ICIC Express Letters
Part B: Applications
Volume **10**, Number **8**, August **2019**

ICIC International ©2019 ISSN 2185-2766
pp. 707–715

EYEGLASSES FRAME SELECTION BASED ON OVAL FACE SHAPE USING CONVOLUTIONAL NEURAL NETWORK

Stephen Young[1], Friska Natalia[1], Sud Sudirman[2] and Chang Seong Ko[3]

[1]Department of Information Systems
Universitas Multimedia Nusantara
Jl. Scientia Boulevard, Gading Serpong, Tangerang, Banten-15811, Indonesia
stephen.young@student.umn.ac.id; friska.natalia@umn.ac.id

[2]Department of Computer Science
Liverpool John Moores University
Liverpool L3 3AF, UK
s.sudirman@ljmu.ac.uk

[3]Department of Industrial and Management Engineering
Kyungsung University
No. 309, Suyeong-ro, Nam-gu, Busan 608-736, Korea
csko@ks.ac.kr

Received January 2019; accepted April 2019

Abstract. *Deep Learning is one part of the Machine Learning which is now widely discussed by researchers. One use of Deep Learning is on the case of image classification. One method that can be used is Convolutional Neural Network. In this case, by using Convolutional Neural Network, it will identify a person's face shape is oval or not. This method will be implemented into Android OS and using Python Language to help in processing the data that retrieve and observe from search engine Google. Viola-Jones Algorithm is useful to analyze the existence of the faces in the image processing. The output that will be accomplished is an application that can give user a choice of frames of glasses based on face shapes. Through this application user can try a variety of frames with practice rather than having to try it in physical form one by one.*

Keywords: Android, Convolutional Neural Network, Eyeglasses frames, Oval face shapes, Viola-Jones

1. Introduction.

Information technology is growing rapidly especially in the use of mobile devices; many applications are designed to meet the needs of its users. The Android operating system is one of the options as a platform in a study due to the Android operating system that is an open source product that was initiated by Google [1]. According to Kremic and Subasi in 2011, Smartphone is a medium-sized machine that is running lots of applications and data behind it very quickly. One example of this study is the face recognition application [2].

The use of face recognition is applied to a number of social media such as Snapchat. Snapchat is an application where users can exchange photos and short video which can be edited to add a filter or text [3]. This application is the basis for expanding the use of face recognition for the business world in the realm of fashion. Broadly speaking, it consists of three kinds of fashion, that is, absolute, millineries and accessories. Eyewear is a millinery, that is, the complement of fashion which is complete absolute fashion that has value and also for beauty [4].

Choose eyeglasses frames for every individual variously to look nice, so first determine the shape of his face [5] because every human being has a different face shape [6] consisting of five basic forms, i.e., oval, round, square, heart and diamond [7]. According to

DOI: 10.24507/icicelb.10.08.707

SharezahHousden, an expert in eyeglasses of Culter and Gross of Knightsbridge, says that choosing eyeglasses not just looks at the design of the frames, but also looks at hair and face shape suitability [8]. Therefore, this becomes the reason to examine how to make it easier for a person in choosing eyeglasses frame that suits face shape.

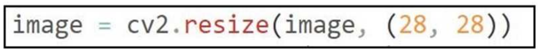
In the previous research, an application that detects faces using Convolutional Neural Network (CNN) was developed. This application can determine the person the face in the input image belongs to with good accuracy. This research will apply the concepts and principles of such research and add an innovation that is able to provide recommendations frame glasses if the input image detected shaped oval face. CNN is a type of Deep Learning in a form of Multilayer Perceptron designed to process data as a two-dimensional image. Multilayer Perceptron (MLP) is a type of neural network model composed of numerous neurons connected by weights of liaison. The concept of CNN is not much different from MLP. The difference lies in the fact that every neuron in the CNN is presented into a two-dimensional form, while MLP every neuron measures only one dimension. In the case of image classification, MLP is less appropriate because it does not store information spatial data from imagery and consider every pixel is an independent feature so that the yield is not good. There are three kinds of primary layer on the CNN concept, i.e., Convolution Layer, Layer Subsampling (Pooling Layer) and Fully Connected Layer. These layers represent the process of building a Convolutional Neural Network model [9]. The output after building the infrastructure is a model that needs to be implemented when building the applications. This model will be a core in the system to become the human-computer interactions application.

