Car Damage Detection

Project Report

Jiangnan Huang	Yu Wu	Yuliang Nie	Wei Zhou	Ruining Wang
03733766	03750271	03752438	03754271	03752560
Xiaoming Zhou	Wenwen Chen	Xiangyuan Peng	Wang Zechen	Zhonghe Ren
03749734	03752435	03750109	03750447	03749679

August 31, 2022

Motivation

Due to the increase in traffic accidents, there are numerous damaged vehicles. Vehicle damage detection is one of the most important primary activities of the vehicle industry. It is particularly important for insurance companies to detect scratches and determine their type. Furthermore, with the rise of new business models like car sharing, it is also essential for service providers to register damages of the rented vehicles.

Commonly, heavy collisions usually cause severe and easily visible damages; moreover, the vehicle may not be operational anyway. The heavy damages are easy to identify by a human. Nevertheless, small damages such as scratches or dents are easily neglected by manual inspection. In addition, manual inspection is labor-intensive and takes more time.

In order to make the detecting process more efficient and precise, we propose a machine intelligence system to detect car damages as assistance and supplement to manual testing. In the past several years, object detection tasks based on the deep neural network have achieved relatively high accuracy. Hence, our work mainly utilizes the YOLOv5 approach to implement damage detection and constructs a benchmark with methods Faster-RCNN and SVM with HoG (Histograms of oriented gradients). With the assistance of our system, the detection of tiny damages caused by low-impact forces can become more efficient, which are probably not immediately visible to the human eye.

---

1 Project Description

The project mainly focuses on the detection of damages to vehicles. A large amount of unlabeled data is provided, which are photos taken from different damaged cars with distinct perspectives. Based on these photos, we need to train a model to distinguish scratches, dents, rims, and other damages. By constantly modifying the annotations and model parameters, we search for the solution with the highest prediction accuracy.

Furthermore, several object detection or classification models are also implemented in our experiments such as YOLOv5, Faster-RCNN, and SVM. A Cross-validation scheme is utilized to decide the best training parameters. Our project is deployed through Docker and is displayed through the web front end.

Finally, we produce a video to clarify the whole project process, including brainstorming, planning and scheduling, group discussions, and work integration. We make use of animation to introduce our implementation scheme and make our training strategy easy to understand. The final operation on the web front end is also presented in our video.

1.1 Research Question

When a vehicle encounters a traffic accident, it would be damaged to different degrees, such as scratches, dents, etc. Traditionally, scratches are judged by the naked eye, but this is often subjective because the human eye's judgment is often disturbed by external factors. With advances in the field of computer vision and artificial intelligence, models can be built to solve these types of problems.

When it comes to the specific project, the first problem is to annotate the images manually, since the available data is unlabeled. The difficulty lies in the precise recognition of the small damages by the naked eye. Some of the small damages are hard to identify because of light or shooting angles. The uneven lighting could also cause annotators to mistake a certain body structure for a dent. Depending on the results of training and testing, the ambiguous labels should be confirmed and corrected by the annotators again to complete the active learning.

The second question is to select a model with high accuracy and high speed to detect vehicle damages. Although there are several excellent machine learning models in the field of object detection, they may no longer be able to guarantee high performance in new application scenarios. Thus, it is necessary to compare the performance among different models and choose the best one.

Problems can also arise at the final stage of integrating the individual work, especially the web front end part that contains several functions such as image labeling, predicting, information downloading, etc. This question needs to be solved with the cooperation of team members.

1.2 Goals

Our task is to build a model that is able to distinguish different types of vehicle damages. The computer vision and machine learning techniques are utilized to automatically identify damages in the photos and determine the types and severity. Our proposed system is supposed to help to re-

duce human error and avoid subjective decision-making, which is able to detect vehicles damages precisely and efficiently.

What's more, the comparison among different machine learning models would also be implemented. The performance and generalization abilities of different methods would be discussed and analyzed.

Finally, in the presentation phase, the web page is supposed to designed to be as beautiful and user-friendly as possible. The video would try to make the explanation lively and interesting, and show the efforts of every team members.

2 Data prepossessing

Data prepossessing is a required first step before any machine learning machinery can be applied, because the algorithms learn from the data and the learning outcome for problem-solving heavily depends on the proper data needed to solve a particular problem.

