Jurnal Sistim Informasi dan Teknologi
https://jsisfotek.org/index.php
2022 Vol. 4 No. 4 Hal: 149-154 e-ISSN: 2686-3154

Identifikasi Masker pada Face Detection dengan Menggunakan Metode Haar Cascade dan CNN

Vebri Dores[1]✉
[1] Independent Researcher
responsiblevebri@gmail.com

Abstract

The World Health Organization (WHO) provides standard recommendations for the prevention and spread of COVID-19 with the importance of face masks for protection from the virus. Technology can be used to help prevent viruses from spreading quickly through direct contact. Utilizing Internet of Things (IoT) based technology tools for public place entrances can be designed to perform rapid inspections. This can be done using image classification methods, namely Haar Cascade and Convolutional Neural Network (CNN). This study aims to detect masks on the face using the python programming language and input data via a webcam camera. A total of 500 datasets of photos of faces without masks and 500 datasets of photos of faces with masks were used as training data. The test results using the Haar Cascade method have an accuracy of 80% while using CNN the accuracy is 100%. So that this study becomes a reference in identifying obedience in wearing masks.

Keywords: Face Detection, Mask, Face Photo, Convolutional Neural Network (CNN), Python.

Abstrak

World Health Organization (WHO) memberikan rekomendasi standar pencegahan dan penyebaran COVID-19 dengan pentingnya masker wajah untuk perlindungan dari virus. Teknologi dapat digunakan untuk membantu mencegah virus menyebar dengan cepat yang melalui kontak langsung. Memanfaatkan alat teknologi berbasis Internet of Things (IoT) untuk pintu masuk tempat umum dapat dirancang untuk melakukan pemerikasaan yang cepat. Hal ini dapat dilakukan dengan menggunakan metode klasifikasi citra yaitu Haar Cascade dan Convolutional Neural Network (CNN). Penelitian ini bertujuan mendeteksi masker pada wajah dengan menggunakan bahasa pemograman python dan *input* data melalui kamera webcam. Sebanyak 500 *dataset* foto wajah tanpa masker dan 500 *dataset* foto wajah dengan masker digunakan sebagai data *training*. Hasil pengujian penggunakan metode *Haar Cascade* memiliki akurasi sebesar 80% sedangkan menggunakan CNN akurasinya sebesar 100%. Sehingga penelitian ini menjadi rujukan dalam mengidentifikasi ketaatan dalam memekai masker.

Kata kunci: Deteksi Wajah, Masker, Foto Wajah, Convolutional Neural Network (CNN), Python.

1. Pendahuluan

Transformasi digital di bidang kesehatan sudah banyak dilakukan baik yang bersifat klinis maupun non klinis. Teknologi dapat digunakan untuk mengurangi kontak langsung karena virus dapat menyebar dengan cepat melalui kontak langsung. Mengenai pencegahan penyebaran COVID-19, *World Health Organization* (WHO) memberikan rekomendasi standar pencegahan dan pentingnya masker wajah untuk perlindungan dari virus [1]. Penggunaan sistem desinfeksi manual yang berlebihan juga menjadi sumber infeksi. Oleh karena itu, merancang dan mengembangkan sistem pengendalian dan skrining penyebaran virus yang murah, cepat, terukur, dan efektif untuk meminimalkan peluang dan risiko penyebaran COVID-19 sangat penting dilakukan. Memanfaatkan alat teknologi berbasis *Internet of Things* (IoT) untuk semua pintu masuk tempat umum dapat dirancang untuk melakukan pemerikasaan yang cepat, termasuk pengukuran suhu menggunakan sensor bebas kontak dan menyimpan catatan individu yang dicurigai. Sistem pemeriksaan berbasis IoT dapat juga diterapkan untuk model *deep learning* deteksi dan klasifikasi masker wajah [2].

