Remove Handwritten Struck-Out Text from Mathematical Answer Sheets via Deep Learning

Han Jiaqi
2022233351
SIST
hanjq2022@shanghaitech.edu.cn

Chen Zui
2022233338
SIST
chenzui2022@shanghaitech.edu.cn

Zhang Tao
2021233352
SIST
zhangtao4@shanghaitech.edu.cn

Chen Zheng
2020533166
SIST
chenzheng@shanghaitech.edu.cn

Zhao Zijun
2022233323
SIST
zhaozj2022@shanghaitech.edu.cn

Abstract

Writing is a classic and commonly used method of representing knowledge and recording information. However, during writing, human usually make writing errors, such as misspelled words, out-of-context wording, incorrect numbers or symbols and so on. Moreover, during writing, the writer's thoughts and ideas may evolve and change, resulting previously written content less valuable or even incorrect. In practice, writers usually cover these errors or unwanted content with a mark, turning them into struck-out text, then write the correction near it. The struck-out text appear in a variety of media and situations, such as letters, manuscripts and exam answer sheets. Such writing errors can be more likely to occur when working out mathematical questions, and these struck-outs may even affect the meaning of the entire formula. We propose a detection model recognizing mathematical struck-outs in the images with high accuracy and quality.

1. Introduction

Writing is a classic and commonly used method of representing knowledge and recording information. However, during writing, human usually make writing errors, such as misspelled words, out-of-context wording, incorrect numbers or symbols and so on. Moreover, during writing, the writer's thoughts and ideas may evolve and change, resulting previously written content less valuable or even incorrect. In practice, writers usually cover these errors or unwanted content with a mark, turning them into struck-out text, then write the correction near it. The struck-out text appear in a variety of media and situations, such as letters, manuscripts and exam answer sheets.

Automatic recognition of handwritten text is an important research topic in the field of digital document analysis. Although the problem is challenging due to the wide diversity of human handwriting, significant progress has been made in recent years. A limitation of most published reports on handwriting recognition is that they consider the input to be ideal, i.e., documents without writing errors. However, a free-form handwritten manuscript may have misspelled or inappropriate words, with the appropriate word usually written next to a struck-out word. Such writing errors can be more likely to occur when working out mathematical questions, and these struck-outs may even affect the meaning of the entire formula. Some examples of inappropriate recognition are as follows:

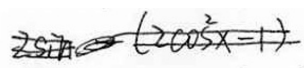

Figure 1. Fully stuck-out text

In Figure 1, we see the text is fully covered with struck-out strikes, which means they should not be recognized.

However, the output given by OCR engine is:

$$zs2ve = 6200^258 \equiv 7 = \lambda$$

Similar results are observed when OCR engine is used to recognize completely struck-out lines.

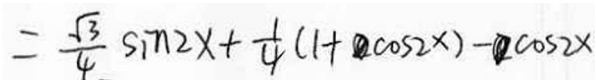

Figure 2. Struck-out characters

In Figure 2, several characters are blackened and should not be recognized. The OCR output of this line is:

$$\frac{\sqrt{3}}{4}\sin 2x + \frac{1}{4}(1 + e\cos 2x) - e\cos 2x$$

Both blackened characters are recognized as e.

There is not much academic work that addresses the identification of scribble marks in handwritten mathematical formulas. We try to investigate how to integrate and improve the methods of handwritten text recognition to achieve good results in this area of mathematical formula struck-out recognition. To be specific, our final model accepts as input an arbitrary image containing a handwritten mathematical formula, and after network processing, outputs the position of bounding boxes that frames the locations of the struck-out in the image, which is also a target detection task from a technical point of view.

The contribution of each members are below.

	工作贡献
韩嘉祺	数据集制作；benchmark（YOLO）；数据增强实现；数据增强实验（YOLO）
陈醉	数据集制作；swin based model 实验及分析
陈正	SSD part，部分ppt与论文写作
赵梓君	论文框架编写及研究背景调研，部分配图和表格统合生成
张涛	Benchmark（3个模型），PPT制作（次），报告制作（次）

Figure 3. Contributions

2. Object Detection Milestone

Compared with deep learning and image recognition, object detection originated relatively late. Around the beginning of this century, the idea of object detection was put forward because deep learning returned to the public's vision. After nearly two decades of development, object detection has become a hot research content at present, especially in the field of computer vision. Object detection algorithms solve more and more practical problems. Its structure has also shifted from traditional algorithms based on manual features to algorithms based on deep learning methods, such as single-stage SSD, YOLO algorithm and two-stage R-CNN series algorithms proposed in recent years. In recent years, with the development of mobile technology, the relevant technology of target detection is gradually oriented to mobile terminal from PC terminal. Now, object detection has become the key research object of universities and research institutions.

Since 2012, target detection has been divided into two important branches,one stage detector and two stage detector, which have some attempts on our tasks.

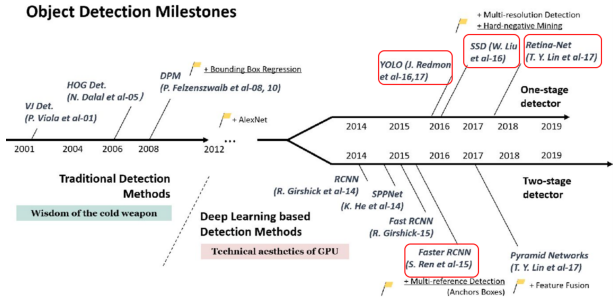

Figure 4. Contributions

The earliest model architecture is the two stage model, which includes two steps of proposal and classification, the first step is to classify each candidate box,

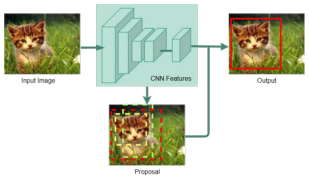

Figure 5. Contributions

while the one stage model is a one-step model, which extracts features directly in the network to predict object classification and location;

Figure 6. Contributions

There are also popular target detection methods based on Transformer, of which the most classic is DETR published on ECCV2020. Traditional detection methods rely on hand-designed components to generate high-quality anchor, or to retain the best one among all candidate boxes. However, DETR simplifies the original complex pipeline, uses the overall total loss to guide the optimization of the whole model, and realizes the end-to-end target detection.

the advantage of transformer is very high confidence and very good masking inference effect, but the disadvantage is that the local information is not as good as RNN or CNN.

3. Related Works

There have been fewer removals, and most of them have nothing to do with scenes that involve lots of mathematical formulas.

These are two representative works:

one is that bounding box is calculated based on connected domains, and then classified. Bounding box is effective for word modification, but it is generally not recognized for alterations embedded in the middle of words

The other is to directly use an OCR recognized pipeline, using the OCR results for comparison testing, the data set is highly unbalanced, and the targeted data set is much different from our expected application scenario

Laurence et al. [1] used a Hidden Markov Model (HMM) to identify words that were crossed out, and they focused on words that were crossed out with wavy lines and straight lines. They used the Baum-Welch algorithm on an training set with non-scribbled words and tested it with a mixed dataset of scribbled and non-scribbled words. The results show that there is more than 90% recognition rate for normal text, about 80% for straight line scribbles, and less than half for wavy lines. B. B. Chaudhary et al. [2] proposed a combined model of identifying and localizing the strike-out strokes as well as removing them. They used a SVM with RBF kernel to identify the words with and without scribbles.