2.1 Data Cleaning

The first step of data prepossessing is to delete invalid data, and manually delete some data that do not contain cars and some data that have no damage to cars.

Figure 1: Delete invalid data

In the category of other, it contains a lot of Images that do not belong to the other three categories, and there are too many types, such as white spot stains, black spot stains, and paint peeling. Too many categories are grouped together, which is inconvenient for model training and learning, so we only leave one kind of black spot stain because it has the largest number.

2.2 Data Conversion

In the second step, since the obtained data format is not consistent, it is also necessary to uniformly convert all the data into the same format JPG.

2.3 Data Combination

The third step, in the process of data prepossessing, we originally planned to divide the pictures into three categories: upper, middle and lower. However, due to the small number of pictures from each perspective, we finally merged all the pictures. The controversial point is that there is a category of dent in the damage, which is not easy to distinguish whether it is the damage caused by the collision or the dent caused by the car's own design. In the end, we decided to combine all possibilities.

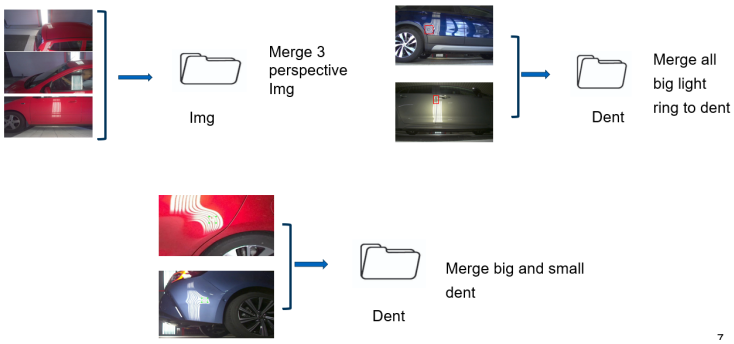


Figure 2: Merge Images

2.4 Challenges

After the annotation of the previous data, new problems also appeared in the training of the model. First, there are too few images in the data set that are obviously damaged and used for training. Amount of valid data is too poor, raindrops on the car could affect the test results, requirements for the environment. The car-body can not be too dirty. Besides, the damage area is usually small and the data for each class is not balanced. There are also some pictures with blurred light and unclear vision. All these make the training of the model face new challenges.

For smaller damages, localization becomes relatively difficult, which is a challenge for target aggregation. And the situation of occlusion, blur and incomplete phenomenon also may appear. So this requires the model to have higher localization capabilities. Therefore, we have adjusted the original plan accordingly.

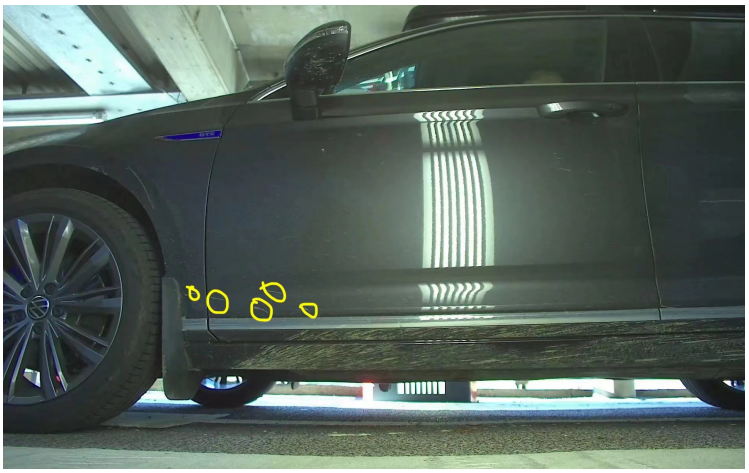


Figure 3: Small damages

3 Implemented Models

3.1 HoG+SVM Model

HOG + SVM is an object detection technique that can be split into two parts: feature extraction and model training; HOG is the full name of Histograms of Oriented Gradients, a feature extraction technique that uses the direction of Gradient in a block to separately count the accumulated Gradient strengths and use them as features. The full name of SVM is Support Vector Machine, which is a machine learning technique. In simple terms, SVM is a technique to find a hyperplane to correctly distinguish different types of data.

