# MACHINE LEARNING PENDETEKSI WAJAH DENGAN OPEN CV MENGGUNAKAN PYTHON

**Sunarsan Sitohang[1,*], Hotma Pangaribuan[2]**
[1,2]Teknik Informatika, Teknik dan Komputer, Putera Batam, Kepulauan Riau
*sunarsan@puterabatam.ac.id

**ABSTRAK**

Machine learning memungkinkan komputer untuk mengidentifikasi pola, membuat prediksi, atau mengambil keputusan berdasarkan data yang diberikan. Dalam essence, machine learning memberikan kemampuan komputer untuk belajar dari pengalaman atau data, dan menggunakan pengetahuan ini untuk menghadapi situasi baru atau tugas yang serupa. Proses ini dilakukan melalui proses konvolusi, dimana sebuah filter digeser ke seluruh area gambar untuk mengenali pola yang ada pada gambar tersebut. Teknologi deteksi wajah dan pembuatan absensi telah menjadi topik yang menarik dalam dunia komputasi visi dan keamanan. Dalam era digital saat ini, penggunaan sistem otomatis untuk mengenali wajah dan memantau kehadiran menjadi semakin penting dalam berbagai bidang, termasuk pendidikan, bisnis, dan keamanan. Penelitian pengembangan sistem deteksi wajah dan pembuatan absensi menggunakan bahasa pemrograman Python dan OpenCV. Python merupakan bahasa pemrograman yang populer dan memiliki banyak library yang mendukung pengembangan aplikasi visi komputer, sedangkan OpenCV menyediakan berbagai fungsi dan algoritma yang berguna dalam analisis dan pemrosesan citra. Sistem deteksi wajah dan pembuatan absensi yang dibangun menggunakan Python dan OpenCV memiliki potensi besar dalam meningkatkan efisiensi dan akurasi proses absensi, serta meminimalkan penyalahgunaan atau kecurangan. Oleh karena itu, dalam implementasinya, perlu mempertimbangkan kebijakan privasi dan perlindungan data yang sesuai agar penggunaan teknologi ini tetap dijalankan dengan etika dan kepatuhan hukum yang berlaku. Dengan demikian, melalui penelitian ini, diharapkan pembaca dapat memperoleh pemahaman yang lebih baik tentang deteksi wajah.

**Kata kunci:** Deteksi Wajah, Machine Learning, Open CV, Python.

***ABSTRACT***

*Machine learning is a branch of artificial intelligence that enables computers to identify patterns, make predictions, or take actions based on provided data. In essence, machine learning grants computers the ability to "learn" from experience or data and use this knowledge to handle new situations or similar tasks. This process is carried out through convolution, where a filter is slid across the entire area of an image to recognize existing patterns. Face detection and attendance systems have become compelling topics in the fields of computer vision and security. In today's digital era, the use of automated systems to recognize faces and monitor attendance is increasingly vital across various sectors, including education, business, and security. Research on the development of face detection and attendance systems utilizes the Python programming language and OpenCV (Open Source Computer Vision Library). Python is a popular programming language with numerous libraries supporting computer vision development, while OpenCV provides various functions and algorithms useful for image analysis and processing. Face detection and attendance systems built with Python and OpenCV hold great potential for increasing efficiency and accuracy in the attendance process, as well as minimizing misuse or fraud. Therefore, in its implementation, it is essential to consider appropriate privacy policies and data protection measures to ensure the technology is used ethically and in compliance with applicable laws. Through this research, it is expected that readers will gain a better understanding of face detection.*

***Keywords***: *Face Detection, Machine Learning, OpenCV, Python.*

## 1. PENDAHULUAN

### 1.1 Latar Belakang

Machine learning adalah cabang dari kecerdasan buatan yang memungkinkan komputer untuk belajar dari data tanpa perlu secara eksplisit diprogram. Dengan menggunakan algoritma dan model komputer yang dapat beradaptasi, machine learning memungkinkan komputer untuk mengidentifikasi pola, membuat prediksi, atau mengambil keputusan berdasarkan data yang diberikan. Dalam essence, machine learning memberikan kemampuan komputer untuk "belajar" dari pengalaman atau data, dan menggunakan pengetahuan ini untuk menghadapi situasi baru atau tugas yang serupa. Sistem biometrika mampu memutuskan apakah hasil pengenalan itu sah atau tidak sah, diterima atau ditolak, dikenali atau tidak dikenali. Sistem biometri yang mulai dikembangkan adalah *fingerprint*, *hand recognition*, *face recognition*, *retina scanning*, dan *DNA scanning*. Salah satu sistem biometri yang banyak digunakan untuk sistem kehadiran adalah *face recognition* (pengenalan wajah) yang banyak digunakan untuk identifikasi personal pada pengguna mesin, akses kontrol, keamanan dan lain-lain.

