Mask Detection based on YOLOv5 with CIOU Loss

Group	Name	Student ID	Distribution
5	SUN Maojun	22046906G	Data processing Coding work and experiments Paper writing (method, result and disscussion) model employ PPT
	SUN Binwen	22049745G	Literature Reading Paper writing(introduction,methods reference) Reduce duplicate checking PPT
	WU Hongfei	22049959G	Literature Reading paper writing(abstract, methods, conclusion, reference) Embellished articles Reduce duplicate checking PPT

Lecture time: 2022.11.15

8:10 PM - 8:30 PM

Abstract—Under the influence of COVID-19, wearing masks has become a daily behavior. Mask detection is a popular task in target detection. We give an efficient mask detection system by using YOLOv5. Label smoothing is performed when calculating the classification loss to reduce the overfitting problem. By using CIOU-NMS, which adds the scale information of boundary frame aspect ratio. With the focal loss function, we cut down the influence of unbalanced sample distribution between the background box and the target box. Our method's feasibility is verified by comparative experiments. We finally achieve a mAP@0.5 of over 0.9 and mAP@0.5:0.95 of 0.61.

I. INTRODUCTION

To stop the spread of COVID-19, people are required to wear masks in public places such as shopping malls and stations. At present, manual inspection is the main method used to check whether people wear masks. That kind of method has drawbacks of low efficiency and easy to miss detection, which has great security risks. A reliable real-time detection system for mask-wearing can better solve these problems. Mask-wearing detection can be classified as a target-detection problem. Target-detection using neural network is a relatively mainstream target detection method at present, which can be classified to two routes. The first route is the target-detection algorithm based on candidate regions represented by Faster RCNN [1] series, whose detection process can be divided into two stages: candidate region classification and coordinate correction. Therefore, it is called a two-stage target detection algorithm, which has good performance on detection accuracy and the drawbacks on slow speed of detecting. The second route is the regression analysis target detection algorithm represented by YOLO [2] series. Its detection only involves one regression analysis process, so it is called a single-stage target detection algorithm. Its advantage is fast detection speed, but its disadvantage is low detection accuracy.

The YOLOv5 was proposed by Glenn Jocher in May 2020. The YOLO series algorithm is widely used in various target detection scenarios due to its advantages of easy deployment and good real-time detection performance.

The YOLOv5 network model follows the structure of YOLOv3 and v4, whose network structure consists of four parts: input, Backbone, Neck, and Prediction. When the model works, the input end carries out data augmentation, image scaling and frame calculation for input images, and the processing results are input to the backbone network, which mainly completes the feature extraction, the feature aggregation end fuses the extracted features, and the prediction end can generate the corresponding target frame according to the features at different scales and calculate the optimal target frame [3].

YOLOv5 has some variant network models: YOLOv5s, YOLOv5m, YOLOv5l and YOLOv5x. They differ in the width and depth of the network. And the number of C3 and CBS structures contained is also different. YOLOv5s is a lightweight network. This paper chooses YOLOv5s and YOLOv5m as the network model of system detection.

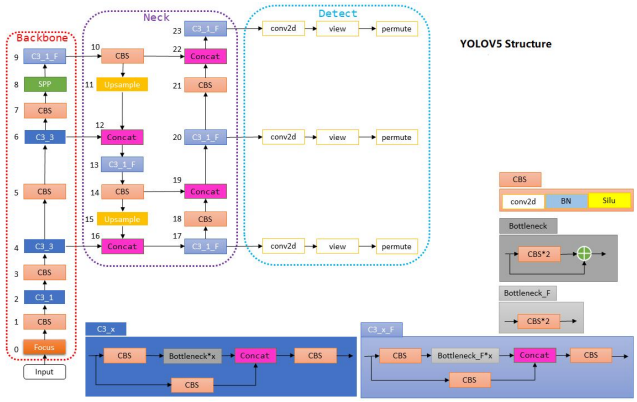

Figure 1. The structure of YOLOv5 Network

A) Input

The Input part mainly includes picture size scaling, Mosaic data augmentation and adaptive anchor box calculation. The commonly used target detection algorithm needs to scale the training picture in the datasets to a uniform size before sending it into the network for training, which can increase the reasoning speed and improve the detection effect. Through means of Mosaic data augmentation, YOLOv5 randomly selected 4 images in the training set and carried out random intercept, flip and zoom operations on them respectively. Then, the 4 images captured were connected to new images respectively and sent to the network for training after processing the label information. Mosaic data augmentation images were added to the training set to improve the detection effect of the network on small targets. Before the training, YOLOv5 calculated the Best Possible Recall (BPR) of the default anchor box from the label files of the training datasets. If BPR was less than 0.98, the anchor box would be recalculated.