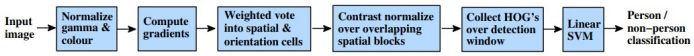


Figure 4: Flowchart for HoG+SVM

So the process of HOG + SVM is to extract the features from the image by HOG and send them to SVM to determine whether there is damage on the block.

3.1.1 HoG

The HOG feature descriptor converts a 3-channel color image into a feature vector of a certain length.

Then we need to define what is "useful" and what is "irrelevant". Obviously, feature vectors are not useful for viewing images, but they are useful for tasks such as image recognition and target detection. When these feature vectors are fed into an image classification algorithm like Support Vector Machine (SVM), better results are obtained.

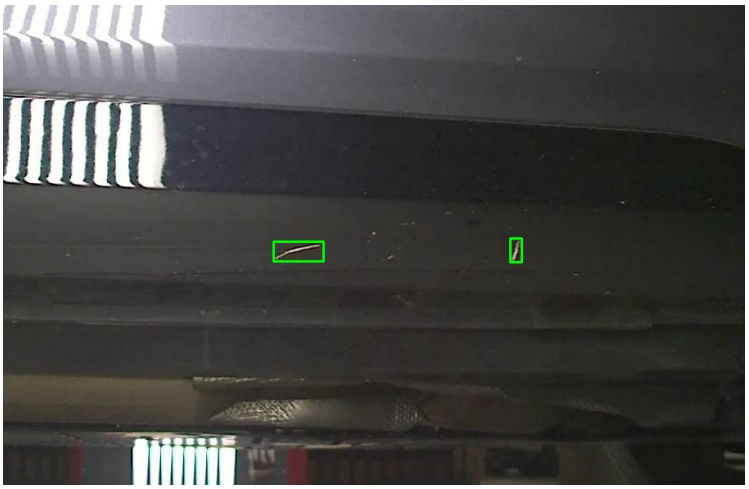


Figure 5: HoG feature extraction for Scratch

For example, if we want to detect lane lines on the road, we can find them by edge detection, in which case the edge information is "useful" and the color information is irrelevant. In this case, the edge information is "useful" and the color information is irrelevant.

In the HOG feature descriptor, the distribution of the gradient direction, i.e., the histogram of the gradient direction, is treated as a feature. The gradient (x and y derivatives) of an image is useful because the gradient magnitude around edges and corners (regions of abrupt intensity change) is large, and edges and corners contain more information about the shape of the object than flat regions.

The histogram of orientation gradients (HOG) feature descriptor is often used in conjunction with a linear support vector machine (SVM) to train a high-precision target classifier.[1]

3.1.2 SVM

SVM is a supervised learning model for analyzing data in classification and regression analysis with associated learning algorithms. Given a set of training instances, each labeled as belonging to one or the other of two categories, the SVM training algorithm creates a model that assigns new instances to one of the two categories, making it a non-probabilistic binary linear classifier. the SVM model is to represent the instances as points in space, such that the mapping makes the instances of the separate categories separated by distinct intervals as wide as possible. The new instances are then mapped into the same space and the categories to which they belong are predicted based on which side of the interval they fall.

However, SVM usually only supports binary classification, for multi-classification tasks we need other skills such as OVA (One VS All), The samples of a certain category are sequentially classified into one class and the remaining samples are classified into another class[2], so that k SVMs are constructed from the samples of k categories, and the unknown samples are classified into the class with the maximum classification function value.