Berdasarkan fakta tersebut maka perlu dilakukan pengawasan kepada masyarakat dalam menjalankan protokol kesehatan COVID-19. Misalnya membuat alat yang dapat mendeteksi secara otomatis menggunakan sistem komputer terhadap orang yang tidak menggunakan masker. Solusi yang dapat diberikan pengawasan yang lebih baik dengan memanfaatkan teknologi *Computer Vision*.

Ilmu yang berhubungan erat dengan *object detection* adalah computer vision [3]. Computer Vision umumnya membahas tentang klasifikasi objek dan deteksi objek pada gambar [4]. Computer Vision yang digunakan umumnya terdari dari pengumpulan

Diterima: 28-08-2022 | Revisi: 30-09-2022 | Diterbitkan: 30-12-2022 | doi: 10.37034/jsisfotek.v4i4.154

gambar, *pre-processing*, *training* dan *deploy* [5]. Setelah semua proses tersebut dilalui maka akan dihasilkan model yang akan digunakan untuk klasifikasi atau deteksi gambar yang baru [6]. Pembuatan model didalam computer visionbiasanya didasari pada metode Convolutional Neural Network(CNN) [7].

Banyak metode telah dibuat dan dikembangkan untuk melakukan deteksi wajah. Dua metode populer yang digunakan saat ini adalah *Haar Cascade* dan *Convolutional Neural Network* (CNN). *Haar Cascade* adalah algoritma yang digunakan untuk mendeteksi wajah secara cepat dan *real time*. Sedangkan CNN memanfaatkan proses konvolusi dengan memindahkan kernel konvolusi (*filter*) dengan ukuran tertentu ke gambar berikutnya dari hasil perkalian gambar dengan *filter* yang digunakan [8].

Metode CNN dapat digunakan untuk mengenali dan mendeteksi sebuah objek pada sebuah citra digital. Algoritma CNN sudah banyak digunakan peneliti dan disebut sebagai algoritma terbaik untuk *object detection* dan *object recognition*. Namun seperti model *deep learning* lainnya, CNN memiliki kelemahan yaitu proses training yang membutuhkan kinerja *hardware* yang mumpuni. *Deep Learning* telah menunjukan performa yang luar biasa beberapa tahun terakhir. Hal Ini dipengaruhi faktor perkembangan komputasi yang lebih baik, *dataset* yang lebih besar dan arsitektur jaringan yang lebih dalam [9].

Metode *Haar Cascade*, yang menggunakan fitur *Haar-like*, diusulkan oleh Paul Viola dan Michael Jones pada tahun 2001. Metode ini sering juga disebut metode *Viola-Jones.* Metode ini banyak digunakan untuk deteksi objek karena strukturnya yang sederhana, tingkat deteksi yang tinggi, dan kecepatan deteksi yang cepat. Hal ini menunjukkan kinerja yang sangat baik dalam deteksi wajah manusia. Fitur *Haar-like* menghitung nilai fitur area melalui perbedaan nilai kecerahan. Fitur *Haar-like* mengklasifikasikan berbagai fitur yang ada pada objek dengan posisi, ukuran, dan bentuk yang berbeda. Saat menggunakan fitur *Haar-like* dimungkinkan untuk mendeteksi objek tertentu dalam sebuah gambar. Wajah manusia memiliki fitur yang dapat digunakan untuk klasifikasi (misalnya, mata, hidung, dan mulut). Oleh karena itu, wajah manusia dapat dideteksi dengan membandingkan nilai fitur yang dihitung dan dilatih sebagai nilai referensi [10].

Penggunaan metode CNN dalam mendeteksi wajah dan mulut terdapat dalam penelitian yang dilakukan oleh Raden Budiarto Hadiprakoso dan Nurul Qomariasih. Setelah melalui proses augmentasi gambar dan *deep transfer learning*, model yang dibangun menggunakan metode CNN dapat mencapai akurasi 98,3% dan skor F1 98,7% pada *dataset* validasi. Berdasarkan hasil pengujian, pendekatan augmentasi gambar terbukti dapat meningkatkan kinerja model dengan melakukan diversifikasi data latih. Selanjutnya, *transfer learning* telah terbukti meningkatkan akurasi model secara keseluruhan [11].

