International Journal Science and Technology
IJST Vol 2 No. 1 |March 2023 | ISSN: 2829-0437 (print), ISSN: 2829-050X (online), Page 44-50

HUMAN GENDER DETECTION SYSTEM BASED ON FACIAL IMAGE USING CONVOLUTIONAL NEURAL NETWORK ALGORITHM

Abdul Roid[1*], Ivan Maurits[2]
[1, 2,3]Faculty of Computer Science and Information Technology, Gunadarma University, Indonesia

Article History
Received : Feb 2023
Revised : Mar 2023
Accepted : Mar 2023
Published : Mar 2023

Corresponding author*:
abdurroid246@gmail.com

Cite This Article:
A. Roid and I. Maurits, "HUMAN GENDER DETECTION SYSTEM BASED ON FACIAL IMAGE USING CONVOLUTIONAL NEURAL NETWORK ALGORITHM", IJST, vol. 2, no. 1, pp. 44–50, Mar. 2023.

DOI:
https://doi.org/10.56127/ijst.v2i1.847

Abstract: The demand for system automation has been continuously increasing with the current technological developments. One of these advancements is in the implementation of face recognition. Camera capabilities have evolved from merely capturing images or videos to being able to process the resulting images. Facial images contain a wealth of information, one of which is the gender information of the individuals. To obtain this information, facial image classification using deep learning is required. In this scientific paper, the author utilizes the Convolutional Neural Network algorithm implemented with the Python programming language and employs TensorFlow as its framework. The research aims to predict human gender based on facial images. The dataset used in this study is obtained from the kaggle.com dataset provider, consisting of 9,600 male facial data and 9,600 female facial data. The data is divided into a training and testing set, with an 80% ratio for training data and a 20% ratio for testing data from the total available data. The model training process is performed for 15 epochs with 768 steps in each epoch. The testing results show that the Convolutional Neural Network method achieves a validation accuracy of approximately 91%. The developed program runs well through a webcam.

Keywords: Image processing, Convolutional Neural Network, deep learning, face recognition, gender, Python, TensorFlow, webcam

INTRODUCTION

Along with the current development of technology and information, its impact on human life is increasing. Thus causing very rapid progress in the field of technology which continues to innovate to follow the needs of everyday life. The need for system automation continues to increase. One of them is the implementation of face recognition applications (face detection).

The image on the human face has a lot of information that can be obtained, such as information in the form of gender, age, race, and expression. In recent years, many people or certain institutions need facial image detection tools for various purposes, including: security systems, customer identification, partner matching, beauty industry, demographic data analysis, and others.

Therefore, a deep learning model that has high accuracy is needed to make face detection applications more accurate. In this study, the authors used the Convolutional Neural Network algorithm to detect human gender based on facial images.

RESEARCH METHOD

The method used in this study uses the following stages:

1. Data Collection
   This is the stage for searching for human facial data obtained from the Kaggle website and dividing the image data into two classes, male and female.
2. Pre-processing of Data
   This is the stage for processing data into variables and augmenting images to increase visual variation in the dataset.
3. Model Architecture Design
   Is the network design stage of the Convolutional Neural Network which consists of determining the number of layers, determining filters, determining kernel size, determining pooling, and determining activation functions.

4. Model Training
   This is the stage for carrying out the training data training process for the model that has been made.
5. Model Implementation
   This is the distribution phase of the CNN model into webcam-based applications
6. Application Testing
   This is a trial stage for gender detection based on faces via a webcam.

RESULT AND DISCUSSION

Research Scheme

This research has several stages which can be seen in Figure 1. The first stage is to collect image data that will be used in the process of training and testing the architectural model of the convolutional neural network. Then a pre-processing stage will be carried out which functions to make data variations and normalize the data so that it is easier to train. After the data is processed, the next step is to design and build the architectural model of the convolutional neural network. This model is made to identify and verify facial image features. Next is the model training stage that has been designed using processed data. After the model is trained, the next step is to perform calculations on the model that has been trained by paying attention to the level of accuracy as a metric for measuring model performance. After the trained model has very good accuracy, the next step is the implementation or distribution of the model to the webcam application.

