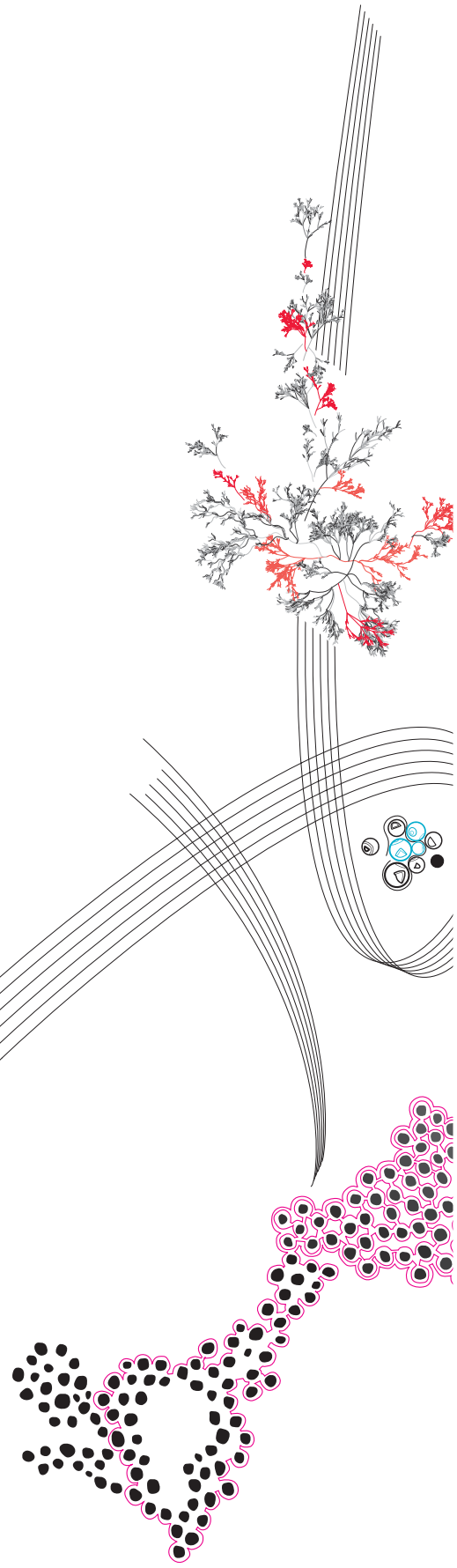

BSc Thesis Electrical Engineering

REAL-TIME PHOTOGRAPHING AND FILMING DETECTION ON MOBILE DEVICES

Wenhao Ben

Supervisor:
Prof.Dr.Raymond Veldhuis
Dr.Chris Zeinstra

July, 2022

Department of Electrical Engineering
Faculty of Electrical Engineering,
Mathematics and Computer Science

UNIVERSITY OF TWENTE.

***Abstract*—With the popularity of cameras and smart devices with built-in cameras, it has become easier to capture face images, the accompanying potential privacy issues have gradually attracted social attention. To protect personal information, a deep learning neural network architecture based on YOLO v5 is designed for real-time object detection system against cameras. The mobile camera fixed on the smart glasses transmits images in real-time to the Android mobile phone for object recognition. The Android mobile phone feeds back the detection signal to the smart glasses, and the smart glasses change the light transmittance to cover the user's eyes to protect personal information. According to the experimental results of the research, the final detection accuracy of glasses objects can reach up to 0.96, meet the basic needs of the product.**

***Index Terms*—Deep Learning Neural Networks, Object detection, Mobile deployment, Communication Systems**

I. Introduction

With the development of modern camera technology, the expensive and bulky old-fashioned cameras are gradually replaced by cheap and lightweight smartphone cameras, so personal cameras have been rapidly popularized along with the growth of smartphone purchases. But the ensuing privacy disputes have also attracted widespread attention, as people find themselves surrounded by cameras that recklessly take pictures containing their identity information, which may be used for some illegal activities that harm their interests.