Penelitian oleh Radimas Putra Muhammad Davi Labib dkk, dengan menggunakan metode *Haar Cascade Classifier* menunjukkan bahwa sistem yang dirancang mampu mendeteksi wajah, hidung, dan bibir pada intensitas cahaya 80-140 lux. Wajah terdeteksi pada jarak 30-120 cm, hidung pada jarak 30-60cm, sedangkan bibir pada jarak 30-70cm. Sistem yang dirancang dapat melakukan proses pendeteksian pada kecepatan 5 fps. Hasil pengujian secara keseluruhan diperoleh tingkat keberhasilan sebesar 88,89% [12].

Berdasarkan hal itu, maka penulis tertarik untuk meneliti metode pendeteksi masker pada wajah dengan menggunakan dua metode klasifikasi citra yaitu *Haar Cascade* dan CNN dengan menggunakan foto wajah pegawai sebagai dataset sebanyak 500 foto bermasker dan 500 foto tanpa masker yang akan ditraining untuk mendapatkan model dan *classifiers*. Hal ini untuk mengetahui proses yang terjadi pada metode *Haar Cascade* dan CNN dalam melakukan deteksi masker pada wajah dan untuk mengetahui perbandingan tingkat akurasi sistem deteksi masker pada wajah dengan menggunakan metode CNN dan *Haar Cascade*. Implementasinya akan menggunakan bahasa pemograman *python*.

2. Metodologi Penelitian

Langkah awal yang dilakukan pada penelitian ini adalah mengidentifikasi masalah, menganalisa masalah, menentukan tujuan, mempelajari literatur, mengumpulkan data dan analisa, mendesain sistem, melakukan implementasi dengan metode *Haar Cascade* dan CNN, dan menguji hasil penelitian. Kerangka kerja dalam penelitian ini dijelaskan seperti pada Gambar 1.

Gambar 1. Kerangka Kerja Penelitian

2.1. Identifikasi Masalah

Tahap ini dilakukan dalam rangka menggali informasi dan untuk mengetahui kendala-kendala yang dihadapi oleh petugas penjaga pintu kantor. Hasil identifikasi masalah akan dianalisa lebih lanjut.

2.2. Menganalisa masalah

Tahap selanjutnya adalah melakukan analisa masalah, dari hasil indentifikasi masalah sebelumnya diketahui bahwa pemeriksaan penggunaan masker menggunakan tenaga manusia dalam melakukan pemeriksaan satu per satu. Tata cara pemeriksaan seperti ini memiliki beberapa keterbatasan yaitu bergantung ke para petugas pemeriksa yang tidak bisa dilakukan setiap waktu dan juga memiliki keterbatasan tenaga, maka dibutuhkan suatu penelitian untuk menerapkan metode yang tepat untuk menyelesaikan permasalahan tersebut.

2.3. Menentukan Tujuan

Tahapan ini menentukan tujuan penelitian yaitu identifikasi masker pada *Face Detection* menggunakan metode CNN dan Haar Cascade. Bagaimana penerapan metode CNN dan Haar Cascade untuk sistem pengenalan wajah dan bagaimana hasil dan analisa akurasi pengenalan wajah dengan menggunakan metode CNN dan *Haar Cascade* tersebut.

2.4. Mempelajari Literatur

Literatur yang digunakan berupa buku referensi atau buku penunjang, jurnal internasional dan nasional serta konsep-konsep yang mendukung dalam menyelesaikan penelitian ini.

2.5. Pengumpulan dan Analisa Data

Penelitian ini membutuhkan data *dataset* berupa foto wajah pegawai yang akan ditraining menggunakan metode *Haar Cascade* dan CNN. Data yang dibutuhkan untuk *dataset* adalah 500 foto wajah pegawai yang memakai masker dan 500 foto wajah pegawai yang tidak memakai masker.

