



Combating Involuntary Information Leakage in Video Conferences Using Computer Vision Algorithms

Jian Song

Master's Thesis
Master of Engineering- Big Data Analytics
2023

MAASTER'S THESIS	
Arcada University of Applied Sciences	
Degree Programme:	Master of Engineering – Big Data Analytics
Identification Number:	9205
Author:	Jian Song
Title:	Against information leakage with online meeting platforms Reflection on eyeglasses
Supervisor (Arcada):	Leonardo Espinosa-Leal
Commissioned by:	
<p>Abstract:</p> <p>Information security is an endless process. With the evolution of technologies, this topic has been broadly discussed over time. During the COVID-19 pandemic, most activities have changed from personal attendance to online, for instance, meetings. This thesis focuses on information leakage when people are attending online meetings with eyeglasses. As the glass itself has a reflective principle, the work is aiming to use a learning-based method to prevent the reflection of sensitive information, especially information leakage via eyeglasses while attending online meetings. The data was collected through simulation of real-world behavior, and three customized deep learning algorithms have been evaluated. With a customized loss function, the area of eyeglasses can be properly detected and blurred, to add noise to the content reflected on the glasses. And with the combination of transfer learning from an advanced pre-trained facial landmarks detection model, the outcome demonstrated the agility and capability to prevent information leakage under the defined scenario. With the consideration of diverse types of devices running meeting software, for example, phones and tablets, the final result is capable to run on different platforms.</p>	
Keywords:	Information Security, Deep learning, Neural networks, Image processing, Computer Vision
Number of pages:	45
Language:	English
Date of acceptance:	18.5.2023

Contents

1. Introduction	8
1.1 Background Introduction	8
1.2 Problem Statement Related to AI	8
1.3 Research Questions	10
1.4 Research Objectives	10
1.5 Limitations	10
2. Related work	12
2.1 Supervised learning with the deep learning method	12
2.2 Facial Landmarks Detection	13
2.3 Eyeglasses Removal from Facial Images	14
3. Research Methods	16
3.1 Research Design	17
3.2 Data Collection	18
3.2.1 Labelme	18
3.2.2 Augmentations	19
3.3 Model Selection	21
3.3.1 Customized Model Output	25
3.3.2 Loss Function Design	26
3.4 POI Blurring	27
3.4.1 Gaussian Blur	28
3.4.2 Facial Landmarks	29
3.5 Experiment	30
3.6 Evaluation of Performance	31
3.6.1 Evaluation of Classification	31
3.6.2 Evaluation of Regression	33
4. Result from the experiment	35
4.1 Customized VGG16	35
4.2 Customized ResNet50	36
4.3 Customized MobileNetV3	37
5. Conclusion	40
5.1 Summary	40
5.2 Work in future	40
References	42

Figures

Figure 1. Zoom Image Collage with Detected Information

Figure 2. Reflection Removal on Eyeglasses Using Deep Learning

Figure 3. Zoom Video Filters

Figure 4. How to put glasses

Figure 5. Illustration of portrait synthesis

Figure 6. Research workflow

Figure 7. Image collection algorithm

Figure 8. Common architecture of CNN

Figure 9. Architecture of VGG-16

Figure 10. Framework of SSD

Figure 11. Binary cross entropy

Figure 12. 68 facial landmarks

Figure 13. Training performance of customized VGG16

Figure 14. Performance of VGG16

Figure 15. Predicted bounding boxes by VGG16

Figure 16. Training performance of customized ResNet50

Figure 17. Predicted bounding boxes by customized ResNet50

Figure 18. Training performance of customized MobileNetV3

Figure 19. Predicted bounding boxes by customized MobileNetV3

Tables

Table 1. The default coordinates manipulation

Table 2. ResNet Architecture

Table 3. Summary of results obtained with the customized models

ABBREVIATIONS

GDPR	General Data Protection Regulation
HOG	Histogram of Oriented Gradients
SVM	Support Vector Machine
CNN	Convolutional Neural Network
VGG	Visual Geometry Group
UAVRS	Unmanned Aerial Vehicle Remote Sensing
SSD	Single-Shot Detector
POI	Point Of Interest
IoU	Intersection over Union
PRH	Patentti ja Rekisterihallitus (Patent and Registration office)
PSNR	Peak signal to noise ratio
API	Application programming interface
TP	True Positive
FN	False Negative
FP	False Positive
TN	True Negative
gt	ground-truth
pd	prediction

Foreword

After working at Helsinki Vantaa Airport for many years, I started to think about how to change my future career. Looking back now, it was a long and difficult way to combine work and study at the same time. In the year 2020, I decided to enter a new field as most people are talking about machine learning and AI. Thankful for the Arcada University of Applied Sciences provided such a great opportunity to enter the big data and machine learning field. In the first year of study, with the first assignment related to stock price prediction, the typical time series task, when I finally completed all requirements with really basic programming skills in Python, the happiness is unforgettable.

From the beginning until now, the courses and study were intensive. There were a lot of time-consuming topics taught in class, especially while writing the thesis. Hereby, I would like to sincerely thank my supervisor Leonardo's support and encouragement, his generous guidance with thesis writing, inspiring advice while making the model selection. At last, the suggestions from my colleagues while making the patent filing to Finnish PRH.

A new journey is waiting ahead, hope to start the challenging but interesting subject in the research field in the near future.

Jian Song

Vantaa, 30.4.2023

1. Introduction

Since the spread of the new coronavirus epidemic, the implementation of various epidemic prevention policies has had a great impact on people's living habits. During locked down times, most countries introduced the recommendation to maintain safe social distancing and encourage people to work remotely at home. It also impacts how people used to work, the most common example is that meetings have been switched from face-to-face to online to ensure the flexibility of working from home and the maintenance of social safety distance. The usage of video call platforms consequently has dramatically increased. The Global Web Conferencing Market size was registered at USD 3.16 billion in 2019 and is estimated to reach USD 8.09 billion by 2030 (Fatpos Global Pvt. Ltd., 2022). This potential market, regarding issues of information security, deserves one's attention.

1.1 Background Introduction

The General Data Protection Regulation (GDPR) was launched in 2016 from the EU area, intending to provide one set of privacy laws for the European Union. This is due to the fast-changing of technology, companies or other organizations are using personal data to maximize their profit. For example, customized website advertisements based on your search history on a search engine, or your most viewed products on an online shop. On the official website, the GDPR has defined personal data as the range of identifiers including Names, Gender, Email Address, and many more. As GDPR official website states, the definition of personal data is any information relating to an "identified or identifiable natural person." In some circumstances, information related to a person's political opinion, occupation, and even hair color can be categorized as personal data. (European Commission, 2023)

1.2 Problem Statement Related to AI

Dima Kagan, Galit Fuhrmann Alpert, and Michael Fire from the Department of Software and Information Systems Engineering, Ben-Gurion University of the Negev, Israel, conducted a study of video conferencing privacy and security threats in July 2020. In their work, the authors computed and demonstrated statistics on video conferencing usage scenarios, the participants covered different age groups, from young people to the elder age as shown in Figure 1. The deep learning-based image processing method algorithm was applied to their task and it is

proven there is a great possibility to identify personal attendance in different meetings via face recognition, and other useful features can be extracted from a user interface of an online meeting platform. (Kagan et al., 2020)

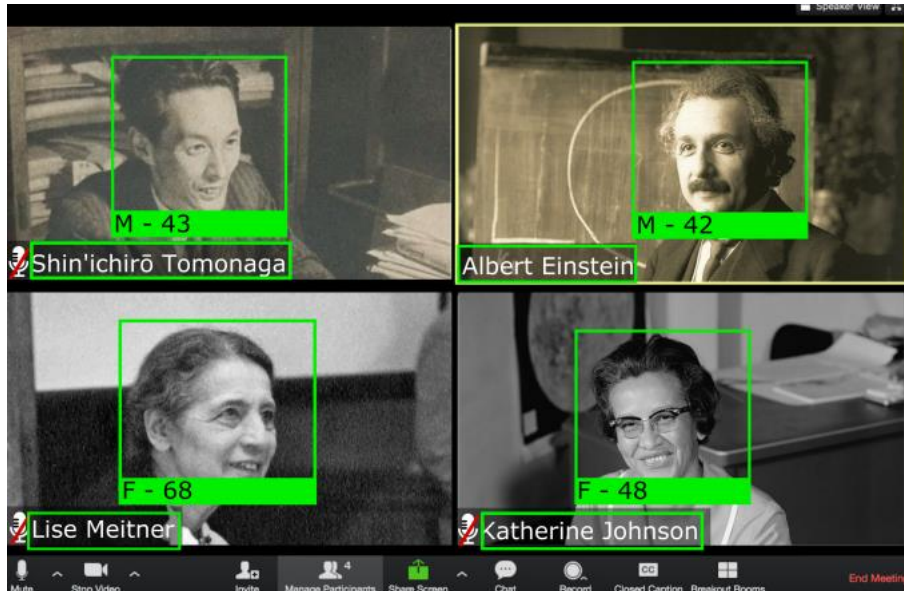


Figure 1. Zoom Image Collage with Detected Information, Along with Extracted Features of Gender, Age, Face, and Username. (Kagan et al., 2020)

Based on the above results, users participating in video conferences are likely to become victims of fake avatars. As same as social network platforms, those who have information about someone else's avatar can easily pretend to be a person they are not.

Furthermore, the University of Michigan worked with its partner Zhejiang University in China as a team, to investigate whether wearing eyeglasses while participating in online video call leads to information leakage. The team has explained how the controlled lab experiment was settled and demonstrated how they achieved over 75% accuracy to reconstruct and recognizing the text information via reflection on eyeglasses, while the webcam is only 720p and the text height is as small as 10mm. The most impressive outcome was that the researchers found a technique to classify which website the participant was surfing, and the accuracy of it reached 94%. (Arghire, 2022)

In the paper the team published, the researchers recommended online meeting platform users take advantage of embedded filter features, such as background blurring or substituting a

sunglass on the face which is reflection free. The researchers proposed both short-term and long-term solutions, including blurring the eyeglass areas of the video stream through the software, to make the reflection area blurry, thus unrecognizable. However, the researchers also stated that potential information leakage level varies from the quality of their reflection quality on eyeglasses, consequently, it would be vulnerable to only implement a single set of protection methods.

1.3 Research Questions

This work is intending to answer the following questions,

1. How to protect against information leakage through reflection on eyeglasses while participating in a video conference?
2. Which solution is optimal to be applied that takes less computational resource?