Adak et al. [3] used a combination of SVM and CNN to classify scribbled handwritten words, but it is uncertain whether this approach is effective for multiple types of scribbles, such as wavy, linear and rounded ones. Chaudhuri et al. [4] used a SVM classifier to identify text with scribbles and a graph-based shortest path algorithm is used to mark the strokes of the strike through. Accordingly, large-sized multi-word and multi-line cross-outs can be segmented into smaller parts and prevented from entering the OCR for the detected deleted text.

Although all the above methods are effective in the field of text scribble recognition, most of them do not take into account the randomness of scribble marks, which are not always standardized straight and wavy lines, depending on the mood of the writer at the time and other factors, and the existence of mathematical symbols such as fraction lines and root signs in mathematical formulas, leading to the low feasibility of these methods.

4. Our dataset

Due to the specific nature of the problem we are studying, the dataset we use in this research is 632 scanned answer sheets collected from a high school exam, the dataset is then annotated by one of our team member and twice reviewed. The detection of struck-out text is regarded as an object detection task, where the struck-out text is expected to be bounded using rectangle bounding boxes. The annotated bounding boxes are annotated according to two principles: 1. The main body of struck-out text should be inside the box, 2. The wanted, non-struck-out text should be avoided to be inside the box. Theoretically, we could effectively remove the struck-out text while preserving the wanted text by setting the pixels inside the bounding box blank.

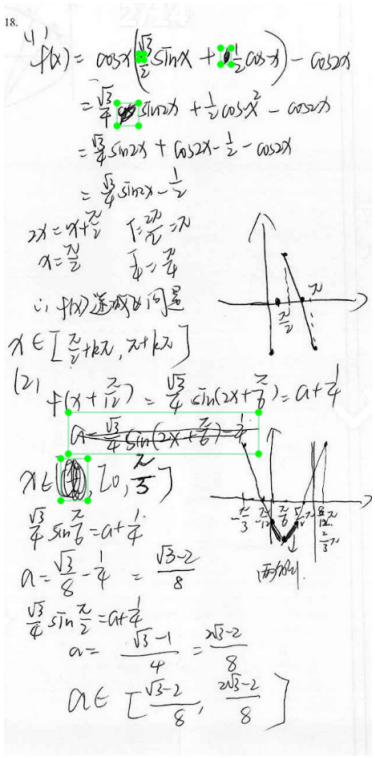

Figure 7. An image in our original dataset

5. Experiment Design

Our experimental design is shown in the figure. We use the self-labeled data set, including 632 math answer sheets,which are randomly divided into train:val:test=8:1:1. We have tried some classic models of one stage,two stage and Transformer as the benchmark, and made improvements on this basis. For yolo and cascade rcnn in benchmark, we tried data enhancement, and there are five main improvement plans for SSD. The first is to conduct ablation experiment, the second is to try common tricks, and the third is to try the previous work on SSD's insufficient detection of small objects. The fourth is to classify the wrong

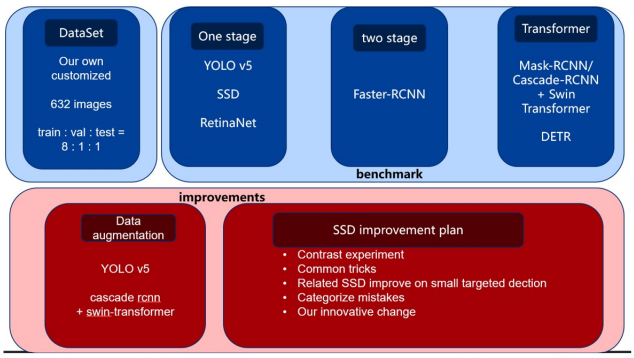

Figure 8. Experiment Design

type, and the fifth is to put forward our own improvement measures.

Our training set would be either a manually labeled or an augmented dataset.

The Experiment Design are show above(Figure 8).

6. Method

6.1. Swin Transformer and YOLOv5

We start our experiments from both model and data perspectives. We choose two detection models, Swin Transformer and YOLOv5. Swin Transformer [5] is a deep learning model based on transformer, which is a transformer with sliding window operations and a hierarchical design. The sliding-window operation includes non-overlapping local window, and overlapping cross-window. limiting the attention computation to one window can not only introduce the localization of CNN convolution operation on the one hand, but also saving the computation.

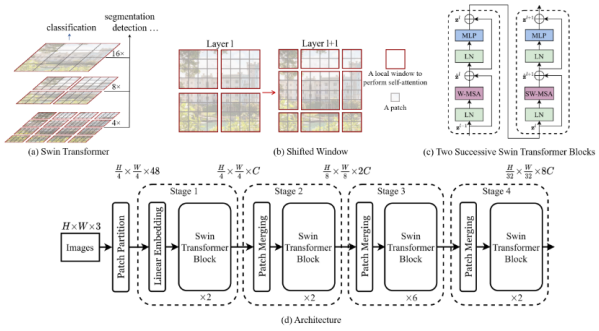

Figure 9. Swin Transformer Architecture

This model still uses patches as in the ViT model. However, instead of using one size as in ViT (16 by 16px), the Swin Transformer first starts with small patches in the first Transformer layer. The model merges these layers into bigger ones in the deeper layers. It takes an image and splits it into 4px by 4px patches. Each patch is a colored image with three channels. Thus, a patch has a total of 48 feature dimensionality. That is, 4 x 4 x 3 = 48. It is then linearly transformed into a dimensionality called C.

So far, compared to ViTs, the image patches are smaller in size. The value, C, determines the size of your Transformer model. For example, we have different BERT model variants. A BERT-Base variant with 748 dimensionalities, and a BERT-Large variant with 1024. In this case, the C is 748 and 1024. The value, C determines the number of hidden parameters in the fully connected layers. The Swin Transformer also has its variants. The Swin-Tiny with C=96, and Swin-Large with C=192. Essentially, 4 x 4 px initial patches are fed as inputs. These patches are converted linearly into C-dimensional vectors. Unlike the ViT model, which processes these vector inputs quadratically, the Swin Transformer employs a clever approach (shifted window approach) when dealing with this issue, preventing the complexity issue from arising.

It adds a linear computation complexity to the image input size. It computes self-attention only within the local window and not globally as is with the ViT model. This feature enables the model to perform dense recognition tasks and allows it to be used for more general-purpose computer vision tasks. The output then gets merged by a merging layer. It concatenates the vectors of groups of 2x2 neighboring patches in the image. Each time the attention window shifts with respect to the previous layer. For example, if in the first layer, the attention was limited to the neighborhood of these regions, in the next layer, the regions are shifted (like in strided convolution). Patches that landed in separate windows in the first layer and could not communicate, can now do so in layer two. These resulting patches are merged by the merging layer. This process is repeated depending on the number of layers chosen.

YOLO v5 does not have many technical improvements compared to YOLO v4, only optimizing the execution speed and model volume, so we give a brief description of the technical advantages of YOLO v4, summarized by an equation,

$$YOLOv4 = CSPDarknet53{+}SPP{+}PAN{+}YOLOv3$$

in which they used a lot of tricks, of which YOLO v4 uses quite a few techniques: at the backbone level, they did CutMix and Mosaic data augmentation, DropBlock regularization, Class label smoothing, etc. At the detector level, they add CIoU-loss, CmBN, DropBlock regularization, Mosaic data augmentation, Self-Adversarial training, and so on.