B) Backbone

Backbone consists of the CBS structure, C3 structure and SPPF structure (Spatial Pyramid Pooling Fast). The CBS structure consists of convolution layer, batch normalization process and Silu activate function [4].

The goal of activation functions is to nonlinearize neural networks. The activation function is continuous and differential. The SiLU function is a combination of the Sigmoid and weighted linearity.

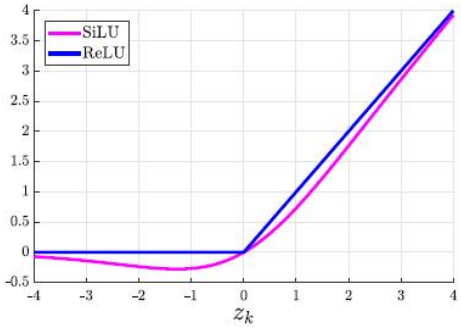

Figure 2. SiLU and ReLU function

The factor a_k of the k th SiLU of the input z_k is calculated by sigmoid function as follows:

$$a_k(z_k) = z_k\sigma(z_k)$$

where the sigmoid function is

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

For larger z_k values, the activation value of SiLU is approximately equal to the value of ReLU. Unlike ReLU, the activation of SiLU is not monotonically increasing. On the contrary, for $z_k \approx -1.28$, its global minimum is about -0.28. One of the characteristics of SiLU is its self-stability. As a regularizer, it suppresses weights of large number parameters.

There are two types of C3 structures. The main deference is the Bottle-Neck structure. The Backbone only uses the BottleNeck1 structure. SPPF compared to SPP [5] can reduce the amount of computation.

C) Neck

The Neck mainly includes FPN (Feature Pyramid Network) structure [6] and PAN (Path Aggregation Network) structure [7]. FPN structure and PAN structure pass top-down strong semantic features and bottom-up strong positioning feature respectively, and carry out feature fusion in the process of feature transmission, so that the network can fuse more feature information, and improve the performance of the next Prediction part. In addition, the Neck part uses the CSP2_X structure.

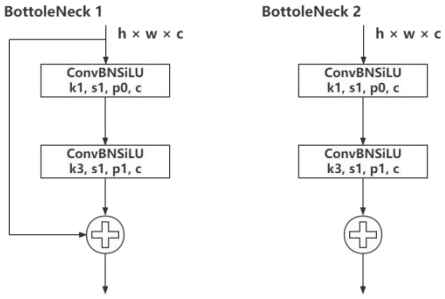

Figure 3. The structure of the Neck part

D) Prediction

The Prediction section includes loss function calculation and Non-Maximum Suppression (NMS). The total loss includes classification loss, box loss and confidence loss. YOLOv5 uses binary cross entropy loss to calculate classification loss and confidence loss, and CIOU loss to calculate location loss, namely boundary box regression loss. The NMS is used to remove redundant detection boxes and reserve the candidate box with the highest prediction probability as the final prediction box.

A. Anchor

Anchor is a box depicted by prior knowledge [8]. Each anchor is representative of a certain class under unsupervised learning. If the sample distribution of the data is close to the total sample distribution, then this anchor is the prior knowledge of the sample.

The anchors in YOLOv5 are obtained by the K-means clustering method, which is composed of weight and height. There are three groups of anchors with different sizes, which process feature maps of different sizes respectively. The smallest anchors are suitable for small target detection tasks, and the largest anchors are suitable for large target detection tasks.

II. METHOD

A. Datasets and experimental environment

The open datasets from Kaggle are used in our experiment. It includes three categories: mask, no mask and Wearing a mask incorrectly. See

FMD:https://www.kaggle.com/datasets/andrewmvd/face-mask-detection.

MFV:https://www.kaggle.com/datasets/trngvhong/mfvt-dataset

Our experiments mainly use FMD (Face Mask Detection) and FMAVT (FMD+MFVT): Randomly select 5147sample from MFVT and add to the FMD.