2. **Problem Statement and Methodology.**

This paper presents a solution to the problem to increase users' satisfaction when using an eyeglasses selection application. In order to do that, we performed the User Acceptance Test to eleven people based on how this application can impact their decision to help them choose the right glasses. In this case, the shape of the face is one of the factors that affected it. This research uses Convolutional Neural Network method to help classify human face shape. The shape of the face which became the object of research is the oval face shape, because there are so many example images that can be obtained through simple search in the Google search engine. In fact, all the data that we used in the experiment were retrieved from the Internet using this search. The data collected in this research contains 1300 images of a human face consisting of 650 images of a human face with oval face shape and 650 images of human face with square face shape.

2.1. **Image processing.**

The data obtained will be resized into 28×28 so that each image has the same size. Figure 1 shows a python script for resizing images.

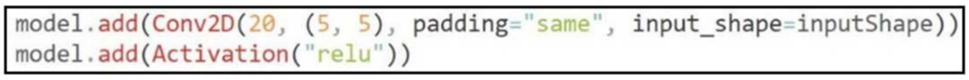


```
image = cv2.resize(image, (28, 28))
```

FIGURE 1. Script image resizing

2.2. **Convolutional Layer & Pooling Layer 1.**

Convolutional Layer composed of neurons is arranged in such a way that it forms a filter with the length and height (pixels) [8]. We used the Conv2D function on the Keras framework for convolution process. The Conv2D function takes 4 arguments, the first is the number of filters, in this case we set 20, the second argument is the shape each filter is going to be, in this case we set 5×5, the third is the input shape and the type of image (RGB or Black and White) of each image, in this case the input image our CNN is going to be taking is of a 28×28 resolution and "3" stands for RGB, which is a color image, the fourth argument is the activation



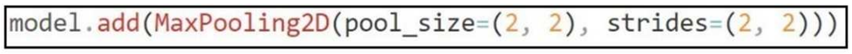
```
model.add(Conv2D(20, (5, 5), padding="same", input_shape=inputShape))
model.add(Activation("relu"))
```

Figure 2. Convolutional Layer 1

function we want to use, and here 'relu' stands for a rectifier function. The function can be seen in Figure 2.

Padding is a parameter that determines the number of zero-valued pixels that will be added on each side of the input. It is used with the intention to manipulate the dimensions of the output of the Convolutional Layer (Featured Map) [10]. By using padding, we can set the dimensions of the output so that it remains the same as the dimension of the input or at least not reduced drastically. In this case, we set this value to "same" to preserve the same output image dimension as the input's. This will allow us to use more Convolutional Layer and deeper network to extract more global features.

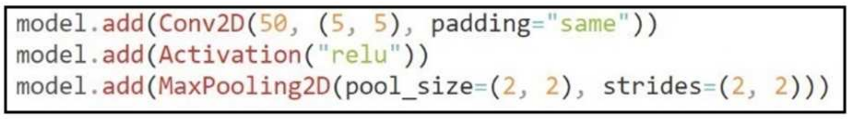
Pooling that is usually used is the Max Pooling and Average Pooling [10]. In this case, we used Max Pooling with pool size as $(2, 2)$ representing the value of the maximum on the 2×2 pixel in Featured Map area that we get from Convolutional Layer and strides as $(2, 2)$ representing how many pixels the filter shifts horizontal and then vertical. The purpose of Pooling Layer is reducing the dimension of Feature Map to accelerate the computation. Figure 3 represented Max Pooling Function.



```
model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))
```