Dalam konteks absensi atau presensi kehadiran, teknologi *face recognition* memungkinkan untuk secara otomatis mengumpulkan data presensi kehadiran, termasuk jam masuk dan jam keluar, dengan menggunakan pengenalan wajah. Teknologi ini dapat membantu instansi yang membutuhkan untuk meningkatkan efisiensi dan akurasi proses absensi, serta mengurangi penipuan dan manipulasi data absensi yang dilakukan oleh oknum.

Penggunaan teknologi *face recognition* dalam presensi kehadiran memiliki beberapa keuntungan yang dapat membantu meningkatkan efisiensi dan akurasi proses absensi (Hajar et al., 2021). Pertama, teknologi ini dapat mengotomatisasi proses absensi, sehingga waktu yang dibutuhkan untuk melakukan absensi dapat lebih efisien dan memungkinkan pengumpulan data absensi yang lebih akurat. Kedua, teknologi *face recognition* dapat membantu mengurangi kecurangan absensi, seperti ada oknum yang membawa kartu absensi orang lain, menitipkan absensi dan lain sebagainya. memberikan tanda tangan palsu pada lembar absensi. Ketiga, teknologi ini dapat membantu meningkatkan keamanan data karena data yang dihasilkan lebih sulit untuk dimanipulasi atau diretas. Keempat, teknologi *face recognition* juga memungkinkan pengumpulan data kehadiran mahasiswa secara real-time, sehingga dapat memberikan informasi yang akurat terkait kehadiran mahasiswa di kelas. Namun, dalam mengimplementasikan teknologi ini, perlu

diperhatikan aspek legal, etis, dan privasi. Evaluasi dan pengkajian harus dilakukan sebelum menggunakan teknologi ini, serta perlu disosialisasikan alternatif lain bagi mahasiswa yang tidak ingin menggunakan teknologi ini.

Penggunaan Bahasa pemograman pada kasus ini adalah Bahasa program python, Bahasa python pertama kali ditemukan pada tahun 1989 oleh guido van Rossum dan termasuk Bahasa programan yang mudah dimengerti dan di operasikan (Putra Prakoso et al., 2024). Pada saat membuat programan absensi menggunakan Bahasa python harus dibantu *extention* yaitu "Open CV" untuk memindahkan wajah. Open CV terhubung langsung ke kamera laptop atau *webcam*. Maka dari itu kita harus menginstall open CV pada laptop kita sebagai *extension* tambahan dari Bahasa program python.

Penelitian ini bertujuan untuk memperkenalkan teknologi terbaru dalam bidang pengolahan citra dan pengenalan pola, serta meningkatkan efisiensi dan efektivitas dalam mengidentifikasi wajah. Dalam penelitian ini akan menggunakan library OpenCV sebagai salah satu komponen utamanya. Berikut merupakan point-point tujuan penelitian: Untuk mengetahui efektivitas teknologi face recognition dalam meningkatkan efisiensi dan akurasi proses presensi kehadiran dan untuk menerapkan Open CV menggunakan Python untuk presensi kehadiran

## 1.2 Landasan Teori

### Python

Python merupakan bahasa pemrograman tingkat tinggi (*high-level programming language*) yang dikembangkan oleh Guido van Rossum dan pertama kali dirilis pada tahun 1991. Python dirancang dengan sintaks yang sederhana dan mudah dipahami sehingga mendukung pengembangan perangkat lunak secara cepat serta meningkatkan produktivitas programmer. Python bersifat *interpreted*, *object-oriented*, dan mendukung berbagai paradigma pemrograman seperti pemrograman prosedural, berorientasi objek, dan fungsional.

Python banyak digunakan dalam berbagai bidang, seperti pengembangan aplikasi web, analisis data, kecerdasan buatan (*Artificial Intelligence*), pembelajaran mesin (*Machine Learning*), otomatisasi sistem, komputasi ilmiah, hingga keamanan siber. Popularitas Python didukung oleh tersedianya banyak pustaka (*library*) dan kerangka kerja (*framework*) yang memudahkan pengembangan aplikasi sesuai kebutuhan. Keunggulan utama Python meliputi kemudahan pembelajaran, keterbacaan kode yang tinggi, portabilitas lintas platform, serta dukungan komunitas yang besar. Selain itu, Python menyediakan berbagai modul bawaan dan pustaka eksternal yang dapat digunakan untuk

mempercepat proses pengembangan perangkat lunak. Dalam bidang kecerdasan buatan dan analisis data, Python sering dipadukan dengan pustaka seperti NumPy, Pandas, TensorFlow, dan OpenCV untuk mendukung pengolahan data dan pengembangan model komputasi (Matthes, 2023).

