AI-BASED IMAGE RECOGNITION ON THE QUALITY OF A COVID-19 TEST BOX

Final Report

Information Technology Capstone Project

COMP5703

Group Members

1. Sicheng Guo (510191768) [Group Leader]
2. Yiming Nie (510032988)
3. Rencong Wang (510061737)
4. Linxin Sun (510020837)
5. Wenyan Hu (500344781)

CONTRIBUTION STATEMENT

Our group, taking project CS08-1, with group members Sicheng Guo, Wenyan Hu, Linxin Sun, Yiming Nie and Rencong Wang, would like to state the contributions each group member has made for this project during semester 2 2022:

- Sicheng Guo: Project scheduling and monitoring, Front-end and Back-end Android development, Model design, building, training and conversion
- Wenyan Hu: Project management, document organisation, App front-end design
- Linxin Sun: Project management, document organisation, App back-end design
- Yiming Nie: Model design, building and training, document organisation
- Rencong Wang: Model design, building and training, document organisation

All group members agreed on the contributions listed in this statement by each group member.

Signatures:

ABSTRACT

The emergence and spread of COVID-19 have entirely changed the world. According to the Australian government's official website, two tests can be applied to detect if the COVID-19 virus has infected people: polymerase chain reaction (PCR or RT-PCR) and rapid antigen self-tests (RATs). Compared with PCR, people can self-test at home using RAT test kits, which is a convenient and efficient way to determine if they are infected. There is a total of 59 brands of self-test kits that the Australian government approves for customers to purchase. Products produced by different brands have different characteristics, such as usage methods, sensitivity to viruses, etc. However, it is difficult for customers to obtain such information intuitively and directly because it does not mention them on the packaging or instruction. The project will provide an Android application with a built-in deep learning model to help users get such information quickly using their mobile phones and assist in making decisions on test box purchases.

Keywords: COVID-19; Self-test; Machine learning; Image recognition; Android Application development; OpenCV; TensorFlow

TABLE OF CONTENTS

1. INTRODUCTION

In Australia, COVID-19 is still a pressing issue that has an impact on society. The Australian Department of Health and Aged Care advises people to have a COVID-19 test even if they only exhibit minor symptoms. Rapid antigen self-tests (RATs) are frequently utilised because they are convenient and portable (Australian Government Department of Health, 2020). The Australian Government has approved 59 different RAT brand names (Therapeutic Goods, 2022). Each RAT possesses unique qualities, such as sensitivity. These features are typically not straightforward, making it challenging for consumers to access them immediately. To create a model with good performance, the project will use a variety of deep learning models and carry out various tests. The project will then create an application and employ the previously trained model to recognise the test box manufacturer. Simply taking a picture of the RAT test kits without opening them will provide users with comprehensive information about them.

2. LITERATURE REVIEW

2.1 Introduction

One of the active areas of artificial intelligence is object detection, where various advanced systems are already in use. It is crucial in the industrial or medical fields since it decreases the demand for labour and boosts productivity. Brand recognition, a subset of object detection, can offer more detailed product information based on identified products. For instance, consumers do not need to know the exact product name to scan it using the camera. The model will automatically give back information like the product name, reviews, and other details that are difficult to find in physical and mortar-stores. The user can conduct information searches more quickly and effectively using these methods.

In this project, the application will recognise the brand of RAT boxes and provide users with information on some aspects, such as accuracy and sensitivity, that are not listed on the box. Therefore, the model is a supervised classification model. More details and preparatory works will be illustrated in the following sections.

2.2 Relevant Knowledge

Artificial intelligence's deep learning (DL) simulates how the human brain's nerve connection mechanism. The performance of DL models will surpass that of conventional techniques as the size of the data set grows, and they can automatically extract features from big data sets. The DL model is also excellent for classifying images. DL models have an advantage over conventional machine learning methods because they can extract features from images from a multidimensional perspective. This will assist in lessening the impact of bias on the photographs and enhance accuracy. Deep learning expertise will be needed for this project to create the classification model.

Moreover, Android development skills will be necessary to develop the application and install the model on the phone. Android platform almost dominates the mobile platform with 71% mobile OS market share in 2022, which means most people can use the application using their cell phones. Besides, an application that can complete complex prediction calculations offline will better meet users' demands because it solves the unavailability of software due to no connection to the network.

According to the client's requirements, the project group must collect information about RAT test boxes for different brands with different sensitivities. After checking the Australian government website, it is known that the Therapeutic Goods Administration (TGA) classifies the sensitivities of RAT test boxes into three types of Positive Percent Agreement (PPA) (Therapeutic Goods, 2022) as follows:

- Acceptable sensitivity - clinical sensitivity greater than 80% PPA
- High sensitivity - clinical sensitivity greater than 90% PPA
- Very high sensitivity - clinical sensitivity greater than 95% PPA

The following figure shows a part of the table listing all government-approved test boxes, including relevant detailed information.

Name of self-test* and how to use the test	Sample type used	Australian Sponsor (supplier)	Manufacturer	ARTG	Clinical Sensitivity
2019 nCoV Ag Test (Latex Chromatography Assay) - Self - test (pdf 1.3Mb) (Rapid antigen test)	Nasal swab	BGI Health (AU) Company Pty Ltd	Innovita (Tangshan) Biological Technology Co Ltd (China)	38750	Very high sensitivity
2019-nCoV Ag Rapid Test Kit (Immunochromatography for self-testing) (pdf,1.56Mb) (Rapid antigen test)	Nasal swab	Big Start Pty Ltd	Guangzhou Decheng Biotechnology Co Ltd (China)	387020	Very high sensitivity
All Test COVID-19 Antigen Rapid Test (Oral Fluid) Self-Test (ICOV-802H) (pdf,366kb) (Rapid antigen test)	Oral fluid	AM Diagnostics	Hangzhou Alltest Biotech Co Ltd (China)	376310	High sensitivity
All Test SARS-CoV-2 Antigen Rapid Test (Nasal Swab) Self-Test (INCP-502H) (pdf,649kb) (Rapid antigen test)	Nasal swab	AM Diagnostics	Hangzhou Alltest Biotech Co Ltd (China)	376310	Very high sensitivity

Figure 2.1 Information about self-test for government-approved brands

Consequently, three brands of self-test kits were selected for this project based on these three different sensitivity levels: RightSign, Clungene and GICA.

2.3 Existing Solutions

2.3.1 ResNet

ResNet is a CNN-based deep learning model. It solves the issue of CNN models having a limit on the number of layers when faced with multi-class classification problems and taking too long to train because of having too many parameters (He, Zhang, Ren, & Sun, 2015). Since the ResNet model allows features to circumvent some layers by establishing shortcuts between each block, its performance does not decrease as the number of the model's layers rises.

The ResNet model can handle multi-category classification issues due to these benefits. Benali Amjoud and Amrouch (2020) used recent convolutional neural network models to extract features from photos in the ImageNet dataset. For instance, VGG and AlexNet. The dataset includes a variety of complex picture types, including bees, flowers, and aircraft, which will present a considerable challenge for the model. ResNet illustrates the benefits of having fewer parameters, which include 25.56 million parameters for ResNet50 and 44.55 million for ResNet100. Compared to AlexNet and VGG16, both of the models have more parameters. Thus, the time of the training is also promising. While other models' accuracy is around 85%, the ResNet models' accuracy is around 90%.

2.3.2 VGG

A larger model with more layers may be better able to handle complex data classification problems. The model may take a long time to update the weights and may find it challenging to converge if there are too many parameters. Kernel size can be decreased from the standard 5*5 to a 3*3 size to increase the number of layers without increasing the number of parameters (Simonyan & Zisserman, 2015). In the ImageNet dataset, the VGG model can increase the depth to a maximum of 19 layers compared to a standard CNN model and also achieved third place in the competition on ImageNet.

Overall, VGG has the advantage of having fewer parameters and more layers while keeping the performance.

2.3.3 InceptionV3

GoogLeNet first introduced InceptionV3 in the 2014 ILSVRC classification challenge, which yielded high performance with a deeper and broader network architecture. It has a broader application domain which means an extraordinary architectural improvement. (Russakovsky et al., 2015) Compared with VGGNet, the Inception architecture of GoogLeNet is capable of performing well with strict memory and computational budget constraints because of fewer parameters. Inception architecture mainly proposed the following two feature extraction and mapping strategies. The first is factorisation into smaller convolutions. For example, it uses double 3x3 convolutions to replace a 5x5 convolution, which decreases calculation costs and improves training efficiency while avoiding losing feature representations. (Szegedy et al., 2015) The other one is spatial factorisation into asymmetric convolutions. A 3x3 convolution, for instance, can be factorised by a 3x1 and a 1x3 convolution, which saves computing resources and enhance the representation.

According to experimental results and comparison, InceptionV3 eventually achieved a good result with 21.2% top-1 and 56.6% top-5 error on the ILSVR 2012 classification with relatively less computational costs. (Szegedy et al., 2015)

2.3.4 DenseNet

An additional deep learning model based on CNN, DenseNet, was first introduced by Gao Huang at the CVPR conference in 2017. Adding a skip-connection that avoids the non-linear transformations, as was previously discussed, allows ResNet to outperform the conventional convolutional feed-forward network. DenseNet has a different connectivity pattern than ResNet's adding operation. It directly connects from any layer to all successive layers, allowing each layer to receive feature mappings from all its preceding layers. Transferring feature maps between layers will not be done by multiplying but by concatenating. As a result, the features from all prior layers will be kept as much as feasible, and their contributions to the final prediction will be equal. (Huang et al., 2018) The following figure shows the DenseNet model structure.