In this context, the DMB (Data Management & Biometrics) research group of the University of Twente hosted the project to design a camera detection system for mobile devices, with the goal of using it to control the light transmittance of a smart glasses to cover the user's face. This research is one of the two parts of that project, the mobile device for deployment is selected as an Android phone, and the other part is to deploy the camera detection system to the Raspberry Pi. In this way, the deployment effects of different platforms can be compared, so as to make the best choice for the final product.

To implement this approach, the following questions need to be answered in this research:

- Which neural network model framework to select?
- How to deploy the model to an Android phone?
- How to select the camera on the smart glasses?
- How to realize the communication between smart glasses and mobile phones?

II. Related work

The main task of this project is to perform real-time object detection on camera and deploy it to mobile devices. To understand the deep learning-based object recognition technology and its mobile deployment, this chapter will introduce work related to neural network models and inference computing frames.

Object detection technology is an important branch in the field of computer vision, which usually refers to detecting the location and corresponding category of objects in an image. The YOLO series are one of the mainstream frameworks in the field of real-time object recognition. It uses the method of meshing the image, generates multiple boxes based on the center point of the network, and predicts whether each box contains object. In 2020, YOLO v4, released by Alexey Bochkovskiy et al., has completed the enhancement of the detection accuracy and speed of small objects, making it more suitable for long-distance real time detection[1].

The mobile deployment of neural network models has always been an important issue in the application of object detection technology. An inference computing framework will be used to implement this. The inference framework includes two parts: model optimizer and inference engine. The model optimizer can simplify the model to a certain extent and reduce the number of parameters. The inference engine is used in the application system to enable the Android system to invoke the model for inference. The current mainstream inference frameworks for Android phones are MNN and NCNN, which both enhance the adaptability to the Android system and make deployment more convenient.

III. Design

The main design process is divided into two parts: the neural network model design and communication design. This chapter, from the perspective of prototyping, will discuss the design ideas and trade-offs included in the research.

A. Neural network

1) YOLO v5: There are many deep learning model frameworks that can be used for object detection, but in order to adapt to the project requirements, the model must have high accuracy and fast detection speed. Therefore, the first step of neural network design is to compare multiple object

detection frameworks based on the above conditions and select the appropriate one for training.

At present, the mainstream model frameworks with fast detection capabilities are the YOLO series and Faster RCNN. The Faster RCNN adopts a two-stage structure and the YOLO series adopts a Single-Shot structure, both of which have real-time detection capabilities. Compare to other choices, YOLO v5 has the characteristics of fast training speed, high accuracy, fast detection speed, and easy deployment, which is very suitable as a model framework for camera detection. For this part of the analysis, please refer to Appendix 1.1.1.

YOLO v5 provides a variety of neural network models of different sizes, from the very large YOLO v5x to the smallest YOLO v5n. The main difference is in the depth and width of the model, so the size of the model is also different. The smaller the model, the faster the detection speed can be obtained, and the more mAP accuracy will be lost. According to the suggestions provided by YOLO v5 official[2], YOLO v5s will be used for training models for mobile devices, which can not only ensure a certain accuracy but also obtain faster detection speed after light-weighting.

2) Data set: For model training, the selection of data sets is extremely important, and a data set with a same characteristics with the detection target can improve the accuracy of the final model. The main detection target of anti-candid system is the camera or smartphone taking a picture with the lens facing the wearer of the glasses. Correspondingly, the main component of the object detection data set should also be "a picture of the camera/mobile phone facing the photographer". The sample is shown in FIG 1.

3) NCNN: One of the research question need to be solved is how to implement the deployment of the model to Android phones. In this link, a suitable inference computing framework must be used to perform inference operations. Whether the selected inference computing framework is appropriate will also greatly affect the speed and accuracy of the deployed model.