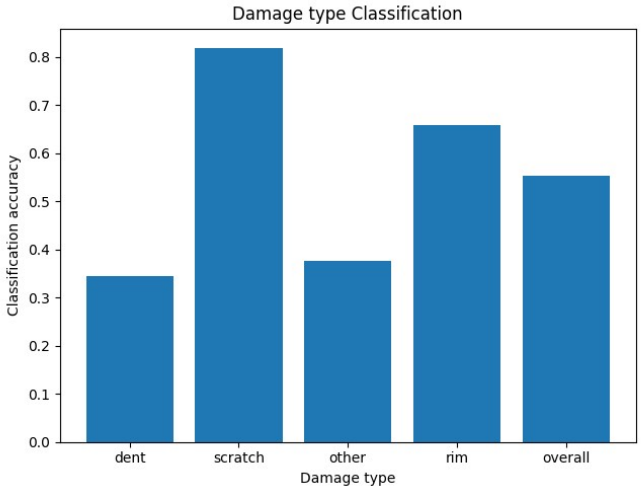


Figure 6: OVA SVM Classification accuracy

3.2 VGG16 Model

VGG model proved to be a significant milestone in the quest of mankind to make computers "see" the world. A lot of effort has been put into improving this ability under the discipline of Computer Vision (CV) for a number of decades. And VGG16 is one of the significant innovations that paved the way for several innovations that followed in this field. The actual model was submitted during the ILSVRC ImageNet Challenge in 2014. The ImageNet Large Scale Visual Recognition Challenge (ILSVRC) was an annual competition that evaluated algorithms for image classification (and object detection) at a large scale[7]. Therefore we try this model on our task.

3.2.1 VGG16

VGG16 was identified to be the best performing model on the ImageNet dataset.

The input to our network configurations is considered to be a fixed size 224 x 224 image with three channels – R, G, and B. The only pre-processing done is normalizing the RGB values for every pixel. This is achieved by subtracting the mean value from every pixel.

Image is passed through the first stack of 2 convolution layers of the very small receptive size of 3 x 3, followed by ReLU activations. Each of these two layers contains 64 filters. The convolution stride is fixed at 1 pixel, and the padding is 1 pixel. This configuration preserves the spatial resolution, and the size of the output activation map is the same as the input image dimensions. The activation maps are then passed through spatial max pooling over a 2 x 2-pixel window, with a stride of 2 pixels. This halves the size of the activations[6].

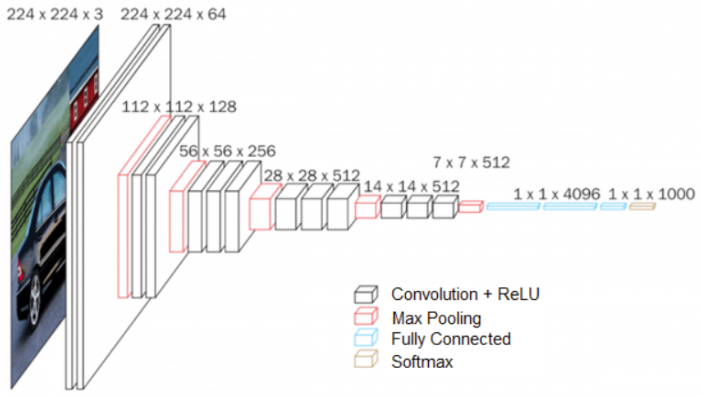


Figure 7: Structure of VGG16

3.2.2 Classification result

Since the VGG16 model usually only performs simple classification but without the function of object detection. Therefore, we use the same preprocessing method as HoG+SVM to trim the marked damaged parts, and then use them as the input of the neural network for training. To enhance the data quality, we also rotated the images.

However, from the results, the training accuracy is not ideal. The features of the rim are more obvious, and the model can easily distinguish it from other classes, but the features of scratches, dents and other classes are not clear enough, and will show relatively poor classification results after training and testing.

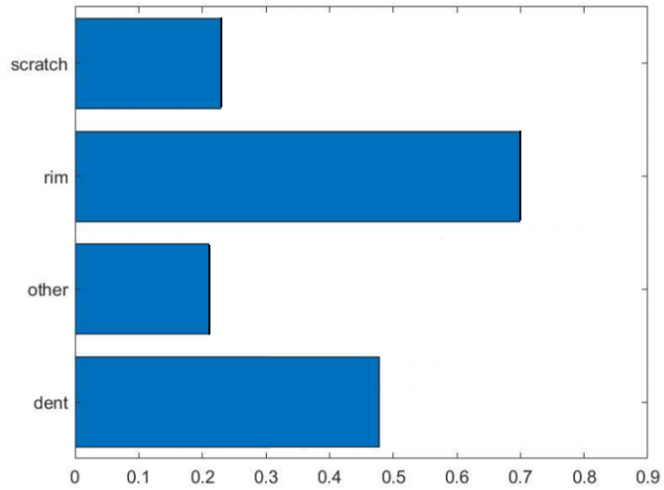


Figure 8: VGG16 Classification accuracy

3.2.3 Challenges

Though this is a very simple, elegant, and easy-to-use model, there are some challenges associated with it. The total number of parameters in this model is over 138M, and the size of the model is over 500MB. This puts some serious limitations on the usage of the model, specifically in edge computing, as the inference time required is higher.