Menurut dokumentasi resmi Python, bahasa pemrograman ini menekankan pada keterbacaan kode (*code readability*) dengan penggunaan sintaks yang jelas dan sederhana sehingga memungkinkan programmer untuk mengekspresikan konsep pemrograman dengan lebih sedikit baris kode dibandingkan bahasa pemrograman lainnya. Python juga mendukung pengelolaan memori secara otomatis melalui mekanisme *garbage collection* sehingga memudahkan pengembangan aplikasi yang kompleks (Python Software Foundation, 2026).

**Face Recognation**

*Face recognition* merupakan teknologi biometrik yang digunakan untuk mengidentifikasi atau memverifikasi identitas seseorang berdasarkan karakteristik unik wajah. Proses pengenalan wajah umumnya terdiri dari beberapa tahapan, yaitu deteksi wajah (face detection), normalisasi citra, ekstraksi fitur (feature extraction), dan pencocokan wajah (face matching). Pada tahap deteksi wajah, sistem mengidentifikasi keberadaan wajah dalam citra atau video menggunakan metode seperti Haar Cascade, *Histogram of Oriented Gradients* (HOG), maupun pendekatan deep learning. Selanjutnya dilakukan normalisasi untuk mengurangi pengaruh variasi pencahayaan, ukuran, dan orientasi wajah. Tahap ekstraksi fitur bertujuan memperoleh ciri-ciri penting wajah menggunakan metode seperti Eigenfaces berbasis Principal Component Analysis (PCA), Fisherfaces berbasis *Linear Discriminant Analysis* (LDA), *Local Binary Pattern* (LBP), maupun representasi fitur berbasis *Convolutional Neural Network* (CNN). Hasil ekstraksi fitur kemudian dibandingkan dengan data yang tersimpan dalam basis data menggunakan teknik pengukuran kemiripan seperti Euclidean *Distance* atau *Cosine Similarity* untuk menentukan identitas seseorang (Wang & Deng, 2021).

Perkembangan teknologi *deep learning* telah meningkatkan kinerja sistem pengenalan wajah secara signifikan dibandingkan metode tradisional. Model berbasis CNN mampu mempelajari representasi wajah secara otomatis dan menghasilkan tingkat akurasi yang tinggi meskipun terdapat variasi pose, ekspresi, maupun pencahayaan. Beberapa model yang banyak digunakan dalam *face recognition* modern adalah *FaceNet*, *VGGFace*, dan *ArcFace* yang memanfaatkan teknik embedding untuk merepresentasikan wajah dalam ruang vektor berdimensi rendah. Pendekatan deep learning terbukti lebih unggul dibandingkan metode klasik seperti PCA,

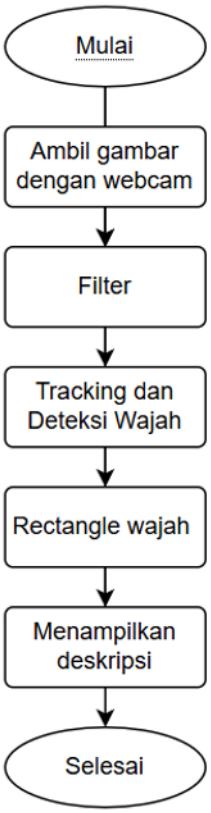
LDA, dan LBP dalam hal akurasi dan ketahanan terhadap kondisi lingkungan yang kompleks, sehingga banyak diterapkan pada sistem keamanan, kontrol akses, absensi, dan berbagai aplikasi biometrik lainnya (Deng et al., 2022), (Tolegenova & Moldagulova, 2025).

Beberapa penelitian terkait topik *face recognation* diantaaranya: Open CV, Tensorflow dan Keras dimanfaatkan untuk mendeteksi pemakaian masker dikala pandemi covid-19 (Tiku et al., 2022) (Syahrial et al., 2023). Open CV dan algoritma Haarcascade dimanfaatkan untuk deteksi wajah (Putra Prakoso et al., 2024) dengan menggunakan pendekatakan CRM (Karunia Rahmadhika & Thantawi, 2021).

