Blink Counter
For AIP Project

Dung. Nguyen Tien[a], Toan. Le Nguyen Khanh[a]

[a]*FPT University, Norges Teknisk-Naturvitenskaplege Universitet, N-7491 Trondheim, Norway.*

June, 2022

Abstract

The eyes are essential to us. They help people how to see the color, to read the writings, or to communicate through gestures or emotions using those eyes. They have contributed to many aspects of daily life. However, the essential things now deteriorate through time nowadays. To counter this problem, the observation to check how the eyes are needed and the eye detection algorithm are implemented. The blink counter is one of the devices using those algorithms. In this project, the blink counter is using the model trained by Convolution Neural Network (CNN) and image-related libraries such as Dlib and OpenCV to scan the video that has been recorded.

Keywords: CNN, Eye Detection, OpenCV, Dlib

1. Introduction

In recent years, artificial intelligence has been developed through many stages from testing some algorithms to letting the computer learn using machine learning algorithms and data mining. There are important projects that have impacted society such as face recognition, eye detection, etc. This project uses how eye detection works to make the blink counter. This blink counter can be used for several applications: Eye check-ups, Drowsiness detection, and many other things. The model used for the project is Convolution Neural Network (CNN). This model is applied to this project to classify the images into open-close state eyes images. Even though the model can process the color, and complex images, the process will take a long time and gradually loss the performance which will make the quality of the images drop for the project. To counter this problem, the image needs to be preprocessed into small, grayscale images to train the model faster and efficiently. To discuss further, the contents below will explain the whole process of the process.

2. Related Works

2.1. Eye Detection with faster R-CNN

As mentioned in the section above, eye detection is one of the applications implemented with machine learning that has impacted society such as biometric security. Faster R-CNN and is an extension of Fast R-CNN[4]. Faster R-CNN is one of the mainstream methods for many object detection applications. Eye detection uses this network to detect the eye from the recorded or live video. Faster R-CNN uses Regional Proposal Network (RPN) to extract region proposals and locate objects. This network can run faster than R-CNN and fast R-CNN and can reach 8-9 frames per second in professional GPU [2].

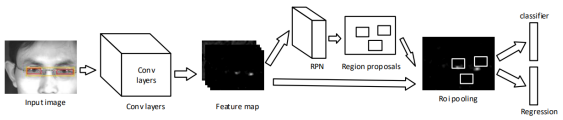


Figure 1: How faster R-CNN works

The monocular model and binocular model are studied to understand how different those models work. To learn those models, the dataset is separated into 3 specific labels: left eye, right eye, and both eyes. The dataset used here is CASIA Iris Image Database Version 4.0, which is used for iris recognition. This dataset is suitable to study the model and apply it to the application. The model used for the eye detection has accuracy detection in the task with 95% accuracy for the binocular and 95% for the monocular. However, there are some drawbacks to generalization ability and speed. This project has proved how effective the R-CNN model can be applied to eye detection.

2.2. Iris recognition with Convolution and Residual Networks

Iris recognition is also one of the eye detection applications. It is using visible and near-infrared

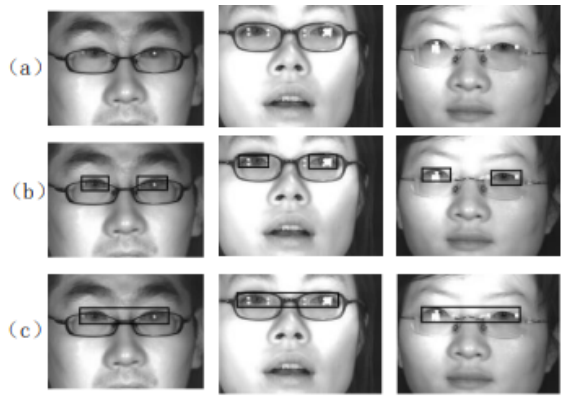


Figure 2: Partial data: (a)original images (b)monocular areas (c) binocular areas