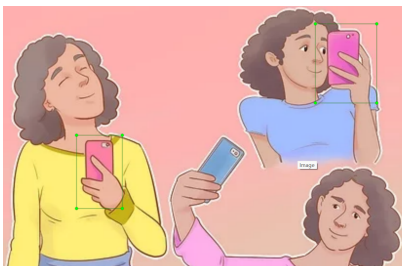

FIG. 1: Data set example[3]

	MNN	TNN	NCNN
Inference Speed	High	High	Medium
Extensibility	Low	Low	High

TABLE I: Comparison between different inference computing framework

The requirement of this research is the Android deployment, so the selected inference framework is also preferentially aimed at the mobile phone. Under this requirement, the mainstream choices are MNN, NCNN, and TNN[4]. These three inference frameworks have their own merits. From the perspective of inference speed, MNN and TNN perform better and seem to be more suitable for the deployment of glasses models. However, the FPS requirement of the project is not that high, because the act of filming cannot be completed instantly, as long as the frame of the candid can be captured, the goal can be completed. Therefore, the first consideration in this scenario is the ease of deployment and the extensibility of its code framework.

NCNN is developed based on the concept of "an ideal framework for mobile phones that can be modified by users"[4]. Its code is very concise and clear, and there are detailed tutorials for beginners to refer to. At the same time, MNN and TNN are more complicated. MNN is provided by Alibaba, its discussion platform is limited, and its code structure is highly encapsulated, which is not suitable for novices to make further modifications to suit the demand. TNN is also not suitable because it has just completed initial construction, and there are not enough tutorials for beginners. The comparison is shown in TABLE I

Since it is necessary to implement the function of communicating with the smart glasses within the original frame, we choose NCNN as the deployment framework of the camera detection model to adapt the needs of the research to further expand the framework.

B. Communication system

The communication system of mobile phones and smart glasses is one of the focuses of this research. The local camera on the front end of the mobile phone cannot meet the product requirements, because if the user is required to keep the handheld mobile phone to obtain the current field of view, it will have a huge impact on the user. It is troublesome and does not meet the original intention of wearable required by product design. Under this condition, it is necessary to design a mobile camera that can capture the frame in real-time and transmit it to the mobile phone for detec-

tion, and a signal transmission circuit that transmits the detection signal from the mobile phone back to the smart glasses. In other words, it is a real-time stable image communication system between smart glasses and Android phones. Android phone is a device for deep learning detection, and the amount of APP code will directly affect the detection speed. Therefore, in this research, the communication server is set up on the smart glasses end, and the client is set up as an Android phone.

There are many ways to design a communication system, it can be wired transmission or wireless transmission, and wireless transmission can also use a variety of methods including Bluetooth, WiFi, NFC (Near-field communication), and UWB (Ultra-wideband). In this research, two different approaches will be tried to compare their pros and cons.

1) Wired transmission + Bluetooth: Because the distance from the mobile phone to the smart glasses is not large, in terms of image transmission, to meet the requirements of short-distance fast and stable transmission, wired transmission is undoubtedly the best choice. All that is required for wired transmission is a data line that is adapted to a mobile phone and a mobile camera, and it does not need to rely on the network environment, nor does it need to perform secondary processing on the transmitted image to achieve stable transmission requirements.

Generally speaking, the data transmission between mobile devices needs to rely on the computer as an intermediary, and the OTG data cable (On-The-Go data cable) can realize the direct connection of mobile devices other than the computer[5]. With UVC (USB video device class) camera, an efficient and stable image transmission system is formed. Another idea is to use a camera extension cable to extend the phone's native camera to the frame of the smart glasses. This method can directly use the Android cameraX library to call pictures, and the quality of the camera is also higher than that of the USB cameras on the market, but the internal circuit of the mobile phone needs to be modified, which requires extremely high technical capabilities.