In our task, because of the dataset is not large enough, the overfitting is very difficult to overcome. The accuracy can oscillate a lot with the epoch increases. So we need to explore for more effective method. Therefore, we consider to turn to object detection model instead of purely classification model.

3.3 SSD Model

3.3.1 Problem Analysis

There are two methods for detecting the objects, they are two stage detector and one stage detector. One-stage detectors predict bounding boxes over the images without the region proposal step. This process consumes less time and can therefore be used in real-time applications. The typical representatives are SSD and YOLO. One unsatisfying example is fast RCNN. However, there might be some problems with fast RCNN: firstly, it has poor detection effect on small targets. The task of this project is to detect vehicle damages, which mainly includes scratch, dent and rim. Among them, rim belongs to larger detection target, while scratch and dent belong to very small detection targets, fast RCNN is not very effective in detecting small targets. In addition, fast RCNN has a larger model, because it is a two-stage detector, therefore, the detection speed is slow. The task of this project is to detect the passing vehicles and serve the car rental companies. If the detection speed is increased, more vehicles can be tested, to improve the efficiency. Therefore, in order to make the small objects get better detection effect and improve the detection speed, this chapter decides to use SSD model algorithm for comparison.

3.3.2 Structure of SSD

Firstly, let's analyze the overall structure of SSD[5]. From left to right, for SSD, the input image size is 300 * 300 * 3. Therefore, when we have an image to input into SSD, we need to resize it to a shape of 300 * 300 * 3, then it can be transferred into SSD network[4]. In order to prevent distortion, gray bars are added around the image. As an example, Figure 8(a) is converted to Figure 8(b).

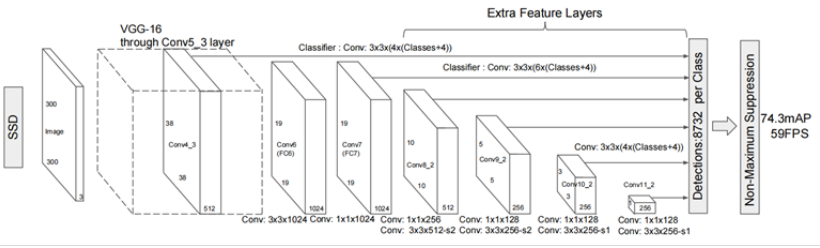
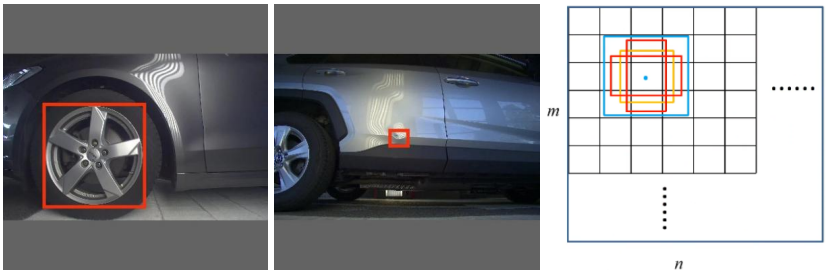


Figure 9: The overall structure of SSD

(a) Before resize (b) After resize

Figure 10: Resize of images

Deep convolution neural network can be used for feature extraction, such as VGG. SSD has made some changes to VGG: ordinary VGG only contains the first two parts. However, the VGG in SSD also contains some convolution layers. Through these convolutions, SSD can continuously extract the features of the input image. In addition to the first two feature layers, the machine also extracts the four feature layers. In this way, SSD continuously extracts features from the four feature layers.



(a) Big feature map (b) Small feature map (c) Prior bounding boxes

Figure 11: Different sizes of feature map for object detection

Figure 11(a) is an image with marked rim, Figure 11(b) is an image with dent, and Figure 11(c) is prior bounding boxes. Compared with the prior bounding box of small size, the sampling degree of prior bounding box will be lower, and some details will be retained by it. The area of the dent in the figure is much smaller than that of the rim. Therefore, the dent is predicted on the tiny feature map. Because the image is convoluted and compressed many times, the features of small objects are easy to disappear. Different size feature objects can be matched on different size feature maps, to improve the detection effect of small targets.