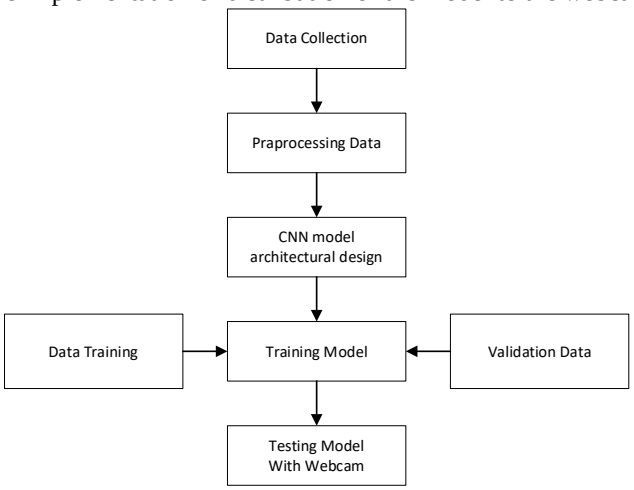

Figure 1. Research Scheme

Application Flowcharts

This flowchart is used to show the flow of the program through each process. Useful for explaining the sequence of procedures and processes for applications to run. The following is an overview of the flowchart of the gender detection application.

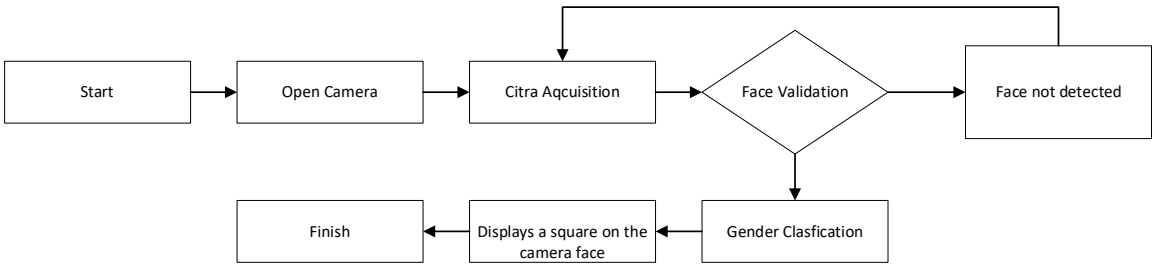

Figure 2. Application Flowchart

In Figure 2 there is an application process flow. When the application is run, it will open the camera, then the camera will take the object that will be in front of it. Next, the program will validate until the camera actually captures facial objects. If no facial objects are captured, the camera will continue to capture images until it gets facial objects. After the camera receives a facial object, it will be classified according to gender, whether male or female. It will then display a square on the face on the camera and display a gender detected label.

Data collection

The image data used in this study is data collected or obtained from the Kaggle website. The total amount of data is 19200 images which are divided into two classes into 9600 images of male faces and 9600 images of female faces.

Data Pre-processing

After the data collection process is complete, the next step is to upload the dataset to Google Drive in an archived format. After the dataset has been successfully uploaded to Google Drive, the next step is to process the data using Google Colaboratory. Here are the steps to connect Google Drive with Google Colaboratory.

```
drive.mount('/content/drive')
```

After connecting to Google Drive, the next process is to load and extract the data into the "tmp" folder in the Google Colaboratory directory. Apart from that, we also need to create special folders to store the male and female images. There are a total of 9600 images for each category, namely boys and girls.