Figure 3. Pooling Layer 1

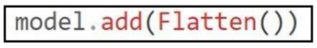
2.3. **Convolutional Layer & Pooling Layer 2.** Just like with the previous phase, in order to increase the accuracy, we create Deep Convolutional Neural Network model by adding one Convolutional Layer and one Pooling Layer. Figure 4 represented Convolutional and Pooling Layer.



```
model.add(Conv2D(50, (5, 5), padding="same"))
model.add(Activation("relu"))
model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))
```

Figure 4. Convolutional and Pooling Layer 2

After that we use flatten function to change Feature Map from 2D vector into 1D vector which will be used in Fully Connected Layer [10]. Figure 5 represented flatten function.

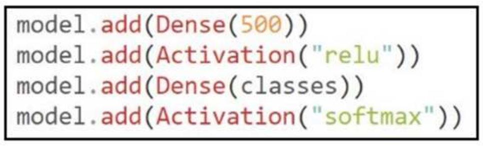


```
model.add(Flatten())
```

Figure 5. Flatten function

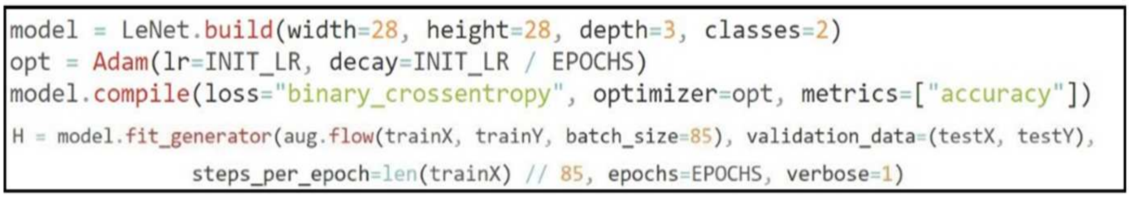
2.4. **Fully connected layer.** On this layer we will link each node obtained after performing the flattening step [10]. By using dense function, we can create a fully connected layer, and we set 500 represented nodes that should be present in this layer. The value will be always between input nodes and output nodes. The activation function will be a rectifier function. Figure 6 details the initialization of the dense function.

After building CNN model, we need to compile the model using Adam function. Then we fit the data to our model by training with iteration as much as 5000 times. Figure 7 shows the script for compiling model and training model.



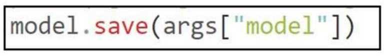
```
model.add(Dense(500))
model.add(Activation("relu"))
model.add(Dense(classes))
model.add(Activation("softmax"))
```

FIGURE 6. Dense function



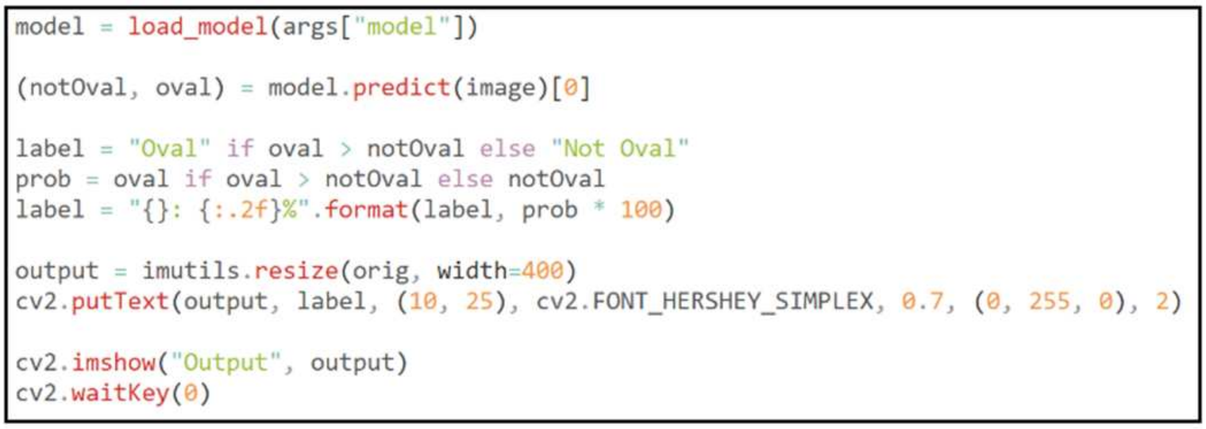
```
model = LeNet.build(width=28, height=28, depth=3, classes=2)
opt = Adam(lr=INIT_LR, decay=INIT_LR / EPOCHS)
model.compile(loss="binary_crossentropy", optimizer=opt, metrics=["accuracy"])
H = model.fit_generator(aug.flow(trainX, trainY, batch_size=85), validation_data=(testX, testY),
                steps_per_epoch=len(trainX) // 85, epochs=EPOCHS, verbose=1)
```