The other method is data augmentation, where a random rectangular cut is made on the large image to obtain a smaller image, where the cut does not go through the black

pixels or cut through the bounding box, and then the cut subimage is pasted onto the blank large image.

Thirdly, we tried to migrate the model to different types of math questions. The first attempt was to use the model trained on large questions to detect scribbles on fill-in-the-blank questions. But the results were very poor and could barely be recognized. So we continued training and testing the trained large question network on the fill-in-the-blank questions, adding data cross-validation.

Besides, after observing the experimental results of the whole test paper, we found that the recognition effect on the small target was very poor compared to the large target. Considering that the features of small, medium and large targets are different, we need to migrate the generic scribble removal detector to a model more suitable for small scales.

6.2. SSD

6.2.1 Why choice SSD

SSD implements a relatively elegant and simple object detection framework, which uses the first-order network to complete the object detection task, reaching a higher level of object detection at the same time. Overall, SSDS have the following three advantages:

(1) Multi-layer feature map is used for prediction, so although it is a first-order network, under some scenarios and data sets, the detection accuracy can still be comparable with Faster RCNN.

(2) The implementation of first-order network enables its detection speed to exceed that of the Faster RCNN and YOLO algorithms of the same period. SSD is a good choice for engineering application scenarios with high speed requirements.

(3) The network is elegant and simple to implement without too many engineering skills, which also opens up a lot of room for subsequent improvement work.

At the same time, the pursuit of higher detection performance will never stop. SSD algorithm also has the following three limitations:

(1) The detection effect of small objects is mediocre. This is because the shallow feature map with high resolution is used to detect small objects, but the shallow semantic information is insufficient, and the classification and regression prediction cannot be well completed.

(2) The size and width and height of each layer of PriorBox depend on manual Settings and cannot be learned automatically. When the detection task is changed, the debugging process is tedious

(3) As it is a first-order detection algorithm, classification and frame regression are only once. In some scenarios where high precision is pursued, SSD series is still in a lower position than Faster RNCN series.

Since many obliterate in mathematics exercises is small,In our next work,we must focus on addressing and improving the SSD's shortcomings in detecting small targets.

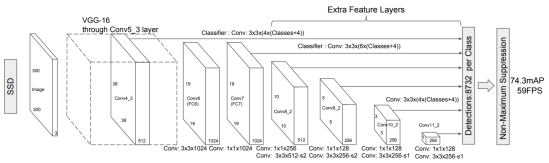

Figure 10. Basic SSD network struct

6.2.2 SSD improving plan

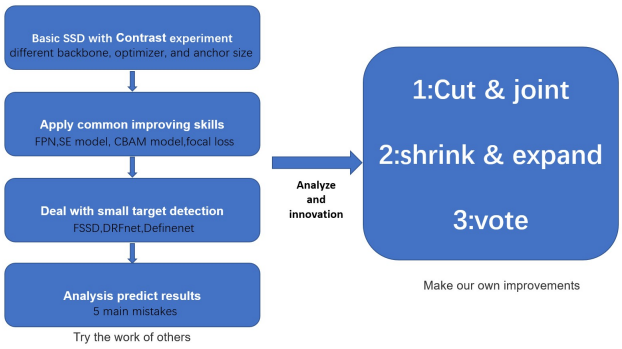

Figure 11. SSD improve flow diagram

The basic net architecture is SSD———Single Shot MultiBox Object Detector based on Pytorch,and I want to make some improvement on it.

First,I do the ablation experiment,with different backbone,different optimizer,different input size and anchor size.I find adjust input size and anchor size will greatly improve the net performance,the best combination we get is vgg+sgd+small achor size+big input size,we can get the Map 75.84.

Second,I apply some common improving skills for SSD,which including FPN,SE model, CBAM model,focal loss.where FPN,SE model, CBAM model will bring a slight improvement,but for focal loss.if we don't carefully adjust its parameters, which will make it less effective.

Third,to deal with the common drawback that SSD in small target detection,we also try DSSD,FSSD,DRFnet and Definenet,which is focus on the improve of SSD in small target.

Fourth, we analyzed the prediction of the existing network prediction results,and also summarize the types and causes of error detection.

Fifth, for several kinds of characteristic detection error cases are found, we have made targeted improvements to these.

6.3. Faster-RCNN, RetinaNet and DETR

6.3.1 Faster-RCNN

As a classical two-stage object detection model, Faster-RCNN [6] introduces a specific end-to-end module Region Proposal Network (RPN). RPN could shares full-image convolutional features with the detection network, thus enabling nearly cost-free region proposals. In nature, like "attention" merchanism, the effect of RPN is tell the whole network where to focus on. By sharing concolutional features, Faster-RCNN combines Fast-RCNN and RPN into a new single model. As for the backbone, unlike the VGG16 [7] used in the vanilla Faster-RCNN, we use the more powerful convolutional extractor ResNet-50 [8] in our experiments.

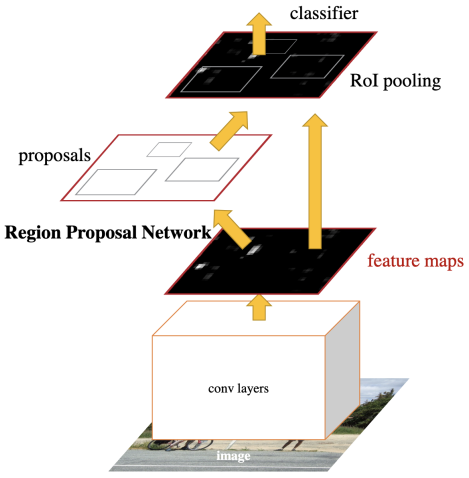

Figure 12. the simple workflow of Faster-RCNN

For the dataset and CUDA device, we apply the strategy identical to DETR [9]' s in the configuration. In the training process, we use two different training strategies. At first, due to the time and GPU limit, we fine-tune the full-size model pretained on VOC2007 [10] dataset. The number of total epochs is 300, and the number of frozen epochs is 150. This means that we train the model in 150 epochs with frozen backbone parameters then train all parameters of the model in the last epochs. As for other hyperparameters, we set initial learning rate as 0.0001, min learning rate as 1e-6, optimizer as Adam [11]. The consuming time is around 8 hours. In the second case, we only use the pretrained ResNet-50 backbone as our initial checkpoint, and the training epochs are also 300.

Regretfully, as for the results, both performances are not satisfactory because their mAPs are below 0.25. The possible reasons are that : 1) the size of dataset is not enough for our tasks. But in fact, the performances for the same dataset used to train YOLO v5 and other models are far better than it. 2) the hyperparameters for training may be more tricky. anchor size, batch size, learning rate and other parameters maybe need to be reset. In the loss curve plot, we find that the downtrend of loss is continuous when the training process is in the end. therefore, more suitable parameter combination is likely necessary.

6.3.2 RetinaNet

Before introducing this model, we need to explain what is focal loss. To decrease the propotion of easy negative (or "background"), the authors of the original paper proposed the new loss function, focal loss. Then in order to verify the effectiveness of focal loss, they presented the new object detection method RetinaNet.