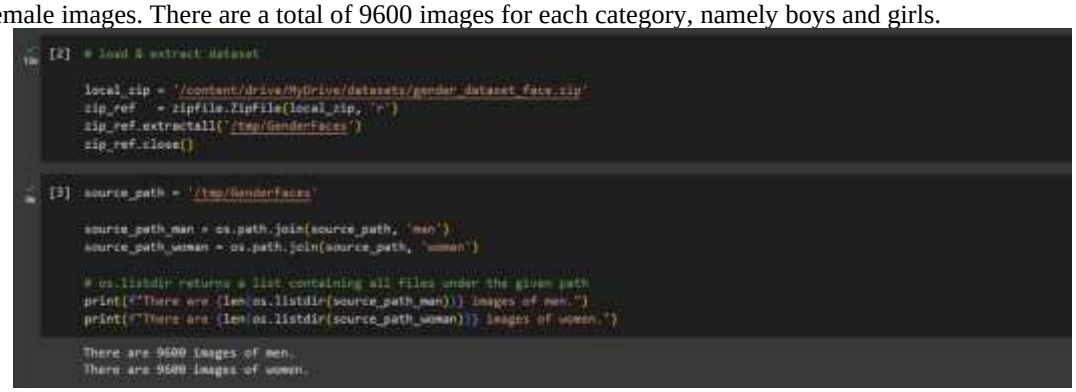

Figure 3. The process of loading and extracting datasets into Google Colaboratory

1) Split Data

At this stage, the data is divided into a training set of 80% and a testing set of 20%. First of all create a folder first to hold the training data and testing data in the Google Colab directory.

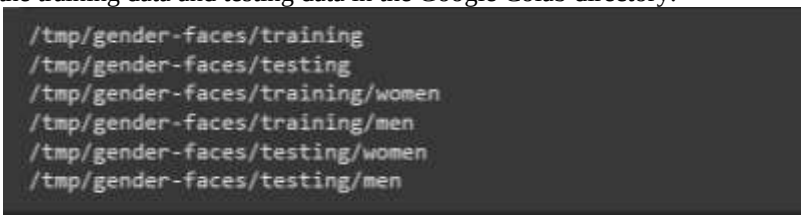

Figure 4. The results of testing the directory that has been created

After the required directories have been created, the next step is to divide the training set and the testing set. After dividing the data, the next step is to check the success of the division. In Figure 8, it can be seen that the data distribution has been successfully carried out, where there are 7680 images included in the training set and 1920 images included in the testing set.

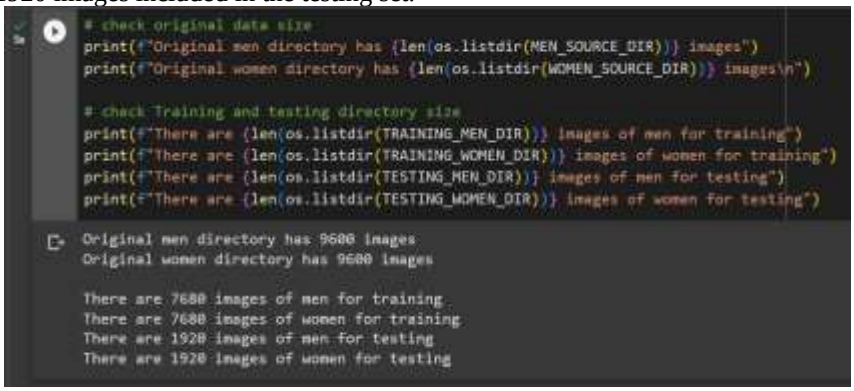

Figure 5. Results of data division

2) Data Augmentation

In this process, image augmentation will be carried out which aims to increase the diversity, variety and quality of training data in image processing. This process involves:

- rescale : to normalize the image pixel values. In this case, the pixel value is divided by 255 so that the value ranges between 0 and 1.
- rotation_range : to specify the angle range in degrees for random rotation of the image. In this example, the image can be rotated up to 40 degrees at random.
- width_shift_range : to specify the random horizontal shift range of the image as a fraction of the image width. In this example, the image can be shifted randomly up to 20% of the image width.
- height_shift_range : to specify the random vertical shift range of the image as a fraction of the image height. In this example, the image can be shifted randomly up to 20% of the image height.
- shear_range : to specify the random stretching range of the image. In this example, the image is subject to random stretching of up to 20%.
- zoom_range : to specify a random zoom range for the image. In this example, the image can be zoomed randomly by a factor of up to 20%.
- horizontal_flip : to set whether to do random horizontal flipping of the image. In this example, the image can be flipped horizontally with 50% probability.
- fill_mode : to determine how to fill the newly formed pixels after augmentation. In this case, 'nearest' fills in the nearest pixel.

Then, new objects are created in the train_generator and testing_generator variables as input image sizes for the model to be used for training and testing. The object has parameters such as batch_size and target_size which are useful for setting the size of the input image.