The research questions are formed in such a way as to provide a novel solution against information leakage through reflections on eyeglasses, under defined scenes when people are participating in the video conference.

1.4 Research Objectives

As the recommendation given by a joint team formed by the University of Michigan and Zhejiang University, this study aims to find a novel solution to conquer information leakage through reflection, the eventual research outcome should not only classify whether the eyeglasses are present but also blur eyeglasses area to add the filter to the associated area. The blurred area also needs to be real-time and synchronized with the random movement of participants. The detailed description of the dataset will be done later in the thesis.

1.5 Limitations

The dataset used in this study is fully self-created. In terms to simulate how participants move around in front of the webcam while attending online meetings, the images were collected through the webcam from multiple angles from left to right direction, but in real life, the movement of participants may be much more diverse, the more completed description of dataset will done later in the thesis.

In terms of information security issues caused by reflection, it is not limited to eyeglasses only in a real-life scenario. Any metallic surfaced object, fine polished stainless-steel thermos cup, or anything that has reflective features, could take into consideration that leads to information leakage. In this research, the target is specially focused on eyeglasses.

Since most video conference platforms are commercial software, to check the real-time capability of research results, the attendance of the video conference was simulated via the cv2 library. The outcome is more convincing if a simple video platform can be created and integrated with the trained model from this research.

2. Related work

2.1 Supervised learning with the deep learning method

It is well known that 80-90% of the time for machine learning development is spent on data preparation. Also, even the best machine learning algorithms cannot perform well without good data or at least handling biased and dirty data during model training (Whang&Lee, 2020).

In June 2021, researchers Sota Watanabe and Makoto Hasegawa from Tokyo Denki University performed a state-of-art method that collected data by blinking the display at a high frequency and marking down the time when the display turned on (Watanabe&Hasegawa, 2021). Through such a procedure, there will be two types of paired images consisting of ground truth (eyeglasses with no reflection when the display is off) and reflection on eyeglasses (display is on) in an identical angle of view. The quality of the data collected in this way will be much better than synthetic data, because, in the meeting scene, the user will move freely at any angle.

Supervised learning is used whenever we want to predict a certain outcome from a given input, and we have examples of input/output pairs (Muller & Sarah, 2017). After obtaining the training data, the researchers mentioned above applied a supervised learning method and trained customized autoencoder and U-Net. The loss function of their neural network was mean squared error (MSE) and the evaluation of accuracy was done through the evaluation of the peak signal-to-noise ratio, as known as PSNR. The result can be seen in Figure 2. As U-Net performs image-to-image translation, while autoencoder aims to find a lower dimensional representation of higher dimensional space via encodes or decodes, the result from autoencoder in (c) is a bit noisier compare with ground truth in row (b).

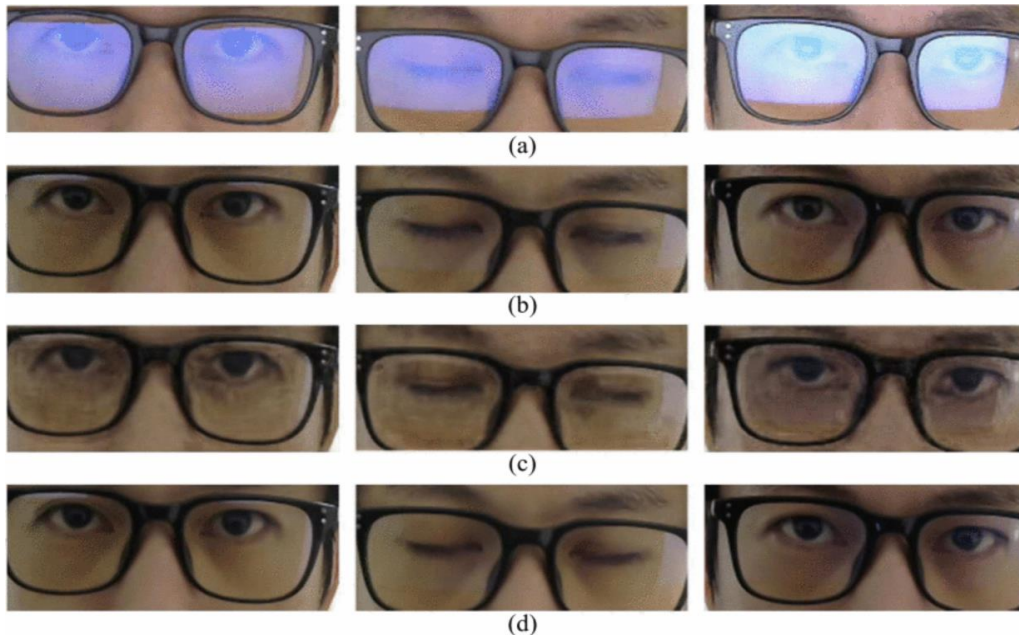


Figure 2. (a) The images with the reflection, (b) the ground truth images without the reflection on the eyeglasses from the test datasets. (c) The results of our autoencoder, and (d) the results of paper’s U-Net. (Watanabe&Hasegawa, 2020)

2.2 Facial Landmarks Detection

The locations of the fiducial facial landmark points around facial components and facial contours capture the rigid and non-rigid facial deformations due to head movements and facial expressions. They are hence important for various facial analysis tasks (Yue Wu&Qiang Ji, 2018). Most camera applications, such as face-swapping effects, adding additional decorations to the face like animated eyeglasses, or other animation effects are based on the detection results of facial landmarks. The online meeting platforms applied such technologies and embedded video filter options to video stream as shown in Figure 3.

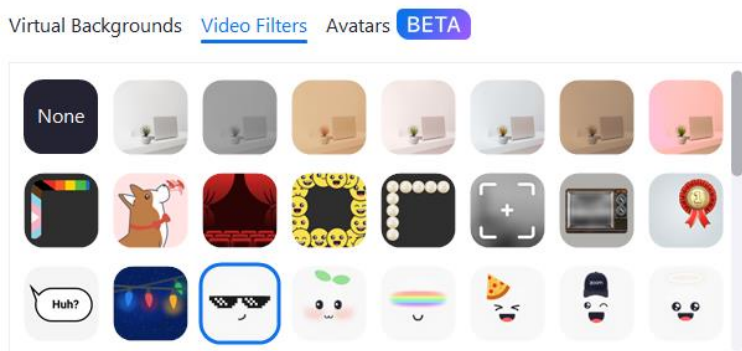


Figure 3. Zoom Video Filters

Irfan Khalid introduced in the year 2021, the prebuilt library like dlib, OpenCV, and mediapipe are ready to use for landmark detection (Khalid, 2021). As the dlib library contains the pre-trained model base on ensemble regression trees that are used to estimate the position of face's landmarks directly from a sparse subset of pixel intensities, the model was produced by Vahid Kazemi and Josephine Sullivan from KTH in the year 2014. The model is trained on the iBUG-300 W dataset (<https://ibug.doc.ic.ac.uk/resources/300-W/>) and includes images and annotations of 68 face landmark points (Kazemi&Sullivan, 2014). Since the dlib face detector is made using classic HOG (Histogram of Oriented Gradients), in combination with an image pyramid, sliding window detection, and a linear classifier scheme, in the real-time video conference simulation, with image manipulation in addition, the performance has a visible postpone from the experiment in a later chapter.

2.3 Eyeglasses Removal from Facial Images

With the advantage of 3D synthesized data, in August 2020, Yu-Hui Lee and Shang-Hong Lai from Tsing Hua University published a paper with their novel solution for eyeglasses removal from face images named ByeGlassesGAN, this algorithm intends to detect the position of eyeglasses automatically and remove the eyeglasses in parallel (Lee&Lai, 2020). By labeling the head pose of all the images from the training dataset, and a thousand eyeglasses segmentation masks of the glasses-images to formulate a glasses pool, the random glasses can be added to those glasses-free images from the training dataset depending on their head pose label and the binary mask, the author also made several steps to those synthetic images to make it photorealistic as shown in Figure 4. The architecture of ByeGlassesGAN consists of one face encoder and two decoders that work for face segmentation. The face encoder works to extract face image information while the face decoder exploits the information retrieved from the face encoder to generate glasses-free images. The segmentation mask of eyeglasses and the face region were predicted by the segmentation decoder. As the feature vectors predicted from the segmentation decoder are shared with the face decoder, thus the result from their experiment demonstrated great capability with eyeglasses even with glare or semi-transparent color.

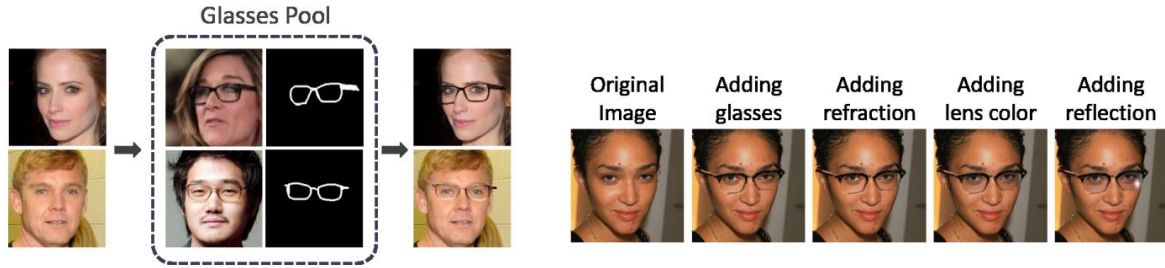


Figure 4. How to put glasses? (Y. Lee & S. Lai, 2020)

On the bases of ByeGlassesGAN, Junfeng Lyn, Zhibo Wang and Feng Xu from Tsinghua University made further improvements in the year 2022 as the light condition and natural shadow caused by glasses were neglected in previous related work. By using the dataset created by Zhibo Wang in the year 2020, which includes 438 subjects of facial scans from different age groups, the researchers built the synthesis of paired data through 3D rendering. The 3D eyeglasses can be attached to facial scans through node-based registration, then the scan with eyeglasses is rendered by randomized illumination. Consequently, the four types of rendered images can be achieved by setting the eyeglasses or enabling and disabling the cast shadows of glasses as shown in Figure 5.