FIGURE 7. Compiling and train model



```
model.save(args["model"])
```

FIGURE 8. Create model



```
model = load_model(args["model"])

(notOval, oval) = model.predict(image)[0]

label = "Oval" if oval > notOval else "Not Oval"
prob = oval if oval > notOval else notOval
label = "{}: {:.2f}%".format(label, prob * 100)

output = imutils.resize(orig, width=400)
cv2.putText(output, label, (10, 25), cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 255, 0), 2)

cv2.imshow("Output", output)
cv2.waitKey(0)
```

FIGURE 9. Testing model

The model itself can be easily created using Keras framework. Figure 8 shows the script to create the model after training process.

After we get the training model, we need to test and predict our data. Figure 9 shows script to test the model. First of all, we need to load the model and then make a predict function, to predict the input image face shape. Labeling will be added if the result shows that the input image is an oval face shape or not with the predictions percentage.

Accuracy Graphic for testing results and predictions from the model which are already built will be explained in the next chapter. Usually, in the CNN features used in ten conventional algorithms are the raw pixel values and their corresponding locations [11]. However, in this study CNN Model will be implemented in the application development on Android Studio where this model is to identify whether the user input image is oval face shape or not.

The flow of the process of the system is as follows.

1) User can input the image by taking a picture directly from the application.
2) The input image will be detected by the system whether having facial characteristics or not using Viola-Jones algorithm in Face API invented by Google. There are eight points which represent each facial characteristic for the detection of the face on the

image that user inputs, and the points are left eye, right eye, left cheek, right cheek, down side of nose, left side of lips, right side of lips and down side of the lips.
3) When the input image detected had characteristic of face, then the system will identify the face shape using models that were already built before.
4) There are two scenarios which occur after the identification of the face shape, the first, if the face is unidentified oval face shape then the system will give a notification to the user that the face is not oval and the second, if the face identified oval face shape, the system will continue to the next step.
5) In the next step, the user will get four glasses frames recommendation [12]. Glasses frames that are being recommended on this research can be seen in Figure 10.
6) User can select and try on the glasses frame recommendation in the application. The system will automatically attach the frames on the image input from the user.