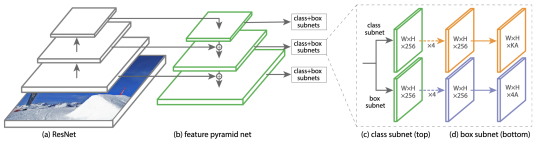

Figure 13. the architecture of RetinaNet

In short, the architecture has two main components: a efficient image feature extraction network (in general, we tend to choose ResNet. In this paper, authors used ResNet-50) and FPN which could implement the feature fusion function. The experimental results show that RetinaNet can reach the detection speed of one-stage approaches but also has the SOTA performance compared to other two-stage methods at that time.

In our experiments, its experimental environment is very similar with Faster-RCNN. The significant configurations are that: initial learning rate is 1e-5, optimizer is Adam, backbone is the ResNet-50 pretrained on COCO 2017 dataset, training epochs are 150. For the training strategy, we choose to fine-tune the whole model based on the pretrained backbone. It is confusing that the performance in validation set and test set is also at the same level as Faster-RCNN's results. Besides the insufficient training, we also argue that the advantage of RetinaNet is to distinguish the negative objects in the multi-classes classification task but there is only one class in our task. Hence, it may lead to the low performance potentially.

6.3.3 DETR

DETR, a powerful Transformer-based end-to-end approach presented by Facebook AI, views object detection as a di-

rect set prediction problem. DETR streamlines the detection pipeline effectively, and does not need to add the anchor generation or other hand-designed components into the workflow. Compared to most previous direct set prediction work, DETR combines the bipartite matching loss funciton and Transformer encoder-decoder architecture. In the COCO 2017 dataset test, the performance is better than Faster-RCNN's.

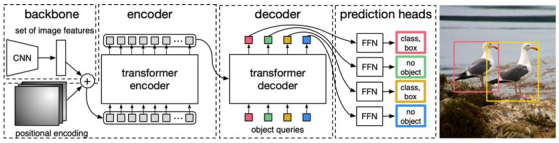

Figure 14. the workflow of DETR

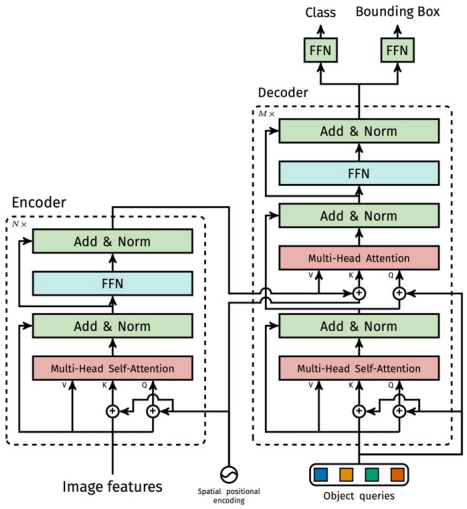

Figure 15. the architecture of DETR

As for the customized dataset in our experiments, it contains X images about hand-written papers. We split the whole dataset into three subsets: training set, validation set and test set, and the ratio is 8:1:1.

In our baseline test, we entirely use the default hyperparameters. The CUDA device is a single node with 4 Tesla K80 gpus. In view of our insufficient computing power for Transformer-based model, we finetune a new model in 300 epochs from a pre-trained DETR-R50 model working on COCO 2017. Therefore, the backbone of feature extractor is the ResNet 50 model pretrained on ImageNet-1K. For some important parameters, we set the initial learning rate as 0.0001, backbone learning rate as 0.00001, optimizer as AdamW. As for other model hyperparameters, they are same as the configurations of vanilla Transformer.

In practice, we find that the training process based on pre-trained backbone model is not stable. Original DETR codes published in Github only provide the COCO dataset pipeline, and there will be bugs in the training process when we transform our own dataset into COCO-style dataset. Hence, we apply the pre-trained DETR-R50 model released by official group into our fine-tuning.

6.4. Mask R-CNN

Mask R-CNN is a paper in ICCV 2017, in general, Mask R-CNN = Faster R-CNN + FCN, by adding a branching network on top of Faster-RCNN to segment the target instances while achieving target detection. Mask R-CNN also uses two steps, the first step is RPN to extract the candidate regions, and in the second step, for each RoI, Mask R-CNN outputs a binary mask. This is different from most of the current systems, where the category classification depends on the prediction of the mask.

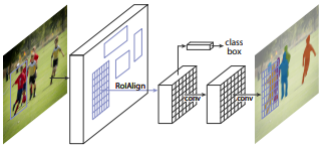

Figure 16. The Mask R-CNN framework for instance segmentation

Mask R-CNN is structurally roughly the same as Faster R-CNN except for some minor changes, and also consists of four parts:

1, Shared Conv Layers(backbone): ResNet and ResNeXt 50/101+FPN are used in Mask R-CNN, and more expressive networks are used to extract features, while FPN is used to extract multi-scale information.

2, Region Proposal Network (RPN): replaces the traditional method to generate region proposal with neural network.

3, RoI Align: Instead of RoI pooling, RoI Align is used to solve the misalignment problem.

4, Three branches: This part is compared with Faster R-CNN only with one more Mask prediction branch, so this part mainly accomplishes three tasks: (1) Classification: after FC layer + softmax for classification, mainly for classification of object proposal, including a total of (K+1) classes, i.e., K classes of objects plus 1 background class. (2) Regression: After FC layer + bbox regressor output, this is the output of 4 values for each of the K classes (K*4), and these 4 values represent the refined bounding box position.(3) Mask prediction: After FCN network to perform pixel-by-pixel mask prediction.

The most critical improvements are RolAlign and the mask prediction branch that follows. Regarding RolAlign:

(1) First map the RoI (solid line) to the feature map (dashed line), where no rounding operation is performed.

(2) After that, the feature map corresponding to RoI is also divided into 2*2 bins, assuming that there are 4 sampling points in each bin, all of which are not integer coordinates and do not correspond to specific values, so bilinear interpolation is needed to value each sampling point separately.

(3) After the interpolation is completed, each bin is pooled, i.e., the four points in the bin are averaged (Average Pooling) or maximized (Max Pooling), and each bin gets a value, so the final 2*2 feature map is obtained after a pooling.

After RoI Align, the feature map is fed into three branches in parallel, where the classification and regression branches are the same as in Faster R-CNN, and the Mask prediction branch is used to predict the object's mask, using FCN full convolutional network to maximize the spatial location information. For each RoI, the branch outputs K m*m binary masks, where K is the number of categories of objects, and in the binary mask, 0 represents the background and 1 represents the object, and the mask value greater than 0.5 is set to 1 and less than 0.5 is set to 0 in the actual operation.

The advantages of Mask R-CNN are:

(1) A strengthened base network, ResNeXt-101 + FPN is used to extract the network, ResNeXt-101 has a stronger feature representation, while FPN is able to extract some multi-scale information, which is also beneficial to detect objects at different scales.

(2) Using RoI Align instead of RoI pooling allows the exact spatial location to be preserved, albeit with a small change, improving the accuracy of mask prediction by 10%-50%.

(3) The network is versatile and able to perform multiple tasks: segmentation, detection, human keypoint recognition, etc.

7. Results

The results of our 8 benchmark are show below.(Table 1)And the results of improve method will be include in improvement part.The Anlysis of the results will be included in Conclusion.

mAP, short for mean of average precision, is a measure of model performance in object detection. In object detection, accuracy in classification is not applicable due to the presence of object location box. Therefore, the unique mAP index of object detection is proposed.