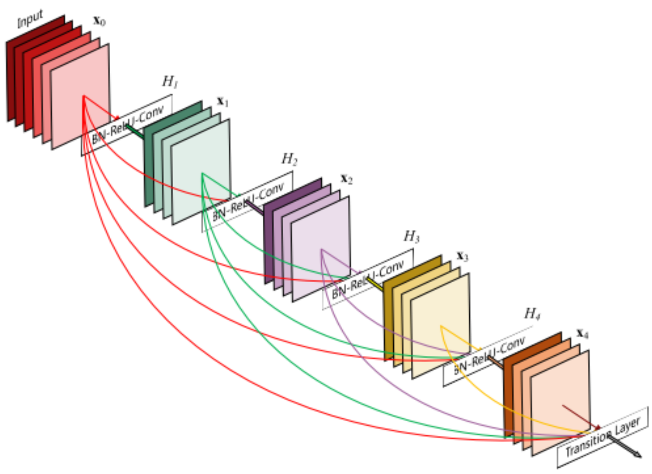

Figure 2.2 DenseNet model structure

Researchers have used DenseNet on several datasets, including CIFAR, SVHN, and ImageNet, in comparative tests with various networks. DenseNetBC with 190 layers achieves error rates on CIFAR10+ and CIFAR100+ that are much lower than wide ResNet design, at 3.46% and 17.18%, respectively. The research demonstrates that the deeper network will not increase error rates on test data, which signifies higher parameter efficiency and better generalisation capabilities. As a result, DenseNet does not have overfitting or optimisation issues and might be adept at coping with the uncertainty and randomness of daily life.

Recently, DenseNet has been used in the field of medicine. It was used by Yukun Bao and Najmul Hasan to train a DenseNet-121 architecture model for COVID-19 prediction and to recognise CT images. They evaluated it compared to other models and concluded that it performed computationally efficiently, taking an average of 195.35 seconds, and had higher accuracy overall (92%). (Hasan et al., 2021)

2.3.5 Mixed Link Network

Mixed Link Network was initially proposed by Wenhai Wang et al. in 2018. They first analysed the advantages and disadvantages of the existing ResNet and DenseNet and concluded that both have similar dense topologies and improvements. For example, ResNet will conduct additions multiple times on the same feature area, which will impede the transmission of information flow between layers; DenseNet has the problem of information redundancy caused by multiple concatenations of the same original features between different layers. Therefore, they proposed Mixed Link Network to absorb the advantages of both networks. This network structure mainly proposes two feature extraction methods: outer connection and inner connection. The inner connection draws on the superposition characteristics of ResNet, while the outer connection draws on the splicing characteristics of DenseNet. The following figure shows the main idea of MixNet.

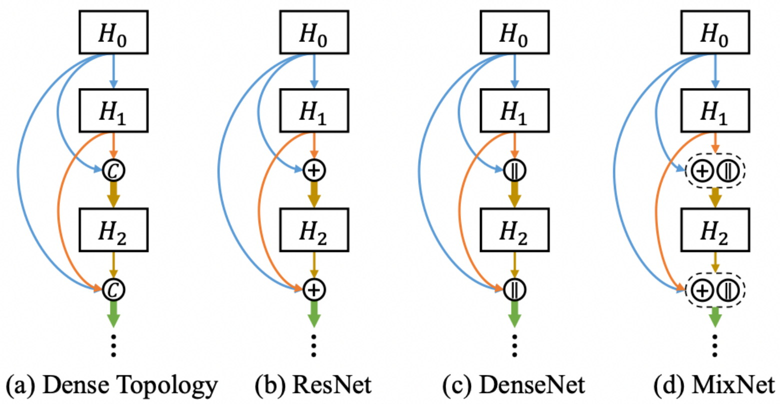

Figure 2.3 MixNet architecture

Wenhai Wang et al.'s research report compared Mixed Link Network with ResNet, DenseNet and other network structures. For the CIFAR-10 dataset, the error rate of MixNet-190 is only 3.13%, which is significantly lower than other models. In addition, MixNet-100 achieved similar results to other models with only 1.5M parameters in the SVHN dataset. (Wang et al., 2018)

2.3.6 Android Application with deep learning models

With the development of machine learning technology and the advancement of mobile phone operating systems, many applications have supported running machine learning models on mobile phones. For example, the built-in portrait recognition function in the album of the newly updated Apple IOS16 system allows users to select a character by long pressing it in a photo. Then, users can copy or share it with other applications. In addition, Apple's album can also identify objects in photos, such as pets, plants, buildings, etc., and obtain relevant information about the object through the Internet and return it to the user. Mobile phone applications equipped with deep learning models are of great significance to the convenience of people's lives. Due to the popularity of mobile devices, most people can obtain the information they want anytime and anywhere through AI-based applications. They do not need relevant background knowledge, significantly improving work efficiency and quality. The following figure shows the object detection function of IOS 16.

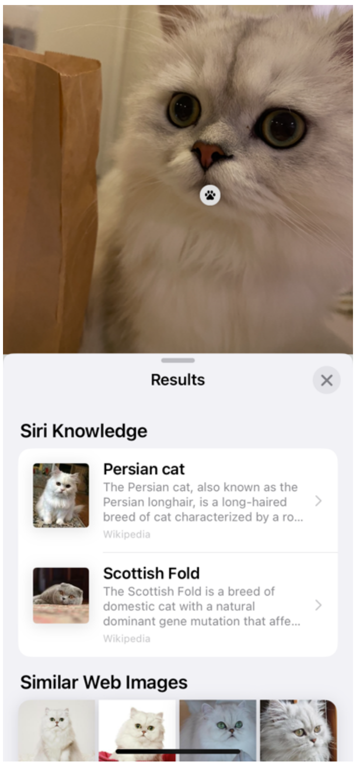

Figure 2.4 IOS 16 object detection

For the Android platform, TensorFlow Lite has launched a corresponding function package, which enables the Keras model trained on the computer to be converted into a model file in .tflite format by TensorFlow Converter to run successfully on the mobile phone. It has a vast application space: whether target recognition, text extraction or speech recognition, it can complete the task excellently. The following figure shows a demo project provided by TensorFlow's official website.

In terms of Android software development, Android Studio is a popular IDE for Android developers since it provides intuitive interfaces and utils. All external packages can be imported into the project and managed through the project lifecycle with a .xml configuration file, which is efficient and easy. Also, its integration with Git enables developers to conduct version control more conveniently.

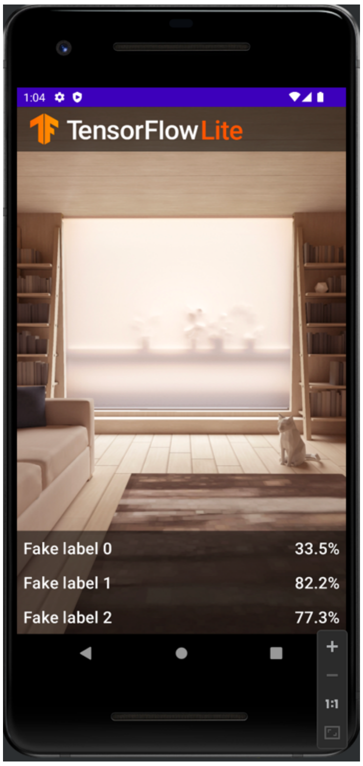

Figure 2.5 TensorFlow Lite demo project

2.4 Proposed Work

In terms of the model, this project will present a deep-learning model that can predict the test boxes' brand names included within the input image. When training, the model may run into issues with small data sets and an imbalanced amount of photos from different classes. The preparation step will also use data augmentation to expand the number of pictures by zooming, flipping, resizing, etc. Furthermore, when confronted with boxes that resemble test boxes but are not, the model may produce incorrect categorisation results. The model will be performed on physical devices running the Android operating system to have an offline recognition capability.

In terms of application, the project will develop Android software compatible with over 98% of the users. The application will be capable of using a camera and system album to upload pictures to feed deep learning models offline. Besides, it should display relevant detailed information according to predictions. The application should have fluent, intuitive, aesthetic interaction interfaces and stable performance to achieve a good user experience.

2.5 Conclusion

Deep learning is a suitable method for classifying images and has been successfully applied in numerous fields. However, there are still issues with model adaptation to data in specific domains in current research. More thorough and theoretical evaluations must be undertaken to choose the optimal model for brand recognition for RAT test boxes in this project. The model's performance can be further enhanced by merging various network models, fine-tuning, and data augmentation to meet the client's needs. Product quality will be ensured by learning from mature models.

Moreover, Android Studio is an ideal and standard tool for Android application development, and TensorFlow Lite is a mature tool for deep learning on mobile applications. After establishing the basic Android Studio project structure and obtaining a deep learning model with good performance, it can ensure compatibility between the model exported from the computer and the Android platform. It also provides a series of easy-to-use functions for developers to use during the development, which provided an excellent solution to the project's objectives and well met the client's needs.

3. PROJECT PROBLEMS

3.1 Project Aims & Objectives

The project aims to develop a way to make it simpler and more natural for consumers who want to purchase RAT test boxes to gain access to data they cannot get directly, like test boxes' sensitivity and government authentication, benefiting their decision-making in choosing the most suitable test kits. Consequently, the project will provide a user-friendly application that includes an offline deep learning model with high accuracy in test box brand domain recognition. Users can use the application to capture or upload the pictures containing RAT test kits they want to know further during the purchase. Afterwards, the application will return detailed information according to the images using built-in recognition models and databases.

3.2 Project Questions

The primary requirement is to train a deep learning model capable of image recognition with relatively high accuracy and create a mobile application for Android that can capture a picture of a RAT test box and notify the user if the brand is one that the Australian Government has certified, how sensitive it was, and what level it fell into. We encountered the following challenges while training the model and creating this application.

3.2.1 How to collect data on several test kit brands

A relatively completed and cleaned dataset is crucial before machine learning model training. Since the client did not provide any data for the project, the project should apply several data acquisition methods such as manual capture, web crawler, data purchase etc.