Therefor the system design is: The remote camera will use OTG cable to realize the image transmission with mobile phone, and if the camera is detected by mobile phone, it will send a detection signal back to the smart glasses, and the smart glasses will be dimmed to prevent the filming. Because the mobile camera itself does not have a circuit to control the potential of the glasses, an additional circuit is required for control, and the mobile phone can control this circuit through

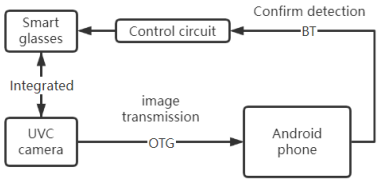

FIG. 2: Wired Communication System Design

Bluetooth. The system is shown in FIG 2

2) WIFI sockets transmission: For the first solution, there is a major drawback: not all Android phones support UVC protocol. Some mobile phone models do not support UVC protocol, so it cannot be connected to an external USB camera. This situation will greatly reduce the universality of the first solution, thereby affecting the final application scope of the product. The second scheme, which uses WIFI for data transmission, is much more suitable on such side. The popularity of the WIFI protocol is much higher than that of the UVC protocol. The main disadvantage of this scheme is that the transmission speed is slow, it depends on the local area network, and the stability is not as good as the first scheme.

There are also many options for WIFI transmission. The research can use an IP camera for video streaming in M-JPEG format, or directly let the camera capture images and then perform socket image transmission. If the video stream is used for transmission, it needs to be processed on the mobile phone, otherwise the deep learning model cannot detect the video in M-JPEG format. Such operations will undoubtedly consume the computing power of the mobile phone. Therefore, it is more suitable to choose socket transmission.

There are currently two mainstream network communications protocols, namely TCP (Transmission Control Protocol) and UDP (User Datagram Protocol). TCP is a connection oriented protocol, and UDP is a connectionless connection, there is no need to establish a double-ended connection, so the transmission speed of the UDP protocol is faster than that of the TCP protocol, but the reliability is weaker than that of the TCP protocol[6].

Unlike the UVC camera designed above and the mobile phone local camera used in the test to the neural network model, the task of socket transmission on Android phones needs to run in a separate sub-thread independent of the main thread (ie UI thread)[5]. Because the app communication part runs on a double sub-thread line, and there is no additional program designed to ensure reliability, it is more important for the reliability requirements to be met, so the TCP protocol is selected as the communications Protocol. The structure of APP is

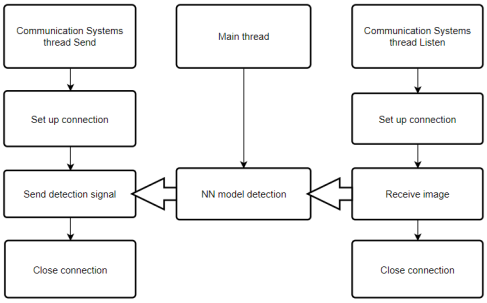

FIG. 3: App threads structure design

shown in FIG 3.

In this research, the Raspberry Pi is directly used for this step. On the one hand, the reason is that the Raspberry Pi itself supports the UVC protocol, a USB camera can be used, and WIFI transmission is supported at the same time, which can complete the integration of three components: WIFI, camera, and control circuit. On the other hand, it is because the Raspberry Pi provides a platform for coding, which is convenient for building a socket server on the Raspberry Pi, reducing the amount of code on the mobile phone and putting more computing power into model inference. The system design is shown in the FIG 4.

IV. Experiments and Results

This chapter will focus on the realization of the methods in the Design chapter.

A. Neural network model

The neural network model used in this project is YOLO v5, which is an open-source object detection model framework based on deep learning neural network algorithms. Before training the data set, the user needs to configure the data set path and modify the training parameters. Also, the code needs to be added to complete the visualization of training results and model metrics. After several training tests, it is found that the model mAP will not increase significantly after more than 150 epochs, so

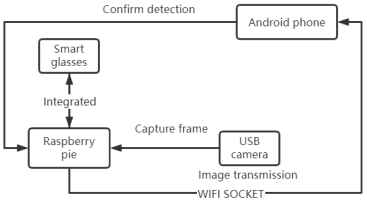

FIG. 4: Wireless Communication System Design

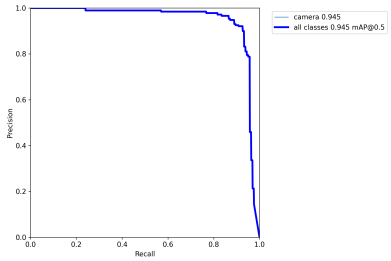

FIG. 5: Recall-Precision curve of test set

the research choose 150 as training epoch number. After 150 epochs of training on the data set of this research, the first experiment is to evaluate the accuracy of the model using an externally test set. The result is shown in the FIG 5.