The AP(Average Precision) calculation is basically the same as calculating the area under the PR curve, but it is slightly different. You need to smooth out the PR curve first. The method is that the precision ratio p corresponding to the recall ratio r is the maximum precision ratio p when the recall ratio is greater than or equal to r.

$$p(r) = \max_{\tilde{r} \geq r} p(\tilde{r})$$

Calculation of AP value is only for one category. After obtaining AP, calculation of mAP becomes very simple. It is to calculate AP of all categories and then take the average value. mAP measures the ability of a trained model to detect all categories. Suppose there are K categories and K¿1, then the formula for calculating mAP is as follows:

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

When K=1, mAP = AP. The mAP calculation method defined by Pascal VOC's new standard can be considered as the standard calculation method of mAP. The COCO Challenge defines 12 mAP calculations, typical of which are:

Map50 Indicates the mAP calculated when the IOU threshold is set to 0.5.

The mAP50:90 (mAP @ [, 5, 95]) said in [0.5, 0.95] threshold range, step length of 0.5 average mAP.

We use various evaluation criteria includes AP, AP50-95, AP75, AP-small, AP-medium, AP-large.

8. Discussion

8.1. Analysis of result

Error types of prediction results can be classified into the following four categories:

Error Category 1: A function curve in a function image is treated as a scratch intended to remove that area.

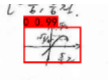

Figure 17. Error category 1

Error Category 2: Text is used to delete scratches.

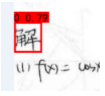

Figure 18. Error category 2

Error Category 3: Some horizontal scratches are difficult to detect. (Too thin, too sparse)

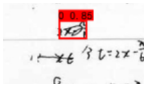

Figure 19. Error category 3

Error Category 4: Only partial targets are detected.

Table 1. Performance of original models on the original dataset

Models	$AP_{50:95}$	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
YOLO v5	0.525	0.826	-	-	-	-
SSD	0.473	0.758	-	-	-	-
DETR (ResNet 50)	0.494	0.859	0.532	0.400	0.552	0.681
RetinaNet (ResNet50)	0.254	0.488	0.235	0.081	0.417	0.370
Faster-RCNN (ResNet50)	0.188	0.475	0.073	0.049	0.252	0.475
Mask-RCNN (Swin-tiny)	0.631	0.903	0.777	0.517	0.704	0.764
Mask-RCNN (Swin-small)	0.613	0.906	0.755	0.509	0.681	0.716
Cascade Mask-RCNN (Swin-small)	0.658	0.915	0.780	0.536	0.737	0.798

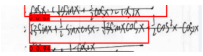

Figure 20. Error category 4

Error Category 5: Special cases cannot be detected. There are some redactions where the change is written directly onto the wrong content, as in the figure below cos, and other things that even the human eye cannot tell.

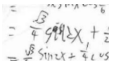

Figure 21. Error category 5

8.2. Improvements(Key point in this report)

8.2.1 Data Augmentation on yolov5swim-transformer

Data augmentation is a technique used to artificially increase the size of a dataset by generating additional data samples. This is done by applying random transformations to the existing data, such as rotating an image, scaling it up or down, cropping it, or adding noise. The idea is that these transformations will preserve the underlying features of the data while adding diversity to the dataset, which can help improve the performance of machine learning models. In image processing, data augmentation is widely used to improve the robustness of models and reduce overfitting.

To seek further improvement of the existing dataset and models, we apply the cut-transform-paste augmentation method. This augmentation method consists of two steps:

(1) For an original image, randomly choose a coordinate as the center, then choose a random width and a random height. Using the 4 parameters (2 for the center, 1 each for width and height) we can locate a rectangle area, which is a cut of the original image. During this process, the ratio of width and height is controlled between 0.5 10. Also, the cut should not intervene black pixels, which is, the boundary of the cut should include no black pixels. Lastly, for each bounding box of the original image, it should either be completely inside the cut image, or completely outside the cut image. The images cut out in this step are called sub-images. Any bounding boxes inside both the original image and the sub-image are transferred to the sub-image.

(2) To generate an augmented image, we use a completely blank image as the basis, then randomly stick the transformed sub-images to the basis to form the augmentation image. For each stick step, first randomly choose a sub-image, then do a series of transformations to the sub-image, including random zooming, random stretch, random Gaussian Filter; if the sub-image contains no bounding box, then it is additionally randomly flipped between -10° and 10°. The sticking process is done by replacing the corresponding pixels of the augmentation image with the corresponding pixels of the transformed sub-images. When sticking each transformed sub-image, we ensure no black pixels are overlapped, which is, if a pixel of the augmentation image is black and the pixel to be overlapped on it is also black, the sticking process is aborted. This ensures no direct text overlap happens during the augmentation process, reducing the noise introduced during the process.

We generated 6320 augmentation images and use them as supplement training set for further training.

We tested the augmentation data by training the 2 of models with best performance, attempting to further improve their performance. We use the augmentation data for the further training of the Cascade Masked-RCNN Swin Transformer and YOLO v5. The hyper-parameters are kept same during the training process.

The result shows that the augmentation data generally do not improve the performance of the two models. The estimated reasons are:

1. The augmentation method is very rough. Especially, the rotation process introduce extra faulty text or clipped pieces, which introduce unnecessary noise to the augmentation data. The distribution of the augmentation data and

Table 2. Performance of two models on the original dataset and the augmentation dataset

Models	$AP_{50:95}$	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Cascade Mask-RCNN (Swin-small)	0.658	0.915	0.780	0.536	0.737	0.798
Cascade Mask-RCNN (Swin-small) + Augmentation	0.596	0.888	0.692	0.507	0.694	0.677
YOLO v5	0.525	0.826	-	-	-	-
YOLO v5 + Augmentation	0.469	0.838	-	-	-	-

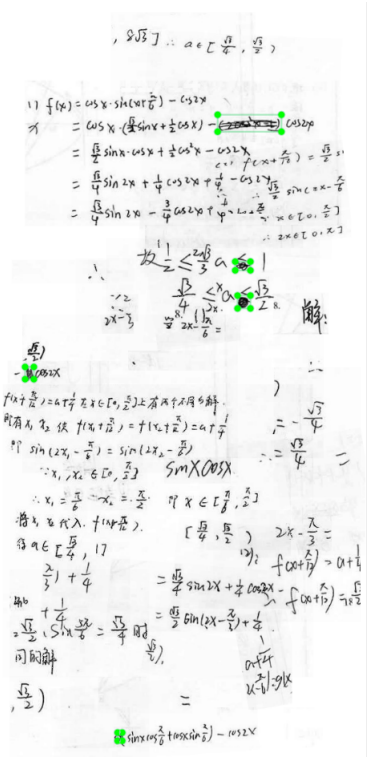

Figure 22. An example of augmentation image

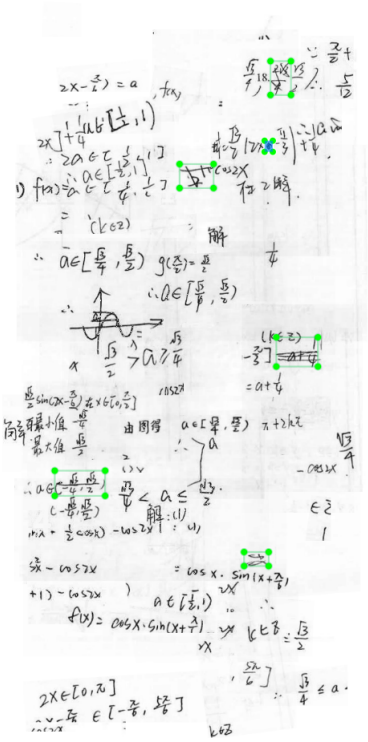

Figure 23. An example of augmentation image

the original data might be very different.