3.2.2 How to pre-process the data and choose the suitable model

After achieving a completed dataset, it is also significant to conduct pre-processing on the dataset. A clean and completed data set can speed up the model's training and make the model's performance more accurate. If the model has relatively high noise or corruption, the model’s performance may also be affected severely. For example, if the training data contains some pictures with moire patterns or tearing may lead to the model’s misclassification of a specific brand.

Moreover, model selection is an essential topic in the project since it is one of the most critical factors in success. Therefore, the project should initially design a baseline model for its tasks and iteratively improve its performance by tuning parameters or finding other solutions and absorbing their advantages. Comparative experiments with several performance identifiers should be conducted to choose the most suitable model better and make results evidential and persuasive.

3.2.3 Model file conversion for mobile devices

Since the application should be capable of recognising text box brands in an offline environment, trained model files should be exported in a format that the Android platform can load. Therefore, the project had to find a specific format and export tools to convert model files to a readable format for the Android platform. Moreover, the project needed to design interfaces connecting TensorFlow models and the application back-end.

3.2.4 Interaction logic of the App between the front-end and back-end

In the application development process, designing the front-end and back-end logic and ensuring the application's stable and smooth operation are very important for the user experience. Therefore, the project needs to choose a back-end design pattern suitable for the project scope and the jump logic between the front-end pages, determining the data transmission logic between the front-end and the back-end. Otherwise, the application may have potential risks, such as low efficiency, taking up much memory, and even crashes.

3.2.5 Testing deliverables and improving if any issues arise

After the demo is delivered at the initial stage of the project, it needs to be tested iteratively, such as unit testing, integration testing and system testing. The testing design needs to be comprehensive and accurate to ensure the high quality of the deliverables, avoiding ignoring potential bugs or problems caused by user misuse.

3.3 Project Scope

The project has been broken down into four main parts to develop mobile software that satisfies our client's needs more effectively and efficiently.

- Data collection and processing. A Python script crawler was used in the project's early phases, but it only produced a dataset with low equality. Members photographed RAT test boxes from pharmacies or stores and then manually cropped out any background that would impact the model's training.
- Model building. The model is designed to predict the test box brands when given user-uploaded photographs. TensorFlow will be used in the

project. Later revisions will presumably be made in light of the larger project dataset and the client's requirements.

- Application development. The project will be created using Android Studio to ensure compatibility with the Android platform. According to the users' input image, the deliverable will automatically crop the image or allow users to manually crop to ensure that background is removed as much as possible, then make predictions and give users detailed information.
- Testing and improvement. Various datasets and models will considerably impact the prediction accuracy. As a result, after the baseline model is created, it will be continually improved to more closely resemble the real-world scenario. The application will undergo repeated testing and correction to achieve effectiveness and stability.

4. METHODOLOGIES

Developing an Android application requires following a proven development process step by step. Overall, the project is divided into two main parts, mobile application development and classification model development. As shown in the figure, the flow chart shows the application construction process.

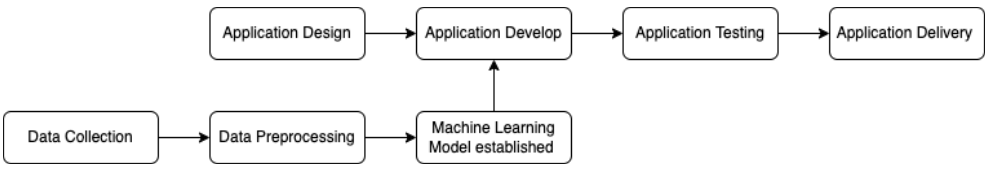

Figure 4.1 Project flow chart

In terms of building models, the first step is data collection, followed by different pre-processing techniques, which will be chosen based on the quality of the data set. Clean, standardised data will be passed to the model built by TensorFlow. In application construction, gathering and identifying client requirements will be the first step. In this phase, the team will output the plan of a minimal visible product and detailed functional expectations of the application. Then start the application development process. During the development stage, the application will be combined with a well-trained model from the model group. After that, the model-containing application will go through unit testing and iterations to collect customer feedback, improving the quality and the final delivery.

4.1 Model Analysis & Implementation

Designing a high-accuracy, small-volume, and rapid artificial intelligence model is the key to the success of this Android application. The researchers have carried out various pre-processing starting from data collection to improve the model's accuracy. After that, the researchers will evaluate various convolutional-based neural networks with different features or advantages and test them in the project's data set. Tunning the parameters until finding the optimal model. The following paragraphs will discuss how the team achieves the final results.

4.1.1 Data Collection & Preprocessing

The dataset's quality is determined by the size of the dataset, the noise, and the quality of the object being identified. Since users' motivation for the application is to find the sensitivity and accuracy of the RAT easily, the group simulates users' use habits, taking photos with their phone with the different possible environments of the RAT box. The project contains three different brands, including high sensitivity, medium sensitivity and low sensitivity.

Overall, the project obtained 150 photos for each brand, around 500 images. To enrich the diversity of samples, team members took images of boxes at different angles, lights, backgrounds, etc. At the same time, the team members used their hands to cover the boxes to a certain extent to simulate the noise in the actual state. The following figure shows some samples of data.

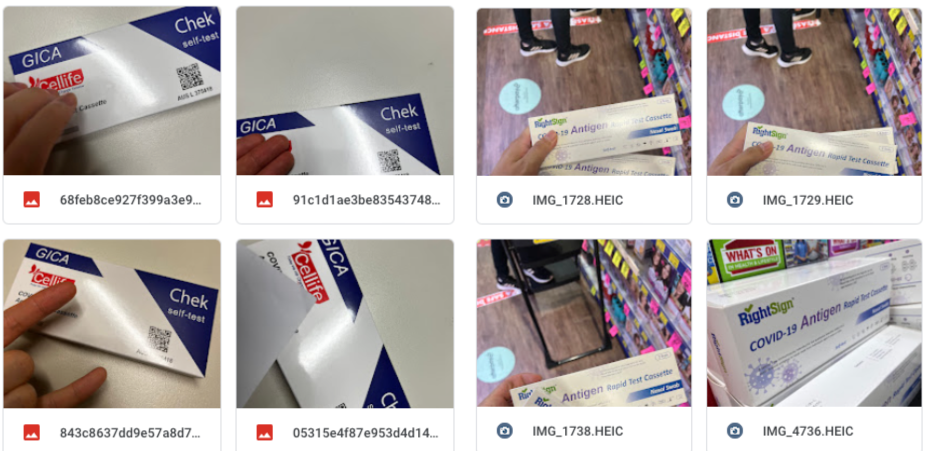

Figure 4.2 Examples of data containing noise (left) and with background (right)

In addition to shooting the images manually, members applied crawlers on the Internet to obtain related pictures. However, the results could have been better, as more information about the COVID-19 test box needs to be more relevant.

The following paragraph will discuss preprocessing strategies applied:

- **Remove images with bad quality:** After the members merged the data obtained by crawlers and manually took, the data would be cleaned. In the data cleaning process, members will go through all the images and manually remove data with massive noise, such as blurred images or images with too many irrelevant backgrounds.

Figure 4.3 Image with too much background

- **Manually crop the RAT boxes:** Since background noise still exists, such as the different coloured labels and products. The right amount of noise will help the model generalise better. However, too much noise will decrease learning ability and make it impossible for the model to identify the main subject that needs to be classified in the images. In order to reduce the learning cost of machine learning models, members manually cropped out the background and rotated the images to a proper level.

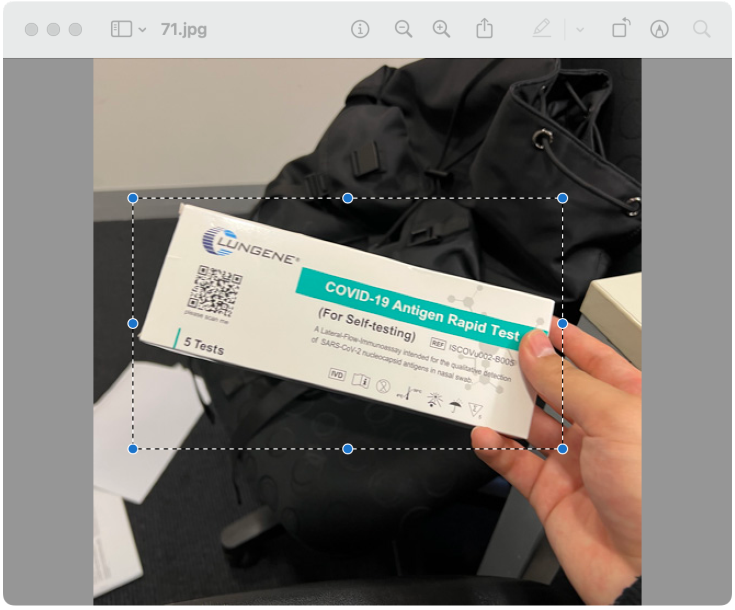

Figure 4.4 Manual crop

- **Resize images:** The cropped image has about 2 million pixels, and each image's size is different due to manual cropping. The number of pixels increases the requirement for computational resources and the training time significantly since the model needs to calculate the value of each pixel. Therefore, the three RGB colours are retained, and the image is resized to 300*300*3.

- **Normalisation:** Each pixel has a value between 0 and 255. Normalisation enables models to converge quickly, so the value of each pixel is divided by 255 and mapped within the range of 0 to 1.

- **Data augmentation:** Each image will have a random scale, shift, zoom in and out, or flip. Each image will be expanded to 5 or 20 images, depending on how many epochs there are in this project. Imagedatagenerator is an encapsulated package that will be applied to perform the augmentation. After augmentation, the size of the training set expands to at least 2,500 images.

The following figure shows preprocessing order strategies applied in the project.