The PR curve is an important indicator to show the accuracy of the model. It reflects the variation in the precision of the model under different recall rates. For the specific definition of this metric, please refer to Appendix 2.1.1.

It is worth mentioning that the PR curve should not extend the point to threshold 1 under normal circumstances. The author of YOLO v5 set a (1, 0) point to let the curve pass through without extending to threshold 1. This leads to the weird straight line at the end of the curve. However, this does not affect the final result. It can be seen from the curve that the training results are basically in line with expectations, and the mean average precision can reach 0.945 when the threshold is 0.5.

Judging from the detection results of the verification pictures given by the model, the performance of the model meets the requirements, only the camera with the lens facing the wearer of glasses is detected, and there is no camera or smartphone facing the wearer is ignored, which is in line with the project application scenario. The result is shown in FIG 6.

FIG. 6: The detection results of real-life pictures

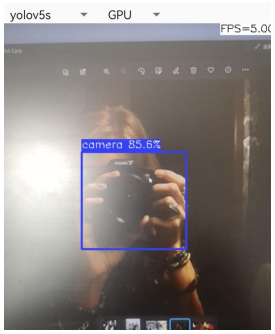

FIG. 7: Preliminary test results using the phone's local camera

B. NCNN

The steps of model conversion are as follows: The official framework of YOLO v5 provides the method to convert it to ONNX (Open Neural Network Exchange) format. After the conversion is completed, the onnx2ncnn tool is used to realize the conversion from ONNX to NCNN.

Then deploy the converted NCNN file to the Android platform, NCNN must be in the corresponding C++ environment, and the Android and NCNN frameworks need to communicate through the JNIEXPORT port. To facilitate the testing of the deployment effect, when the remote camera communication system has not yet been developed, the local camera of the mobile phone is used for testing. The result is shown in FIG 7.

As the results show, its FPS can reach 5 under the condition of ensuring a certain precision. At the level of deep learning neural networks and their Android deployment, the current model meets the requirements.

C. Wired transmission + Bluetooth

During the implementation of this system, it was found that the mobile phone used in this research did not support the UVC protocol. At the same time, it is also very difficult to measure the speed of Bluetooth in the absence of equipment, so reference data is used for analysis, the result is shown in TABLE II

The transmission speed of Bluetooth is far less than that of WIFI, but it consumes less energy, which is advantageous for smart glasses that need to be used for a long time[5].

	WIFI	Bluetooth
Energy consumption	5.2W	1W
Communication speed	409Mbps	2Mbps

TABLE II: Comparison between WIFI-5 and Bluetooth 5.0 (Power rating)[5]

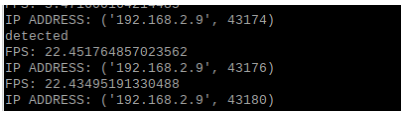

FIG. 8: Communication System Test Results

D. WIFI sockets transmission

As mentioned in the design chapter, the main body of the communication system is an image transmission system based on the TCP protocol. To reduce the amount of code on the mobile phone, a socket server is set up on the Raspberry Pi. When the Raspberry Pi is turned on, the Socket server will be automatically opened and wait for the mobile phone to respond. After the response, the frame captured by the Raspberry Pi camera is obtained through OpenCV, and the passed socket is transmitted to the mobile phone for detection. In synchronization with this, the receiving code also runs on the Raspberry Pi. It will wait for the detection signal from the mobile phone. If the signal is received, it will control the GPIO port of the Raspberry Pi to make the smart glasses change to avoid being photographed. The test results of the communication system are shown in FIG 8.