2.6. Desain Sistem

Tahap desain sistem diawali dengan cara menentukan *software* dan *hardware* apa yang akan digunakan untuk penelitian berdasarkan analisa masalah yang sebelumnya sudah dilakukan.

2.7. Implementasi Sistem

Aplikasi untuk mendeteksi masker pada wajah dibuat dengan menggunakan bahasa pemrograman *Python.* Aplikasi yang dibuat terdiri dari aplikasi untuk mendeteksi wajah dengan data sumber dari video *webcam* secara *live*.

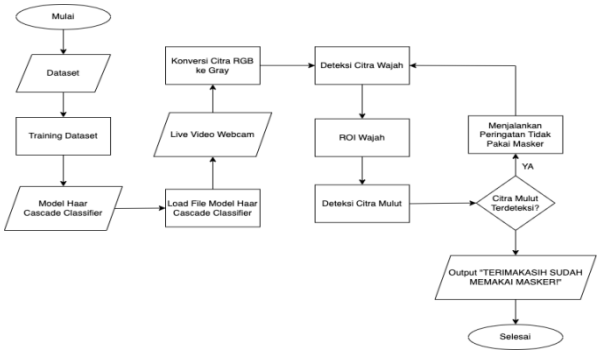
2.8. Pengujian Hasil

Tahap pengujian dilakukan menggunakan aplikasi yang sudah dibuat. Aplikasi akan diuji menggunakan menggunakan data dari *webcam* secara *live*.

3. Hasil dan Pembahasan

3.1. *Haar Cascade*

Tahapan yang ditempuh untuk klasifikasi citra menggunakan metode *Haar Cascade* adalah sebagaimana ditampilkan dalam *flowchart* disajikan pada Gambar 2.



Gambar 2. *Flowchart* Metode *Haar Cascade*

Penjelasan lebih langkap dari *flowchart* pada Gambar 2 diatas adalah sebagai berikut:

a. *Training dataset*

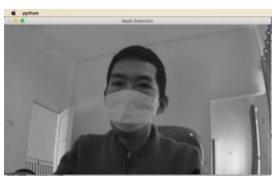
Proses *training dataset* pada metode *Haar Cascade Classifier* dalam Bahasa pemograman *python* adalah dengan menggunakan *file haarcascade_frontalface_default.xml* yang sudah disediakan dalam *library OpenCV*. Proses melatih data menjadi *classifier* membutuhkan sejumlah gambar positif dan negatif. Hal ini sudah tersedia langsung dalam *OpenCV* sehingga pengguna program tidak perlu men*training* ulang karena sudah siap untuk digunakan.

b. *Load file model haar cascade classifier*

Load file model haar cascade cassifier dapat dilakukan dengan cara, pertama perintah mengimport *library* yang dibutuhkan yaitu *numpy*, *cv2* dan *imtulis* kemudian mendeklarasikan *classifier* yaitu *classifier* wajah dan mulut, membaca *input* gambar dari video *webcam* dengan perintah *cv2.VideoCapture(1)* dan selanjutnya menampilkan pada layar.

c. Konversi citra *RGB* ke Gray

Konversi citra RGB ke *Gray* dilakukan setelah input video dari *webcam* berhasil dilakukan. Perintah yang digunakan adalah *cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)* akan mengubah gambar RGB menjadi *Grayscale.*



Gambar 3. Hasil Konversi Citra RGB ke *Gray*

d. Deteksi citra wajah

Tahap selanjutnya perintah untuk mendeteksi wajah. Berikut perintah yang digunakan untuk deteksi wajah menggunakan bahasa *python* disajikan pada Algoritma 1.