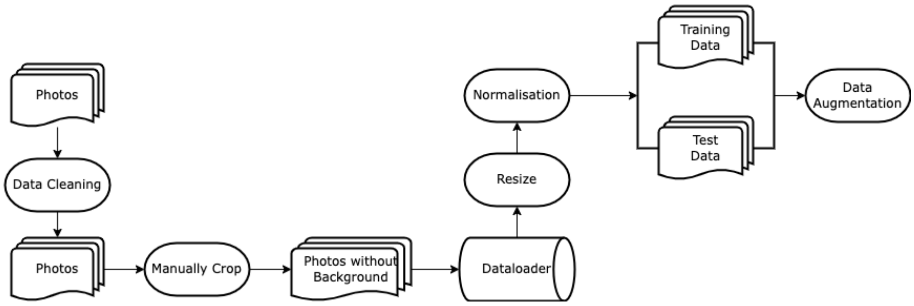

Figure 4.5 Data preprocessing strategy

4.1.2 Model Selection & Building

Transfer Learning Model: In the transfer learning model, pre-trained models such as ResNet, VGG, DenseNet, and InceptionV3 were tested. The models will be exported directly from the Keras package, selected and tested with different layers. All models will follow the typical sequences and structure. Moreover, the output from models will be connected with an average pooling, a dropout layer, a fully-connected layer and an activation layer. The sigmoid function will calculate the probability of each class and map them into the range of 0 to 1. The final prediction will be applied with a threshold. The threshold will remove the prediction with low confidence. The application will ask the user to upload a new image.

The following figure shows the transfer learning model architecture.

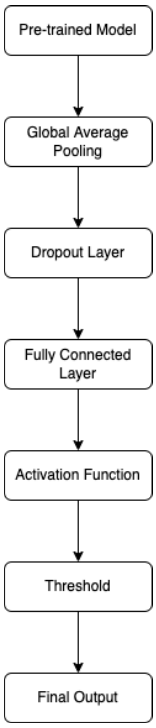

Figure 4.6 Transfer learning architecture

MixNet: The MixNet model combines image features and words on the image. In the model, the group applied DenseNet121 to perform feature extraction of images and used a pre-trained text recognition model to extract the word on the box. In the pre-processing step, all words in the training set are counted for frequency by the model. Since misspellings may occur but have a low probability of occurring, the team chooses words that appear more than once to compile a dictionary. Based on the dictionary, a computer-readable numeric label is generated for each word. Rather than applying the words to vectors technique, which converts the words to meaningful vectors, the team will apply one hot method, which converts each word to a unique figure. In the training process, the model will obtain the feature from DenseNet, extract the words on the image, convert the words to vectors based on the dictionary and concatenate the word feature with image features.

The following figure shows the MixNet model architecture.

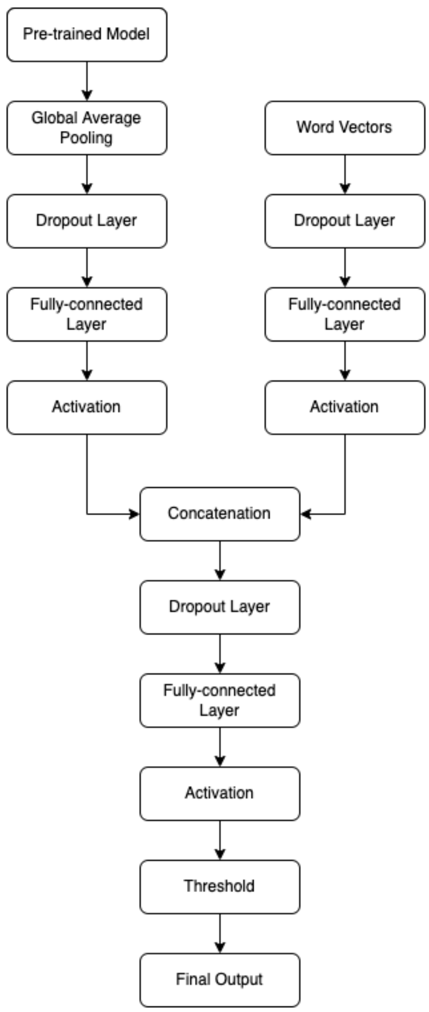

Figure 4.7 MixNet architecture

4.1.3 Model Evaluation & Tuning

This project's performance indicators will be f1 score, accuracy and precision. The hyperparameters, such as learning rate, epochs, batch size, and the number of layers, will be tested with different combinations until obtaining the optimal performance.

4.2 Mobile Application

4.2.1 Front-end Page Design & Jump Logic

In Android Studio, the design and implementation of the front-end page are similar to that of a general web application. It uses XML files of HTML-like language to design the page content layout, and at the same time, it performs personalised art design on the components of the page, such as colour, background, size, animation effect and so on. In this project, all pages are stored in a container with a back button and a menu bar, which ensures the overall consistency of the style of all pages. In addition, all pages use a linear layout, which ensures that the program can adaptively scale and scale the page's content according to the resolution of the user's mobile phone. The colour tone of the page is cyan, and it can automatically switch between night and day themes according to the time of the user's mobile phone. All components adopt a flat design, ensuring aesthetics and making the page concise.

This project uses the Navigation component of Android Jetpack for the jump logic between pages, which avoids boilerplate code and simplifies the development process. The component needs to define all the pages included in the jump logic in the XML file and the jump relationships' names between them. After the jump relationship is established, the back-end code can call the relationship name directly to realise the page jump. The following figure shows the visualisation of the Navigation Graph.

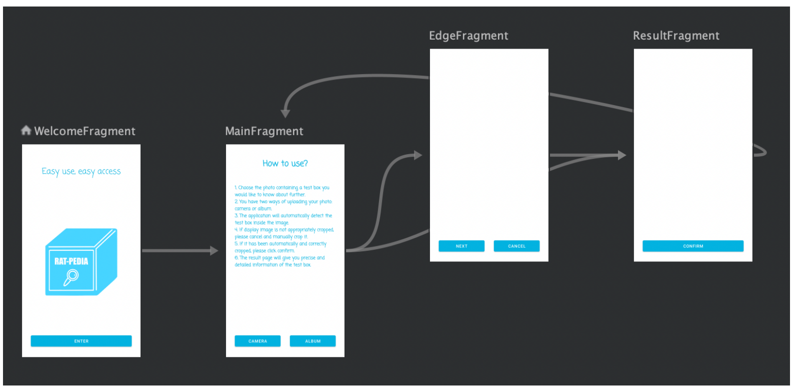

Figure 4.8 Navigation Graph

As shown in the figure, the application entrance is a welcome page with the LOGO and slogan of the application on it. The LOGO was designed and exported through Procreate on iPad. After the user clicks the enter button below, it will jump to the application's main interface, including the instruction and two buttons: camera and photo album. After the user decides which method to upload the image, the application will jump to the system application of the corresponding method, performing edge recognition on the image returned by the application and then presenting the processing result on the cropping page. If the picture on the cropping page correctly frames the test box part in the picture uploaded by the user, the user can select the confirmation button and make a prediction. The user can manually crop the image if the automatic cropping result is unsatisfactory. After the prediction, the page will jump to the result page. The result page will display the predicted picture and related results to the user. After obtaining the information, the user can click the confirm button to return to the main page.

4.2.2 Back-end Modules Design

The project structure is based on the single-activity with multi-fragment development idea recommended in the official Google Android development document, which is light and suitable for small projects. It also calls multiple system applications such as camera, photo album and clipping functions. First, in realising front-end and back-end data interaction, the project uses the data binding function provided by Android Studio. It can register all components in the front-end page into a binding container. When the backend needs to pass values to the front-end to change its display content dynamically, the back-end code can directly refer to the front-end component ID registered in the binding container. This not only optimises the readability of the code but also prevents the back-end from repeatedly searching for components through the findById method. The traditional findById method needs to consume much memory when there are many application components, which is detrimental to the fluency of the application and the user experience. To control the application's access to files in the user's mobile phone, the project adopts the FileProvider function released in the Android API 21 version. It can uniformly create a cache folder inside the application and manage the cache files integrally.

The project first determined the basic logic flow chart in the program design stage. This figure clearly defines each program module's life cycle and calling relationship. The modules required for this project can be roughly divided into the following modules:

Camera module: This module is mainly responsible for successfully invoking the system camera function and creating a new file in the cache folder of the application when the user chooses to use the camera function to upload pictures, and finally saving the photos taken by the camera into the file. This module will return the image's uniform resource identifier (URI).

Album module: This module is mainly responsible for successfully calling the system album function when the user chooses to use the album to upload pictures and copying the album's pictures to the application's cache folder to unify the access rights. This module needs to return the URI of the image selected by the user.

Automatic crop module: This module is mainly responsible for identifying the edge of the image after the user successfully uploads it, then returns the largest rectangular edge in the image and automatically crops it based on the coordinates. This module needs to return the automatically cropped image.

Crop module: This module is mainly responsible for giving users the channel to crop after the result of the automatic cropping function manually is not ideal. It will call the image cropping function of the system and return the cropped image after the user crops.

Prediction module: This module is mainly responsible for inputting the final processed image to the built-in TensorFlow Lite model of the application and converting the calculation result into the corresponding brand name. At the same time, the module also needs to look up the corresponding detailed information in the built-in database through the brand name and present all the information to the user in a package.