3.3.3 Simulation Result

From the simulation results, the overall mAP value is 0.59 if the “score threshold” is set as 0.5.

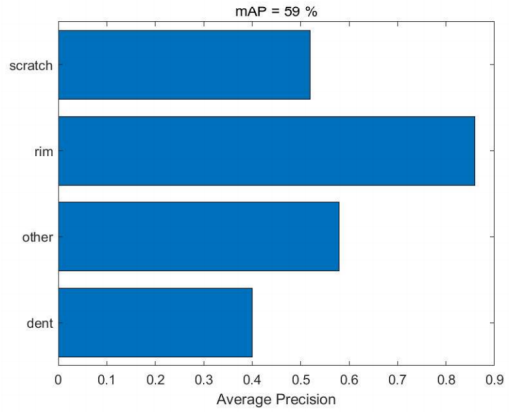


Figure 12: Simulation result for SSD Model

3.4 YOLOv5 Model

YOLOv5 is a single-stage target detection algorithm. This algorithm added some new improvements on the basis of YOLOv4, so that it's speed and accuracy have been greatly improved.

3.4.1 Structure

The input of YOLOv5 has the same enhancement method (Mosaic) as Yolov4. In the YOLO algorithm, there will be anchor boxes with initial length and width for different data sets. In the process of network training, the network outputs the prediction frame based on the initial anchor frame and then compares it with the ground truth, calculates the difference between them, and then reversely updates and iterates the network parameters. The calculation of the value of initial anchor box is run through a separate part In Yolov3[8] and Yolov4 when training different data sets. However, this calculation is embedded in YOLOv5 and the optimal anchor box value in different training sets is adaptively calculated during each training[3]. The function of the focus module is to slice the image before it is putted into the backbone. The detailed process is to get a value for every other pixel in one image. it is similar to adjacent down sampling, so that four images are all obtained, and no information is lost. In this way, the W and H information are concentrated on the channel space, so that the input channel is expanded by 4 times. That means the spliced image has 12 channels compared to the original RGB which only have 3 channels. Then the new image obtained is subjected to a convolution operation and finally a double down sampling feature map without information loss is obtained. The CSP structure of YOLOv5 is to divide the original input into two branches, perform convolution operations to reduce the number of channels by half and then perform Bottleneck * N operation on one branch and then merge the two branches, so that the input and output of BottleneckCSP have the same size. The purpose is to allow the model to learn more features. The Neck of YOLOv5 is like Yolov4 with the structure of FPN+PAN. But in the Neck structure of YOLOv4, the author used ordinary convolution operations. In the Neck structure of YOLOv5, the author used the CSP2 structure designed by CSPnet, which enhances the ability of network feature fusion. In the detection stage, YOLOv5 uses CIOU-Loss as the loss function of the Bounding box.

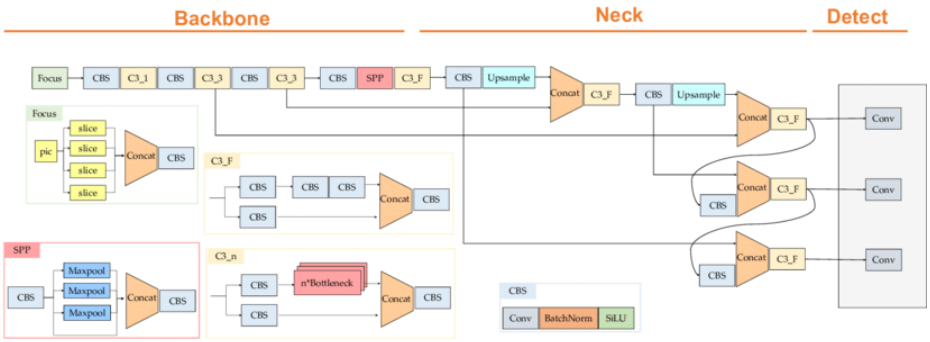


Figure 13: Structure of YOLOv5

3.4.2 Models

YOLOv5 has four models of different sizes, namely v5s, v5m, v5l, and v5x. The weight sizes are 14, 42, 93 and 170 MB respectively. As the model size increases, the model detection accuracy will continue to improve. Because our dataset is not enough, we selected v5x.