light to take a high-contrast photograph of a person's iris and uses this feature to make geometric security such as face recognition or ear recogniton[3]. Convolution and Residual Networks are combined to be used for iris recognition. The CNN learns image feature representations automatically. Residual Network is used here to counter accuracy loss for image-related tasks, which can use the knowledge acquired in previous layers[6]. Each has its advantages and disadvantages when extracting and classifying images. The MiCoRe-Net is the neural network that combines the ConvNet and ResNet by starting with a convolutional layer and inserting a convolutional layer between every two residual layers. It obtains advantages from both architectures, which is not only to learn the model faster than CNN but also to overcome the saturation issue caused by the plain CNN.

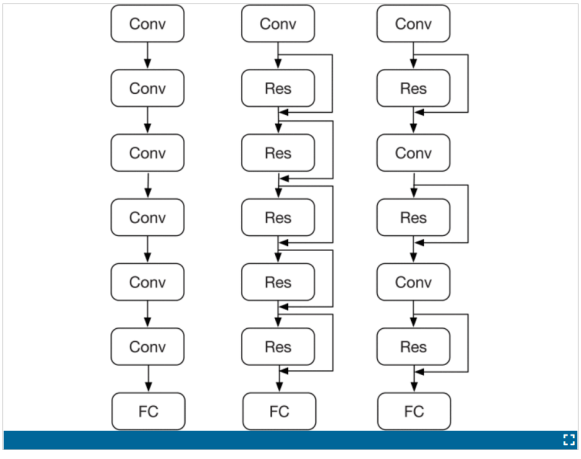


Figure 3: An illustration of the pure convolutional neural network (Left), residual network (Middle), and proposed MiCoRe-Net (Right).[8]

The model that has been trained using MiCoRe-Net has achieved performance exceeding the performances of traditional methods and other previously proposed deep learning-based architectures. The MiCoRe-Net is one of the neural networks that can achieve excellent results in eye detection projects.

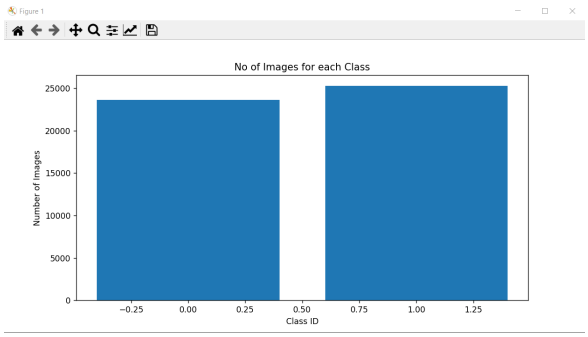


Figure 4: How the train and test dataset split

3. Method

3.1. Dataset

The dataset used for the project is the OACE dataset (Open and Close Eyes) from Kaggle. It contains 100.000 eye images that have been classified and put into two separate folders (open and close). This dataset is very helpful when used for iris localization. However, the images are in color, which makes the processing run slower. To prevent this from happening, turning the images into grayscale images is necessary. The images also need to be resized on a smaller scale (In this project, the shape of the images is rescaled into (32, 32)) to reduce the load when processing.

The images in the dataset also have external features such as glasses and the images are modified such as shifting, and rotating. The initial dataset has been split into two separate lists: the training dataset and the test dataset. To validate and test the model that has been made, the dataset is split into two lists with a ratio of 50:50.

The dataset is also separated based on the folders into two lists and those lists are labeled into the binary number (1 for the open images, 0 for the close images). To improve the model, enriching the dataset by data augmenting using Keras from the Tensorflow library, which contains many built-in tools to help for machine learning and artificial intelligence project. The images are also rescaled by a fraction of 255 (the reason is the minimum-maximum value for the images is 0-255) to reduce images into ones with values 1 and 0. This preprocessing help to increase the accuracy and reduce the load for processing the data.