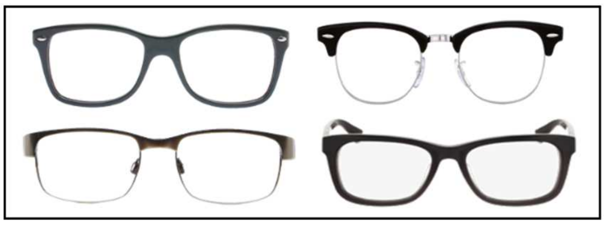
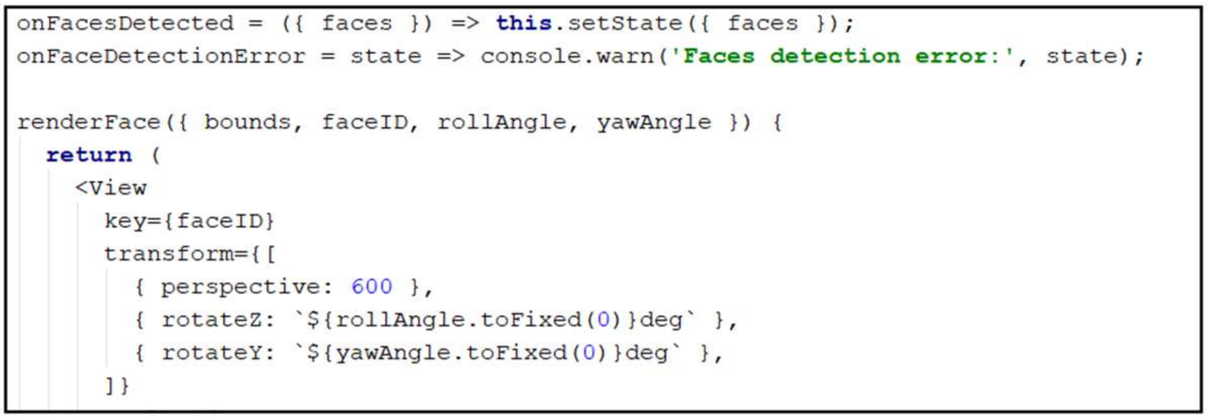


Figure 10. List of frames

3. **Analysis and Discussion.** In order that the frame can be attached to the face, it is necessary to determine the location of the eyes in the image. Using face API, we can detect the face and its associated facial landmarks such as the eyes [13]. First of all, we need to create a face detector. Script for face detector can be seen in Figure 11.



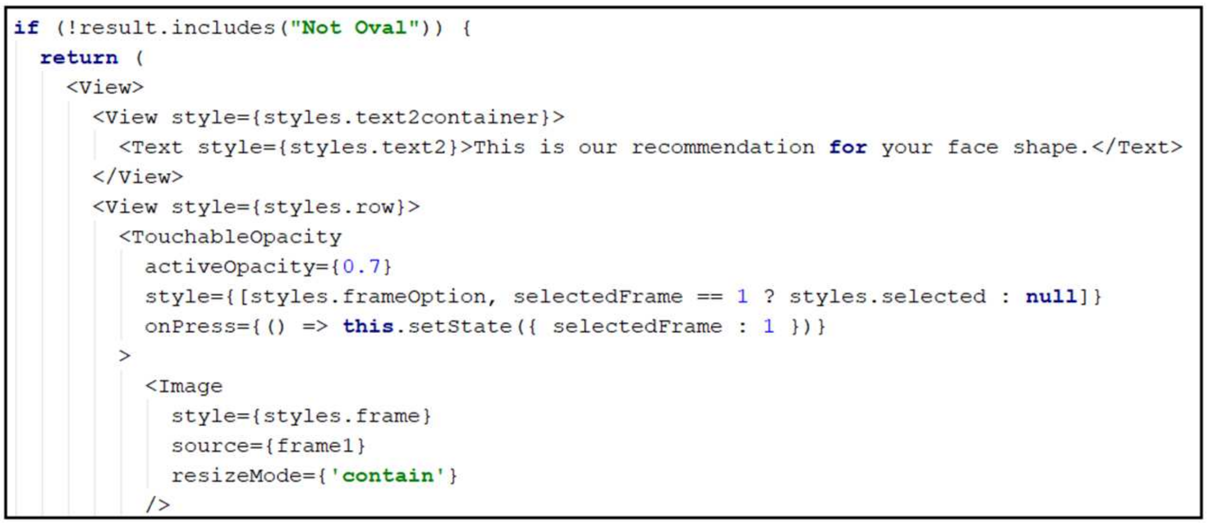
```
onFacesDetected = ({ faces }) => this.setState({ faces });
onFaceDetectionError = state => console.warn('Faces detection error:', state);

renderFace({ bounds, faceID, rollAngle, yawAngle }) {
  return (
    <View
      key={faceID}
      transform={[
        { perspective: 600 },
        { rotateZ: `${rollAngle.toFixed(0)}deg` },
        { rotateY: `${yawAngle.toFixed(0)}deg` },
      ]}
```