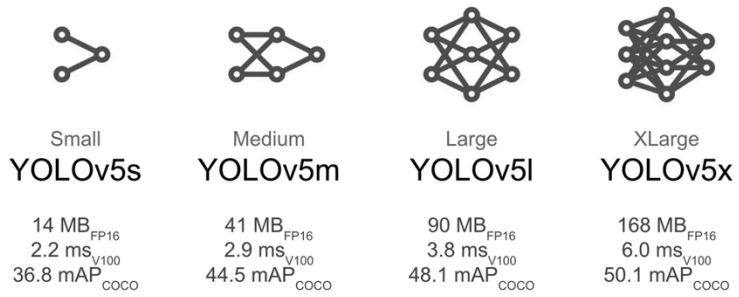


Figure 14: Models of YOLOv5

3.4.3 Results

Although we have selected the largest YOLOv5 model, the result is still not very satisfactory. The main reason is that the amount of data in each class is not enough. The accuracy of “others” is very low, because the type of “others” is a black dot with very small area. The network structure of YOLO is not friendly to the detection of small targets. At the same time, we also see that the effect of the model on multi-target detection is acceptable when the confidence is 0.4.

4 Conclusion

This project aims to detect damages to vehicles using learning-based approaches. We first preprocess and label images provided by the company WENN. Based on these images, we tried different

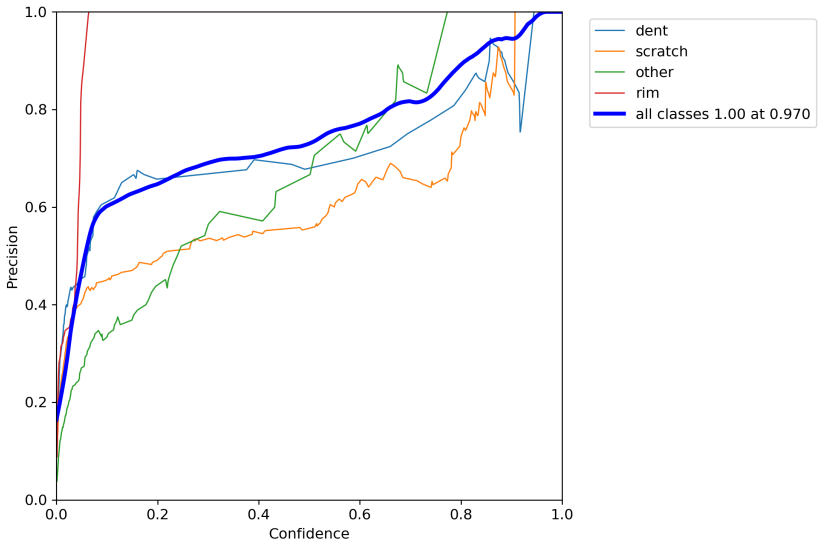


Figure 15: Precision of model

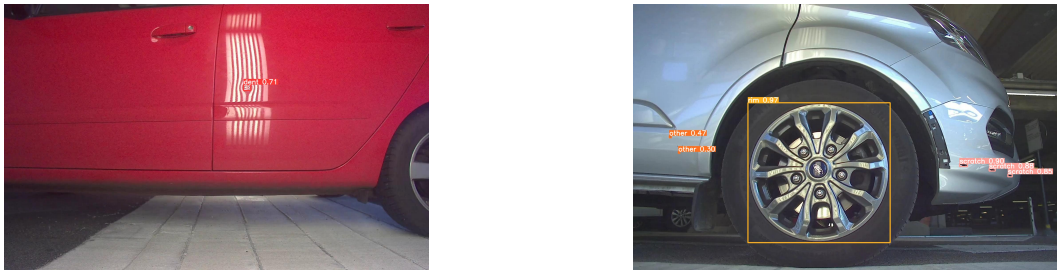


Figure 16: Results of detection

models like YOLOv5, Faster-RCNN, and SVM to classify different types of damages. We also apply cross validation scheme to select the best parameters. After evaluation and comparison of all models, we propose to use YOLOv5 as our final choice, since it outperforms all other models.

In addition, we provide a Django-based web page that implements three functions. First, users can label their data and download the annotated datasets. They can also use our model to predict damages in uploaded car picture. Besides, they can correct incorrect or inaccurate results.