## 2. METODE PELAKSANAAN

Desain penelitian menjadi tahap awal dan tahap yang penting dalam proses penelitian. Penyusunan desain penelitian adalah tahap penelitian yang biasanya disusun dan mampu membuat gagasan atau rencana dalam proses penelitian secara mudah. Secara menyeluruh, desain penelitian adalah semua proses yang digunakan dalam merencanakan dan melakukan sebuah penelitian. Didalam hal ini unsur desain dapat meliputi semua struktur penelitian yang diawali saat menemukan ide, tujuan kemudian membuat konsep penelitian, permasalahan, perumusan, sumber informasi yang menjadi kajian pustaka.

Desain penelitian yang saling berhubungan membuat penelitian mudah dipahami seperti desain hubungan antar variabel, *collection* data dan *analysis* data, sehingga dengan adanya desain penelitian yang baik peneliti dan pihak yang terlibat serta orang yang membaca mempunyai gambaran yang jelas dan terarah tentang hubungan variabel yang ada dalam konteks penelitian yang akan dilakukan oleh seorang peneliti dalam melakukan penelitian.




**Gambar 1** Desain Penelitian

### 2.1 Operasional Variabel

Operasional variable adalah suatu cara atau unsur yang membantu komunikasi Variabel Yang digunakan dalam penggunaan algoritma YOLO ialah jenis-jenis sayuran Yang sering di jumpai di

pasar. Namun, untuk menentukan objek bentuk sayuran, variabel-variabel ini perlu disesuaikan dengan fitur-fitur khusus yang dimiliki oleh bentuk sayuran tersebut. Misalnya, untuk mengenali bentuk wortel, variabel-variabel YOLO perlu disesuaikan dengan fitur-fitur seperti panjang, lebar, dan bentuk ujung wortel.Oleh karena itu, selain variabel YOLO, dataset yang digunakan untuk melatih YOLO juga harus mencakup berbagai bentuk sayuran dengan fitur-fitur yang berbeda-beda.

### 2.2 Teknik Pengumpulan Data

Data dikumpulkan oleh peneliti menggunakan cara mengambil referensi data dari Kaggle, roboflow, dan ultralytic dari berbagai macam sudut dan bentuk.

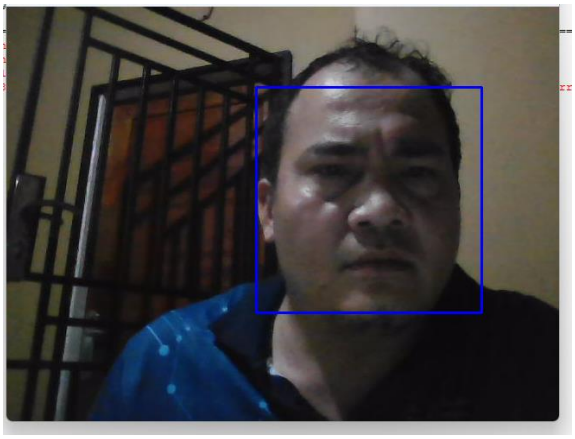
## 3. HASIL DAN PEMBAHASAN

Pada penelitian ini, langkah pertama yang dilakukan adalah menjalankan program deteksi wajah terlebih dahulu, dengan kode perintah yang tertera pada

```
import cv2

# Load model deteksi wajah
face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

# Buka kamera
cap = cv2.VideoCapture(0)

while True:
    # Baca frame dari kamera
    ret, frame = cap.read()

    # Ubah ke grayscale (model deteksi wajah bekerja lebih baik di grayscale)
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    # Deteksi wajah
    faces = face_cascade.detectMultiScale(gray, scaleFactor=1.3, minNeighbors=5)

    # Gambar kotak di sekitar wajah yang terdeteksi
    for (x, y, w, h) in faces:
        cv2.rectangle(frame, (x, y), (x + w, y + h), (255, 0, 0), 2)

    # Tampilkan hasil
    cv2.imshow('Deteksi Wajah', frame)

    # Tekan 'q' untuk keluar
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

# Bersihkan setelah selesai
cap.release()
cv2.destroyAllWindows()
```

gambar 2 dengan hasil program pada gambar 3 dibawah ini.