As shown in the figure, every time the image transmission is completed, the Raspberry Pi segment will close the socket connection in time and wait for the next connection. The FPS shown in the figure is the speed at which the Raspberry Pi takes photos and completes the sending and receiving, not includes the time of the object detection. It is about 20, which is much higher than the detection speed of the neural network. This means that the impact of the communication system on the overall speed will be much less than the detection speed.

To be able to achieve this speed, a lot of adjustments have been made to the code. First, in the original design, the work of taking pictures was done by the Picamera class. After the shooting is completed, it is stored in a fixed address, and then the open function is used to read the data of this address. This design makes the shooting speed of the Raspberry Pi very slow. To improve the speed, in the later design, the OpenCV cap.read function directly reads the pi camera data. Secondly, the project chooses to use OpenCV's imencode function instead of read equation to convert image data to binary to save time for reading data. Finally, without affecting the detection accuracy of the neural network model, the size of the captured picture is adjusted so that it is sized appropriately and can be sent in one packet without transmitting multiple packets. The FPS comparison is shown in the TABLE III

Speed (FPS)	Before	After
Sending speed	4	25
Communication speed	2	20

TABLE III: Speed comparison before and after adjustment of the communication system sends and receives images

As the speed of communication systems increases, delays appear in detection. This is because the camera is shooting faster than the WIFI socket can send, so OpenCV will put the captured image into a cache. After the sending is completed, OpenCV will read the picture in the cache, not the picture taken by the camera in real-time, so the picture received by the mobile phone is not the picture taken by the current view, resulting in a delay in detection.

V. Comparative experiments and results

Raul Ismayilov is working on a similar project. The difference is that the platform he arranged to deploy is the Raspberry Pi, so there is no need to design a communication system, and the corresponding computing power is not as good as that of an Android phone. He chose yolov2 faster as the deep learning model framework to obtain faster inference speed, but at the same time lost a certain inference accuracy. In this section, a comparison is made with the work of Raul Ismayilov to compare the performance differences brought about by different deep learning models and system designs.

A. Neural network model

The first experiment is to use Raul Ismayilov's test set to test the YOLO v5 model used in this project to observe the difference in accuracy between the two[6]. Raul Ismayilov's test set is mixed with more background images that do not contain cameras, and also includes many smaller camera objectes, so it is more complex and harder to detect than the previous test set. The results are shown in the TABLE IV

As can be seen from the table, before deploying to the platform, only comparing two different neural network models under ideal conditions, the accuracy of YOLO v5 is higher than that of YOLOv2 faster. At the same time, this experiment also shows that YOLO v5's detection ability for background and small objects is also good through a different test set.

NN Model	Original test set	Raul's test set
YOLO v5	mAP = 0.96	mAP = 0.92
YOLOv2 faster	mAP = 0.66	mAP = 0.78

TABLE IV: Comparison of model detection precision results for the test set[7]

B. Real-world experiments

To further test the performance of the two designs in real scenes, the project team conducted a series of comparative experiments, including shooting under different types of smartphone cameras, different distances, different lighting conditions, different shooting poses, and different backgrounds. Observe the recognition speed and accuracy of the model. The experiment result example is shown in FIG 9.

The final result of the experiment is in line with the expectations of the project team. YOLO v5 performs better than YOLOv2 faster in terms of accuracy and recognition range, but not as fast as it in terms of recognition speed. In addition, the two models perform well under different mobile phone models, different shooting poses, and different lighting conditions, and the change in the background does not affect their performance.

C. Discussion

There are two main reasons for the difference in performance between the two designs: differences in neural network models and differences in communication systems.

As can be seen from the TABLE V, the size of the YOLO v5 model is much larger than that of the YOLOv2 faster, which is one of the important reasons for the delay. Another main reason is the difference in the communication system. Raul uses the Raspberry Pi to detect objects directly, without the need for cross-device communication, and there is no situation where the OpenCV shooting speed is faster than the transmission speed and causes caching.