2. The original data is already plenty in number and variance, while the augmentation data is relatively poor in quality, thus the augmentation data does no help.

3. The hyper-parameters of the models are not changed and tuned when training using the augmentation data. However, with extra amount of data, the old hyper-parameters might not be optimal, further causing the low performance of the models trained on augmentation data.

8.2.2 ablation experiment on basic SSD

Advantages of each of the three backbone:

Vgg16: multiple 3 * 3 convolution kernels are stacked to replace large-scale convolution kernels and reduce required parameters

Resnet50: easy to optimize and can improve accuracy by adding considerable depth.

Mobilenetv2: greatly reduce the amount of computation and the number of parameters

For SSD,through our experiments, we found that vgg is the best backbone, even though it is the slowest of the three,it has the highest map. I find adjust input size and anchor size will greatly improve the net performance, We only use the default detection of small objects input size and anchor size provided by SSD, which could achieve an 11.37% improvement in map. The best one results are shown in the figure 5-8. And the all result show in table 1.

8.2.3 some common improving skills on SSD

8.2.3.1 combine SSD with FPN

firstly we combine SSD with FPN. FPN is a network structure that utilizes multi-scale feature graphs inherent in deep convolutional neural networks, and by adding lat-

Table 3. ablation experiment on basic SSD

frame	*BackBone*	*BackBone*	*BackBone*	*Adjust*	*Adjust*	*optimizer*	*optimizer*		
	vgg16	*resnet50*	*mobilenetv2*	*anchors_size*	*input_shape*	*sgd*	*adam*	AP_{50}	FPS
			✓			✓		33.76	83.17
			✓	✓		✓		41.34	77.67
SSD		✓		✓		✓		47.63	96.36
	✓			✓		✓		61.41	105.33
	✓			✓			✓	62.23	104.93
	✓			✓	✓		✓	74.98	50.60
	✓			✓	✓	✓		75.84	51.53

1 run on GPU:NVIDIA TITAN X(250W/12196MiB)

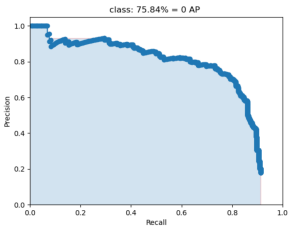

Figure 24. mAP

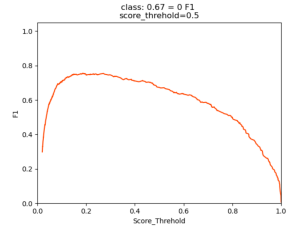

Figure 25. F1-score

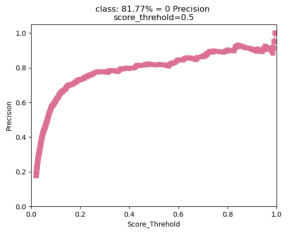

Figure 26. Precision

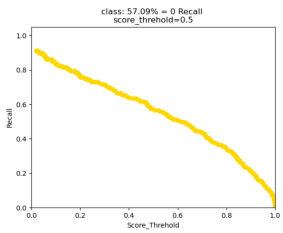

Figure 27. Recall

eral connections and up-sampling, constructs feature pyramids of different scales with high-level semantic information with minimal additional computation.

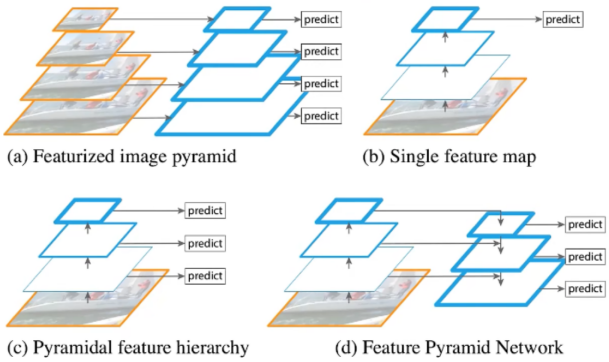

Figure 28. Error category 1

where

(a) is the pyramid structure of feature image. When detecting targets of different scales, the images will be scaled, and the images of each scale will be predicted in turn. (Slow speed)

(b) is a single feature map structure. The final feature map is obtained by backbone, and the prediction is made on the final feature map. (Not friendly to small targets)

(c) For the pyramid feature hierarchy, images are input to the backbone, and prediction is made on different feature maps obtained by the forward transmission process of the backbone. (Small goals are easily misclassified)

(d) is the feature graph pyramid network. The features on different feature graphs are fused, and then the prediction is made on the fused feature graphs. FPN is a network structure that utilizes multi-scale feature graphs inherent in deep convolutional neural networks, and by adding lateral connections and up-sampling, constructs feature pyramids of different scales with high-level semantic information with minimal additional computation.

For object detection model, FPN structure is not an independent module in the model, but an additional item of the original Backbone, which is integrated in the convolutional neural network.

8.2.3.2 replace cross-entropy loss with focal loss

We implemented a focal loss function and ability to easily select between cross-entropy and focal loss using command line arguments.

which means replace

$$\begin{aligned} \mathrm{L} &= -y\log y' - (1-y)\log(1-y') \\ &= \begin{cases} -\log y', & y=1 \\ -\log(1-y'), & y=0 \end{cases} \end{aligned}$$

to

$$\mathrm{L}_{fl} = \begin{cases} -\alpha\left(1-y'\right)^{\gamma}\log y', & y=1 \\ -(1-\alpha)y'^{\gamma}\log\left(1-y'\right), & y=0 \end{cases}$$

Focal loss has two main functions: Prevent the detector from being disabled. This is very common in the one-stage detection network, the network running running will not converge, the final effect is not good. focal loss can reduce the contribution of a well-predicted sample, such as background, to loss, so that the network can spend more effort to learn the normal target that is difficult to predict. Automatically solve the problem of unbalanced positive and negative samples of a stage detector.

But in our experiment,focal loss makes it less effective, after discussion, we think that focal loss actually has two adjustable parameters, alpha and gamma. A lot of people should use the default values, but these two values actually have a lot to do with the data set, and it is obviously not effective to set without experiment

8.2.3.3 add self-attention model

We tried to add some self-attention models, SE and CBAM.

SE model

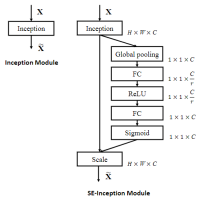

Figure 29. SE model

The SE module first Squeeze the feature map obtained by convolution to get the channel-level global features. Then, the global features are Excitation Excitation to learn the relationship between all channels and get the weights of different channels. Finally, multiply the original feature map to get the final features. Essentially, the SE module provides attention or gating for the channel dimension. This attention mechanism allows the model to focus on the most informative channel features while suppressing the less important channel features. Another point is that the SE module is generic, which means it can be embedded into existing network architectures.