**Gambar 2.** Kode Program Deteksi Wajah



**Gambar 3.** Hasil Deteksi Wajah

Setelah melakukan deteksi wajah, langkah selanjutnya adalah mengumpulkan data wajah dengan kode program pada gambar 4 dan hasilnya adalah pada gambar 5 dibawah ini.

```
Kumpul_Data_Wajah.py - C:\Penelitian Python\Kumpul_Data_Wajah.py (3.10.11)
File  Edit  Format  Run  Options  Window  Help
import cv2
import os

# Masukkan ID pengguna
user_id = input('Masukkan ID pengguna: ')
folder = f'data/user_{user_id}'

# Buat folder jika belum ada
os.makedirs(folder, exist_ok=True)

# Load Haar cascade
face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

# Akses webcam
cap = cv2.VideoCapture(0)

count = 0
while count < 50:  # Ambil 50 gambar wajah
    ret, frame = cap.read()
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    faces = face_cascade.detectMultiScale(gray, 1.3, 5)

    for (x, y, w, h) in faces:
        count += 1
        face_img = gray[y:y+h, x:x+w]
        cv2.imwrite(f"{folder}/{count}.jpg", face_img)
        cv2.rectangle(frame, (x, y), (x+w, y+h), (255, 0, 0), 2)

    cv2.imshow('Mengambil Gambar Wajah', frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()
```

**Gambar 4.** Kode Program Kumpulkan Data Wajah

```
IDLE Shell 3.10.11
File  Edit  Shell  Debug  Options  Window  Help
Python 3.10.11 (tags/v3.10.11:7d4cc5a, Apr  5 2023, 00:38:17) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:\Penelitian Python\Kumpul_Data_Wajah.py ==========
Masukkan ID pengguna: 1
>>>
```

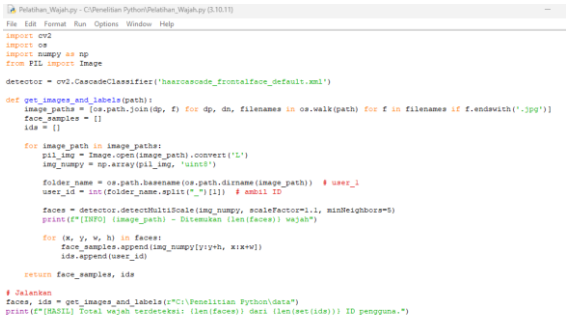
**Gambar 5.** Hasil Eksekusi Program

Dari kode perintah pada gambar 4 diatas, jika kita jalankan maka akan memerintahkan untuk memasukkan id pengguna seperti yang tertera pada gambar 5 diatas. Saat id pengguna diisi dengan sembarang nomor dengan digit tertentu dan ditekan enter maka akan menghasilkan 50 gambar tangkapan yang berbasis gambar keabuan seperti gambar 6 dibawah ini.

**Gambar 6.** Kumpulan Data Wajah

Data gambar tangkapan menggunakan 4 id yaitu user_1, user_12, user_123 dan user_1234, sehingga data gambar yang tersimpan yaitu sebanyak 200 gambar.

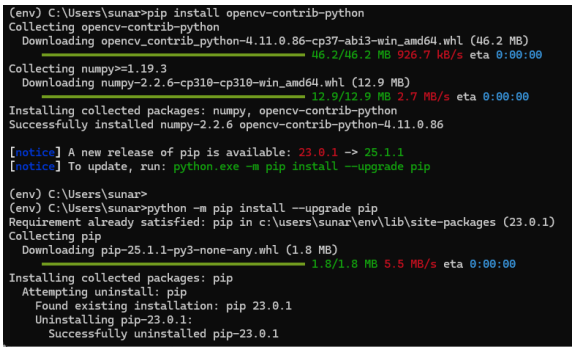
Setelah data wajah sudah tercapture, langkah berikutnya adalah melatih model pengenalah wajah dengan menggunakan kode perintah pada gambar 7 dibawah ini



```
import cv2
import os
import numpy as np
from PIL import Image

detector = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

def get_images_and_labels(path):
    image_paths = [os.path.join(dp, f) for dp, dn, filenames in os.walk(path) for f in filenames if f.endswith('.jpg')]
    face_samples = []
    ids = []

    for image_path in image_paths:
        pil_img = Image.open(image_path).convert('L')
        img_numpy = np.array(pil_img, 'uint8')

        folder_name = os.path.basename(os.path.dirname(image_path))  # user_1
        user_id = int(folder_name.split("_")[1])  # ambil ID

        faces = detector.detectMultiScale(img_numpy, scaleFactor=1.1, minNeighbors=5)
        print(f"[INFO] {image_path} - Ditemukan {len(faces)} wajah")

        for (x, y, w, h) in faces:
            face_samples.append(img_numpy[y:y+h, x:x+w])
            ids.append(user_id)

    return face_samples, ids

# Jalankan
faces, ids = get_images_and_labels(r"C:\Penelitian Python\data")
print(f"[HASIL] Total wajah terdeteksi: {len(faces)} dari {len(set(ids))} ID pengguna.")
```