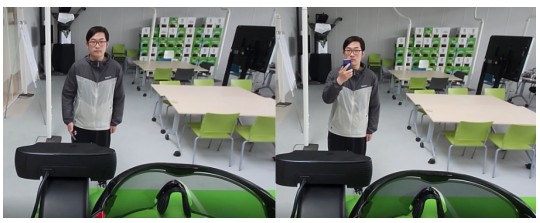

FIG. 9: Real-world experiment result example

NN Model	Parameter quantity	Input image size
YOLO v5	7012k	640×640
YOLOFastestV2	250k	352×352

TABLE V: Comparison of model size[8]

VI. Future work

While completing the study, the researcher came up with some ideas for future work. The main problem with the current product is the detection delay, which is partly due to OpenCV's caching behavior.

Due to the limitation of equipment, this study did not realize the idea of OTG+Bluetooth, and this design may be solved in future research, which will solve the buffering problem of OpenCV since the system do not need socket to do transmission and significantly improve the communication speed of the communication system. In terms of neural network models, YOLO v6 developed by Meituan has just completed its open-source release[9], which focuses on improving the detection speed of the mobile terminal, and can replace the current model in future research.

Also, while this product is designed to protect personal privacy, the way it captures the view of the smart glasses could violate the privacy of others. Therefore, in future research, a program could be designed to automatically detect and blur human faces. This avoids privacy disputes that the product may cause.

VII. Conclusion

This research completes real-time object detection for cameras and Android deployment, which can be used for project on anti-candid camera glasses.

To answer the research questions about the neural network model framework selection and deployment methods were done through the use of YOLO v5 and NCNN. The camera on the smart glasses was chosen to be a USB camera connected to a Raspberry Pi. The communication system was built through a WIFI socket.

The experimental data shows that the design of this study meets the initial product requirements in terms of accuracy, real-time, and applicability. Despite the defect of detection delay, considering the replaceability of the deep learning model network model, these studies are still valuable. The efforts of this research in communication systems can also provide a reference for the mobile deployment of related products in the future.

References

[1] Alexey Bochkovskiy, Chien-Yao Wang. and Hong-Yuan Mark Liao, 2020. YOLOv4: Optimal Speed and Accuracy of Object Detection.

[2] ultralytics. 2020. GitHub - ultralytics/yolov5: YOLOv5 in PyTorch > ONNX > CoreML > TFLite. [online] Available at: <https://github.com/ultralytics/yolov5>.

[3] Cartoon pivture. 2020. [online] Available at: <https://cn.vsolcn.com/blogs-detail/wifi-6-vs-wifi-5>.

[4] Nihui. 2022. GitHub - Tencent/ncnn: ncnn is a high-performance neural network inference framework optimized for the mobile platform. [online] Available at: <https://github.com/Tencent/ncnn>.

[5] chen yurun. 2020.PP developer [online] Available at: <https://zhuanlan.zhihu.com/p/33329187: :text>.

[6] Lifesize. 2022. TCP vs. UDP: What's the Difference?. [online] Available at: <https://www.lifesize.com/en/blog/tcp-vs-udp/: :text=TCP

[7] Raul Ismayilov. 2022. REAL-TIME DETECTION OF PHOTOGRAPHING AND FILMING ON EMBEDDED SYSTEMS

[8] Roboflow Blog. 2020. Responding to the Controversy about YOLOv5. [online] Available at: <https://blog.roboflow.com/yolov4-versus-yolov5/>

[9] GitHub. 2022. GitHub - meituan/YOLOv6: YOLOv6: a single-stage object detection framework dedicated to industrial applications.. [online] Available at: <https://github.com/meituan/YOLOv6>.

APPENDIX

A. 1.1.1: Neural network framework selection

The working process of the R-CNN series is mainly divided into two stages, namely the generation of Region Proposal and the classification of Region Proposal. In the generation stage, the idea of selective search is used to perform layering and merging based on similarity. In the classification stage, the deep network model in the image domain is used, and the external Softmax is connected to complete the classification and calibration of the candidate regions. However, the selective search method to generate region proposals runs slowly and cannot meet the requirements of real-time detection. Therefore, with Faster R-CNN, it directly uses CNN to generate region proposals by introducing RPN, shares convolution layer with the second-stage CNN to improve the detection speed, and achieved remarkable results.