Table 1. Our datasets

Title	FMD	MFVT	FMAVT
Format	VOC	YOLO	YOLO
Total Set	853	13507	6000
Training Set	689	10445	4320
Validation Set	76	1706	1080
Test Set	88	1356	600

In our experiment, we use Windows 10 operating system, PyTorch framework and NVIDIA GeForce 1060 graphics card for calculation.

Table 2. Environment

Parameter	Configuration
CPU	Intel Core i7-8750H
GPU	NVIDIA GeForce GTX 1060
System Environment	Windows 10
Language	Python 3.8
CUDA Version	11.1
PyTorch Version	1.8.0

B. Data preprocessing

The FMD is VOC format and the MFVT is the YOLO format. We firstly convert the FMD to YOLO format. Take an image in the datasets as an example, which width=1800, height=380.

Assume the coordinates of the box area in VOC are:

$x_{min} = 725, x_{max} = 985, y_{min} = 160, y_{max}$=380.

Identify applicable sponsor/s here. *(sponsors)*

Figure 4. A sample of box area

Converting to YOLO given by:

$$x = \frac{x_{center}}{width}, y = \frac{y_{center}}{height},$$

$$w = \frac{x_{max} - x_{min}}{width}, h = \frac{y_{max} - y_{min}}{height}$$

C. Loss Function

The loss function is composed of classification loss, bounding box loss and confidence loss.

A) Bounding box loss

The IOU evaluates the overlap between two areas. It is the ratio of the overlapping area of two areas to the total area.

In target detection, IOU is the quotient of the intersection and union of the prediction box and the actual box.

$$IOU = \frac{|B \cap B^{gt}|}{|B \cup B^{gt}|}$$

Where B^{gt} is the ground truth box.

We use IOU loss to make the IOU metric better:

$$\mathcal{L}_{IOU} = 1 - IOU$$

We use CIOU loss in the experiment [9].

$$\mathcal{L}_{Box} = \mathcal{L}_{CIOU} = 1 - IOU + \frac{\rho^2(b, b^{gt})}{c^2} + \frac{v^2}{(1 - IOU) + v}$$

where b and b^{gt} denote the central points of B and $B^{(gt)}$, $\rho(\cdot)$ is the Euclidean distance, and c is the diagonal length of the smallest area containing the two boxes.

And α is a positive trade-off parameter, v measures the consistency of aspect ratio [9]:

$$v = \frac{4}{\pi^2}\left(\arctan\frac{w^{gt}}{h^{gt}} - \arctan\frac{w}{h}\right)^2$$

In the object box regression function, The CIOU Loss considers these essential geometric elements: overlapping area, center point distance, and aspect ratio.

The distinctions between some loss function are analyzed comprehensively:

IOU Loss: Mainly considers the overlap area between the detection frame and the target frame.

GIOU Loss: Based on IOU, considered the area of the minimum external rectangle of two boxes, which solves the problem of when two boxes have no Intersection, whatever how far they are, the loss is 0 and the gradient is also 0.

DIOU Loss: Based on IOU and GIOU, added the distance information of the center point in the box.

CIOU Loss: Based on DIOU, added the scale information of boundary frame aspect ratio.

B) Confidence loss.

In classification problems we often use binary cross entropy as loss function to evaluate the quality of the prediction results. YOLOv5 performs sigmoid processing on the confidence of the output, which is the BCELogitsLoss [10].

$$\mathcal{L}_{conf} = -\frac{1}{N}\sum_{i=1}^{N}\Big(y_i \cdot \log\big(\sigma(C(y_i))\big) + (1 - y_i)\cdot\log\big(1 - \sigma(C(y_i))\big)\Big),$$

where y is 0 or 1, which means if the box contains the target or not. σ is the sigmoid function, and $C(y_i)$ is the CIOU corresponding to the prediction box and the target box. We apply CIOU as the confidence label of the prediction box.

If p(y) of label y approaches 1, the loss value should be close to 0.

C) Classification loss

The classification loss is also based on BCELogitsLoss,

$$\mathcal{L}_{BCELogitsLoss} = -\frac{1}{N}\sum_{i=1}^{N}\begin{pmatrix} y_i \cdot \log\big(p(y_i)\big) \\ + (1 - y_i)\cdot\log\big(1 - p(y_i)\big)\end{pmatrix}$$

However, there are some problems. the relationship between real tags and other tags is ignored, and the model cannot learn more. The generalization ability of the model trained in this way will be poor. The model is easy to be affected when dealing with classification problems such as datasets with high sample similarity and high data noise.