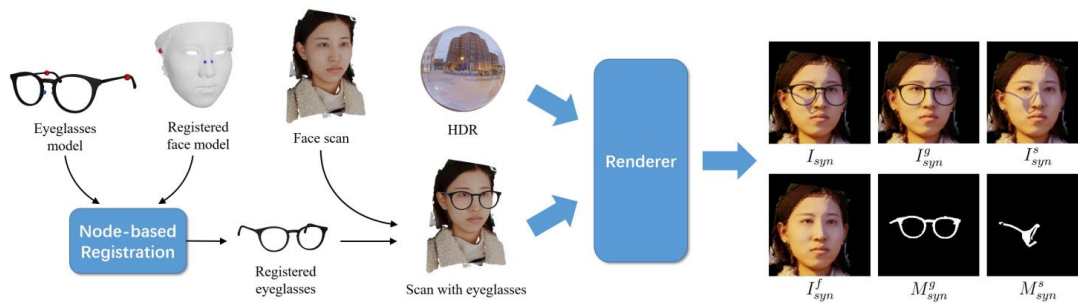


Figure 5. Illustration of portrait synthesis. (Junfeng Lyn & Zhibo Wang & Feng Xu, 2022)

The experiment of this research achieves promising performance but unfortunately, with the real-world data, it doesn't overcome the extreme head positions and lenses with a special color.

3. Research Methods

In previous chapters, the privacy and security problems of online conference platforms have been introduced, and several state-of-arts solutions made by scholars are also briefly summarized. Instead of seeking the opportunity to overcome the gap between synthesis data and real-world data, in addition to taking into consideration of different glasses style and lens colors, the research method of this work is designed to take advantage of object detection techniques and apply a blurring algorithm to the detected area, where the eyeglasses are located. Thus, no matter at which angle the reflection happens on eyeglasses, the content is blurred.

Object detection involves detecting instances of objects from one or several classes in an image. Each detection is reported with some form of pose information. This could be as simple as the location of the object, a location and scale, a bounding box, or a segmentation mask (Yali et al, 2020). Once the location of the eyeglasses is fetched, with the corresponding coordinates, the eyeglasses area can be blurred via Gaussian blur or other similar methods from the image processing field.

The devices and libraries used in this research are listed as follows, devices,

- Lenovo ThinkPad P1 Gen5, 32 GB, RTX3070Ti 8G, 21DDS23U00
- Logitech C615 HD web camera
- Regular black-framed anti-blue light eyeglasses

libraries,

- Python 3.10.9 <https://www.python.org/downloads/release/python-310>
- Tensorflow 2.9.0 <https://github.com/tensorflow/tensorflow/releases/tag/v2.9.0>
- Tensorboard 2.9.1 <https://github.com/tensorflow/tensorboard/releases?page=2>
- Keras 2.9.0 <https://pypi.org/project/keras/2.9.0/>
- Labelme 5.1.1 <https://libraries.io/pypi/labelme>
- Split-folders 0.5.1 <https://pypi.org/project/split-folders/>
- Albumentations 1.3.0 <https://pypi.org/project/albumentations/>
- Dlib 19.24.0 <https://libraries.io/pypi/dlib>
- Opencv-python 4.7.0.68 <https://pypi.org/project/opencv-python/4.7.0.68/>
- UUID <https://pypi.org/project/uuid/>

3.1 Research Design

In this section, the detailed research method of this thesis is described. First of all, a sufficient amount of data is collected to train a deep learning model, all those images need to be labeled either with or without eyeglasses. After fine-tuning and the trained model saved, it should be capable to classify if the glasses are present or not, and able to locate where the glasses are located in the image, in the other hand, means it should properly predict the coordinates of the glasses area. Consequently, if the participants are not wearing glasses, there will be no action needed, and the video frame has remained the same. If the eyeglasses are detected, the image processing step will be carried out and the frame of the video will be blurred with the coordinates predicted by the deep learning model. Due to the flexibility of joining video meetings, the participants may attend via phone or tablet instead of sitting in front of a laptop or desktop, there will be several deep learning models evaluated as a comparison to save the computational resource. The full workflow is displayed in Figure 6.

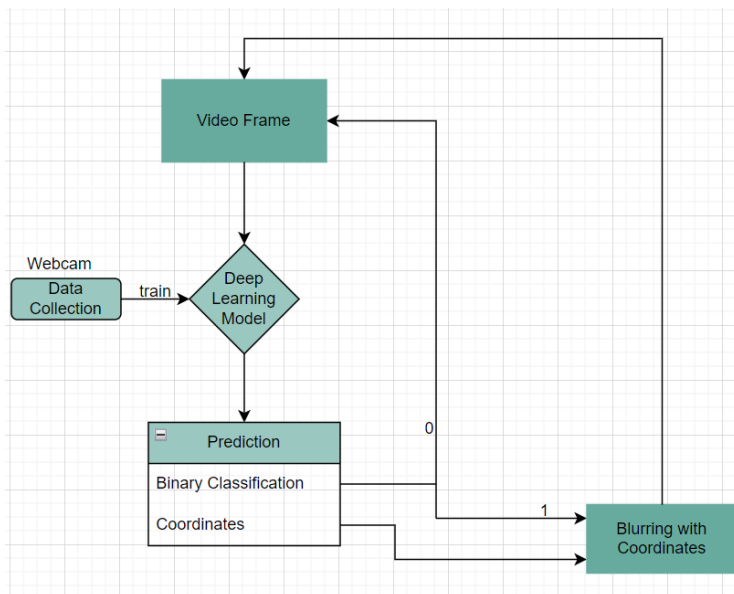


Figure 6. Research workflow

The training data will be collected through a webcam, and deep-learning models can detect the presence of eyeglasses after the model is trained. The model is used to make two predictions, classification, and regression. If the binary classification value is 0, the model will return the original frame image, if the binary classification value is 1, the blurred frame image will be returned based on the coordinates the model predicted.

3.2 Data Collection

The data from this research was collected via a webcam, the Logitech C615. By placing it on the top middle side of the monitor, around 300 images were collected using windows embedded camera. The size of the collected image is 720 x 480 pixels by default. To simulate the diversity of user positions during a video meeting, the base images are taken by moving the upper body left and right, also totally absent from the camera, including head-turning which only left or right glasses can be seen from the camera. Then the webcam was shifted to a lower position to simulate the laptop or smaller smart devices such as tablets and photos.

The algorithm was written to automatically take images via webcam, as demonstrated in Figure 7. The cv2 library has the *VideoCapture* method, by setting the time interval to 0.5 seconds to have a bit of time to adjust body position, and another library *UUID* was applied to provide unique file names to those images for later manipulation. The total image number is set within the loop until sufficient numbers of base images are collected. Those images will be saved to a dedicated directory by the python os library.

```
#build connection with video camera using cv2
#number of camera index may differ

cap = cv2.VideoCapture(1)

for imgnum in range(number_images):
    print('Collecting image {}'.format(imgnum))
    #return value whether it's successful and frame itself
    ret, frame = cap.read()
    imgname = os.path.join(IMAGES_PATH, f'{str(uuid.uuid1())}.jpg')
    cv2.imwrite(imgname, frame)
    cv2.imshow('frame', frame)
    time.sleep(0.5)

    #CV2 break the Loop
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
cap.release()
cv2.destroyAllWindows()
```

Figure 7. Image collection algorithm

3.2.1 Labelme

Labelme is an open-source annotation tool developed by MIT. It has been broadly used to support manual image annotation with object detection, segmentation, and object classification tasks. It provides plenty of shape options to annotate the objects within the image including

circles, rectangles, polygons, and so on. In this research work, as most eyeglasses are commonly round or rectangular shape, the rectangular shape was selected to annotate the base images collected from the previous chapter. If the participant is absent within the image, there will be no annotation made, or if there are two eyeglasses visible, two associated rectangular annotations are manually added to the image. In case the image has only one eyeglass visible while the participant's head was turning fully to the left or right position, only one annotation is made. All those annotations are saved in separate JSON files and the file name of those JSON files are identical to the associated base image, which was created via UUID in the previous chapter during base image collection.

3.2.2 Augmentations

As it was mentioned in Chapter 2.1, machine learning algorithms, especially deep learning neural network usually requires a sufficient amount of data. The author Cem Dilmegani indicated in his research that the most common challenge in machine learning implementation is insufficient data (Dilmegani, 2022). Also, it has a direct influence on model performance depending on the quality, quantity, and relevancy of training data. Data augmentation is a common method to increase the size of training data and helps to prevent overfitting of the model. In this research work, in total 6800 images were collected after data augmentation.

The Python library *Albumentations* is used for image augmentation in this work. The new images are generated from existing base images collected in previous steps. It supports various computer vision tasks such as object detection, classification, instance segmentation, and so on. The second algorithm was prepared to apply image augmentation based on different pixel-level transforms, for instance, *Random Crop*, *Horizontal Flip*, and *Random Brightness Contrast*. Especially, the *RandomCrop* will be cropping the base image randomly with given image width and height parameters, the size of the base image needs to be bigger than the parameters set to the *RandomCrop* method.

The main reason why *Albumentations* was selected as the data augmentation method is that the images will be augmented along with associated bounding boxes. In other words, the image will be flipped, cropped, or any other operation that happened with the base image, and the bounding boxes drawn via *Labelme* will also simultaneously change. To regularize the coordinates of the bounding boxes, the default format of the coordinates was selected as

‘albumentations’, which is calculated with original coordinates divided by the width and height of the original image, as a result, the output of the coordinates is in the range between 0 and 1 accordingly.

In the end, the outcome from augmentation contains the image vectors of augmented images, the regularized coordinates of the eyeglasses area, and a binary classification value of either 0 or 1. The most important part of augmentation is the coordinates initiation because the coordinates generated from the Labelme library have a few conditions as shown in Table 1.

Initial coordinates created from Labelme	
Yes	No
Both glasses are visible, keep the original coordinates [[x1, y1, x2, y2], [x3, y3, x4, y4]], insert binary class value ‘1’	Created default coordinate, [[0, 0, 0, 0], [0, 0, 0, 0]], insert binary class value ‘0’
Only left glass visible, keep original coordinates, set default coordinate for the right glass [[x1, y1, x2, y2], [0, 0, 0, 0]], insert binary class value ‘1’	Created default coordinate, [[0, 0, 0, 0], [0, 0, 0, 0]], insert binary class value ‘0’
Only the right glass is visible, keep the original coordinates, and set the default coordinate for the left glass [[0, 0, 0, 0], [x1, y1, x2, y2]], insert binary class value ‘1’	Created default coordinate, [[0, 0, 0, 0], [0, 0, 0, 0]], insert binary class value ‘0’

Table 1. The default coordinates manipulation

The deep learning model takes input and feeds it into the neural network, the output are following the format depending on the number of neurons from the dense layer. Since the eyeglasses contain two pieces of glass and during data collection and annotation, there are situations that only one eyeglass is visible, the missing coordinates of the invisible glass need to be initialized to unify the standard of the input shape. The details of the neural network will be explained in a later chapter.