YOLO uses the method of meshing the image, generates multiple boxes based on the center point of the network, and predicts whether each box contains objection and the type of objection. Its accuracy is not as good as R-CNN, but the detection speed has improved. For YOLO, further processing of the images in the grid helps to improve the efficiency of the convolutional layer[4]. Since YOLO v4, the YOLO series has begun to use data enhancement techniques such as geometric distortion, illumination distortion and image occlusion, and the most effective technique is Mosaic Augmentation, which greatly improves the training performance of YOLO v4[4], especially for smaller detection objects. The performance comparison with other object detection frameworks can be seen from the figure below.

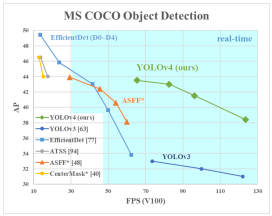

FIG. 10: Compassion of proposed YOLOv4 and other state-of-art object detectors[4]

YOLO v5 has made changes on the basis of YOLO v4, adding an adaptive anchor frame, and changing the activation functions mish and leakyReLU selected by YOLO v4, replacing them with leakyReLU and Sigmoid with lower complexity. At the same time, the application of data enhancement technology is reduced, and only the scaling technology, color space adjustment and Mosaic Augmentation are retained[7].

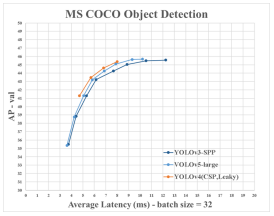

FIG. 11: V5 vs V4 1[7]

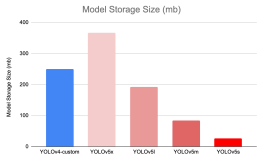

FIG. 12: V5 vs V4 2[7]

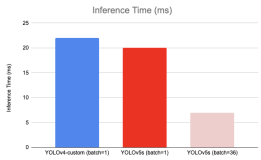

FIG. 13: V5 vs V4 3[7]

FIG. 14: V5 vs V4 3[7]

As can be seen from the above figures, the performance of YOLO v5 is lower than that of YOLO v4. However, it is implemented based on PyTorch, and its flexibility is much stronger than that of V4. At the same time, the model size of YOLO v5s is also much smaller than that of V4, which is more conducive to the deployment of mobile devices. Also YOLO v5 has much less training time than YOLO v4, which is friendly to personal user. As for the choice between YOLO v5 and Faster RCNN, refer to Priya Dwivedi's article "YOLO v5 compared to Faster RCNN. Who wins?"[reference] published in Towards Data Science, which mentions that YOLO V5 maintains almost the same accuracy as Faster RCNN with speed is increased by 2.5 times, and the detection of small targets is more accurate[7].

B. 2.1.1: Recall-Precision curve

The PR (precision-recall) curve is an extension of the basic indicators of a deep learning model, which specifically reflects the relationship between model precision and recall.

In the process of deep learning, labels are positive samples, and samples classified as positive are True Positive, abbreviated as TP. Labels are positive samples, and samples classified as negative are False Negative, abbreviated as FN. Labels are negative samples, and samples classified as positive are False Positive, abbreviated as FP. Labels are negative samples, and samples classified as negative are True Negative, abbreviated as TN. These four basic metrics, together with the probability threshold T, are used to detect the performance of the model. In actual operation, the predicted probability greater than the threshold T is the positive class, and the predicted probability less than the threshold T is the negative class, which is 0.5 by default. If we reduce this threshold T, more samples will be identified as positive classes, which can improve the recall rate of positive classes, but at the same time, more negative classes will be mistakenly classified as positive classes. If the threshold T is increased, the recall of the positive class decreases and the precision increases.

The PR curve is to observe the precision of the model under different thresholds by constantly changing the threshold to changing the recall rate. Generally speaking, the area framed by the PR curve represents the average precision of the model, so the closer the curve is to the upper right corner, the better.