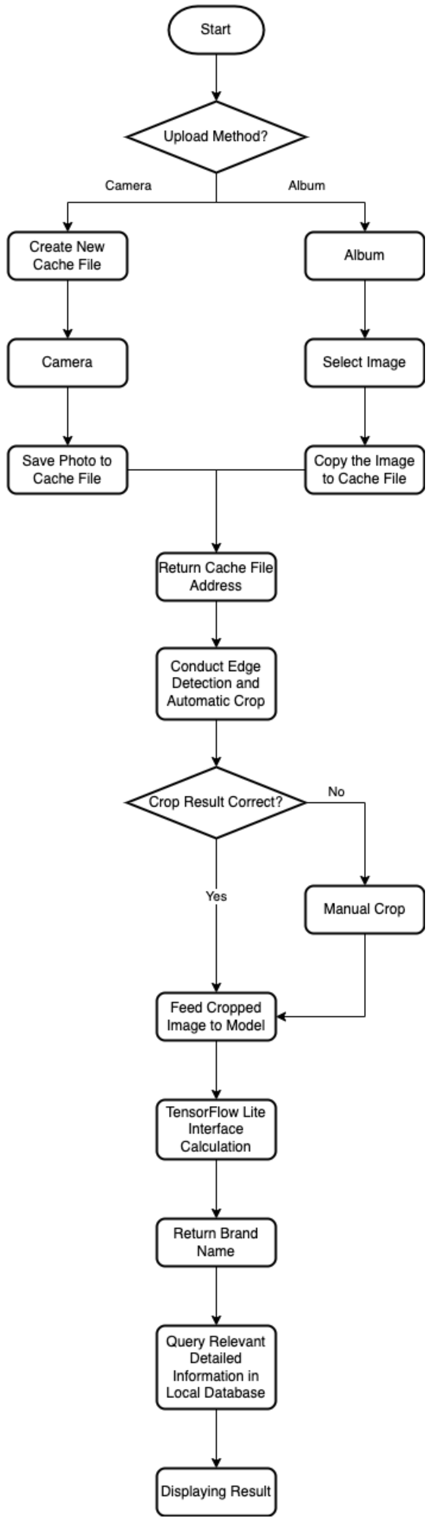

Figure 4.9 Program logic

4.2.3 Application Testing

In order to ensure the quality of deliverables and customer satisfaction, four testing methods have been adopted in this project: unit testing, integration testing, system testing and acceptance testing. The following will mainly introduce the unit test content for each functional module in the application.

Camera module: For the camera module, the test mainly simulates the user's behaviour after entering the camera, such as taking pictures, confirming, cancelling, etc. By printing the application log, it can be found that the camera module can successfully generate a new file in the cache folder and store the photos taken by the camera in the new file.

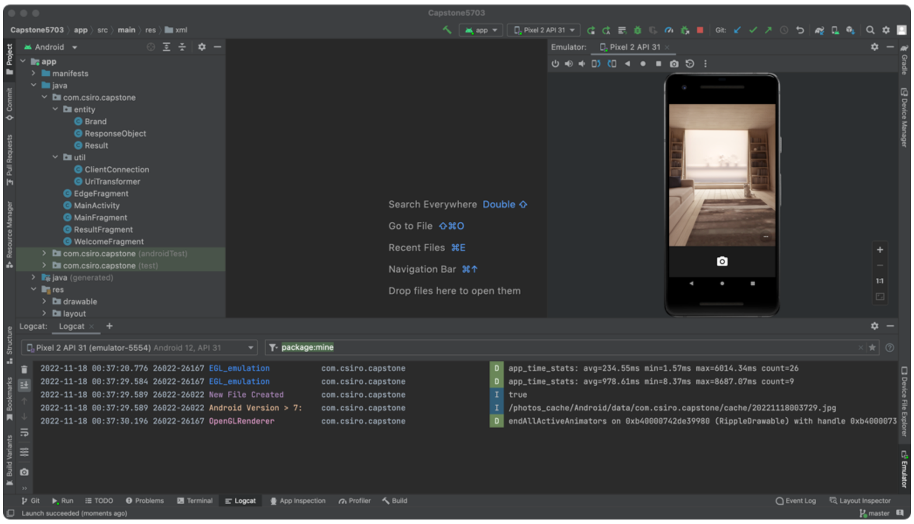

Figure 4.10 Camera module test

Album module: For the album module, the test mainly simulates the user's behaviour after entering the album, such as selecting a photo, confirming, returning, etc. By printing the application log, it can be found that the photo album module can successfully copy the pictures in the system photo album to the cache folder.

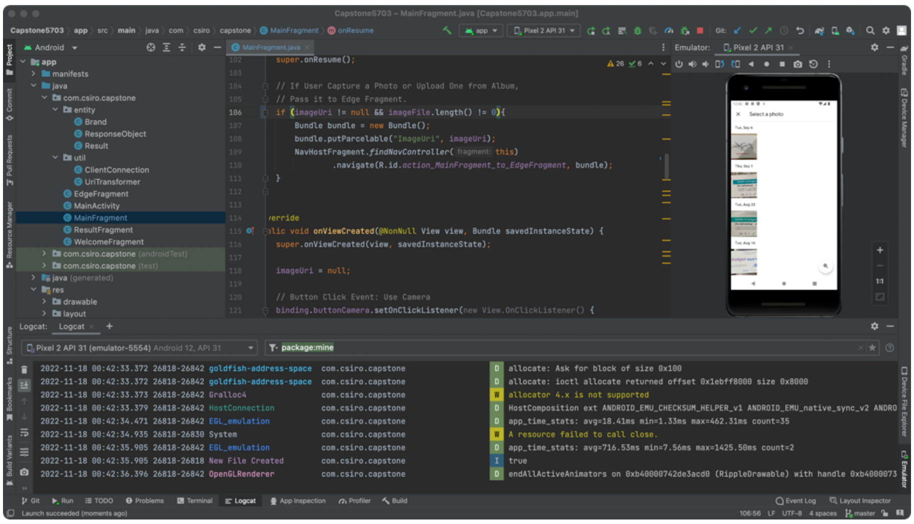

Figure 4.11 Album module test

Automatic crop module: For the automatic cropping module, the test mainly simulates the user's behaviour after the uploaded image is successful, such as confirming the automatic cropping result and cancelling it. By printing the application log, it can be found that the automatic cropping module can identify the largest rectangular edge for each picture and return the processed picture.

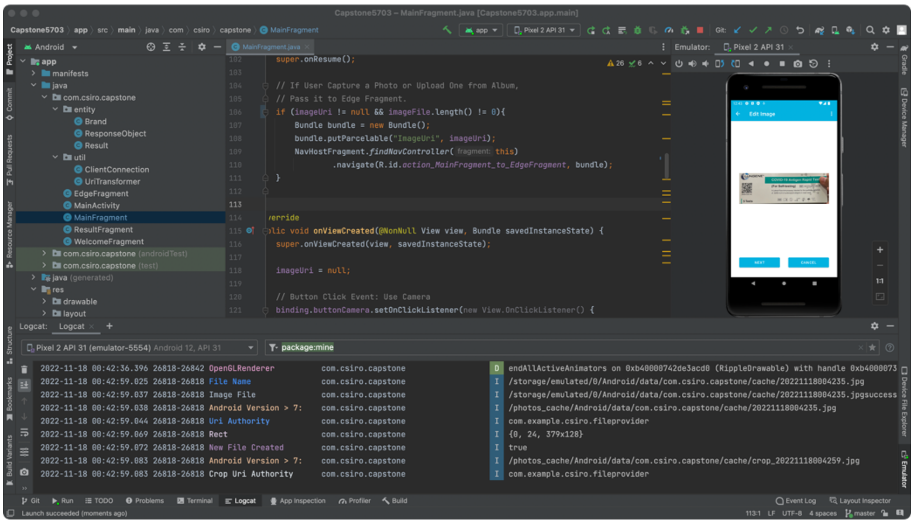

Figure 4.12 Automatic crop module test

Crop module: For the cropping module, the test mainly simulates the behaviours that may occur when the user enters the manual cropping function when he is not satisfied with the automatically cropped picture, such as zooming in and out of the picture frame, rotating the picture, etc. By printing the application log, it can be found that the module can successfully save the manually cropped pictures.

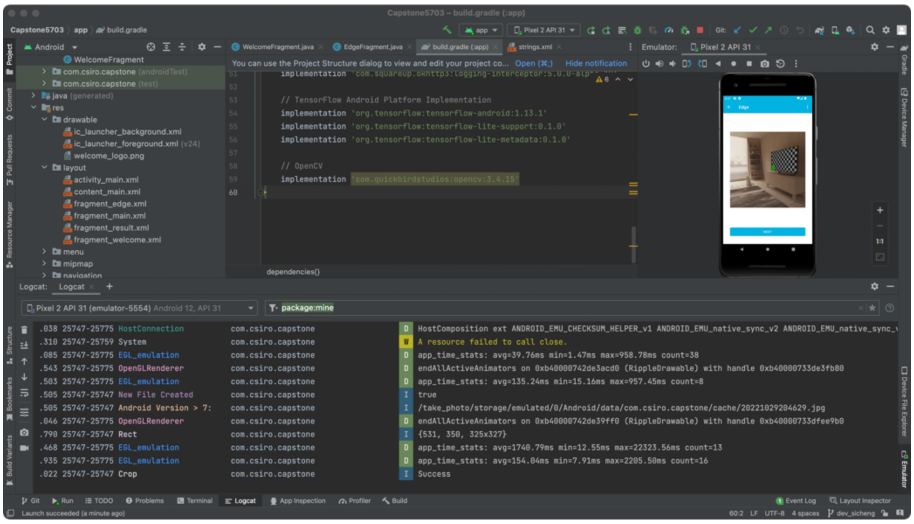

Figure 4.13 Crop module test

Prediction module: For the prediction module, the test mainly simulates the possible behaviours of the user when the prediction result is obtained, such as confirmation, return, etc. By printing the application log, it can be found that the module can successfully call the deep learning model and fully display the prediction results to the user.