In this way, we introduce label smoothing [11] with the ε =0.1, which mainly reduces the overfitting problem by reducing the weight of the class of real sample labels in calculating the loss function. After adding label smoothing, the probability distribution becomes:

$$p_i = \begin{cases}(1 - \varepsilon), & i = target \\ \dfrac{\varepsilon}{N - 1}, & i \neq target\end{cases}$$

The cross-entropy loss function becomes:

$$\mathcal{L}_{cla} = \begin{cases}(1 - \varepsilon)\cdot Loss, & i = target \\ \varepsilon \cdot Loss, & i \neq target\end{cases}$$

So, the total loss function defines as the weighted sum of confidence loss, box loss and classification loss

$$\mathcal{L} = a\cdot\mathcal{L}_{conf} + b\cdot\mathcal{L}_{box} + c\cdot\mathcal{L}_{cla}$$

In our experiment a = 1.0, b = 0.05, c = 0.5.

Then, we use focal loss with gamma=1.1 to give a different weight to target and background, because there are more background boxes than the target box.
We define p_t:

$$p_t = \begin{cases} p, & if\ y = 1 \\ 1 - p, & otherwise \end{cases},$$

and Focal loss:

$$FL(pt) = -(1 - p_t)^{\gamma} log(p_t)$$

The higher γ will cause a higher loss gap between the well-fitted category and the bad-fitted category.

D. NMS: non-maximum suppression.

The NMS [12] is used to filter duplicate or similar prediction boxes. In our experiment, CIOU-NMS is the evaluation indicator. We conjecture that two boxes with far distance from the center correspond to different objects and should not be removed.

$$s_i = \begin{cases} s_i, & CIOU(M, b_i) < N_i \\ 0, & CIOU(M, b_i) \geq N_i \end{cases}$$

s_i represents the score of each box, M is the box with the highest confidence. b_i is the other box, N_i for setting a threshold. We can see when $CIOU$ is greater than N_i. The box will be dropped.

E. Optimizer

A) AdamW

In traditional SGD, if we want to increase the generalization ability of the model, the L2 regularization term is usually introduced to bring more restrictions on parameters with large weights and achieve the effect of weight decay. After the introduction of the L2 regularization term, the result of calculating the gradient of the regularization term will be added when calculating the gradient. Therefore, if the parameter itself is relatively large, the corresponding gradient will be relatively large, and the gradient will fall faster naturally.

However, adding L2 regularization term directly in front of Adam optimizer cannot achieve the effect of weight decay. Because there is a normalization operation in Adam, weights with large gradient are also normalized. As a result, Adam + L2 cannot punish parameters with large weights. This makes the Adam + L2 optimizer less effective.

We use AdamW [13] to achieve the effect of weight decay in Adam optimization algorithm. According to the weight decay, after the Adam optimizer is used, the punishment for larger parameters is carried out, that is, the weight decay part is separated from the gradient update.

Specifically, let $\nabla f_t(x_{t-1})$ stand for the gradient at time t and the weight values of all parameters at time t-1, then:

$$g_t = \nabla f_t(x_{t-1}) + w x_{t-1}$$

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) {g_t}^2$$

$$\widehat{m}_t = \frac{m_t}{1 - {\beta_1}^t}$$

$$\hat{v}_t = \frac{v_t}{1 - {\beta_2}^t}$$

$$x_t = x_{t-1} - \eta_t \left(\frac{\alpha \widehat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} + w x_{t-1} \right)$$

The learning rate η_t means using a decay strategy from the initial learning rate α. The main difference of g_t lies in not only about gradient, but also the parameter values of the moment on the minuend joined a moment in the values of the parameters x_{t-1}, ensuring that the parameters whose value is too large to the faster attenuation.

We set the initial learning rate with 0.001, β_1 with 0.937, and weight decay with 0.0005.

F. Learning rate

Warm restart and Cosine Anneal learning rate

To improve the anytime property of the trained neural model, we use warm restart and cosine anneal for adjusting the learning rate dynamically. First, determine the upper and lower bound of the learning rate and the cycle step size [14].

$$\eta_t = \eta_{min} + \frac{1}{2}(\eta_{max} - \eta_{min}) \left(1 + \cos\left(\frac{T_{\text{cur}}}{T_{max}} \pi \right) \right)$$

Second, use the maximum learning rate to reduce the learning rate after each step of training, and then restore the learning rate to the maximum learning rate.