Model Architecture Design

At this stage, the author uses a Sequential model which consists of several layers arranged sequentially. The input layer uses a convolutional layer with 16 filters, each with a 3x3 dimension. The activation function used is ReLU. The input_shape parameter specifies the dimensions of the input model, which is a 150x150 pixel image with 3 channels (RGB). Then, it is continued with the MaxPooling2D layer which is used for downsampling the feature map generated by the previous Convolutional layer. In this case, MaxPooling is done with filters measuring 2x2 and stride 2. This process is repeated for the next few layers with the addition of double the number of filters from the previous layer.

On the 11th layer, use the Flatten layer which functions to flatten the output from the previous layer into a one-dimensional vector. This output will be the input for the next layer, namely the fully connected layer. This layer has 1024 neurons in the hidden layer with the ReLU activation function. In the output layer, there is a fully connected layer with 1 neuron and a sigmoid activation function. This layer is used to generate model output in the form of binary class probabilities, where the output is close to 0 or 1 to represent different classes. Thus, the fully connected layer on the 11th layer and the output layer are responsible for classifying based on the features previously extracted.

```
model = tf.keras.models.Sequential([
tf.keras.layers.Conv2D(16, (3,3), activation='relu', input_shape=(150, 150, 3)),
tf.keras.layers.MaxPooling2D(2, 2),
tf.keras.layers.Conv2D(32, (3,3), activation='relu'),
tf.keras.layers.MaxPooling2D(2,2),
tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
tf.keras.layers.MaxPooling2D(2,2),
tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
tf.keras.layers.MaxPooling2D(2,2),
tf.keras.layers.Conv2D(128, (3,3), activation='relu'),
tf.keras.layers.MaxPooling2D(2,2),
tf.keras.layers.Flatten(),
tf.keras.layers.Dense(1024, activation='relu'),
tf.keras.layers.Dense(1, activation='sigmoid')
])
```

After the model is defined, the model is compiled using the Adam optimizer with a learning rate of 0.001. The loss function used is binary_crossentropy, which is suitable for binary classification tasks. The evaluation metric used is accuracy, which measures the accuracy of the model in classifying. By compiling

the model, the model is ready to be trained and evaluated using the appropriate data. By using model.summary(), we can see the overall model structure. In this model, the total parameters trained are 660,705.

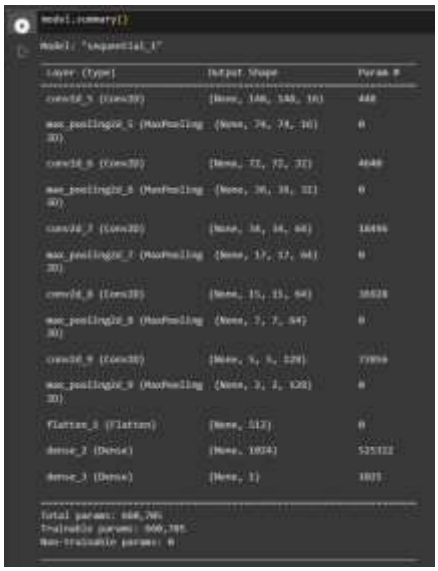

Figure 6. Summary of Model Structure

Models Training

At this stage, pre-compiled model training is carried out. Training is done using the 'fit()' method on the model object. In the parameters, there is train_generator as a training set that has been augmented before and testing_generator as a testing set.

The training process involves providing training data to the model repeatedly over several epochs or iterations. In this case, the model will be trained for 15 epochs through the training data. Each cycle or epoch consists of a number of steps that are determined based on the batch size used, in this case as many as 768 steps. At each step, the model will receive a certain amount of training data according to the batch size for processing and updating of model parameters.

Using the specified number of epochs, steps per epoch, and evaluation step, the model will be continuously updated based on the given training set, and then tested on the test set to see how well the model performs. This process repeats until it reaches a specified number of epochs or satisfies a predefined stopping condition.

