools like the videoProc converter. After those images are created, the only way to classify them is to check each image by eyes. For example, some images without the characteristics that is used to detect, or extract need to be deleted by hands. Once the images and the related .txt file is done and put into the right position. It is the time to start working the yolov5. From Figure 1, the expectation of the detection is not ideal, as the curve tend to be a line. The most ideal expectation will be a concave curve. Due to this circumstance, the number of images and epochs was increased (500 and 450 respectively). The result of this detection looks perfect in Figure 2. In the second graph, the precision and the recall have a great improvement. From Figure 2, the curve become more concave which means the results are better. After those process has finished running, this study records another video, and use it for the detection. From Figure 3, the finial detection of the CSGO character is quite good, but this is only for the second conditions (500 images and 450 epochs).

In this case, the number of pictures used, and the epochs used in the training section will affect the results significantly. It is recommended to have as more epochs as people can.

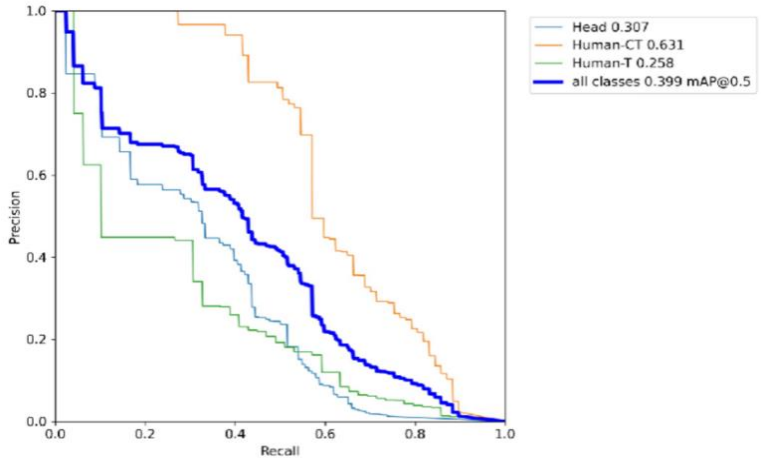

Figure 1. The relation between the precision and the recalls.

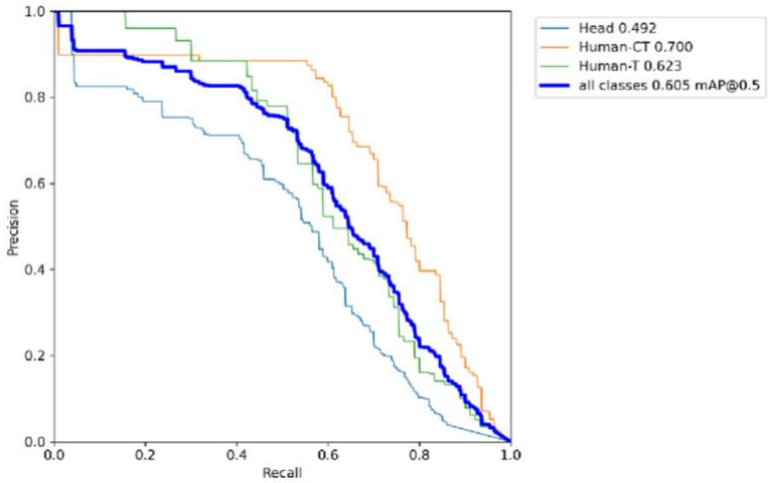

Figure 2. The result of using 500 pictures and 450 epochs.

Figure 3. the detection of the all characteristics that have been labelled (Human-CT, Human-T, and Head).

3.2. Improvements and cautions

First of all, one small caution is ‘remember to name the images and the related .txt file in the same way otherwise, there will be a bug. During the part of labelling images, some parts of the images should not be labelled to prevent any error. If people want to achieve an effective method of labelling, the rules below should be obeyed: 1) for some parts of the images that are too small for the computer to extract characteristics should not be labelled, even though people can identify the parts of the images by themselves. 2) for the parts of the images that have been covered by other things partially should not be labelled. For example, the things that want to be labelled only show 3/4 should not be labelled.

On top of that, for the dataset, in order to achieve a higher accuracy after training the dataset, more dimensions and different environments of the images should be used if the images are in 3D. In addition, adding more images to the dataset is another way to increase the accuracy, but it will take the computer more time to train the dataset. At last, people can repeat to train the datasets more times in order to achieve higher accuracy.

In addition, the only format that is accepted by yolov5 is the .txt file. If the image-related file is in another form such as .xml or .json, some code should be used to change the format. The code for changing the .xml file and the .json file to the .txt file can be found in [10], which is at topics 5 and 4 respectively.

At last, when the power of the video card is not enough to make the algorithm, people can change a few lines of code in the train.py. The code on lines 479, 480, 481, 482, 483, 484, 485, 487, 496, and 501 are the main code that uses up the computer. If someone says they want to obtain the best result and they do not have a strong video card and CPU, they can use the online server to make the algorithm of the yolov5. The article about using this server can be found in [10], and sadly, this is only in Chinese.

4. Conclusions

In this experiment, a detection of the characters in CSGO (Human-T, Human-CT, and Head) has made in order to find out the ways to increase the accuracy of the detection. In addition, there are a cornucopia of factors which will affect the the accuracy in the end. The factors that have the highest effects on the accuracy are both the number of epochs and images used. From the results, an increase of 260 images and 300 epochs has a much better accuracy. Therefore, in order to have a better accuracy more images and epochs can be used, but the power of the video card also need to be considered. In the future, it is hopefully that some improvements can be made at the algorithm by using the characteristics either from the real world or in the game world.

References

[1] Ge, Z, et al. Yolox: Exceeding yolo series in 2021 [R]. arXiv preprint arXiv:2107.08430 (2021).

[2] Jiang, P, et al. A Review of Yolo algorithm developments [J]. Procedia Computer Science 199 (2022): 1066-1073.

[3] Thuan, D. Evolution of Yolo algorithm and Yolov5: The State-of-the-Art object detention algorithm [R]. (2021).

[4] Li, S, et al. YOLO-FIRI: Improved YOLOv5 for Infrared Image Object Detection [J]. IEEE Access 9 (2021): 141861-141875.

[5] Kayalibay, B, et al. CNN-based segmentation of medical imaging data [R]. arXiv preprint arXiv:1701.03056 (2017).

[6] Yu, Q, et al. Improved denoising autoencoder for maritime image denoising and semantic segmentation of USV [J]. China Communications 17.3 (2020): 46-57.

[7] Barade, A. Automatic Traffic Sign Recognition System Using CNN [J]. International Journal of Information Retrieval Research (IJIRR) 12.1 (2022): 1-14.

[8] Thaiyalnayaki, K., et al. Automatic classification of cassava using data augmentation and CNN [C]. AIP Conference Proceedings. Vol. 2405. No. 1. AIP Publishing LLC, 2022.

[9] Makesense [R], 2022, https://www.makesense.ai

[10] CSDN, Run deep learning using Hengyuan cloud server [R], 2022, https://blog.csdn.net/m0_53392188/article/details/124339442