We set warm up epochs as 3.0, warm up moment as 0.0005, and initial bias learning rate as 0.1.

G. Grading Criteria

A) Mean Average Precision

Considering not only the need to detect the target and the output target category, but also locate the target position. The simple accuracy index in the classification problem can not reflect the accuracy of the result of the target detection problem. Mean Average Precision (mAP) is a common index used to determine the quality of the target detection model.

The algorithm usually outputs a bounding box to identify the location of the detected target. To measure the accuracy of the prediction box and the ground truth of the target, the IoU can be used.

We can set a threshold, usually 0.5, and the prediction results can be divided into:

1) $IOU \geq 0.5$,
Classify the object as True Positive (TP).

2) $IOU < 0.5$,
Classify the object as False Positive (FP) when it is a wrong detection, but it is truly negative.

Classify it as False Negative (FN)) when there is ground truth in the box, but the model does not recognize it.

True Negative (TN) - All parts of the image that do not contain the actual box and the detection box. It is not typically used in calculations.

Then, we use precision and recall for evaluation. The precision refers to the correct rate of prediction in all results predicted as positive. It is formally defined as

$$Precision = \frac{TP}{TP + FP}$$

The recall refers to the ratio correctly predicted in all positive cases. It is formally defined as

$$Recall = \frac{TP}{TP + FN}$$

Then, average precision (AP) is the average accuracy of the model for a single category. For each different recall value, select the maximum of precision when it is not worse than these recall values,

$$p_{interp(r)} = \max_{\hat{r}:\hat{r}\geq r} p(r),$$

and then calculate the area under the PR curve as the AP value

$$AP = \int_0^1 p_{interp(r)} dr$$

In our experiment, we first calculate the precision and recall of each class to get P-R Curve, and AP is equal to the integral value of P-R Curve. If the AP value of an algorithm is large, that is, the area under the P-R curve is relatively large, it can be considered that the precision and recall of this algorithm are relatively good as a whole.

mAP is the average of AP values for all categories, which is defined as

$$mAP = \frac{1}{C}\sum_{i=1}^{c} AP_i$$

Then, *mAP@0.5* represents the mAP of each type of image when IoU is set to 0.5. And *mAP@0.5:.095* represents the average mAP at different IoU thresholds (from 0.5 to 0.95, step size 0.05).

B) Relation Curve

P-Curve represents the relationship between precision and confidence (IoU in YOLOv5).

R-Curve represents the relationship between recall and confidence (IoU in YOLOv5).

PR-Curve represents the relationship between precision and recall.

F1-Curve is a metric for classification, a harmonic mean function of precision and recall, between 0 and 1. It is formally defined as

$$F1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$$

H. Experiment

Our experiment details are as follow.

Table 3. Experiment setting.

Exp	DSSL	TSACF	TSACCF	TMACCF
Datasets	FMD	FMAVT	FMAVT	FMAVT
Model	Yolov5s	Yolov5s	Yolov5s	Yolov5m
Batch size	16	16	16	16
Epochs	300	200	200	200
Optimizer	SGDR	AdamW	AdamW	AdamW
LR	Linear	Cosine	Cosine	Cosine
NMS	IOU	IOU	CIOU	CIOU
Loss	Label smoothing	Label smoothing+ focal loss	With label smoothing+ focal loss	With label smoothing+ focal loss

III. RESULTS AND DISCUSSION

The training result, which is valid on the validation sets can be seen as follow.

Table 4. Experiment Result

Experiment	DSSL	TSACF	TSACCF	TMACCF
Precision	**0.9625**	0.90893	0.91562	0.92161
Recall	0.8543	0.87008	0.87563	**0.88636**
mAP@0.5	0.8692	0.89141	0.89634	**0.90584**
mAP@.5:.95	0.5755	0.59512	0.59506	**0.60918**

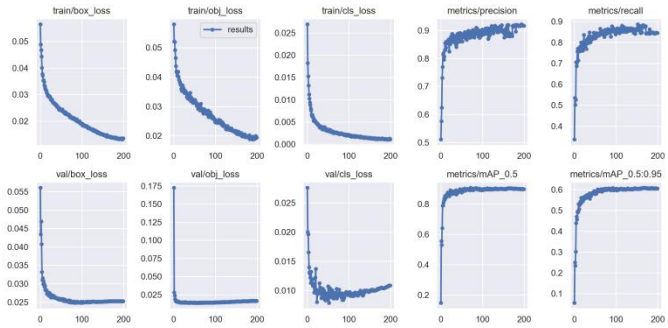

Figure 5. Training process and validation result of TMACCF

A. Discussion of network structure

TMACCF perform best in recall and mAP, this is mainly attributed to the Yolov5m, which has a deeper and wider network structure than Yolov5s and means that it has a stronger ability to extract features.