Figure 11. Face detector

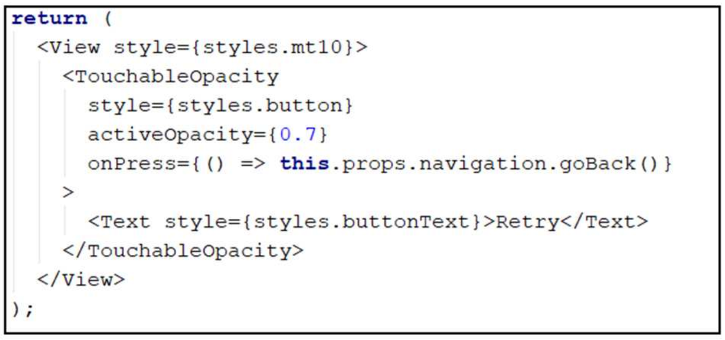
After that, landmark will determine the objects on a face as well as the return value of the coordinates at each object on a face. The coordinates will be pointing the objects directly to the input image when user tries to capture the picture. The application can identify the input image face shape is oval or not, by parsing the input image into the server. In the server there is only a script like as in testing script. This script needs to add a Flask function because the script will operate in web-based platform. Then system will retrieve the result and parse the result into frame selection page in the application.

Assets such as frame glasses, have previously been input into application in advance and were called at the time of frames selection page. There are two conditions in frame selection. If the input image is identified as having an oval face shape, frame recommendation will be shown in the list and also in the identification result. User can select one



```
if (!result.includes("Not Oval")) {
  return (
    <View>
      <View style={styles.text2container}>
        <Text style={styles.text2}>This is our recommendation for your face shape.</Text>
      </View>
      <View style={styles.row}>
        <TouchableOpacity
          activeOpacity={0.7}
          style={[styles.frameOption, selectedFrame == 1 ? styles.selected : null]}
          onPress={() => this.setState({ selectedFrame : 1 })}
        >
          <Image
            style={styles.frame}
            source={frame1}
            resizeMode={'contain'}
          />
```

Figure 12. Frame selection



```
return (
  <View style={styles.mt10}>
    <TouchableOpacity
      style={styles.button}
      activeOpacity={0.7}
      onPress={() => this.props.navigation.goBack()}
    >
      <Text style={styles.buttonText}>Retry</Text>
    </TouchableOpacity>
  </View>
);
```

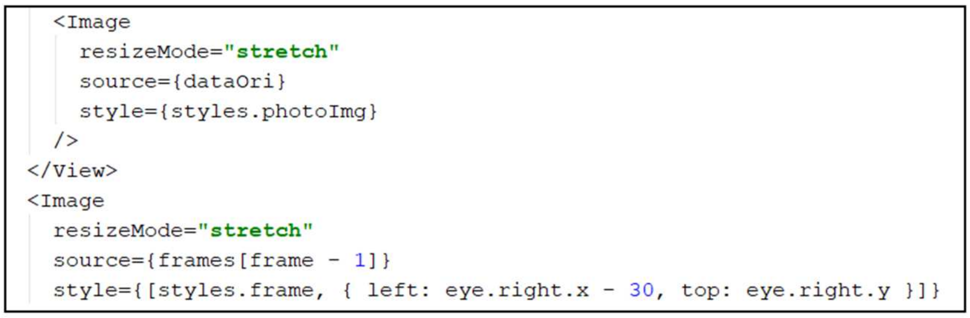
Figure 13. Not oval face shape

frame on the list to try it on. The application will parse the frame into try frame page in the application. The script of frame selection if oval face shape can be seen in Figure 12.

The second condition is that the users will not be able to see and choose the recommendations frames if the results of the identification are not oval face shape. The user can try to capture an image again to let the system identify their face again. The script can be seen in Figure 13.