**Gambar 7.** Koding Pelatihan Wajah

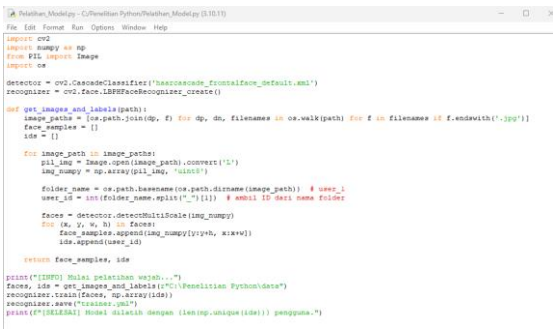
Sebelum melakukan pelatihan kita wajib instal terlebih dahulu opencv, numpy, dan pillow seperti gambar 8 dibawah ini.



```
(env) C:\Users\sunar>pip install opencv-contrib-python
Collecting opencv-contrib-python
  Downloading opencv_contrib_python-4.11.0.86-cp37-abi3-win_amd64.whl (46.2 MB)
     ---------------------------------------- 46.2/46.2 MB 926.7 kB/s eta 0:00:00
Collecting numpy>=1.19.3
  Downloading numpy-2.2.6-cp310-cp310-win_amd64.whl (12.9 MB)
     ---------------------------------------- 12.9/12.9 MB 2.7 MB/s eta 0:00:00
Installing collected packages: numpy, opencv-contrib-python
Successfully installed numpy-2.2.6 opencv-contrib-python-4.11.0.86

[notice] A new release of pip is available: 23.0.1 -> 25.1.1
[notice] To update, run: python.exe -m pip install --upgrade pip

(env) C:\Users\sunar>
(env) C:\Users\sunar>python -m pip install --upgrade pip
Requirement already satisfied: pip in c:\users\sunar\env\lib\site-packages (23.0.1)
Collecting pip
  Downloading pip-25.1.1-py3-none-any.whl (1.8 MB)
     ---------------------------------------- 1.8/1.8 MB 5.5 MB/s eta 0:00:00
Installing collected packages: pip
  Attempting uninstall: pip
    Found existing installation: pip 23.0.1
    Uninstalling pip-23.0.1:
      Successfully uninstalled pip-23.0.1
```

**Gambar 8.** Instal Library Yang Dibutuhkan

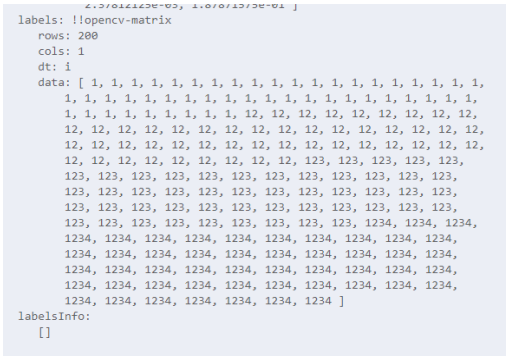
Setelah pelatihan wajah, dilanjutkan dengan pelatihan model dengan penggunaan koding pada gambar 9 dibawah ini.



```
import cv2
import numpy as np
from PIL import Image
import os

detector = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
recognizer = cv2.face.LBPHFaceRecognizer_create()

def get_images_and_labels(path):
    image_paths = [os.path.join(dp, f) for dp, dn, filenames in os.walk(path) for f in filenames if f.endswith('.jpg')]
    face_samples = []
    ids = []

    for image_path in image_paths:
        pil_img = Image.open(image_path).convert('L')
        img_numpy = np.array(pil_img, 'uint8')

        folder_name = os.path.basename(os.path.dirname(image_path))  # user_1
        user_id = int(folder_name.split("_")[1])  # ambil ID dari nama folder

        faces = detector.detectMultiScale(img_numpy)
        for (x, y, w, h) in faces:
            face_samples.append(img_numpy[y:y+h, x:x+w])
            ids.append(user_id)

    return face_samples, ids

print("[INFO] Mulai pelatihan wajah...")
faces, ids = get_images_and_labels(r"C:\Penelitian Python\data")
recognizer.train(faces, np.array(ids))
recognizer.save("trainer.yml")
print(f"[SELESAI] Model dilatih dengan {len(np.unique(ids))} pengguna.")
```