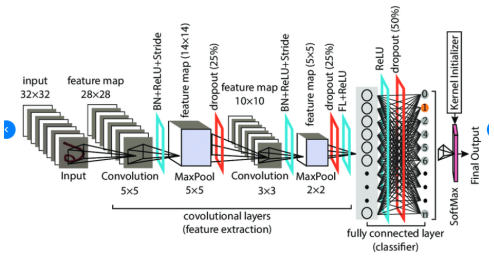


Figure 5: How CNN works

3.2. Train CNN model

In this project, the CNN model is used for image classification. The CNN is one of the deep learning methods to process the images by extracting features from them, assigning weights, and convolving the images with a suitable filter to identify the images based on the results. The model consists of four Conv2D layers in total, two Conv2D layers to extract features and one MaxPooling2D layer followed to remember the results and give them to the other layers and reduce the dimension of the map. After finishing the procedure, the Dropout layer shut down half of the previous layers to prevent all the neurons from converging to the same goal, thus decorrelating the weights. The Flatten follows after the Dropout layer to unroll the results that are suitable for the last two-layer (Dense layers) to compute the final results. The final output is a binary value whose values range from 0 to 1 using the sigmoid function which determines the open or close eye images.

The input for the model needs to be the images with width-size-channel, which is (32-32-1). The first convolution layer use kernel with a size of 5x5, which improves the performance of the model, and extracts features with 60 filters. The Max Pooling layer will highlight by taking the highest values with kernel size 2x2 and giving those values to the next convolution layers[7]. The next two convolution layers will use kernel size 3x3, which is different from the first convolution layer used, to extract the edge features with 30 filters from the given values. All convolution layers are using the Relu activation function since the Relu activation function is simple, fast to compute, and doesn't suffer from vanishing gradients, like sigmoid functions. The Dropout will shut down half of the nodes in the model to prevent overfitting, which makes the accuracy of the model reduced when compared to the different images. The Flatten layer will take the extracted edges and flatten them into 480 weights to process for further layers. The FC layers (Fully Connected layers) will use those weights to calculate and make the final result for the model, which classifies the data into

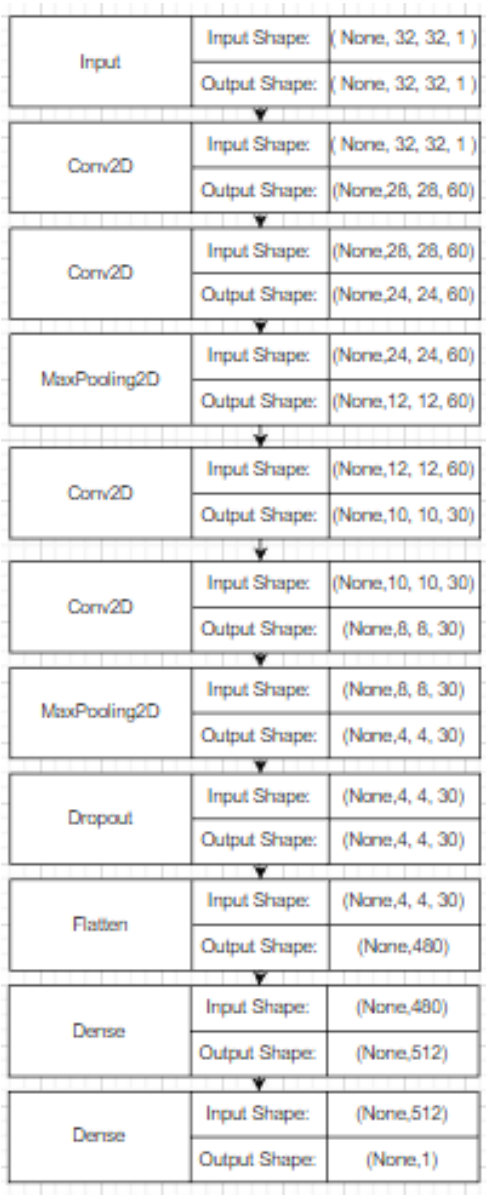


Figure 6: The architecture of the CNN

binary classes. The first dense layer used Relu activation, which is non-linear activation, to classify and choose which feature is relevant for a class or not using 512 layers. Since the binary classification approach is used for the model, the sigmoid function, which is equivalent to 2 class in multi-class function softmax where the second element is assumed to be zero, is preferably used. The final Dense layer, which used Sigmoid activation, will output only one value in the range of 0 to 1, which will determine the state of the eye. The loss function used for the model is Binary Cross-Entropy to calculate the loss between binary values.