CBAM model

Convolutional Block Attention Module (CBAM) indicates the attention mechanism module of the convolutional block. It is a kind of attention mechanism module combining spatial and channel. Compared with SeNet's attention

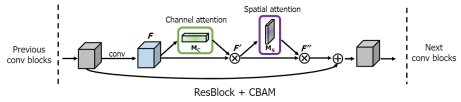

Figure 30. CBAM model

mechanism that only focuses on channels, better results can be achieved. The arbitrary structure of the models and test results are shown below,in which

$$\begin{aligned}\mathbf{M_c}(\mathbf{F}) &= \sigma(\mathrm{MLP}(\mathrm{AvgPool}(\mathbf{F})) + \mathrm{MLP}(\mathrm{MaxPool}(\mathbf{F}))) \\ &= \sigma\left(\mathbf{W_1}\left(\mathbf{W_0}\left(\mathbf{F^c_{avg}}\right)\right) + \mathbf{W_1}\left(\mathbf{W_0}\left(\mathbf{F^c_{\max}}\right)\right),\right.\end{aligned}$$

where σ denotes the sigmoid function, $\mathbf{W_0} \in \mathbb{R}^{C/r \times C}$, and $\mathbf{W_1} \in \mathbb{R}^{C \times C/r}$. Note that the MLP weights, $mathbfW_0$ and $\mathbf{W_1}$, are shared for both inputs and the ReLU activation function is followed by $\mathbf{W_0}$.

$$\begin{aligned}\mathbf{M_s}(\mathbf{F}) &= \sigma\left(f^{7\times7}([\mathrm{AvgPool}(\mathbf{F}); \mathrm{MaxPool}(\mathbf{F})])\right) \\ &= \sigma\left(f^{7\times7}\left(\left[\mathbf{F^s_{avg}}; \mathbf{F^s_{max}}\right]\right)\right)\end{aligned}$$

where σ denotes the sigmoid function and $f^{7\times7}$ represents a convolution operation with the filter size of 7×7 .

The way we use these tricks is shown below.On the basis of the original SSD architecture, we added: one is FPN, which can fuse information of different layers. We chose to splice the layers of the adjacent three layers together. One is to add an attention module after the convolutional layer, CE module focuses on channel, CBAM module takes care of both channel and space

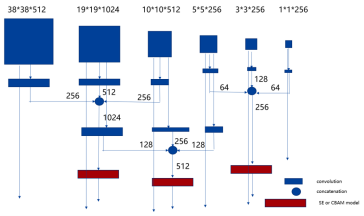

Figure 31. The way we apply these tricks

For FPN and self-attention module, there is about 1 point of improvement We also tried to replace the cross information entropy with focal loss, but it went down by 3 points

8.2.4 Deal with small target detection on SSD

The predecessors already did some work on improving SSD specifically in small targets detection. We tested DRFNet, FSSD and Refinenet.

DRFnet

Table 4. common tricks on basic SSD

$frame$	$CBAM$	SE		
	$+FPN$	$+FPN$	$Focalloss$	AP_{50}
	✓			76.96
SSD		✓		76.74
			✓	73.14
baseline				**75.84**

The core idea of DRFNet consists of a shared feature extractor and two effective refinement heads. By decoupling details and context information, a refinement header adopts a globally aware feature pyramid. It can enhance spatial details and narrow the gap between high-level semantics and low-level details without adding too much computational burden. Meanwhile, the other refinement head uses mixed expanded convolution blocks and grouped upsampling, which is very effective in extracting context information. Based on double refinement, our method can enlarge the receptive field and obtain more discriminant features from high resolution images.

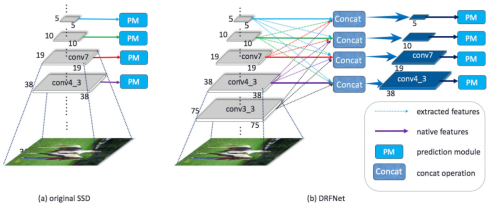

Figure 32. SSD and DRFNet

RefineNet

The core ideal of RefineNet consists of two interconnected modules, the anchor refinement module and the object detection module. Specifically, the purpose of the former is: (1) to filter out the negative anchor points and reduce the search space of the classifier; (2) The position and size of the anchor point are roughly adjusted to provide a better initial position for the subsequent regressors. The latter module takes the improved anchor as input to further improve regression and predict multiple classes of tags. At the same time, we design a transmission link block to transmit the features in the anchor refinement module to predict the position, size and class label of the target in the object detection module. The multi-task loss function enables us to train the entire network in an end-to-end manner.

FSSD

FSSD is SSD with a lightweight and efficient feature fusion module. The architecture of the feature fusion module is as follows: The features of different scales in different layers are projected together, and then the batch normalization layer is used to normalize the feature values. Then

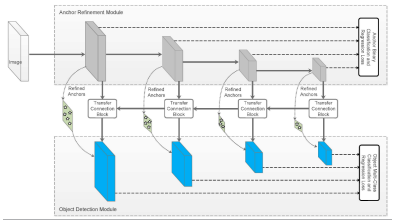

Figure 33. RefineNet

some additional drop sampling blocks are attached to generate the new feature pyramid and fed back to the multi-box detector to produce the final detection weight result.

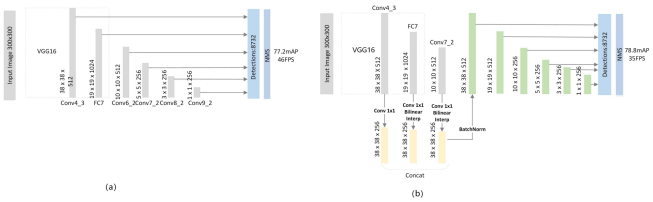

Figure 34. SSD vs F-SSD

The final results are show below.

Table 5. improve for SSD in small target detection

$frame$	$FSSD$	$DRFnet$	$Definenet$	AP_{50}
			✓	82.26
SSD		✓		80.17
	✓			79.45
baseline				**75.84**

8.2.5 Our innovative work on SSD

1:Cut the whole picture into small pieces and examine them separately then joint them back.

Origin picture is 1297*636,we cut many 300*300 patches with a vertical and horizontal stride (shift) of 150 pixels in the origin graph,where the numbers all are hyperparameters.This cut operator makes Strike-Out text large

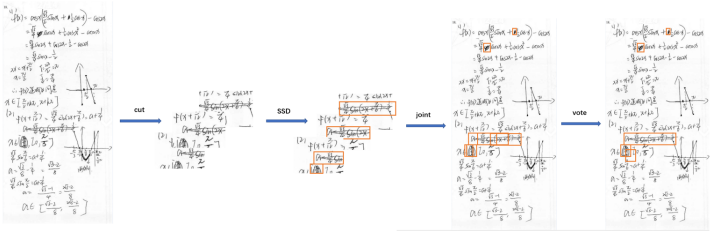

Figure 35. innovative improvement flow on SSD

enough for SSD more easily to detect.If we do this, we also need to adjust ground truth at the same time to fit each patch.And since the input and patches are fixed size,total patch number and each position is fixed,so it's easy to put the cut pieces back in origin place. And since the input size can't be exactly divided by patches,e.t. 636 %300=36,for this part,at the start,we try to fill it with zero,and when we joint it,we just cut the part we fill,but this will result that maybe a bounding box will be cut in the half,it's hard to fix it,so I choice to give up this part, obviously, some places can't be inspected,which will lead to a decrease in effectiveness.