3.3 Model Selection

Model selection is the process of selecting one final machine learning model from among a collection of candidate machine learning models for a training dataset (Brownlee, 2019). In general, there is no such ranking of machine learning models and it is impossible to get the conclusion which model works best for the problem. Each machine learning model has its advantage and disadvantage, those were designed to solve the issue under pre-defined scenarios and may not perform well in solving other issues. For instance, Support Vector Machine (SVM) performs nicely in high dimensional spaces, especially when the number of data samples is less than the number of dimensions, but it is not suitable to apply to a big dataset, and the performance is underperforming when the number of features for each data points exceeds the number of training data samples (Dhiraj K, 2019).

In this research, the model selection is mainly considered with the fact of the complexity of the model, as well as the computational cost. The reason was, that online meeting platforms usually provide flexibility in joining, the participants can join freely via a laptop, or smart devices such as a tablet or phone, at the end, the trained model should be capable to run on those devices which won't reserve a lot of computational resources. To see how the complexity of models affects the blurring performance, the VGG16 (Boesch, 2023), ResNet50 (He et al., 2015), and MobileNetV3 (PA, 2020) were chosen to make the experiments.

The above mentioned models all belong to the convolutional neural network (CNN), the CNN is specifically useful for image patterns finding, and recognizing objects, categories, and classes. In 2020, Mayank Mishra indicated in his article that CNN is a class of neural networks that specializes in processing data that has a grid-like topology, an image for instance. The most common CNN architecture incorporates three layers, a convolutional layer, a pooling layer, and a fully connected layer. The input images were put through a set of convolutional filters by the convolution layer. The pooling layer is a nonlinear downsampling process that aims to reduce the number of parameters (weights) that the neural network needs to learn. The fully connected layer (hidden layer) is the last layer in the CNN architecture, it contains the combination of the Affine function ($y=Wx+b$) and Non-Linear function (activation function such as ReLu, Sigmoid). Figure 8 demonstrates the most common architecture of CNN.

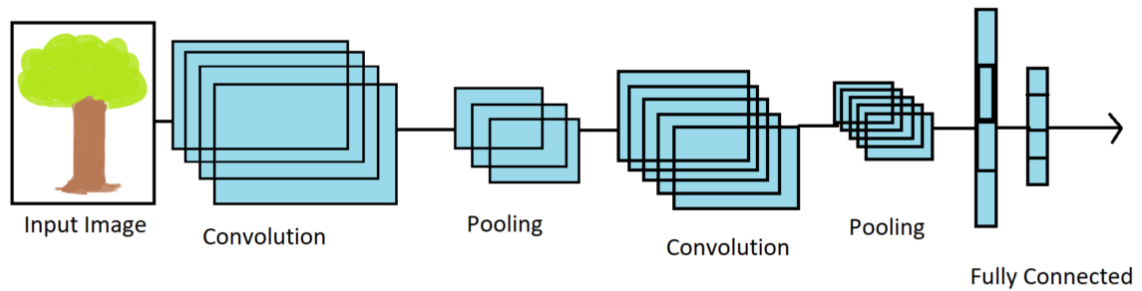


Figure 8. Common architecture of CNN (Rath, 2019)

The VGG model was proposed by Andrew Zisserman and Karen Simonyan in the year 2013, and the prototype was created during the ImageNet challenge in 2014 (Zisserman&Simonyan, 2014). Both of the inventors belong to the Visual Geometry Group (VGG) at Oxford. The number followed by VGG indicated the number of convolutional layers within the deep learning algorithm, for example, VGG16 refers to 16 layers deep neural network whereas VGG 19 refers to 19 layers deep. The VGG16 was chosen in this work due to the fewer layers it has compared with VGG19, although the number of CNN layers increasement can potentially enhance the model's ability to fit more complicated functions, in other words, better performance can be promised with more layers in CNN. Overall, the VGG16 is a better choice considering with training effort, the main reason was the weights of the neural network are updated through the backpropagation mechanism, which makes weights change and consequently minimizes the loss of the model, the deeper the layer will cause training time significantly increase. As shown in Figure 9, the VGG16 has in total of 21 layers but only 16 weighted layers are learnable with parameters.

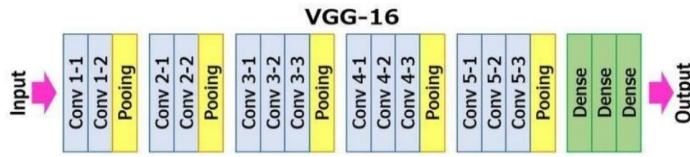
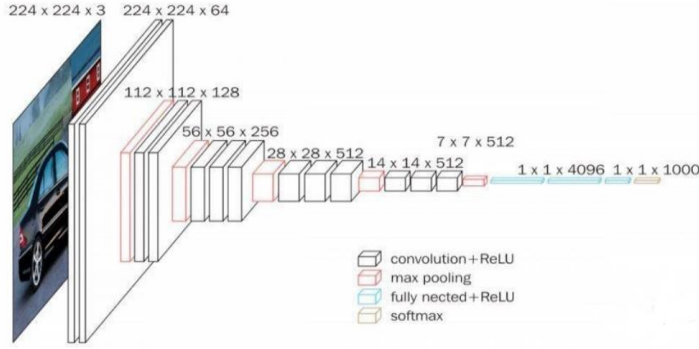


Figure 9. The architecture of VGG-16 (Rohini, 2021)

The ResNet model was introduced in the paper “Deep Residual Learning for Image Recognition” in the year 2015 by Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (He et al., 2015). ResNet50 is a specific type of CNN that forms the model by stacking residual blocks and it refers to a 50-layers convolutional neural network consisting of one MaxPool layer, one average pool layer, and 48 convolutional layers. From an architecture point of view, the ResNet has fewer filters, thus not as complex as VGG models. Especially, the ResNet50, according to the authors, they have used a special bottleneck design for the building block. The bottleneck residual blocks use 1×1 convolutions, which causes less calculation of parameters and matrix. The architecture of ResNet is demonstrated in Table 2.

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
		3×3 max pool, stride 2				
conv2_x	56×56	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$
conv3_x	28×28	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 8$
conv4_x	14×14	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 23$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 36$
conv5_x	7×7	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Table 2. ResNet architecture (He, et al., 2015)

The third model selected in this research is MobileNetV3. The MobileNet

was developed by Google and it is TensorFlow's first computer vision model specifically designed for mobile (Howard et al., 2019). It is a class of convolutional neural networks and uses depth-wise convolutions to reduce the number of parameters significantly in comparison with other CNNs. This model is well known for its less computational insensitivity and widely used in real-world mobile applications including object detection, face attributes, and localization problems, which perfectly fulfill the expectations of this research work.

The design of depth-wise separable convolution incorporates two layers, the depth-wise convolution layer, and the point-wise convolutional layer. The main reason why depthwise separable convolutions can be applied to mobile devices can be explained as follows, assume there is an input tensor with size $10 \times 10 \times 3$, and the expected output tensor is $10 \times 10 \times 256$,

1. In 2D convolutions, the number of calculations with the above-mentioned input and output tensor is, $(10 \times 10 \times 3) * (5 \times 5) * (256)$, where 5×5 is a kernel size, which can be changed case by case, in total the multiplications needed is 1,920,000.

2. With the depthwise separable convolution, the first step is filtering, which will split the original input tensor into single channels, which leads to $(10 \times 10 \times 3)$ tensor into $(10 \times 10 \times 1)$, since there are 3 RGB channels, in total the multiplications is $(10 \times 10 \times 1) * (5 * 5) * 3$, which equals 7,500. The second step is the combination, as the total number of channels required from the output tensor is 256, the multiplications needed from the second step can be calculated as $(10 \times 10) * (1 \times 1 \times 3) * 256$ which is 76,800. In summary, from the filtering step and combining step, the total multiplication is 84,300 instead of the regular convolution of 1,920,000, reaching nearly 23 times fewer multiplications required.

As Sarkar emphasized in his article in the year 2021, the MobileNets are small and very efficient with its deep learning architecture, and it has been widely applied in mobile devices. Because of its small size, the sacrifice from its performance is accuracy, in comparison with other fully convolutional architectures, but the accuracy tradeoff is very minute (Sarkar, 2021). The author has indicated the performance of MobileNet and Inception V3 model on the Stanford dogs dataset, that the Inception model reached an accuracy of 84%, while the largest MobileNet gets an accuracy of 83.3%. But in comparison of the number of parameters from

each model’s architecture, the Inception V3 has 23.2 million parameters while MobileNet has only 3.3 million parameters. The fewer parameters allow MobileNets to become the most popular deep learning model that runs on mobile or other embedded devices.

3.3.1 Customized Model Output

In this research work, the aim is to detect the position of the eyeglasses, which is the regression problem, and whether the eyeglasses are presented or not, is a binary classification problem. Instead of training two separate machine learning models, and loading the trained model individually for each problem, the customized multi-output model is used as the main solution to solve the research problem. The reason is loading two separately trained deep learning neural networks will nearly double the processing time.

In the year 2019, Yundong Li and other researchers conducted research regarding small object detection and classification in railway scenes, involved in a field called Unmanned Aerial Vehicle Remote Sensing (UAVRS). The UAVRS has been widely used in plenty of areas due to its low cost, high resolution, real-time video transmission, and flexibility. The author has accomplished the work based on Single-Shot Detector (SSD) and applied it in a railway scenario to detect abnormal objects, such as pedestrians, animals, vehicles, and rocks, which can impose high risks to the trains’ safe operation. Their work has adopted a part of VGG16 architecture and the outputs of the algorithm are similar issues compared with this research work, Figure 10 demonstrates the customized output of their research.