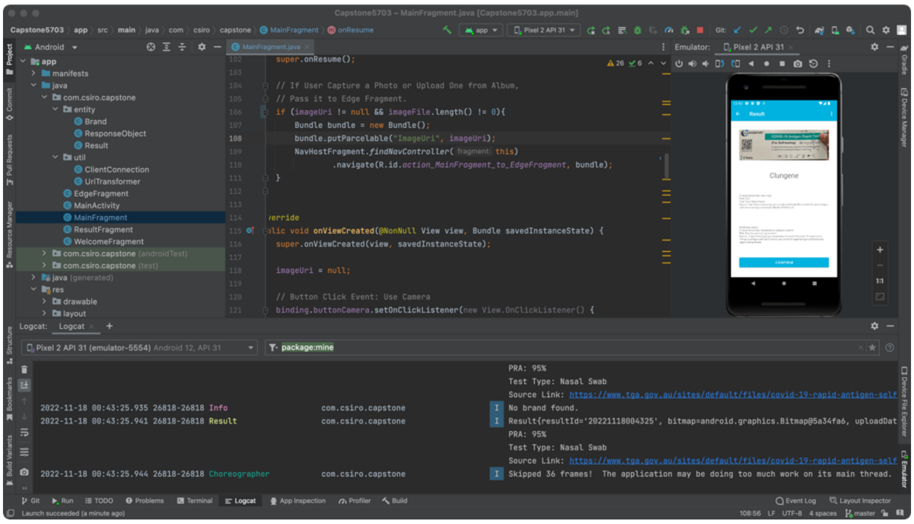

Figure 4.14 Prediction module test

5. RESOURCES

5.1 Hardware & Software

The project's initial primary deliverable expectation was an Android application that could be used with mobile devices running Android versions higher than 5.1 (API 21). The app was created using Android Studio Chipmunk 2021.2.1 Patch 1 and tested on a Pixel XL-model Android virtual device (API 31) and a Xiaomi 8 physical device (API 28). Additionally, the development was carried out on a Macbook Pro equipped with an M1 Pro (ARM) CPU and a Surface Pro equipped with an i5-1135G7 chip (x86).

Group members first used iPhone 13 Pro Max cameras to take pictures of RAT test boxes during data gathering, and all information was then transferred to Google Drive. During system analysis and design, the project used Visio and Draw.io to create diagrams such as UML class diagrams and system flow charts. During development, the project was pushed to the GitHub repository for version control.

In addition, the project utilised Procreate on iPad with IOS 15 and an Apple Pencil 1st generation for UI design and refinement. All feed images were also manually cropped through Apple Preview and Adobe Photoshop.

Moreover, model training was mainly processed on a Macbook Pro with M1 Pro GPU and a Macbook Pro with M1 GPU. Since some external packages are not supported on the ARM architecture, the project also used Google Colab's Tesla T4 with 16 GB GDDR6 memory for accelerating workload.

Finally, project scheduling and progress status monitoring were conducted through JIRA and Monday.com. Group members used Slack to connect with clients and the tutor. Document organisation was performed through Google Drive, Google Docs, Microsoft Office Word, Excel and PowerPoint. The project presentation video was recorded through Zoom and trimmed by Final Cut Pro.

5.2 Materials

To improve the model performance, the project may need extra data (i.e. photos of RAT test boxes from different brands) from clients in the future so that the model will better meet their requirements.

Virtual machine learning hardware membership, such as Google Colab Pro, may also be needed for higher model training efficiency.

Moreover, the project applied several open-source external packages for efficient and convenient development. During model design and training, the project used Numpy, OpenCV, Keras, EasyOCR, TensorFlow and TFLite for data loading, cleaning, preprocessing, model building training and exporting. In terms of Android application development, the project used TensorFlow Lite Android and QuickBirdStudios OpenCV implementations for creating interfaces to read .tflite files and conduct edge detection.

5.3 Roles & Responsibilities

The group's members are split into two subgroups due to the project's characteristics: the model design and implementation group and the application development group. Each participant will be given a group assignment and be in charge of a particular region.

- Sicheng Guo is the project leader.

1. Leader: Project scheduling and group coordination; task breakdown and assignment; technology stack determination; monitoring progress status and project performance; reporting project progress status and proposing improvement solutions with client and tutor; obtaining feedback, recommendations and requirements.
2. Application Developer: Project structure and back-end logic design; interface design, implementation and testing; development knowledge introduction; version control.
3. Model Designer: Conduct comparative experiments among InceptionV3, ResNet, DenseNet and MixNet; write codes for Dataloader, Word Vectorisation, MixNet model structure and training and TFLite export function.

- Yiming Nie belongs to the model group.

1. Model Designer: Data collection and preprocessing; ResNet, VGG architecture building and parameter tuning; cooperation with the application team to ensure the final product's availability and stability; Design model performance evaluation metrics; Generate model results and performance evaluation visualisation diagrams.

- Rencong Wang belongs to the model group.

1. Model Designer: Data collection and preprocessing; OCR technology implementation, evaluation and testing.
2. Diagram Maker: Drawing diagrams for data collection and preprocessing illustration benefiting report comprehension.

- Linxin Sun belongs to the application group.

1. Application Developer: Write back-end code for the product, such as interface calling; values transference of the application.
2. Management Assistant: Arrange requirement documents; Apply agile development to assist project management by Jira; promptly update tasks and propose follow-up plans.

- Wenyan Hu belongs to the application group.

1. Data Collector and Cleaner: Collection and classification of images from three brands of RAT boxes; sorting them into usable and non-usable images; pre-processing image data for better model training.
2. Management Assistant: Apply agile development to assist project management by Jira; promptly update tasks and propose follow-up plans. Record and summarise weekly meetings; upload meeting minutes on Slack or Google Drive.
3. Application Developer: System flow and front-end UI design; database creation, including detailed information about each test box brand; user instruction writing.

6. MILESTONES / SCHEDULE

Milestone	Tasks	Reporting	Date
Week-1	Group formation and member introduction	None	03-08-2022
Week-2	Roles and responsibilities assignment, group division, task assignment	Meeting with client to report task allocation and obtain requirements	12-08-2022
Week-3	Project scope definition, requirements document generation	None	17-08-2022
Week-4	Foundational knowledge acquisition and practice training, machine learning literature review	None	26-08-2022
Week-5	Proposal Report Due, project technology stack familiarity, project structure and logic design, future plan and risk management	Proposal Report	31-08-2022
Week-6	Data collection & preprocessing, baseline model building	None	07-09-2022
Week-7	Model improvement background literature review, Android application design	Meeting with client to report application design and model performance	14-09-2022
Week-8	Communicate with client to check project progress and request for recommendations, conduct experiments on improved models, Android application implementation	Mid check with client to report progress status	21-09-2022
Week-9	Progress Report Due, improvements solution check, conduct comparison with previous model	Progress Report	05-10-2022

Week-10	Model export to the application, conduct unit tesing for each modules	Unit testing report	12-10-2022
Week-11	Conduct system testing, fix revealed bugs	System testing report	19-10-2022
Week-12	Final Presentation	Final Presentation	26-10-2022
Week-13	Final Report	Final Report	02-11-2022

7. RESULTS

7.1 Deep Learning Model

All models' convergence speeds are acceptable regarding the training set's accuracy. The accuracy increases steadily to above 90% after the third epoch, which means the model can effectively identify features or different classes of apparent box features.

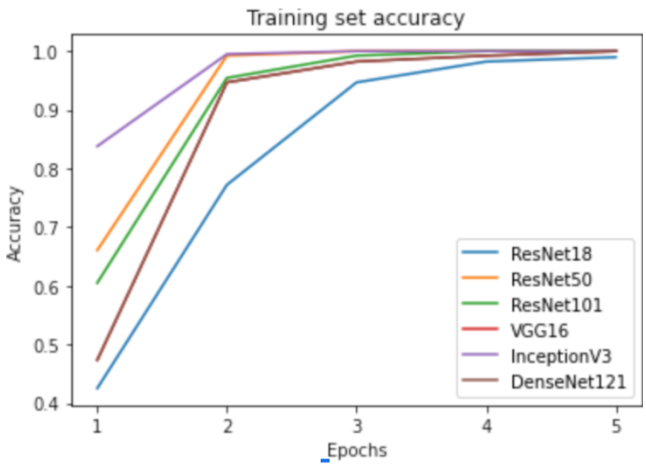

Figure 7.1 Training accuracy

All models have an accuracy of almost 100% and a high confidence value for any prediction. In addition to accuracy, precision is one of the primary measures. Precision measures how accurate the model is for each prediction. For instance, precision indicates the probability of correct prediction for a given class. Regarding APP performance, the group wanted the model to have high accuracy for brand predictions and not provide results for uncertain or low confidence predictions to prevent reducing APP user satisfaction due to incorrect predictions.

Model (Number of layers)	F1 score	Precision
ResNet 18	0.95~0.98	0.96~0.98
ResNet 50	0.94~0.98	0.95~0.97
ResNet 101	0.92~0.98	0.93~0.96
VGG 16	0.97~0.99	0.97~0.99
InceptionV3	0.98~0.99	0.98~0.99
DensetNet 121	0.98~0.99	0.98~0.99
MixNet(OCR+DensetNet169)	0.98~0.99	0.98~0.99

Figure 7.2 Comparison evaluation

The ResNet model's prediction errors occurred mainly between Clungene and GICA. The model incorrectly classifies Clungene and GICA, but after several training epochs, the model's accuracy rises to 100%. False predictions are mainly due to added noise, such as blocking part of the image with a hand, data enhancement removing image features, etc., which prevents the model from capturing enough features.