2:adjust the ratio of default bounding box.

The original SSD architecture uses aspect ratios (width/height) of 1, 2, 3, 1/2, 1/3.but in our Error Category 4,we will see some Strike-Out text is very long,so we add aspect ratios 5, 7, 10 after the default ratio,which makes the the default bounding box can cover longer areas.

3:Scale the border to fit the writing

expand and shrink the upper, lower, left and right boundary of detection area to touch the text.We do Scale twise before voting on mutiple detection box and after voting on final detection box.

Before begin,we set all the pixel which is background to 0,else to 1.

First we shrink the bounding box.we use np.any on the area circled by it.Where there is a writing,it will be targeted.We find the max and the min of the coordinate of them,and shrink the bounding box.

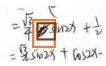

Figure 36. shrink

where orange bounding box is origin box,brown bounding box is after shrinking.

Second we expand the bounding box.we set all pixels inside it to targeted 1.Then we use very traditional search algorithm BFS,start on the pixels on the outermost of the box,to search until we find all around is 0,stop.

Figure 37. expand

where orange bounding box is origin box,brown bounding box is after expanding.

4:Finally, we use the voting function designed by us to determine the final border.

Let B represent the set of all bounding box we product.B_i be one of bounding box,and C_i is the confidence of it. if one pixel belongs to B_i,we will have

$$L_{ab}^i = \begin{cases} 1 & \text{if } P_{ab} \in B_i \\ 0 & \text{if } P_{ab} \notin B_i \end{cases}$$

Then,for each pixel,we will recalculate its confidence by average,which is our vote function.

$$\sum_{i=0}^{|B|} L_{ab}^i C_i / \sum_{i=0}^{|B|} L_{ab}^i$$

The origin SSD work on a misstion with 21 class,each class has its own confidence,then do softmax on them.But now we only need classify one class,so each pixel in one detection area share one confidence,if one pixel belongs to mang detection areas,we will add them together and then average them.Later,we will pick and throw away the pixel wich confidence less than the threshold that we set 0.53(we also try some other threshold,but this one is the best).we set them to 0 just like background. Then we use scipy.ndimage.measurements.label,which divide whole picture into few Connecting area,for each area,we caculate the max and the min of the coordinate of the pixels,use this to represent the final voting result.

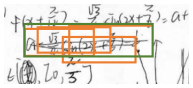

Figure 38. vote

where orange bounding box is origin mutiple boxs on same target,grean one is final result after voting.

The final result is shown below.

Table 6. The result of our innovative work

$frame$	AP_{50}
Baseline	75.84
our innovation	34.13

9. Other related work we done

We collected and annotated this math homework handwriting dataset ourselves and write a script,which allow dataset transformation between yolo,voc and coco format.

10. Conclusion

Since there are few previous studies in this field and we use self-labeled data set, it is difficult for us to find a suitable baseline to compare with our experiment, so we can only compare it among various models we have tried. The results of each Benchmark are shown in the figure. We find that Mask-RCNN based on Swin as backbone can achieve excellent results, while Faster-RCNN has poor effect no matter using original parameters or heavy training. However,

we also understand that there is a huge difference between the parameters of each model, and the high accuracy of a model may be because it needs longer quantitative time and more data. So the more scientific comparison is between our improvements to each model and the original model.

For our first attempt at data augmentation, In the two models with the best performance, Cascade Swin and Yolov5, we used the augmenting data for supplementary training and found that AP performance basically decreased. The possible reasons are: The amplification method is rough, introduces too much noise, and may even be different from the distribution of the original data. And the original data may have been sufficient in quantity and quality, so that augmentation can not bring effect improvement; In addition, in the data enhancement test, we used additional enhanced data, and other parameters remained unchanged, so it is possible that the original superparameter is no longer optimal due to the change of data.

The second attempt we made was an incremental improvement to the SSD.

The first is a comparison experiment on the original SSD, We found that vgg16 is the best backbone, Also, increasing the input size and reducing the anchor box size can greatly improve the map, but it is also more time consuming. Adn there is little difference between optimizers.

Second, I tried the tircks common to SSDS As shown in the figure on the right, we added FPN on the basis of the original SSD architecture: one is FPN, which can fuse the information of different layers. We chose to splice the layers of the three adjacent layers together. One is to add an attention module after the convolutional layer, CE module focuses on channel, CBAM takes care of both the channel and the space, For FPN and self-attention module, there is about 1 point of improvement We also tried to replace the cross information entropy with focal loss, but it went down by 3 points.

The third attempt is the insufficient work in SSD small object detection by predecessors Because small objects can only be detected by the shallow anchor frame, and the shallow layer information is not very rich, the effect of the original SSD is not good All three are designed to solve this problem.

Refinenet uses chain residuals, FSSD uses other layer information in vgg16, and DRFnet uses more scientific FPNS We then categorized the top 5 mistakes and made innovative improvements based on them.

We ended up doing something innovative of our own, We cut the original image into different pieces, input them into the network respectively, and then put the output back in place. Finally, we use voting to combine multiple anchor boxes predicting the same area into one We vote on the average confidence level, which is reserved only if it is greater than the threshold of 0.52, We used the shrink module in two places, when cutting into multiple pieces and when final voting, First, it shrinks the anchor frame to the boundary of the words inside the frame, Then, after expansion, mark the inside of the anchor frame, and BFS search from the boundary until a pixel is surrounded by a white background, to determine the expansion boundary, But the effect is diminished.

References

[1] Likforman-Sulem, Laurence and Alessandro Vinciarelli. "HMM-based Offline Recognition of Handwritten Words Crossed Out with Different Kinds of Strokes." (2008). 3

[2] Mioulet, B. B. Chaudhary, C. Chatelain, T. Paquet, "Unconstrained Bengali handwriting recognition with recurrent models", Proc. ICDAR, pp. 1056-1060, 2015. 3

[3] Adak, C., Chaudhuri, B.B., Blumenstein, M.: Impact of struck-out text on writer identification. In: Proceedings of IJCNN, pp. 1465-1471 (2017). 3

[4] Chaudhuri, B.B., Adak, C.: An approach for detecting and cleaning of struck-out handwritten text. Pattern Recogn. 61, 282–294 (2017). 3

[5] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021. 4

[6] Ren S, He K, Girshick R, et al. Faster r-cnn: Towards real-time object detection with region proposal networks[J]. Advances in neural information processing systems, 2015, 28. 6

[7] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition[J]. arXiv preprint arXiv:1409.1556, 2014. 6

[8] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 770-778. 6

[9] Carion N, Massa F, Synnaeve G, et al. End-to-end object detection with transformers[C]//European conference on computer vision. Springer, Cham, 2020: 213-229. 6

[10] Everingham M. The pascal visual object classes challenge,(voc2007) results[J]. http://pascallin. ecs. soton. ac. uk/challenges/VOC/voc2007/index. html., 2007. 6

[11] Kingma, Diederik P., and Jimmy Ba. ”Adam: A method for stochastic optimization.” arXiv preprint arXiv:1412.6980 (2014) 6

[12] 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.