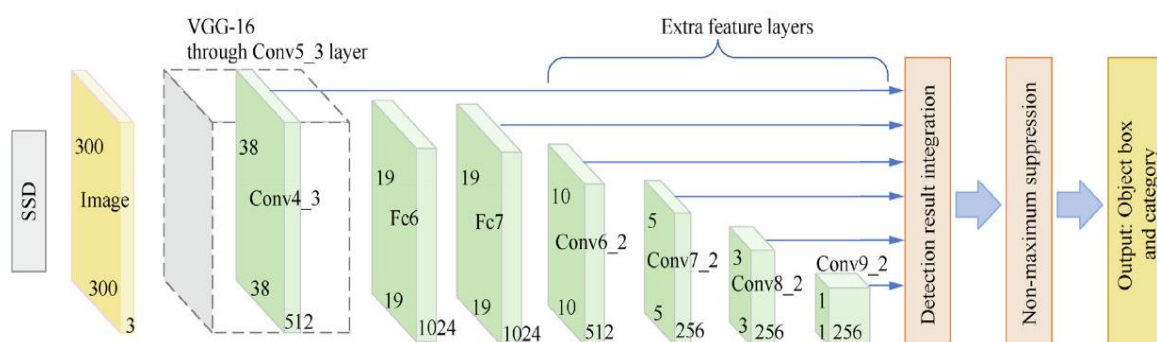


Figure 10. Framework of SSD (Yundong, et al.,2019)

In the figure above, the output of the model can address the bounding boxes and category of the object at the same time. In this research, the aim is to detect the presence of the eyeglasses

and further blurring of the eyeglasses area also requires the regression of bounding box coordinates, which is similar to the work conducted in small object detection mentioned above. In the article written by Jason Brownlee in the year 2020, the author indicated that the sequential API from Keras allows its user to create a model layer by layer for most of the problems, but the limitation is when the model has multiple inputs or outputs, or share the layers created before. By taking advantage of Keras functional API, it is possible to build the shared layer and create the multi-output model to fit the research requirements (Brownlee, 2020).

Inspired by the findings in previous paragraphs, the selected models were downloaded with the *include_top* argument set to 'False', then two of the GlobalMaxPooling2D layers were stacked separately, the first GlobalMaxPooling2D layer is connected with a Dense layer which has just one neuron, corresponding with binary classification value, either 1 or 0. The second GlobalMaxPooling2D layer was connected with another fully connected dense layer that has eight neurons and represents eight coordinates values since the bounding boxes were labeled individually in the data preparation step.

3.3.2 Loss Function Design

The loss function is an important part to train a deep learning neural network because the neural networks are trained based on stochastic gradient descent theory, it requires a loss function to be chosen and thus the model can find a way to minimize the loss, in other words, reduce the cost of the function and measure how well the model can perform with the given data. In this work, the loss function of the eyeglasses' presence is chosen with binary cross-entropy loss. The loss is calculated under Equation 1,

$$L = \frac{1}{N} \sum_i L_i = \frac{1}{N} \sum_i -[y_i * \log(p_i) + (1 - y_i) * \log(1 - p_i)] \quad (1)$$

Where y_i represents the label of sample i , either 0 or 1, p_i represents the probability of sample i belongs to class 1. Suppose the label of y is 1, then Loss is then $-\log(p(y))$, when the forecast is close to 1, the Loss is 0, or otherwise, it reaches infinite. If the label of y is 0, the loss equals $-\log(1-p(y))$, when prediction is close to 0, the Loss is 0, otherwise, it tends to positive infinity. Figure 11 demonstrates the relationship between the text explained above.

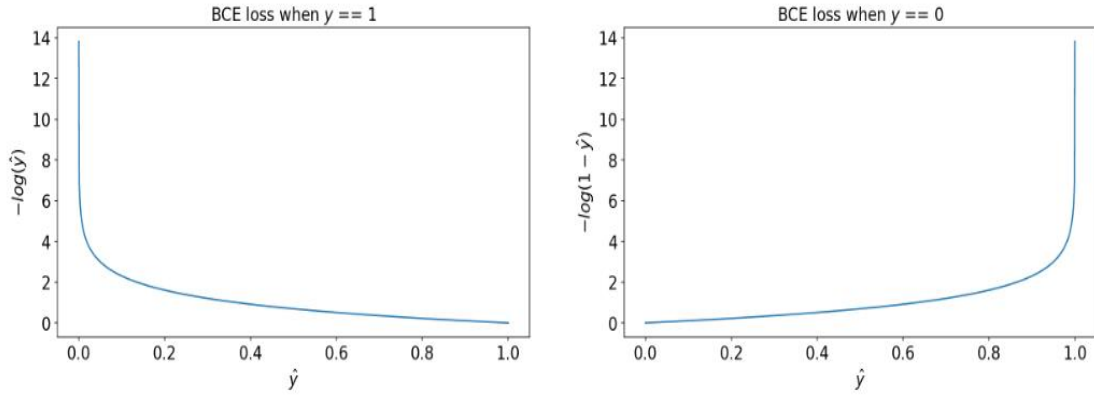


Figure 11. Binary cross entropy

Regarding with bounding box of eyeglasses position within the image, the coordinates prediction is a regression problem. In this research, the loss of the bounding box position is simply calculated by the discrepancy between real coordinates and predicted coordinates. In total, drawing a bounding box requires only two points, the upper left point, and the bottom right point. The rest of the two points can be calculated by adding the width and height of the image size. By calculating the distance of the real upper left point and bottom right point, then making the summation of those two differences, the loss is quite intuitive to understand in a 2D data space. As a result, the overall loss of the model is calculated as the sum of binary cross entropy loss and the summation of the distance of those two points in Equation 2.

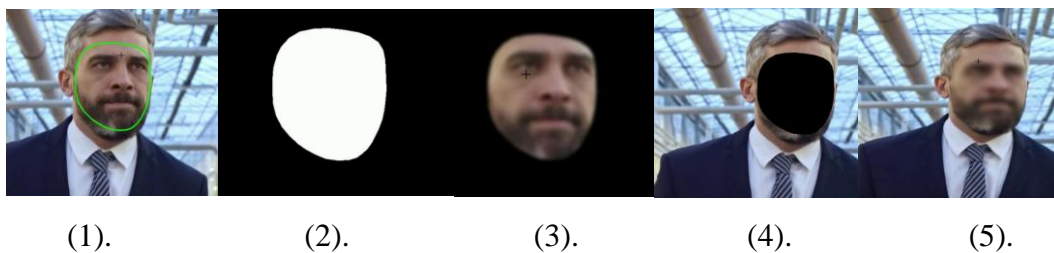
$$\sum_i (wi_{true} - wi_{pred})^2 + (hi_{true} - hi_{pred})^2 + L \quad (2)$$

3.4 POI Blurring

POI refers to a point of interest. In the image processing field, it happens that people may want to blur a dedicated area of the image to protect one's privacy. For example, Zoom offered an option that only shows participants' faces and all image background is blurred, or during the social interview, the attendee may want their face to be pixelated.¹ In this work, the main task is to blur the eyeglasses area only, the POI area from the image needed to be selected before the blurring is applied, and the selected model will predict the coordinates for this use.

¹Zoom version 5.9.3 or higher, in windows environment.

In the classic POI blurring, the most common method is to create a pure black mask image with the exactly same size as the original image, then use the coordinates to select the POI of the original image area and stack it to the mask image created. Then select the same POI from the original image, and applied the coordinates from the mask image, and stack to the original one. Consequently, the background image can be extracted and one can easily blur the POI area from the mask image and stack everything together at the end. The detailed process is shown below (Canu, 2021).



- (1). Select the POI area from the original image
- (2). Create the black mask image with the same POI area
- (3). Attach POI from the original image to mask the image
- (4). Extract the background
- (5). Stack back the blurred POI to the original image

3.4.1 Gaussian Blur

The Gaussian blur is the most classic algorithm among image blurring algorithms. As all the images are presented as pixels, the value varies from 0 to 255. The image is a matrix of pixels with values ranging from 0 to 255. If the filter is applied to each pixel, with the odd number of width and height kernel (consequently each pixel of the image will be the center of the filter kernel), and calculate the new value of this pixel by the average of the surrounding pixels, by sliding the filter one by one and keep each pixel at the center of the filter, the new pixel value will be calculated to represent the original image.

In graphics, it means the centralized pixel is losing its details by getting the average of the surrounding pixels, which generated the blurring effect. When the filter kernel size increased, the blurrier the image will be. But simply getting the average of surrounding pixels is not that reasonable, because the image pixels are consecutive, the pixel close to the centralized pixel

has a more closed relationship compared with those pixels which are far away from the centralized pixel.

In the mathematical aspect, the Gaussian Distribution is a “Bell curve”, those points close to the center of the bell curve have a larger value, and those far from the center of the bell curve will have a smaller value. The shape of the function can be described by the mean value μ and the variance value σ^2 . The mean determines the center point of Gaussian distribution on the x-axis and the variance demonstrates how the ‘Bell Curve’ is spread.

3.4.2 Facial Landmarks

Considering the localization of the eyeglasses is predicted via regression approach, since the loss is unavoidable from machine learning algorithms, the minor loss of the localization predicted from the model may cause strange outcomes on the original video frames. To be specific, the human face is symmetric, if the model predicted the position of the glasses is slightly away from central, either shifting a bit to the left or right side will cause the image looks abnormal. To solve this potential issue, facial landmarks detection has also been taken into consideration to determine the POI of the frame image.

In Python, the Dlib library provides a pre-trained 68 facial landmarks detection model, which was specially developed with HOG (Histogram of Oriented Gradients) and SVM (Support Vector Machine), the model was trained with dataset iBUG300-W, which was annotated with 68 (x, y) coordinates that map the key facial points on a person’s face as displayed in the Figure 12,

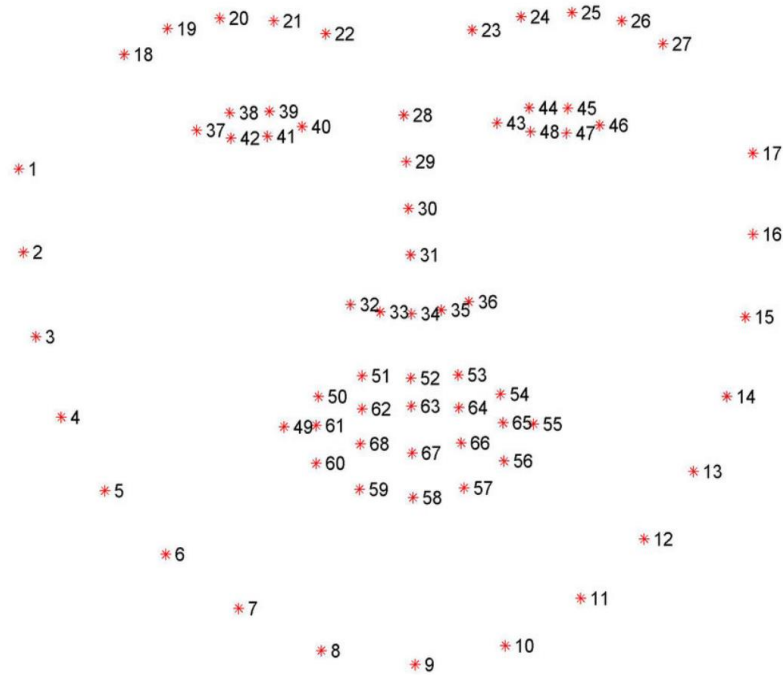


Figure 12. 68 facial landmarks (Giuseppe et al., 2018)

In the condition that the eyeglasses are presented, the most common area where the eyeglass is located ranges from landmarks index 1 to 3, 3 to 30, 30 to 16, then from 16 across both eyebrows and back to index 1. Those indexes formulated an area to cover the eyeglass position and split the issue into a binary classification problem and an image processing task with transfer learning. Based on the high accuracy of Dlib facial landmarks detection, this approach will be carried out in the experiment part to make the comparison with the original solution.