Algoritma 1. Deteksi citra wajah

```
if(len(faces)==0 and len(faces_bw)==0):

  cv2.putText(gray,'WAJAH TIDAK
  TERDETEKSI',org,font,font_scale,
  weared_mask_font_color,thickness,
  cv2.LINE_AA)

else:

  cv2.putText(gray, 'WAJAH TERDETEKSI', org,
  font,font_scale,
  not_weared_mask_font_color, thickness,
  cv2.LINE_AA)
```

e. Membuat ROI pada wajah

Tahap selanjutnya membuat ROI (*Region of Interest*) berupa kotak persegi yang akan berada pada wajah. Hal ini digunakan untuk memudahkan mengenali jika benar wajah terdeteksi dan di bagian mana terdeteksi. Menampilkannya ke layar output akan diubah kembali menjadi image RGB karena proses menggunakan *image gray* tidak diperlukan lagi. Perintah program yang digunakan disajikan pada Algoritma 2

Algoritma 2. ROI Wajah

```
for (x, y, w, h) in faces:
  cv2.rectangle(img, (x, y), (x + w, y + h),
(255, 255, 255), 2)
  roi_gray = gray[y:y + h, x:x + w]
  roi_color = img[y:y + h, x:x + w]
```

Berikut hasil yang ditampilkan dari program yang dijalankan disajikan pada Gambar 4.

Gambar 4. Membuat ROI pada Wajah

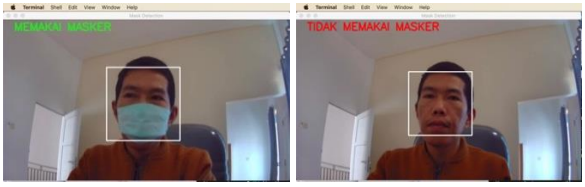
f. Deteksi citra mulut

Tahap selanjutnya mendeteksi apakah memakai masker atau tidak. Jika terdapat mulut maka *output*nya tidak memakai masker sebaliknya jika tidak terdeteksi mulut maka *output*nya memakai masker. Perintah program yang digunakan yang digunakan disajikan pada Algoritma 3.

Algoritma 3. Deteksi citra mulut

```
  mouth_rects =
mouth_cascade.detectMultiScale(gray, 1.5, 5)
  if(len(mouth_rects) == 0):
      cv2.putText(img, weared_mask, org,
font, font_scale, weared_mask_font_color,
thickness, cv2.LINE_AA)
  else:
      for (mx, my, mw, mh) in mouth_rects:
          if(y < my < y + h):
              cv2.putText(img,
not_weared_mask, org, font, font_scale,
not_weared_mask_font_color, thickness,
cv2.LINE_AA)
              break
```

Berikut hasil yang ditampilkan dari program yang dijalankan pada Gambar 5.



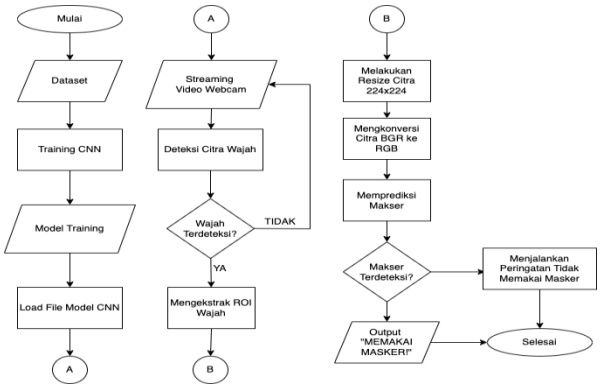
Gambar 5. Program Mampu Membedakan yang Memakai Masker dan yang Tidak Memakai Masker

g. Menampilkan *output* teks atau menyalakan alarm

Tapah terakhir menampilkan hasil keputusan dari deteksi program, jika program tidak mendeteksi masker pada wajah maka akan menyalakan suara alarm dan jika masker terdeteksi tidak menyalakan alaram, hanya menampikan pesan Memakai Masker di layar.

3.2. CNN

Tahapan yang ditempuh untuk klasifikasi citra menggunakan metode CNN adalah sebagaimana ditampilkan dalam *flowchart* yang disajikan pada Gambar 6.