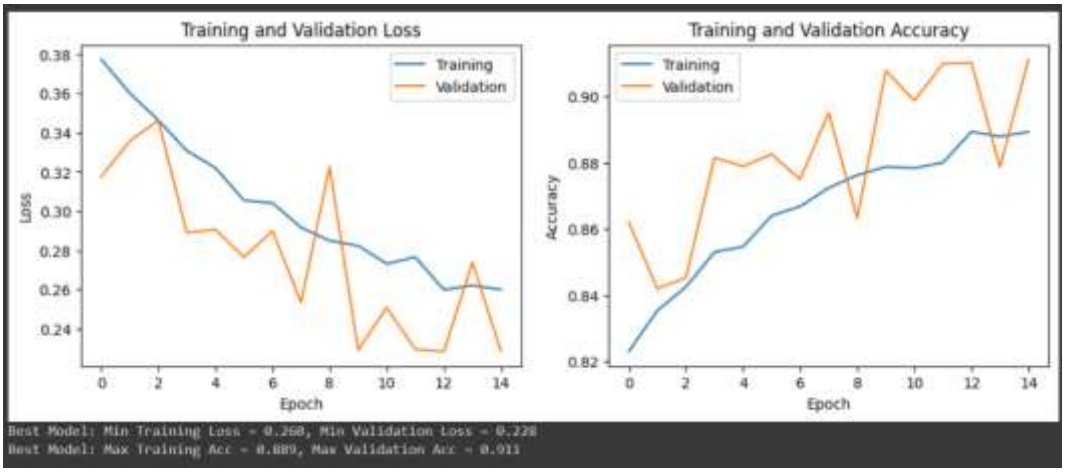

Figure 7. Graph loss and model accuracy

In Figure 7, it can be seen that there are two different curves, namely the loss model on the left and the accuracy of the model on the right. The blue line represents training for loss and accuracy, while the orange line is the result of validation for loss and accuracy. Based on the epoch journey in the figure, it can be concluded that the accuracy of the train model increases gradually from the first to the 15th epoch, reaching 88%. Meanwhile, the loss training model continued to decline from the first to the 15th epoch. On the other

hand, model validation accuracy fluctuated unstable with a tendency to increase up to 91%, while loss validation also fluctuated unstable but tended to decrease. This model does not experience overfitting or underfitting, which shows a very good performance in detecting gender based on facial image.

After finding a model with good performance, the next step is to save the model in h5 format so that it can be applied to the webcam.

```
model.save('model-pi.h5')
```

The purpose of the above program is to detect the sex of the face detected in the image frame taken from the webcam in real-time. To display the webcam, the program uses the open-cv library. In addition, the program also loads a pre-trained Convolutional Neural Network model for sex detection, uses the model to predict the gender of each detected face, and displays a gender label above the rectangle surrounding the face in the image frame displayed in the application window.

If the prediction results indicate that the detected face is male, then the frame surrounding the face and the gender label will be green. However, if the prediction results show that the detected face is female, then the frame and gender label will be pink.

CONCLUSION AND SUGGESTIONS

Conclusion

Based on the research that has been done, the author has succeeded in developing a simple application that uses a webcam as its basis. This application uses the Convolutional Neural Network algorithm and the Python programming language to identify a person's gender based on their face, with a high degree of accuracy. The results showed that the model built had a validation accuracy of 91% in predicting two classes, namely boys and girls. This program can work properly and displays the output via webcam.

Suggestion

Based on the research that has been done, there are several suggestions for this research as an effort to improve and develop or further research are as follows:

1. It is expected to improve accuracy for the better by collecting more training data or doing hyperparameter tuning to improve model performance.
2. It is hoped that further development will be carried out by trying to implement the application on several devices such as websites, mobile apps, and IOT (internet of things) devices to improve user experience.