3.5 Experiments

The experiment of this research is designed in several phases. In the first phase, the selected three convolutional neural networks will be trained separately with multi-output, classification of eyeglasses presence, and coordinates of eyeglasses' position. Taking the advantage of *cv2* Python library, the external webcam will be used to simulate the real-world meeting platform. After models are trained with the same dataset collected in the previous chapter and fine-tuned, the *cv2* library is used to extract the frames of the video, and the performance of the eyeglasses regression and eyeglass presence will be examined in a real-time scenario. The image process/blurring happens within the detection, by using the coordinates predicted from the regression of the given dataset, the bounding box will be drawn to check if the position of the

predicted eyeglass is accurate. Of course, by taking off eyeglasses and wearing eyeglasses while the webcam is running, the accuracy of the binary classification can be measured as well.

In case the model classified the eyeglasses is not presented, there is no image processing needed, and the default coordinates of eyeglasses will be signed as zeroes. If the model predicted that the eyeglasses are presented (depending on the users' head position), the coordinates predicted by the trained model will be used to map the POI area on each video frame, at the end, the blurring process took place, the blurred video frame will be returned to the stream.

To measure the performance of each model, the head position is designed from various angles and the webcam remained in the same location on the monitor. After all, the model that shows the best capability will be chosen as the solution for the research questions, and the Gaussian blurring method will be compared with Dlib facial landmarks, which aims to find the optimal regression method of eyeglasses position.

3.6 Evaluation of Performance

TensorFlow, as one of the most popular machine frameworks, provides a tool named TensorBoard that offers the measurements and visualizations needed in machine learning workflow, consequently, the performance can be improved based on the measurements. It supports the possibility of tracking the metrics such as loss and accuracy during the experiment process, as well as visualizing model graphics. Since the loss function was built by summing up both binary classification loss and localization loss, the evaluation of selected models will be mainly based on analytics of total loss while the model was trained, also the performance of classification loss and localization loss in comparison between training dataset and validation dataset.

3.6.1 Evaluation of Classification

Since the classification task and regression task are not the same issue, this research work aims to use a single deep learning algorithm to solve the complex problem simultaneously, the evaluation of each outcome from the dense layer will be evaluated separately. For binary classification, simply using accuracy is a bit risky and vulnerable. For instance, if the dataset has exactly one thousand data points, in the situation that those data points are imbalanced, over 800 data points are belonging to the same category, even though the model can predict or

classify all test data as the above-mentioned class/category, it will reach over 80% of accuracy despite the data is imbalanced and the testing dataset may no longer the same imbalance distribution. To overcome this issue, the F1-score is used to evaluate how well the model can perform with the classification task.

As Kundu introduced in his article in 2023, the F1-score is a machine learning metric that assesses the capability of the model by calculating class-wise performance rather than simply calculating overall accuracy (Kundu, 2023). The metrics used to calculate the F1-score are called precision and recall. The definition and formulas are demonstrated below,

- True Positive (TP) is, the number of times the model correctly classified a positive data sample as positive
- False Negative (FN) is, the number of times the model incorrectly classified a positive sample as negative
- False Positive (FP) is, the number of times the model incorrectly classified a negative sample as positive
- True Negative (TN) is, the number of times the model correctly classified a negative sample as negative

Usually, the accuracy is calculated through the formulation in Equation 3,

$$Accuracy = \frac{True_{positive} + True_{negative}}{True_{positive} + True_{negative} + False_{positive} + False_{negative}} \quad (3)$$

To avoid the scenario mentioned at the beginning of this sub-chapter, the metric to evaluate the performance of classification of the model has been determined to use the F1-score. The f1 is calculated via precision and recall. Precision refers to the ratio between the number of positive samples correctly classified to the total number of samples classified as positive, including both correctly and incorrectly predicted samples. This calculation simply measures how the model performs in classifying a sample as positive in terms of accuracy. The formula is displayed in Equation 4,

$$Precision = \frac{True_{positive}}{True_{positive} + False_{positive}} \quad (4)$$

From the formula above, it's reliable to tell how is the model performance in classifying data samples as positive.

The recall is calculated as the ratio between the number of positive data samples correctly classified as Positive to the total number of positive samples. The recall can measure the model's ability to detect positive samples. The formula is demonstrated in Equation 5,

$$Recall = \frac{True_{positive}}{True_{positive} + False_{negative}} \quad (5)$$

Finally, the F1-score can be calculated by Equation 6,

$$F1 \ score = 2 * \frac{Precision * Recall}{Precision + Recall} \quad (6)$$

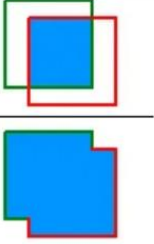
As the formula demonstrated above, it takes into consideration both precision and recall, which combines two metrics as a single one metrics. In the real-world scenario, the data is not evenly distributed and f1 is one of the good matrices to conquer the imbalanced dataset.

3.6.2 Evaluation of Regression

Unlike the classification issue, simply measuring if the detection is correct or not using precision or recall is not enough. The reason was the coordinates of bounding boxes can be miss drawn after the objects were detected, this can be explained in the situation where the ground truth is an object within a vertical rectangle, but the predicted object location is a horizontal rectangle that covers the same object.

The Intersection over Union (IoU) is a metric in object detection that evaluates the degree of overlap between the ground truth (gt) and prediction (pd). The ground truth and the prediction can be any shape as a rectangular box, circle, or irregular shape (Kiprono, 2020).

The formula in Equation 7 demonstrates how IoU is calculated,

$$IOU = \frac{\text{area of overlap}}{\text{area of union}} = \frac{\text{Diagram 1}}{\text{Diagram 2}} \quad (7)$$


The value of IoU ranges from 0 to 1, when ground truth and predicted bounding box perfectly overlap each other, the value of IoU is 1, otherwise, there is no overlap at all, and the value of IoU will be 0. Usually, the threshold is defined in object detection tasks to determine whether the predicted coordinates are considered a TP, FP, TN, or FN. In this research, the threshold was set to 0.5 as it is the commonly used threshold in object detection.

Considering the position of one's body may move in a multi-angle during a meeting conference, the models' ability to overcome the complex situation and make rather sufficient estimations will be measured. After a model is trained and fine-tuned, the performance of the model will be checked with different head positions. Furthermore, the blurring algorithm will be applied while the video stream is running, and the regular Gaussian blur method will be compared with Dlib facial landmarks detection based on their runtime. The optimal blur solution will take less time to return the blurred video frame to ensure the smoothness of the video stream flow.

4. Results and Discussion

In this chapter, the evaluation was made in two parts. Firstly, the performance of the selected models is checked via metrics discussed in previous chapter 3.6, the F1-score (binary classification), and IoU (localization of object detection). Secondly, the blurring methods will be compared regarding the execution of time. The test dataset includes 684 images, split from those images collected during data collection, containing both scenarios where eyeglasses are present and otherwise. Thirdly, the model was trained and fine-tuned, saved to use for coordinates prediction and classification of eyeglasses. The image processing step tested Gaussian blur and dlib facial landmarks detection to blur the dedicated area, compared in respect of processing time.

4.1 Customized VGG16

The result from the VGG16 model reached the optimal training result from epoch 10, after epoch 10, the loss of the validation dataset starts to increase. The same situation occurred compared with total loss, the yellow line indicates how the loss of the validation dataset starts to raise. This is the indicator that the model will be exposed to overfitting with the increasement of epochs it's trained on. In case when ground truth of the class is 0, and the predicted value is not 0, but infinitely close to the ground truth, the F1-score was calculated by setting the threshold to 0.5. The threshold can help the model to determine whether the probabilities predicted from the sigmoid layer is a true positives or not. If the probability is bigger than 0.5, the model considers the eyeglasses are presented, or otherwise it's not presented.

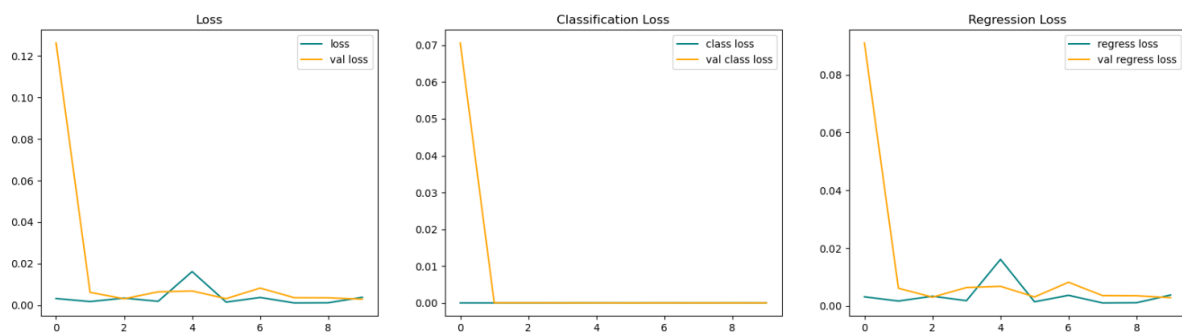


Figure 13. Training performance of customized VGG16

Overall, the classification task performed nicely after the first epoch, the classification loss is infinitely close to 0. The regression loss started to drop after the first epoch, after 10 epochs

the regression loss reached 0.0028 on the validation dataset. By randomly selecting 4 images from the test dataset, the predicted bounding boxes are shown in Figure 14.