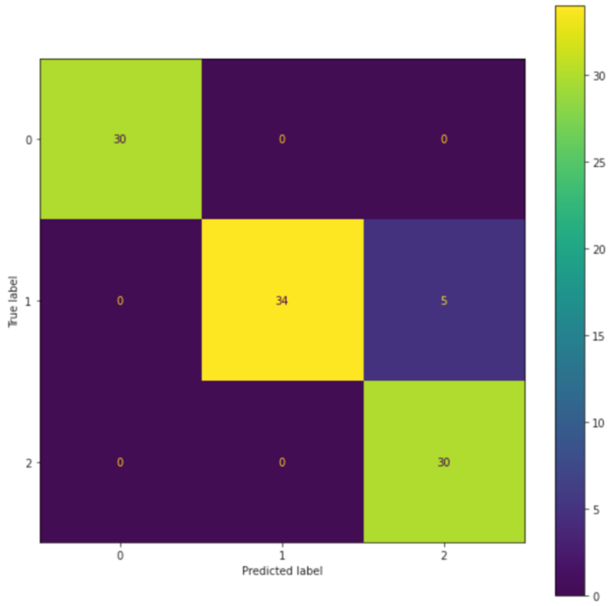

Figure 7.3 Confusion Matrix for ResNet

All images in the test set were pre-processed, clear, noise-free and complete, without obscuring any features on the images. As a result, the model performed well and achieved 100% accuracy.

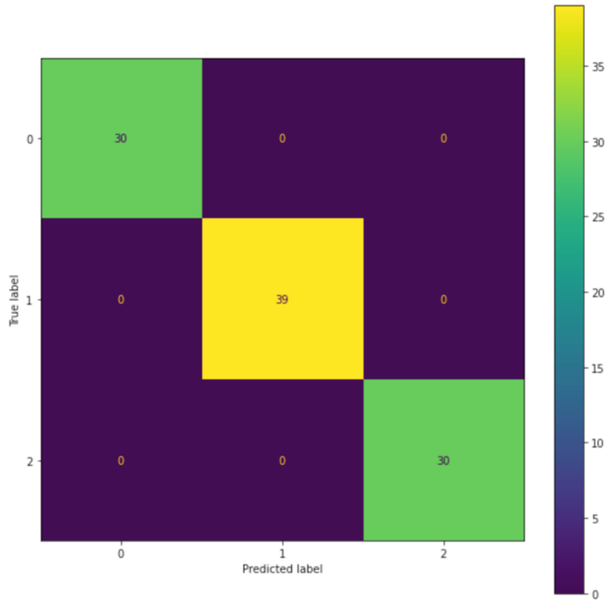

Figure 7.4 Confusion Matrix for InceptionV3, VGG, DenseNet, and MixNet

The following table shows achieved each model's best parameters. Considering the performance of each model's architecture (F1 score), the project determined to use MixNet as the final model.

Best Hyperparameters			
Model	Learning Rate	Batch Size	Epochs
ResNet	0.001	40	14
VGG	0.001	40	16
InceptionV3	0.0001	40	45
DenseNet	0.0001	40	23
MixNet	0.0001	40	26

Figure 7.5 Best set of hyperparameters

7.2 Android Application

The project finally delivered an Android application that has gone through unit testing, integration testing and system testing and can run smoothly on virtual and physical devices. The corresponding Android target version is API 31, and the minimum supported version is API 26. It fulfils all the requirements put forward by customers and has good performance and aesthetics. The following will introduce each interface and function of the final version of the application.

Welcome Page: The welcome page is mainly composed of the application LOGO, slogan and entry button. When the user opens the app, this page will first appear in view. When designing the page layout, the constituent elements are reduced as much as possible to reduce material loading and increase user experience without losing the unique style of the application.

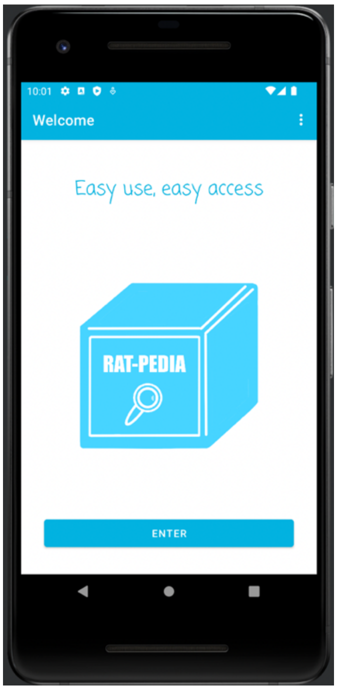

Figure 7.6 Welcome page

Main Page: The main page contains three components: the user guide, camera, and album button. The user guide tries to explain the usage of the application as simply and clearly as possible so that users can use the application correctly. The two buttons below allow users to use the camera or album to upload the photos they want to predict.

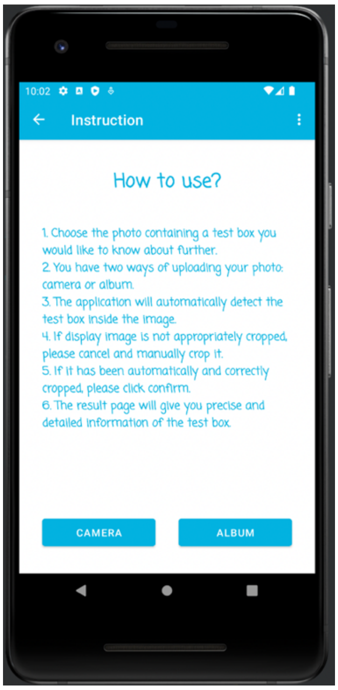

Figure 7.7 Main page

Camera Function: The camera function is provided by the Android system. Users can use this function to take pictures of the test box in the natural environment and save it to the phone storage. Due to different systems, the camera interface may also be different.

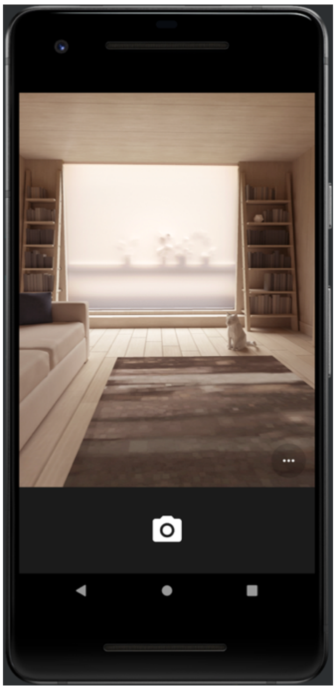

Figure 7.8 Camera

Album Function: The photo album function is also provided by the Android system. Users can select existing photos in the system album and submit them to the program for prediction.

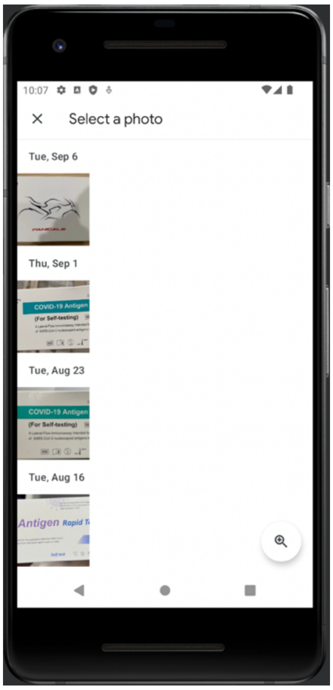

Figure 7.9 Album

Edit Image: The image editing function includes the automatic and manual cropping functions based on the edge recognition algorithm of OpenCV. After the user uploads the picture, the application will crop the object similar to the test box in the picture and present the automatic cropping result on the picture editing page. If the user is unsatisfied with the result, it can be cropped manually, provided by the Android system.

Figure 7.10 Edit image

Prediction Page: The prediction page presents the results calculated and automatically packaged based on the images passed in by the user. It contains a picture box, detailed description text, and confirm button. Through this page, the user can intuitively obtain relevant information about the test box in the picture.

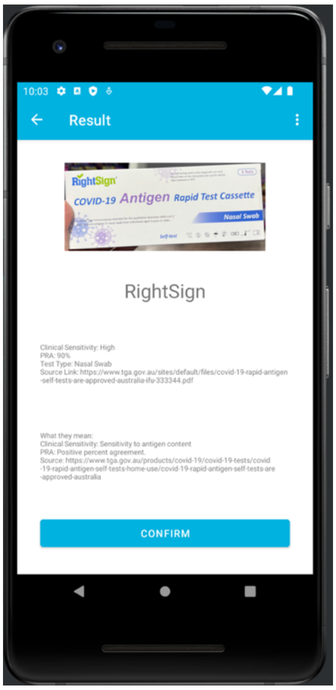

Figure 7.11 Prediction Page

8. DISCUSSION

In order to fulfil the project requirements set forth by the client and to accomplish the project's delivery goal of providing an Android application that could identify three brands of test boxes. The project milestone plan assigned by the project team can be divided into two parts: the model group and the application group. This section will be developed from the above two parts, detailing the approach and required resources and knowledge for the different parts of this project, discussing in detail the significant impact of these approaches in the project, and showing the presentation and discussion of the final deliverables for each part. This culminates in a discussion of the role of the methodologies used in the above project in the areas covered.

8.1 Model

When building the machine learning model needed for the project, the plan is to divide this module again into data processing, writing machine learning model algorithms, comparing each algorithm's training results and selecting the algorithm with the highest correct rate to be applied to the project.

First of all, in the data processing phase, since the client did not provide data before the project started, all the data in this project were collected by manually shooting COVID-19 detection boxes by comparing the two methods of crawler technology and manual data collection. Specifically, the data collected by the above method in this project provided data diversity through different shooting angles of the user, the distance between the user and the detection box, and the different luminance of the user's shooting environment, which also helped prevent overfitting of the subsequent machine learning algorithm.