B. Discussion of optimizer and learning rate

TSACF perform better than DSSL. Because the cosine anneal and AdamW are easily to find a better minimizer than linear and SGD. The Linear method is difficult to correctly describe the optimal solution position and the structure of the loss function, which makes the model tend to converge to a local minimizer. Finally, due to the attenuation of the learning rate, the model eventually falls into a local optimal solution, rather than the global minimizer.

The cosine function is characterized by the fact that as the independent variable x increases, the function value decreases slowly, then accelerates, and then slows down. Using this feature, after the learning rate decays to a certain value, the Cosine learning rate method readjusts the recovery learning rate, jumps out of the current local optimal solution and re-finds the global optimal solution. Besides, Adam works well with weight decay just as we analysed in last section.

C. *Discussion of NMS*

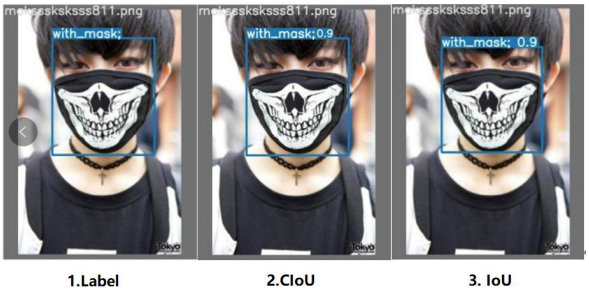

Figure 6. A sample detected by the IOU-NMS and CIOU-NMS.

We can see that the box predicted by CIOU is closer to the label. Because CIOU combines the information of distance, ratio of height and width. So, more similar bounding boxes are retained after NMS.

D. *Discussion of focal loss*

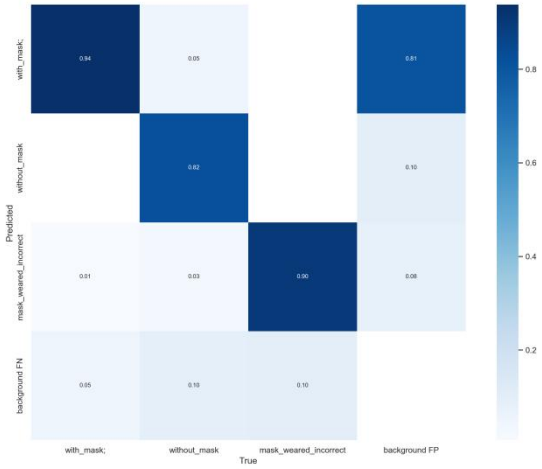

Figure 7. The confusion matrix of DSSL

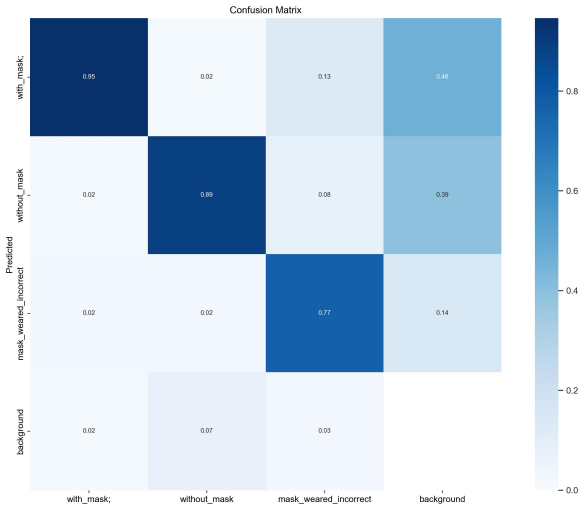

Figure 8. Confusion matrix of TSACF

We noted that the FN in figure 7 is 0.81 and FN in figure 8 is 0.46. We guess that the FN is quite high in DSSL because the huge number of sample gap between the target box and the background box, so there are many losses generated by the background. However, in TSACF, we use the focal loss with γ=1.5, in the training process, if the background box is well classified, the model will pay more attention to the target box. That is why FN is declining after the focal loss.