Figure 14. Predicted bounding boxes by customized VGG16

Finally, the cv2 library was applied to simulate the real-world video streaming scenario. With the Gaussian blurring method, the video went smoothly in the capture, and the average process time for a single frame decomposed from the video source took an average of 14 ms per frame. As defined in the Computer Graphics field, if a video is streamed with 30 frames per second, it is considered real-time. From the experiment above, classify the presence of the eyeglasses and predict the associated coordinates for 30 frames will consume around 420ms, which is still less than half a second. Besides, using dlib facial landmarks to determine the blurring area causes clear detection, along with the image blurring process, a single frame took an average of 25ms. The simulated video console activated from the cv2 library is nearly frozen while performing the same task compare with the Gaussian blur method.

4.2 Customized ResNet50

During the training process, the customized ResNet50 has been sensitive to a certain extent. The trends of loss figures are convergent with the increasement in epochs, but the model seems too sensitive and the loss of the validation dataset started to increase after the 12th epoch, which means lead to overfitting. The reason behind this is due to its residual architecture, the model itself is deep but slim. To overcome this situation, various fine-tuning methods were applied. During the training process, different optimizers were tested, and learning decay changed to a much smaller value compared with the VGG16 model mentioned in the previous paragraph. Also, freezing the trainable layers and only training the remaining two dense layers were examined as well. The performance during model training can be found in Figure 15.

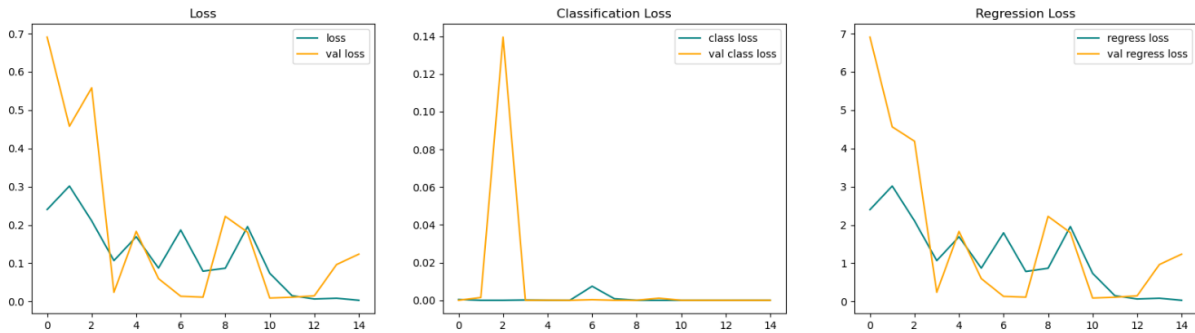


Figure 15. Training performance of customized ResNet50

To check how the trained ResNet50 performs, the numpy iterator method was called to randomly select 4 images from the test dataset. The performance of the model is as expected, that detected rather good coordinates when eyeglasses are presented, but the IoU is not as precise compared with the customized VGG16 model, the predicted bounding boxes can be found in Figure 16 as follows,

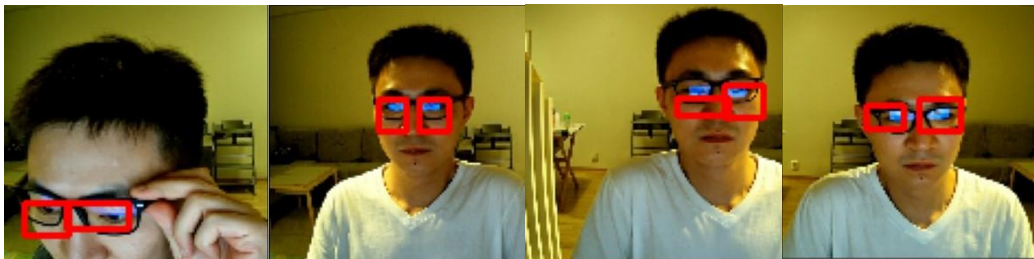


Figure 16. Predicted bounding boxes by customized ResNet50

Because the performance evaluated upon dlib facial landmarks detection demonstrated significant detention on the image processing part, the experiment from this procedure was deprecated. The image blurring was carried out with the Gaussian blur method directly. Under the same setup, the customized ResNet50 model can correctly classify whether the participant is wearing glasses or not. On average, a single frame decoded from captured video took around 15ms per frame. To manipulate 30 frames, the entire procedure will take half a second as estimated.

4.3 Customized MobileNetV3

The training process of MobileNetV3 took longer compare with both customized VGG16 and ResNet50. The previous models demonstrated overfitting through the increasement of the validation dataset, but MobiNetV3 was taking more epochs to complete the training procedure. The ResNet50 started overfitting after running 12 epochs, but MobileNetV3 keeps convergent

from the loss curve displayed in Figure 17, the regression loss still keeps decreasing even after 30 epochs.

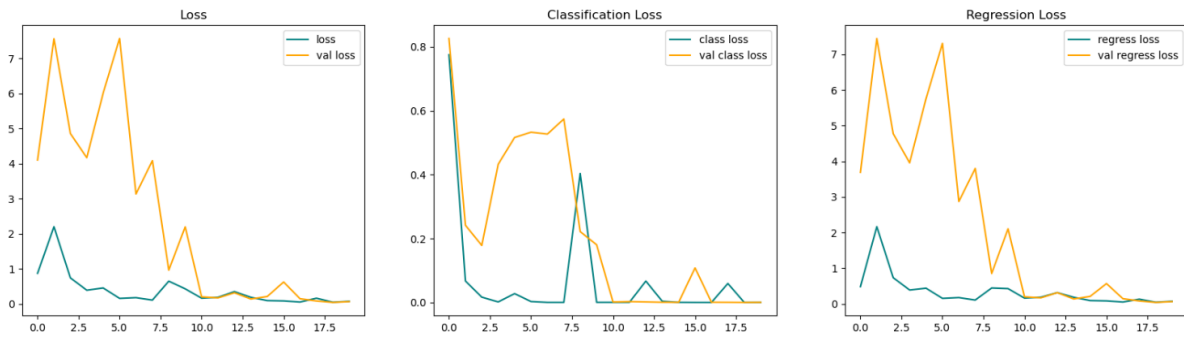


Figure 17. Training performance of customized MobileNetV3

The performance was evaluated again via the same procedure as applied to previous models, the trained model achieved a reasonable outcome as well. The model can detect the correct location where the eyeglasses are located in the image, the reflective areas are covered by the bounding boxes the model predicted. The random 4 images from the test dataset were used to evaluate how is the model performance, the result is shown in Figure 18.



Figure 18. Predicted bounding boxes by customized MobileNetV3

The real-time simulation is also applied at the end, the model can perfectly detect whether the participant is away from the camera or not, and the eyeglasses were detected as well. The average consumed time with the Gaussian blur processing is about 12ms. To complete 30 frames from a captured video, the overall time will take around 360ms. Table 3 summarizes the F1-score, IoU, and processing time to blur a POI area.

Models	F1-score	IoU(threshold 0.5)	The time needed for 30 frames/ms
VGG16	0.721	0.49	≈ 420
ResNet50	0.489	0.53	≈ 450
MobileNetV3	0.614	0.62	≈ 360

Table 3. Summary of results obtained with the customized models

Overall, those customized models can detect the presence of the eyeglasses while the video cam is running, and blurring the area where the eyeglasses are located was succeeded under reasonably time-consuming. Especially the MobileNetV3, because of its architectural design, the latency is the smallest among all selected models. The reflective areas from eyeglasses were blurred according, without sensitive information can be seen. When a participant appears in the video streaming, all those models can simultaneously detect the presence of eyeglasses, and blur the area without visible latency, when the participant took off their eyeglasses, the model stopped the blurring effect, and the video streaming runs normally. Last, the blurring of the reflective area doesn't seem to be unusual as well, since the kernel size (9, 9) from the Gaussian blur was specially chosen to blur the reflection and avoid too much blurring effect that makes the image looks abnormal.

5. Conclusion

5.1 Summary

In this thesis work, the aim was to prevent information leakage while participating in online meeting platforms. To overcome the existing issue, the data was collected to train the deep learning models. The dataset contains thousands of self-created images, labeled and annotated according to research requirements. In total, there were three convolutional neural networks selected, the classic VGG16, one of the principal residual convolutional neural networks ResNet50, and MobileNet which is well known to run on smart devices with sufficient performance. Each model was modified to produce two outputs, the binary classification output indicates the presence of the eyeglasses, and the regression output predicts the coordinates of both the left and right eyeglasses in a single image.

The blurring effect was tested with traditional dlib facial landmarks detection and the Gaussian blur, the optimal method was chosen by taking into consideration of real-time video streaming, and several kernel size have experimented with to ensure the ordinary look of the image returned by the model. Although the F1-score of MobileNetV3 and ResNet50 is less than VGG16, all of those models performed well to classify the presence of the eyeglasses while hiding the entire body away from the video and taking the eyeglasses off. There was no latency while the video stream is running, and all the models were able to handle the designated tasks, eventually, the reflective area can be blurred with the regular representation.

5.2 Future work

In this research, the data was self-collected to overcome the issue that synthesized eyeglasses data doesn't contain the reflection. In purpose to guarantee the quality of the detection, the eyeglasses were separated and annotated to the left and right. The images were collected by the webcam while the participant was sitting in front of the camera, the complexity was added by moving the upper body and head-turning. Once the participant is moving far from the webcam, the model may fail to detect the presence of the eyeglasses as it's too small to be seen.

Furthermore, the reflective items are not limited to eyeglasses only, the information leakage can happen from any reflective surface, for instance, a small metallic water bottle. In future research, one may try to add anchor boxes introduced by Fast RNN papers, the powerful

method to detect multiple objects. Regarding dlib facial landmarks detection, as it was developed with SVM and HOG many years ago, there has been a lot of deep-learning-based facial landmarks detection algorithm developed, and Google Face Mesh which provides 468 face landmarks in 3D with multi-face support that runs on mobile phone even.

Furthermore, instead of annotating each eyeglass separately, it will be better to compare with data that has only one area annotated. Since the blurring happened twice with both the left and right eyeglasses regions, the time needed in the blurring process is certainly more than a single region blurring. But on the other hand, blurring a single area when the position of the eyes is not at the same horizontal level, the result may look abnormal, still it may be worth making additional comparisons with the solution this thesis work provided.