From the landmark process, application already returns coordinates of each of face objects. To determine the position of the eyes in the input image, it will parse the coordinates and calculate again the coordinate because the input image needs to be resized to 150×150 pixels and the result stored to use on the try-frames page. Then on the try-frame page, the system will parse two objects, that is, user input image and the frame of glasses which are selected by the user. The value of the coordinates of the area of the eye on the image is useful for attaching the frame in the picture of the face. Script can be seen in Figure 14.

Prior to this, there are research applications that can only identify the face shapes but none gives recommendations on suitable eyeglass frames. In this study, the users can, not only find out the shape of their face, but also the recommendations frame glasses that suits the shape of her face, and in this case is oval face shape. Figure 15 represented training accuracy after training the model. On this graph, it can be concluded that the accuracy is increased in proportion to the magnitude of the iteration. Not only value of iteration, the amount of data that is used for training phases also affect the level of accuracy. The prediction will automatically generate when testing was done. In this case,



```
    <Image
      resizeMode="stretch"
      source={dataOri}
      style={styles.photoImg}
    />
  </View>
  <Image
    resizeMode="stretch"
    source={frames[frame - 1]}
    style={[styles.frame, { left: eye.right.x - 30, top: eye.right.y }]}
```

FIGURE 14. Try frame

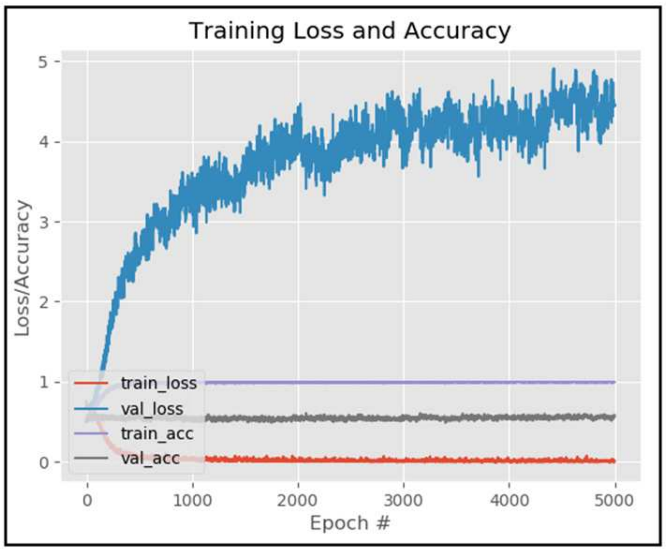


FIGURE 15. Accuracy graphic

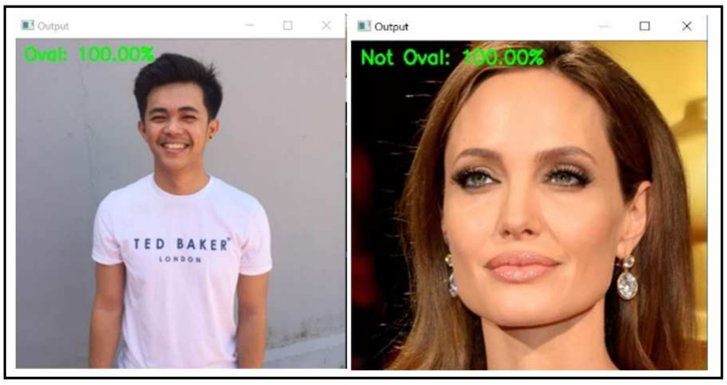


FIGURE 16. Prediction result

we used two categories of faces namely oval and non-oval face shape to determine the eyeglass frame recommendation. The result can be seen in Figure 16.

This model is implemented into Android so user can use the application by download from Android Play Store, the application name is Justfit. The application can be downloaded in Android Play Store for free at this link https://tinyurl.com/unduhjustfit.