$$\text{Log loss} = \frac{1}{N}\sum_{i=1}^{N} -(y_i * \log(p_i) + (1-y_i) * \log(1-p_i))$$

Figure 7: Binary Cross Entropy[5]

The loss function will calculate the loss and adjust the model based on the test data for validation. To prevent the model deteriorate when processing, the function to save the best result (weigh) is used and discard the wrong predictions.

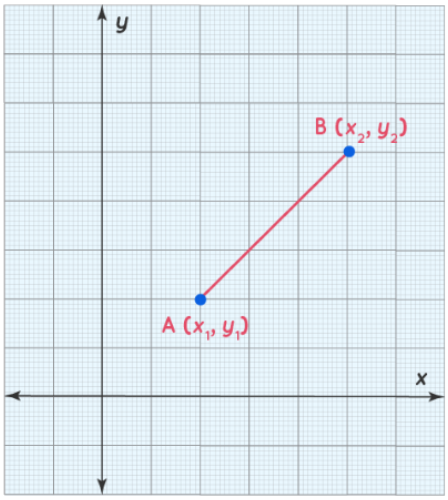
3.3. Testing the images

To use the model, the images need to be processed similar to the preprocessing that has been discussed above. The video that has been recorded or live feed will be processed as images by the OpenCV library, which is used mainly for computer vision. The images then will be localized using landmarks provided and employing Dlib to mark those landmarks onto the images. The OpenCV will use the landmarks to localize the eyes and extract the eye images. The input images need to have similar images that the model is given to predict the state of the eyes (For this model, the size 32x32x1 is necessary). The video will be grayscaled to reduce the load of processing when using the model.

3.4. How to determine which state is blinking

The model is trained to determine how to predict using one eye-grayscaled image. The model will return confidence values varying between 0 and 1. The application will set the model to return confidence values from each eye separately, so the way to determine the blink from the images is by using each confidence value to make the counts for the final result. To determine how many did one people in the image blink, the limit will be set appropriately from values between 0 and 1. The settings will be as followings: Both eyes will be closed if the confidence values from both eyes are less than 0.25, and the state of those eyes will be open if both eyes have confidence values more than 0.75; For the counter for one eye each side, if both eyes have different states, each eye has open state if the confidence value more than 0.75 and close state if the confidence value less than 0.25. To determine whether the eyes open or not, count the blinks, assigning a variable with a binary value such as True/False to see the state of the eyes. For this application, when the variable is set from False to True, the counter will increment. The setup for this application has accounted for how many the left eye, right eye, or both eyes blink separately.

Dlib uses the landmarks file given to mark all faces on the image, which make the counter hardly gives accurate result when other people use it. To resolve the issue, Dlib has provided some methods, and one of those methods is one of the basic formulas on distance, Euclidian Distance. By using this formula, the distance can be calculated and if the distance when comparing landmarks between two images is close, the face in the image is likely to be the same person (For Dlib, they set the value of 0.6 to be the standard to distinguish the face from other people).