References

- Long, Y. *et al.* (2023) *Private eye: On the limits of textual screen peeking via eyeglass reflections in Video conferencing*, *arXiv.org*. Available at: <https://arxiv.org/abs/2205.03971> (Accessed: 20 January 2023).
- Arghire, B. (2022) *Eyeglass reflections can leak information during video calls*, *SecurityWeek*. Available at: <https://www.securityweek.com/eyeglass-reflections-can-leak-information-during-video-calls/> (Accessed: 20 January 2023).
- Kagan, D., Alpert, G.F. and Fire, M. (2020) *Zooming into video conferencing privacy and security threats*, *arXiv.org*. Available at: <https://arxiv.org/abs/2007.01059> (Accessed: 23 January 2023).
- Fatpos Global Pvt. Ltd. *Covid-19 impact on global web conferencing market by type; by downstream fields and region –analysis of market size, share and trends for 2014 – 2019 and forecasts to 2030* (2022) *ReportLinker*. Available at: https://www.reportlinker.com/p06191508/COVID-19-Impact-on-Global-Web-Conferencing-Market-By-Type-By-Downstream-Fields-and-Region-Analysis-of-Market-Size-Share-and-Trends-for-and-Forecasts-to.html?utm_source=GNW (Accessed: 20 January 2023).
- Watanabe, S. and Hasegawa, M. (2021) *Reflection removal on eyeglasses using Deep Learning* / *IEEE conference, Reflection Removal on Eyeglasses Using Deep Learning*. Available at: <https://ieeexplore.ieee.org/abstract/document/9501489/> (Accessed: 7 February 2023).
- Whang, S.E. *et al.* (2023) *Data Collection and quality challenges in Deep learning: A data-centric AI perspective - the VLDB Journal*, *SpringerLink*. Available at: <https://link.springer.com/article/10.1007/s00778-022-00775-9> (Accessed: 20 February 2023).
- Du, C. and Su, G. (2005) *Eyeglasses removal from facial images*. Available at: <https://www.sciencedirect.com/science/article/abs/pii/S0167865505001133> (Accessed: 20 February 2023).
- Khabarлак, K.S. and Koriashkina, L.S. (2022) *Fast Facial Landmark Detection and Applications: A Survey*. Available at: https://www.researchgate.net/publication/346444595_Fast_Facial_Landmark_Detection_and_Applications_A_Survey (Accessed: 24 February 2023).
- Wu, Y. and Ji, Q. (2018) *Facial Landmark Detection: A Literature Survey - International Journal of Computer Vision*, *SpringerLink*. Available at: <https://link.springer.com/article/10.1007/s11263-018-1097-z> (Accessed: 24 February 2023).
- Rosebrock, A. (2021) *Facial landmarks with dlib, opencv, and python*, *PyImageSearch*. Available at: <https://pyimagesearch.com/2017/04/03/facial-landmarks-dlib-opencv-python/> (Accessed: 2 March 2023).

- Suresh, V. (2021) *Face detection and landmarks using dlib and opencv, Face Detection and Landmarks using dlib and OpenCV*. Available at: <https://vigneshs4499.medium.com/face-detection-and-landmarks-using-dlib-and-opencv-8c824f50cc78> (Accessed: 6 March 2023).
- Kazemi, V. and Sullivan, J. (2014) *One millisecond face alignment with an ensemble of regression trees, One Millisecond Face Alignment with an Ensemble of Regression Trees*. Available at: https://www.researchgate.net/publication/264419855_One_Millisecond_Face_Alignment_with_an_Ensemble_of_Regression_Trees (Accessed: 7 March 2023).
- Tang, X. *et al.* (2018) *Facial landmark detection by semi-supervised Deep Learning, Neurocomputing*. Available at: <https://www.sciencedirect.com/science/article/pii/S0925231218301139> (Accessed: 11 March 2023).
- Lyu, J., Wang, Z. and Xu, F. (2022) *Portrait eyeglasses and shadow removal by leveraging 3D Synthetic Data, Computer Vision and Pattern Recognition*. Available at: <https://arxiv.org/abs/2203.10474> (Accessed: 11 March 2023).
- Wang, Z. *et al.* (2021) *Single image portrait relighting via explicit multiple reflectance channel modeling, OPUS at UTS | Open Publications of UTS Scholars*. Available at: <https://opus.lib.uts.edu.au/handle/10453/147134> (Accessed: 13 March 2023).
- Lee, Y.-H. and Lai, S.-H. (2020) *Byeglassesgan: Identity preserving eyeglasses removal for face images, arXiv.org*. Available at: <https://arxiv.org/abs/2008.11042> (Accessed: 13 March 2023).
- Amit, Y., Felzenszwalb, P. and Girshick, R. (2020) *Object detection, SpringerLink*. Available at: https://link.springer.com/referenceworkentry/10.1007/978-3-030-03243-2_660-1#Sec2 (Accessed: 17 March 2023).
- Dilmegani, C. (2022) *What is data augmentation? techniques & examples in 2023, AIMultiple*. Available at: <https://research.aimultiple.com/data-augmentation/> (Accessed: 17 March 2023).
- Howard, A.G. *et al.* (2017) *MobileNets: Efficient convolutional neural networks for Mobile Vision Applications, arXiv.org*. Available at: <https://arxiv.org/abs/1704.04861> (Accessed: 24 March 2023).
- Brownlee, J. (2019) *A gentle introduction to model selection for Machine Learning, MachineLearningMastery.com*. Available at: <https://machinelearningmastery.com/a-gentle-introduction-to-model-selection-for-machine-learning/> (Accessed: 24 March 2023).
- K, D. (2023) *Top 4 advantages and disadvantages of support vector machine or SVM, Medium*. Available at: <https://dhirajkumarblog.medium.com/top-4-advantages-and-disadvantages-of-support-vector-machine-or-svm-a3c06a2b107> (Accessed: 20 March 2023).

- Boesch, G. (2023) *VGG very deep convolutional networks (vggnet) - what you need to know*, *viso.ai*. Available at: <https://viso.ai/deep-learning/vgg-very-deep-convolutional-networks/> (Accessed: 26 March 2023).
- Mishra, M. (2020) *Convolutional Neural Networks, explained*, *Medium*. Available at: <https://towardsdatascience.com/convolutional-neural-networks-explained-9cc5188c4939> (Accessed: 26 March 2023).
- Jordan, J. (2018) *Common architectures in convolutional neural networks.*, *Data Science*. Available at: <https://www.jeremyjordan.me/convnet-architectures/#vgg16> (Accessed: 26 February 2023).
- Rath, S.R. (2019) *Convolutional neural network architectures and variants*, *DebuggerCafe*. Available at: <https://debuggercafe.com/convolutional-neural-network-architectures-and-variants/> (Accessed: 2 April 2023).
- Learning, G. (2021) *Everything you need to know about VGG16*, *Medium*. Available at: <https://medium.com/@mygreatlearning/everything-you-need-to-know-about-vgg16-7315defb5918#:~:text=What%20is%20VGG16%20used%20for,to%20use%20with%20transfer%20learning.> (Accessed: 2 April 2023).
- He, K. *et al.* (2015) *Deep residual learning for image recognition*, *arXiv.org*. Available at: <https://arxiv.org/abs/1512.03385> (Accessed: 5 April 2023).
- Simonyan, K. and Zisserman, A. (2015) *Very deep convolutional networks for large-scale image recognition*, *arXiv.org*. Available at: <https://arxiv.org/abs/1409.1556> (Accessed: 5 April 2023).
- PA, S. (2020) *An overview on MobileNet: An Efficient Mobile Vision CNN*, *Medium*. Available at: <https://medium.com/@godeep48/an-overview-on-mobilenet-an-efficient-mobile-vision-cnn-f301141db94d> (Accessed: 5 April 2023).
- Li, Y. *et al.* (2020) *Multi-block SSD based on small object detection for UAV railway scene surveillance*, *Chinese Journal of Aeronautics*. Available at: <https://www.sciencedirect.com/science/article/pii/S1000936120301126> (Accessed: 1 March 2023).
- Amato, G. *et al.* (2018) *A comparison of face verification with facial landmarks and deep features*, *A Comparison of Face Verification with Facial Landmarks and Deep Features*. Available at: https://www.researchgate.net/publication/338048224_A_Comparison_of_Face_Verification_with_Facial_Landmarks_and_Deep_Features (Accessed: 14 April 2023).
- Gad, A.F. (2021) *Accuracy, precision, and recall in deep learning*, *Paperspace Blog*. Available at: <https://blog.paperspace.com/deep-learning-metrics-precision-recall-accuracy/> (Accessed: 9 April 2023).

- Kundu, R. (2022) *F1-score in Machine Learning: Intro & Calculation, V7*. Available at: [https://www.v7labs.com/blog/f1-score-guide#:~:text=for%20Machine%20Learning-,What%20is%20F1%20score%3F,predicti on%20across%20the%20entire%20dataset](https://www.v7labs.com/blog/f1-score-guide#:~:text=for%20Machine%20Learning-,What%20is%20F1%20score%3F,predicti on%20across%20the%20entire%20dataset.). (Accessed: 14 March 2023).
- Koech, K.E. (2020) *On object detection metrics with worked example, Medium*. Available at: <https://towardsdatascience.com/on-object-detection-metrics-with-worked-example-216f173ed31e> (Accessed: 16 March 2023).
- Jose, I. (2018) *Facial mapping (landmarks) with dlib + python, Medium*. Available at: [https://towardsdatascience.com/facial-mapping-landmarks-with-dlib-python-160abcf7d672#:~:text=What%20is%20Dlib%3F,iBUG300%2DW%20dataset%20was%20used](https://towardsdatascience.com/facial-mapping-landmarks-with-dlib-python-160abcf7d672#:~:text=What%20is%20Dlib%3F,iBUG300%2DW%20dataset%20was%20used.). (Accessed: 13 April 2023).
- Sarkar, A. (2021) *Understanding depthwise separable convolutions and the efficiency of MobileNets, Medium*. Available at: <https://towardsdatascience.com/understanding-depthwise-separable-convolutions-and-the-efficiency-of-mobilenets-6de3d6b62503> (Accessed: 20 April 2023).
- Müller, A.C. and Guido, S. (2017) *Introduction to machine learning with python: A guide for data scientists*. Sebastopol, United States: O'Reilly Media. Page 27-129.