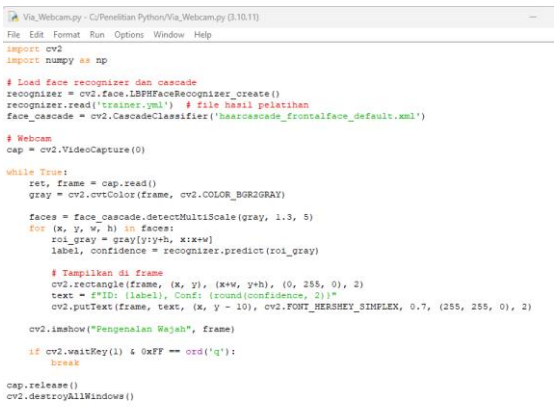
**Gambar 9.** Pelatihan Model

Kode perintah pada gambar 9 diatas akan menghasilkan hasil seperti gambar 10 dibawah ini.

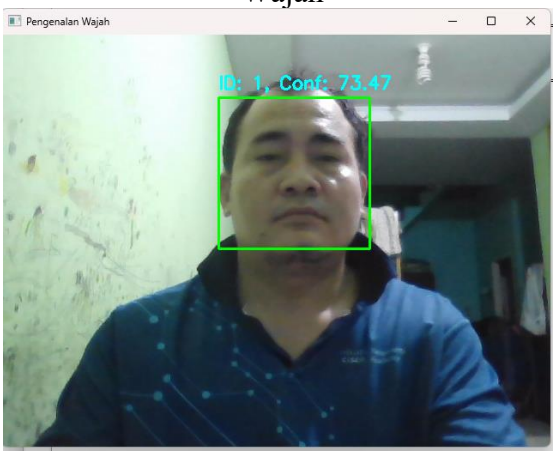



**Gambar 10.** Hasil Pelatihan Model

Setelah model dilatih selanjutnya kita uji, apakah model mengenali wajah yang dideteksi, maka kita tambahkan *confidence* dan ini didapat dari kode program pada gambar 11 dan hasilnya pada gambar 12 dibawah ini.




**Gambar 11.** Kode Program Pendeteksi Wajah




**Gambar 12.** Hasil Deteksi Wajah

Berdasarkan hasil deteksi maka wajah berhasil dideteksi dengan nilai konfiden 73,47% dan dapat dikatakan model yang dibangun akurasinya sudah baik dan dapat diterapkan.

Penelitian ini berhasil mengimplementasikan sistem pendeteksi wajah menggunakan pustaka OpenCV dengan algoritma deteksi berbasis Haar *Cascade Classifier*. Sistem dikembangkan menggunakan bahasa pemrograman Python dan diuji menggunakan dataset gambar wajah dari kamera secara real-time serta dataset gambar statis. Berikut tabel 1 hasil dari pengujian yang dilakukan:

**Tabel 1**Hasil Pengujian

| No | Skenario Uji | Qty | AD (%) | FP | FN |
|---|---|---|---|---|---|
| 1 | Gambar wajah frontal (1 orang) | 50 | 96% | 2 | 0 |
| 2 | Gambar dengan banyak wajah | 30 | 91% | 1 | 3 |
| 3 | Gambar wajah tidak frontal | 30 | 85% | 3 | 5 |
| 4 | Real-time webcam | 100 frame | 94% | 4 | 2 |

Hasil pengujia1 pada tabel 1 di atas menunjukkan bahwa sistem cukup efektif dalam mendeteksi wajah terutama dalam kondisi pencahayaan baik dan sudut wajah frontal. Berikut adalah hasil visual deteksi wajah pada beberapa skenario:

1. Deteksi wajah tunggal dengan pencahayaan baik berhasil dengan akurasi tinggi.

2. Deteksi pada kondisi ramai (multi wajah) masih cukup baik meskipun terdapat beberapa wajah yang tidak terdeteksi.
3. Deteksi wajah menyamping atau miring mengalami penurunan akurasi.
4. Dalam kondisi gelap atau pencahayaan rendah, akurasi menurun signifikan.

3.1 Pembahasan

Sistem pendeteksi wajah yang dibangun menggunakan OpenCV dengan Haar Cascade menunjukkan performa yang memadai untuk penggunaan dasar seperti:

1. Pengenalan wajah secara real-time
2. Sistem absensi otomatis
3. Aplikasi keamanan dasar

Namun, sistem memiliki beberapa keterbatasan:

1. Sensitif terhadap Pencahayaan, akurasi deteksi menurun drastis saat pencahayaan buruk. *Haar Cascade* bekerja optimal dalam kondisi terang.
2. Sudut Wajah dan Orientasi, wajah yang tidak frontal (misalnya menyamping) sulit dideteksi karena Haar Cascade dilatih untuk wajah frontal.
3. Cepat dan ringan namun terbatas, *haar cascade* sangat cepat, cocok untuk aplikasi real-time, namun tidak sekuat metode modern seperti *CNN-based detectors* (misalnya MTCNN, SSD, YOLO) dalam hal akurasi dan fleksibilitas.