$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Figure 8: Euclidean Distance Formula[1]

4. Results

After training, the model has achieved near 100% (about 0.998%) accuracy when compared to the data for validation. This model also can be used even including external factors such as shifting, glasses, and reflection in the images. From the confusion matrix, the data is not biased in the model. The model uses simple CNN to achieve performance, not the complexity of the details. The

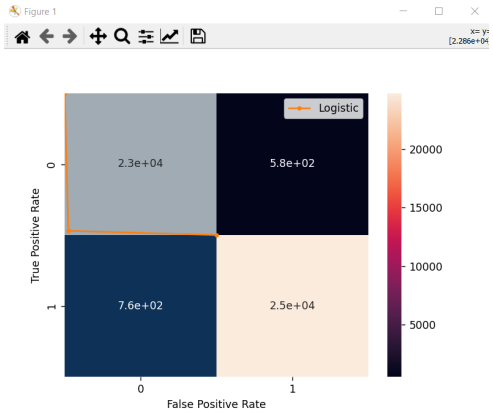


Figure 9: Confusion Matrix

model has achieved a satisfactory result that can be used for the application to apply for applications, which are eye detection-related applications such as blink counter.

The accuracy of this model is very high, however, the precision of this one is lower than other metrics. The True positive rate in this model is double the False-negative rate, which shows how biased the model is to the open eye images. This result can be caused by many factors: reducing images in size which reduce the pixels or the dataset has some images with a state open slightly which confuses the model. Even though there are some aspects of the model that need to be improved, the error is acceptable with the Recall and F-Score is very high, which achieved near 100% (about 0.998%)

4.1. The application

Figure 10: The state when the eyes are open

Figure 11: The state when both eyes are closed

Figure 12: The counter for both eyes increments after closing

Although the application created is simple and is not able to detect unexpected cases, it has achieved satisfactory results for this project. By using the landmarks provided by Dlib and the computer vision library OpenCV, the model can be used for the application with high accuracy. The application can detect images despite having external factors such as shifting, glasses, blurring, and angles.

Figure 13: The state when the eyes are open

Figure 14: The state when left eye is closed

Figure 15: The counter for the left eye is increments after closing

Using the method that has been discussed in the method, section, the counter in the application

works in each video with face images. However, the application cannot work with some videos such as the video which only has eye images. The model has achieved many satisfactory results such as high accuracy (99%) when compared to the data for validation. The application built for using this model is small but nearly accurately detects the blink and the counter for the blink works for each situation: Both eyes blink, left eye blinks, right eye blinks. Applying facial landmarks that were given, the application can be applied for tracking many faces at once. However, the application needs some improvements to apply to other videos and the model needs to be improved to counter the unexpected problems.

References

[1]CUEMATH, *Euclidean distance*, 2017.

[2]J. Cui1, F. Chen, D. Shi, and L. Liu, *Eye detection with faster r-cnn* (Oxford University Press, 2021).

[3]EFF, *Iris recognition*, 2019.

[4]A. F. Gad, *Faster r-cnn explained for object detection tasks*, 2020.

[5]D. Godoy, *Understanding binary cross-entropy / log loss: a visual explanation*, 2018.

[6]*Introduction to residual networks*, 2020.

[7]M. Tripathi, *Image processing using cnn: a beginners guide*, 2021.

[8]H. S. Z. Wang C. Li and J. Sun, *Eye recognition with mixed convolutional and residual network (micore-net)*, vol 6 (IEEE Xplore, 2018).

Appendix A. Contribution

Task	Priority	Owner	Time	Status	Self Assessment
Planning and designing the model	High	Dung, Toan	Week 1, 2	Completed	10
Collecting Data	High	Dung	Week 3	Completed	10
Evaluate methods	High	Dung, Toan	Week 4	Completed	10
Preprocessing dataset and training the model	High	Dung	Week 5, 6	Completed	10
Evaluate and Testing the model	Medium	Dung, Toan	Week 7	Completed	10
Writing Report and Presentation	High	Dung, Toan	Week 8, 9	Completed	10

Appendix B. Additional Information

The resources including the application and the model are in the following link:
https://fptuniversity-my.sharepoint.com/:f:/g/personal/dungntse150614_fpt_edu_vn/Enkw82NwYdtBp86hv5w4r7ABRz3ytdz62XLxEiuxXLtKkw?e=rNrTCy