REFERENCES

[1] Bronshtein, A. (2017, May 17). Train/Test Split And Cross Validation In Python.
[2] Brownlee, J. (2019). Deep Learning For Computer Vision: Image Classification, Object Detection, And Face Recognition In Python. Machine Learning Mastery.
[3] Budi, A., Suma'inna, & Maulana, H. (2016). Pengenalan Citra Wajah Sebagai Identifier Menggunakan Metode Principal Component Analysis (PCA). JURNAL TEKNIK INFORMATIKA UIN Syarif Hidayatullah, 9(2), 166-175. Doi:Https://Doi.Org/10.15408/Jti.V9i2.5608
[4] Cholissodin, I., & Soebroto, A. A. (2019). Ai, Machine Learning & Deep Learning (Teori & Implementasi). Malang: Fakultas Ilmu Komputer (Filkom), Universitas Brawijaya (Ub).
[5] Dewi, S. R. (2018). Deep Learning Object Detection Pada Video Menggunakan Tensorflow Dan Convolutional Neural Network. Yogyakarta.
[6] Fibrianto, A. S. (2016, April). Kesetaraan Gender Dalam Lingkup Organisasi Mahasiswa Universitas Sebelas Maret Surakarta Tahun 2016. Jurnal Analisa Sosiologi, 5(1), 10-27.
[7] Godoy, D. (2018, November 22). Understanding Binary Cross-Entropy / Log Loss: A Visual Explanation.
[8] Jamaaluddin, & Sulistyowati, I. (2021). Buku Ajar Kecerdasan Buatan (Artificial Intelligence). Sidoarjo: UMSIDA PRESS.
[9] Kumar, A. (2020). Great Mind Maps For Learning Machine Learning.
[10] Mahmud, M., Kaiser, M. S., Hussain, A., & Vassanelli, S. (2018). Applications Of Deep Learning And Reinforcement Learning To Biological Data. IEEE Transactions On Neural Networks And Learning System, 29(6), 2063-2079.
[11] Nugroho, P. A., Fenriana, I., & Arijanto, R. (2020). Implementasi Deep Learning Menggunakan Convolutional Neural Network (Cnn) Pada Ekspresi Manusia. Jurnal Algor, Vol. 2 No. 1 (2020): Data And System, 12-21.
[12] Nur Syahrudin, A., & Kurniawan, T. (2018). Input Dan Output Pada Bahasa Pemrograman Python. Jurnal Dasar Pemrograman Python STMIK.

[13] Nurfita, R. D., & Ariyanto, G. (2018). Implementasi Deep Learning Berbasis Tensorflow Untuk Pengenalan Sidik Jari. Jurnal Teknik Elektro, Universitas Muhammadiyah Surakarta, 18(1), 22-27.
[14] Prasetio, A., Hasibuan, M. H., & Sitompul, P. (2021). Implementasi Metode RELU, SIGMOID, MSE, Dan SGD Dalam Memprediksi Tingkat Pemahaman Siswa/I Pada Mata Pelajaran Pemrograman Di SMKS Mustafa Perbaungan. Jurnal Nasional Komputasi Dan Teknologi Informasi, Vol 4, No 3 (2021), 185-194.
[15] Putra, J. W. (2020). Pengenalan Konsep Pembelajaran Mesin Dan Deep Learning (1.4 Ed.).
[16] Qiang, W., & Zhongli, Z. (2011). Reinforcement Learning Model, Algorithms And Its Application. 2011 International Conference On Mechatronic Science, Electric Engineering And Computer (MEC), 1143-1146.
[17] Rahman, S. E. (2021). Convolutional Neural Network Untuk Visi Komputer Jaringan Saraf Konvolusional Untuk Visi Komputer (Arsitektur Baru, Transfer Learning, Fine Tuning, Dan Pruning). Sleman, Yogyakarta: Deepublish.
[18] Riadi, R. A. (2020, November 23). Mengenal Artificial Intelligence.
[19] Roihan, A., Sunarya, P. A., & Rafika, A. S. (2020). Pemanfaatan Machine Learning Dalam Berbagai Bidang. IJCIT (Indonesian Journal On Computer And Information Technology), 5(1), 75-82.
[20] Setiawan, W. (2020). Deep Learning Menggunakan Convolutional Neural Network: Teori Dan Aplikasi. Malang: Media Nusa Creative.
[21] Shahriar, N. (2023). What Is Convolutional Neural Network — CNN (Deep Learning).
[22] Shulur, S. P. (2015). Perancangan Aplikasi Deteksi Wajah Menggunakan Algoritma Viola-Jones.
[23] Soni Putra, N. (2023). Implementasi Algoritma Convolutional Neural Network Untuk Identifikasi Jenis Kelamin Dan Ras Pda Citra Wajah. Universitas Jambi, Fakultas Sains Dan Teknologi, Jambi.