1. False Detection

Kadang sistem mendeteksi objek non-wajah sebagai wajah (false positive), khususnya dalam gambar dengan banyak tekstur atau objek kompleks.

Sebagai solusi pengembangan ke depan, beberapa pendekatan yang bisa digunakan adalah:

1. Mengganti Haar Cascade dengan HOG + SVM, Dlib, atau model deep learning seperti MTCNN, FaceNet, atau YOLO face detector.
2. Menambahkan preprocessing seperti normalisasi histogram untuk meningkatkan deteksi dalam pencahayaan buruk.
3. Melakukan augmentasi data untuk memperkuat model terhadap berbagai kondisi orientasi wajah.

## 4. KESIMPULAN

Berdasarkan analisis data wajah dan pengujian atas kinerja dapat disimpulkan bahwa sistem pendeteksi wajah menggunakan pustaka OpenCV dengan algoritma deteksi berbasis Haar *Cascade Classifier* dapat bekerja dengan baik adapun kelebihan dari model yang dibangun ini adalah system dapat mendeteksi secara real-time, dapat digunakan langsung via webcam, ringan karena menggunakan metode deteksi sederhana. Modular karena mudah ditambahkan pengguna baru dan retrain model. Ditinjau dari kelemahan dari model sistem ini yaitu bergantung pada kualitas kamera dan pencahayaan. Haar Cascade deteksi wajah kurang akurat untuk wajah

miring atau sebagian tertutup. Model LBPH bisa menurun akurasinya jika wajah pengguna mirip atau kondisi berubah drastis (misal: memakai masker, pencahayaan ekstrem).

## DAFTAR PUSTAKA

Deng, J., Guo, J., Yang, J., Xue, N., Kotsia, I., & Zafeiriou, S. (2022). ArcFace: Additive Angular Margin Loss for Deep Face Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *44*(10), 5962–5979. https://doi.org/10.1109/TPAMI.2021.3087709

Hajar, R. R., Sejati, P., Mardhiyyah, R., Yogyakarta, T., Ring, J. S., Utara, R., Lor, J., & Yogyakarta, S. (2021). Deteksi Wajah Berbasis Facial Landmark Menggunakan Opencv Dan DLIB. *Jurnal Teknologi Informasi)*, *5*(2).

Karunia Rahmadhika, M., & Thantawi, A. M. (2021). Rancang Bangun Aplikasi Face Recognition Pada Pendekatan CRM Menggunakan Opencv Dan Algoritma Haarcascade. *Jurnal IKRA-ITH Informatika*, *5*(1), 109–118.

Matthes, E. (2023). *Python Crash Course: A Hands-On, Project-Based Introduction to Programming* (3rd ed.). No Starch Press.

Putra Prakoso, A., Rasyid, A., Deannova, A., & Rahmawan, A. E. (2024). *Deteksi Wajah Menggunakan Cascade Classifier Dengan Opencv-Python*. *2*(1), 23–29.

Python Software Foundation. (2026). *Python Software Foundation*. https://docs.python.org/3/

Syahrial, R., Sukmawati, T., & Dewi, E. N. (2023). Face Mask Detection Menggunakan Python Dan Opencv Untuk Mendeteksi Pelanggaran Protokol Kesehatan Covid-19. *Jurnal Elektro & Informatika Swadharma (JEIS)*, *3*(1), 77–86.

Tiku, J. C., Saputra, W. A., & Prasetyo, N. A. (2022). Pengembangan Sistem Deteksi Memakai Masker Menggunakan Open CV, Tensorflow dan Keras. *JURIKOM (Jurnal Riset Komputer)*, *9*(4), 1183. https://doi.org/10.30865/jurikom.v9i4.4739

Tolegenova, D., & Moldagulova, A. (2025). Comparative Analysis of Face Biometric Identification Methods: From Traditional Algorithms to Modern Deep Learning Models. *Computing & Engineering*, *3*, 34–40. https://doi.org/10.51301/ce.2025.i2.06

Wang, M., & Deng, W. (2021). Deep face recognition: A survey. *Neurocomputing*, *429*, 215–244. https://doi.org/10.1016/j.neucom.2020.10.081