Figure 9. Prediction result

E. *Speed test*

The average cost of per image is approximately 18ms. We use OpenCV and TensorRT to deploy the model. In real-world scenes, the model can achieve 60 fps real-time detect assignment.

IV. CONCLUSION

In this paper, we are well done the mask detection assignment by YOLOv5, and we focus on how to improve the mAP. We compare different datasets, network structures, optimizers, loss functions and NMS. We verified the bigger backbone cloud leads to a better effect in recall and mAP. We proved CIOU-NMS has a visible improvement compared to IOU-NMS, and the focal loss can tackle the problem of unbalanced

data distribution. By label smoothing, Cosine Anneal and AdamW, we obtained a model with mAP@0.5 of over 0.9 and mAP@0.5:0.95 of 0.61. Finally, we deployed our model in real-world scenes and perform an excellent result.

REFERENCES

[1] GIRSHICK R, DONAHUE J, DARRELL T, et al. Region-based convolutional networks for accurate object detection and segmentation[J].IEEE Trans Pattern Anal Mach Intell,2016,38(1):142-158.

[2] REDMON J, DIVVALA S, GIRSHICK R, et al. You only look once: unified, real-time object detection [C]//2016 IEEE Conference on Computer Vision and Pattern Recognition. June 27-30, 2016, Las Vegas, NV, USA.IEEE,2016:779-788.

[3] LIN T Y, GOYAL P, GIRSHICK R, et al. Focal loss for dense object detection[C]//2017 IEEE International Conference on Computer Vision. New York: IEEE Press, 2017: 2999-3007.

[4] Elfwing S, Uchibe E, Doya K. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning[J]. Neural Networks, 2018, 107: 3-11.

[5] HE K M, ZHANG X Y, REN S Q, et al. Spatial pyramid pooling in deep convolutional networks for visual recognition[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2015, 37(9): 1904-1916.

[6] LIN T Y, DOLLÁR P, GIRSHICK R, et al. Feature pyramid networks for object detection[C]//2017 IEEE Conference on Computer Vision and Pattern Recognition. New York: IEEE Press, 2017: 936-944.

[7] LIU S, QI L, QIN H F, et al. Path aggregation network for instance segmentation[C]//2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. New York: IEEE Press, 2018: 8759-8768.

[8] Jia X, Wang H, Wang M, et al. SAR image ship target detection based on sea-land segmentation and YOLO anchor free[C]//Fourteenth International Conference on Machine Vision (ICMV 2021). SPIE, 2022, 12084: 200-207.

[9] Zheng Z, Wang P, Ren D, et al. Enhancing geometric factors in model learning and inference for object detection and instance segmentation[J]. IEEE Transactions on Cybernetics, 2021.

[10] Zhang W, Xia X, Du J, et al. Recognition and detection of Wolfberry in the natural background based on improved YOLOv5 network[C]//2022 3rd International Conference on Computer Vision, Image and Deep Learning & International Conference on Computer Engineering and Applications (CVIDL & ICCEA). IEEE, 2022: 256-262.

[11] Guo G, Zhang Z. Road damage detection algorithm for improved YOLOv5[J]. Scientific reports, 2022, 12(1): 1-12.

[12] Zheng Z, Wang P, Liu W, et al. Distance-IoU loss: Faster and better learning for bounding box regression[C]//Proceedings of the AAAI conference on artificial intelligence. 2020, 34(07): 12993-13000.

[13] Loshchilov I, Hutter F. Decoupled weight decay regularization[J]. arXiv preprint arXiv:1711.05101, 2017.

[14] Loshchilov I, Hutter F. Sgdr: Stochastic gradient descent with warm restarts[J]. arXiv preprint arXiv:1608.03983, 2016.

[15] Lin T Y, Goyal P, Girshick R, et al. Focal loss for dense object detection[C]//Proceedings of the IEEE international conference on computer vision. 2017: 2980-2988.

Appendix for code:

We delete the dataset and model because it is too large.

The dataprocess.py is to convert the img format.

The Split_data.py is for split the datasets

The VOC_label.py is for convert the

The train.py is for train the model

The test.py is for test the model

The detect.py is for real detection

The data dir is the config of datastes.