We developed our project within Docker container in an agile way. Finally, we made a short video to show our whole project.

4.1 Summary of the Results

Although the subgroup for data preprocessing has put a lot of effort on data cleaning, format unification and data combination according to damage class, some problems due to dirty car body, light

reflection, still remain and lead to fewer useful images. This brings many challenges to the next step, modeling, such as difficult segmentation of small damage.

The subgroup for model training and test applies multiple approaches on our data set. The performance is not perfect, but still satisfied. After comparison, we decide to choose YOLOv5 as the best option. Even though there is much space to improve the accuracy and robustness, the overall precision of YOLOv5 seems to be the best with high confidence.

The website frontend enables convenient interaction as required. Some integrations, like the labeling tool *labelImg*, become difficult due to compatibility in Docker container. Thus, we take compromised solutions in this case, and give up fancy layout for quick development. In the future, the website can be developed further to include more functionalities and a better design.

Furthermore, our source code is well containerized, as Docker reduces the need for more infrastructure resources comparing to virtual machines and simplifies installation of app at user-end.

Finally, we present our entire teamwork in the uploaded video.

4.2 Future Work

In terms of the model, due to the limitation of data volume, the accuracy of YOLOv5 that we finally used was around 0.65 at confidence 0.4. We can refine our model by collecting more data in the next step.

On the web page side, the function of labelling uses a third-party web page due to capacity limitations, and perhaps in the future this can be achieved on our own.

More features can be added to docker, such as git.

5 Comments to the Group Work Experience

Xiaoming Zhou: Talent wins games, but teamwork and intelligence win championships. Although we have made arrangements before we act. However, there are still various problems in the implementation. It is very difficult to solve them in time. When we realize the problem, several weeks have gone.

Wei Zhou: Thanks for the team work. Having friends I can talk to when I run out of ideas will help me get inspired. With partners can quickly correct me and answer my questions when I don't know how to do.

Jiangnan Huang: Thanks for all help from my team.

Yuliang Nie: Thanks to all my teammates, without their efforts and help, it is difficult for a novice like me to complete a project like this, it was a great experience!

Zechen Wang: Despite the many challenges, we all worked together to complete the project and I learn a lot during the teamwork.

Wenwen Chen: Working together is brilliant. Communication and cooperation are essential to success.

Ruining Wang: It was a wonderful time to work with so many nice people. The success of the project is inseparable from the efforts of all team members.

Xiangyuan Peng: Nice experience to work with everyone. We are a great team. Happy to get knowledge of machine learning, docker and git. A good preparation for the future work.

Zhonghe Ren: Thanks to all my teammates worked together to solve different problems we encountered. I learned a lot from this course!

Yu Wu: It is a nice experience.Through group work, we not only learned a lot of professional knowledge, but also gained friendship.

References

[1] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. Ieee, 2005.

[2] Kai-Bo Duan, Jagath C Rajapakse, and Minh N Nguyen. One-versus-one and one-versus-all multiclass svm-rfe for gene selection in cancer classification. In *European conference on evolutionary computation, machine learning and data mining in bioinformatics*, pages 47–56. Springer, 2007.

[3] Rachel Huang, Jonathan Pedoeem, and Cuixian Chen. Yolo-lite: a real-time object detection algorithm optimized for non-gpu computers. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 2503–2510. IEEE, 2018.

[4] Jisoo Jeong, Hyojin Park, and Nojun Kwak. Enhancement of ssd by concatenating feature maps for object detection. *arXiv preprint arXiv:1705.09587*, 2017.

[5] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[6] Edmar Rezende, Guilherme Ruppert, Tiago Carvalho, Antonio Theophilo, Fabio Ramos, and Paulo de Geus. Malicious software classification using vgg16 deep neural network's bottleneck features. In *Information technology-new generations*, pages 51–59. Springer, 2018.

[7] Dhananjay Theckedath and RR Sedamkar. Detecting affect states using vgg16, resnet50 and se-resnet50 networks. *SN Computer Science*, 1(2):1–7, 2020.

[8] Yunong Tian, Guodong Yang, Zhe Wang, Hao Wang, En Li, and Zize Liang. Apple detection during different growth stages in orchards using the improved yolo-v3 model. *Computers and electronics in agriculture*, 157:417–426, 2019.