Secondly, in the cleaning and processing phase of the data, the noise distribution of the data was reduced as much as possible because of the efficiency of the subsequent model learning. In order to achieve this goal, the image format was standardised by manually cropping the images. In addition, because of the small amount of raw data collected in this project, data enhancement techniques were used to further improve the accuracy of the subsequent machine learning, specifically by rotating, flipping, and scaling the images. The final data collection for this project resulted in 750 RAT test boxes of three brands certified by the Australian government.

Afterwards, the project requirements were analysed when locating the machine learning method. The convolutional model for supervised classification was used for transfer learning in the primary phase of the project, specifically ResNet, VGG and InceptionV3 machine learning algorithms. After the construction of the above three convolutional models is completed, it can be obtained that the three algorithms perform equally for this project. However, having a good generalisation ability in the dataset is challenging. The reason for analysing the results of these three algorithms is that the contour of the selected RAT test box is very similar. So the project plans to improve the machine learning model further. After conducting comparative experiments of transfer learning and fine-tuning, members found that whether or not the parameters of the pre-training model were frozen had no significant effect on performance, which may attribute to CNN's concentration on the object's contour and ignorance of higher-level features such as text. Through the observation of different test boxes, it is found that the text font on the box symbolising different brands is the easiest mark to distinguish different test boxes. Hence, the project will apply the text recognition model, specifically the OCR model. This model is a word extraction model, which can achieve the purpose of automatically extracting the text on the image. Precisely for this project, after extracting the text on the box, the text will be extracted into a word vector that can be processed by the computer and then processed in different word classifiers. Using this model in the project data, it was concluded that it dramatically improves the model's generalisation ability and increases the correct rate of recognising the box brand.

In the MixNet model, the team chose a one-hot method to compile the word into vectors, which has the following consideration. The usual word-to-vector method will output vectors that are related to each other. In other words, the similarity or relation between words can be calculated. In the classification, the relationship between words does not affect the model's prediction, and the brand of the box in the image can be determined by extracting the brand name, e.g. 'GICA'. With its smaller size and more direct understanding, the one-hot approach is the ideal solution for this project.

In terms of the number of layers of each model, more layers allow the model to handle more complex classification tasks. However, more parameters increase the computational effort and the possibility of overfitting. In practice, the most intuitive result is that InceptionV3 takes the longest time to train and achieve convergence, and its size is larger than DenseNet with the same number of layers. VGG also consumes more GPU resources than ResNet18 due to its three fully-connected layer structures.

8.2 Application

When developing the application, the plan was to subdivide this module into defining the application requirements, front-end interface, requirements implementation and integration with the machine learning model, completing the demo running on a physical Android device and finally initialising the server project.

In the first part of the application requirements definition, a comparison with the client and the initial project plan finally identified two main requirements to be completed for this project cycle, specifically to take photos using the camera function, save them in an album, and select photos from the album for prediction. These two main requirements will be tested throughout the project.

Then the front-end interface was written to match the two main requirements, using a minimalist blue and white colour scheme to complete the front-end interface of the project.

For the requirements implementation phase, for the two requirements of this project, the overall Android framework is required to achieve page jumping and conversion between different requirements, especially in the first requirement to achieve the camera function. For the page-jumping architecture, the navigation component method of the Jetpack will be used to implement it. The navigation component follows a single activity structure on the Android platform, allowing the application to run faster and more efficiently.

Once the application's front-end and back-end logic code are completed, it must be combined with the trained machine learning model. In particular, for the machine learning model, the entire model is trained based on the TensorFlow model. However, for the Android platform required for this project, TensorFlow is not supported, and with this in mind, the project will use the TensorFlow interface to solve such a problem.

Specifically, we will use a channel to convert the TensorFlow model to TensorFlow lite in the project, which means after the model training is completed, the model file will be converted to TensorflowLite format, and then the trained model can be used on the Android platform through the TensorFlow Lite interface to complete the combination of the application and the model.

Through the previous project plan, it can be run very successfully on real Android devices For server project initialisation to achieve the purpose of unifying model files and future management of customer application updates.

At the final customer delivery, the customer verifies the project completion by finding any photo of the test box on the internet and transferring it to this product, with the final result that the name of the test box can be successfully displayed.

The above are all the project results achieved in this project, and it can be seen that all the initial requirements proposed by the customer were completed and confirmed by the customer. The project's final deliverable is an application that can run on Android phones to identify the three popular COVID-19 test kits, which will be commercially viable for the client and the Australian government to manage the spread of the new crown epidemic in the country. Such a mobile application provides users with a convenient and efficient way to access the details of the test kits in their hands anytime, anywhere via their Android phones, thus providing great help to their health situation. For example, some contract workers need more accurate test kits, so they do not have to risk losing their jobs due to false positive results from low-sensitivity kits. For example, hospitals and clinics need to provide high-sensitivity kits to their patients because they do not want false-negative patients to transfer COVID-19 to vulnerable populations, thus spreading the new coronavirus faster and more widely.

It is worth mentioning that for some particular groups of people that most products on the market do not take into account, such as blind users, it is not convenient to directly view or access the details of the test box through the Internet information. For users in environments with poor cell phone signal quality, it is straightforward to check whether the test kits are approved by the Australian government and the medical information needed without having to access complex and varied results via the Internet.

9. LIMITATIONS AND FUTURE WORKS

9.1 Limitations

As the project progressed, although it was completed, there were still some limitations in the project's progress.

When talking about machine learning, the Dataset is anyways the first. Unfortunately, the group did not have any data at the beginning of the project. The team found some data on the internet through web crawling techniques, but there were few relevant data sources on the web, so the data had to be collected manually by taking pictures. When taking the pictures, it was also necessary to consider the realism of the data, simulating the natural environment in which the user would use the application, such as in a pharmacy, clinic and other different scenarios, so the pictures needed to include different camera angles, distances, brightness etc.

However, when applying this data to the model, the training results of the model are not satisfactory because there is too much noise in the actual situation picture data. So in order to get a more accurate result, the data needs to be cleaned.

As these images have different angles, distances, etc., it is difficult to process them uniformly, so the group chose to do the cropping manually to reduce the noise and standardise the format. This was a significant amount of work.

After cropping, some of the original images may not have been usable due to high noise levels, so the data needed to be re-collected. The final size of the dataset for the project was determined to be around 150 images for each of the three brands of RAT test boxes. However, this dataset was insufficient for machine learning to get a perfect result.

For Mixnet, there will likely be some spelling errors when performing the word extraction. Correcting the spelling is essential, and lack of mature practice. Current open-source solutions are limited by their out-of-date word bank. For example, COVID, RightSign cannot be found in the word bank. Similar situations happen to the bank of words to vectors as well. Thus, the current solution would be one hot coding or training a vector model with our current words.

In addition to using more data to build a complete model. In the future, adding more functions to assist the disabled or elder in the application is possible. Imagine that after the disabled open the camera, they can quickly know the product information in their hand, which will be a perfect product for such people.

9.2 Future Works

In future work, the project team will further optimise the project. Firstly, the expansion of the dataset and the acquisition of data support, both manual and non-manual, is a high priority, as an expanded dataset will not only facilitate the training of the model.

Furthermore, implementing a more target group-friendly design will significantly improve user satisfaction with AI-based software. For example, introducing a voice broadcast feature that uses voice announcements to guide the use of the software for people who are blind or dyslexic and to broadcast recognition results.

The accuracy of the model for word extraction can still be improved. The model was trained from an untimely updated training set. Therefore some of the keywords, such as COVID and GICA, may have spelling errors, leading to the model not providing helpful word vectors.

Finally, in terms of Application, it will continue to update and learn to be more in tune with user habits, increase user satisfaction and improve software usage.

REFERENCES

Australian Government Department of Health. (2020, March 24). *Protect yourself and others from COVID-19*. Australian Government Department of Health. https://www.health.gov.au/health-alerts/covid-19/protect-yourself-and-others

Australian Government Department of Health. (2021, December 10). *Testing for COVID-19*. Australian Government Department of Health. https://www.health.gov.au/health-alerts/covid-19/testing

Benali Amjoud, A., & Amrouch, M. (2020). Convolutional Neural Networks Backbones for Object Detection. *Lecture Notes in Computer Science*, 282–289. https://doi.org/10.1007/978-3-030-51935-3_30

Hasan, N., Bao, Y., Shawon, A., & Huang, Y. (2021). DenseNet Convolutional Neural Networks Application for Predicting COVID-19 Using CT Image. *SN Computer Science*, *2*(5). https://doi.org/10.1007/s42979-021-00782-7

He, K., Zhang, X., Ren, S., & Sun, J. (2015). *Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification*. https://arxiv.org/pdf/1502.01852.pdf

He, K., Zhang, X., Ren, S., & Sun, J. (2015b, December 10). *Deep Residual Learning for Image Recognition*. https://arxiv.org/abs/1512.03385

Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. (2018). *Densely Connected Convolutional Networks*. https://arxiv.org/pdf/1608.06993.pdf

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., & Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, *115*(3), 211–252. https://doi.org/10.1007/s11263-015-0816-y

Simonyan, K., & Zisserman, A. (2015, April 10). *Very Deep Convolutional Networks for Large-Scale Image Recognition*. https://arxiv.org/abs/1409.1556

Szegedy, C., Vanhoucke, V., Ioffe, S., & Shlens, J. (2015). *Rethinking the Inception Architecture for Computer Vision*. https://arxiv.org/pdf/1512.00567.pdf

Therapeutic Goods. (2022, June 21). *COVID-19 rapid antigen self-tests that are approved in Australia*. Therapeutic Goods Administration (TGA). https://www.tga.gov.au/products/covid-19/covid-19-tests/covid-19-rapid-antigen-self-tests-home-use/covid-19-rapid-antigen-self-tests-are-approved-australia

Wang, W., Li, X., Yang, J., & Lu, T. (2018). *Mixed Link Networks*. https://arxiv.org/pdf/1802.01808.pdf