4. **Conclusions.** The Deep Convolutional Neural Network was applied in this study to solving the problem and using Python Language to create the model. This model can make it easier for a person in choosing eyeglasses frame that suits face shape rather than having to try it in physical form one by one. From User Acceptance Test result,

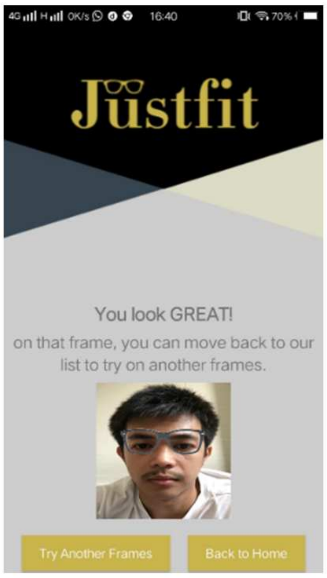

Jüstfit
You look GREAT!
on that frame, you can move back to our list to try on another frames.
Try Another Frames
Back to Home

Figure 17. Try frame result

the level of satisfaction of respondents did not reach satisfactory results, there are eight respondents who experienced failure and three successful respondents in terms of accuracy of glasses frames attached to the eye area, so our satisfaction level respondents against the accuracy of these frames only reached 27%. An example of the final recommendation is shown in Figure 17. This figure shows the accuracy of the positioning of the eyeglasses frame relative to the eye area. Positive feedback from respondents is that applications can properly detect the oval face. The average scale for satisfaction of respondents within the application recommendations frame glasses is 3.36 out of 5. Furthermore, other problems in identifying other human face shapes and used in real time will be studied in future research.

Acknowledgment. All computations are performed in Big Data Laboratory of Universitas Multimedia Nusantara.

REFERENCES

[1] B. Patil and P. Ramteke, Development of Android based cloud server for efficient implementation of platform as a service, *International Journal of Advanced Research in Computer Science and Software Engineering*, 2014.
[2] E. Kremic and A. Subasi, The implementation of face security for authentification implemented on mobile phone, *The International Arab Journal of Information Technology*, 2011.
[3] J. I. Olszewska, Automated face recognition: Challenges and solutions, *Intech*, pp.70-71, 2016.
[4] R. Ridhiawaty, The results of the study design material invisibility technique of wet settlement through model STAD, *Action Research and Education Journals*, vol.3, no.2, 2017.
[5] H. A. Stein, R. M. Stein and M. I. Freeman, *The Ophthalmic Assistant: A Text for Allied and Associated Ophthalmic Personnel*, Elsevier Health Sciences, 2012.
[6] A. Andiyanto and A. I. Karim, *The Make Over*, PT Gramedia Pustaka Utama, Jakarta, 2010.
[7] V. Ekanem, *Beauty and Wellness: A Compilation of All You Need to Know to Proceed in Style, Beauty, Good Health and Confidence*, Publiseer Publishing, Nigeria, 2017.
[8] C. Harding, *How to Choose Glasses That Suit You*, http://www.dailymail.co.uk/health/article-77313/How-choose-glasses-suit-you.html, accessed on September 26, 2011.
[9] W. S. E. Putra, Image classification using Convolutional Neural Network (CNN) on the Caltech 101, *ITS Engineering Journal*, vol.5, no.1, 2016.
[10] W. Rawat and Z. Wang, Deep convolutional neural networks for image classification: A comprehensive review, *Neural Computation*, vol.29, no.9, pp.2352-2449, 2017.
[11] A. S. Al-Kafri et al., Segmentation of lumbar spine MRI images for stenosis detection using patch-based pixel classification neural network, *IEEE Congress on Evolutionary Computation*, 2018.

[12] B. Brown, *Everything Eyes: Professional Techniques * Essential Tools * Gorgeous Makeup Looks*, Chronicle Books, San Francisco, 2014.

[13] Google, Inc., *Introduction to Mobile Vision*, https://developers.google.com/vision/introduction, accessed on December 22, 2017.