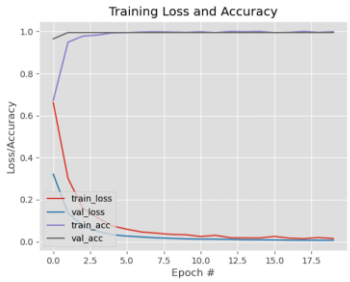
Gambar 6. *Flowchart* Metode CNN

Penjelasan lebih langkap dari *flowchart* pada Gambar 6 di atas adalah sebagai berikut:

a. *Training* CNN

Proses training ini akan dijalankan menggunakan bahasa pemograman *python* di mana data yang akan digunakan untuk *input training* adalah foto pegawai

dalam *folder with_mask* dan *without_mask*, sehingga mendapatkan hasil *training* seperti Gambar 7 berikut:



Gambar 7. Hasil *Trining* CNN

Setelah proses *training* selesai, didapatkan nilai *loss* terkecil sebesar 0.0144, akurasi tertinggi sebesar 1.0000, *loss* pada data validasi terkecil sebesar 0.0065, dan akurasi tertinggi pada data validasi sebesar 0.9950. Model ini nantinya akan digunakan untuk melakukan klasifikasi masker pada wajah.

b. Load file model CNN

Tahap ini dilakukan untuk mengimport *library* yang dibutuhkan diantaranya *tensorflow*, *keras*, *numpy*, *cv2* dan *imtulis* kemudian mendeklarasikan classifier yaitu *classifier* wajah dan mulut.

c. Input citra video webcam

Proses selanjutnya adalah membaca input gambar dari video webcam, dengan perintah *vs=VideoStream(src=1).start()* lalu menampilkan pada layar.

d. Deteksi citra wajah dan mengekstrak ROI wajah

Tapah selanjutanya adalah mendeteksi wajah yang dilakaukan dengan perintah if len(faces) > 0: dimana jika wajah ditemukan maka akan dilanjutkkan dengan perintah selanjutnya dan jika tidak ditemkukan yang digunakan disajikan pada Algoritma 4.

Algoritma 4. Deteksi wajah dan ROI

```
    face = frame[startY:endY, startX:endX]
    face = img_to_array(face)
    face = preprocess_input(face)
    faces.append(face)
    locs.append((startX, startY, endX, endY))

    if len(faces) > 0:
      faces = np.array(faces, dtype="float32")
      preds      =      maskNet.predict(faces,
batch_size=32)
    return (locs, preds)
```

e. Melakukan *resize* citra 224x224

Tahap *resize* ini dilakukan dengan perintah *face = cv2.resize(face, (224, 224))*. Hasil dari perintah ini berupa gambar dengan ukuran 224x224 *pixel* seperti pada Gambar 8.

Gambar 8. Hasil *Resize* Citra ke 224x224 *Pixel*

f. Mengkonversi citra BGR ke RGB

Tahap selanjutnya adalah proses konversi warna citra dari BGR ke RGB. Perintah yang digunakan adalah *face = cv2.cvtColor(face, cv2.COLOR_BGR2RGB).*

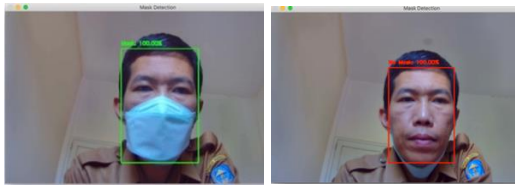
g. Memprediksi Masker

Tahap terakhir adalah memprediksi masker dengan menggunakan perintah *(locs, preds) = detect_and_predict_mask (frame, faceNet, maskNet)* kemudian dilakukan *loop* dari data prediksi wajah untuk menemukan masker dengan perintah yang digunakan disajikan pada Algoritma 5.

Algoritma 5. Deteksi wajah dan ROI

```
    for (box, pred) in zip(locs, preds):
    (startX, startY, endX, endY) = box
(mask, withoutMask) = pred
```

Tahap selanjutkan dilakukan pelabelan untuk prediksi wajah ditemukan dibuat label Mask dan yang tidak ditemukan dibuat label *No Mask*. Hasil yang diperoleh dari menjalankan program ini disjaikan pada Gambar 9.



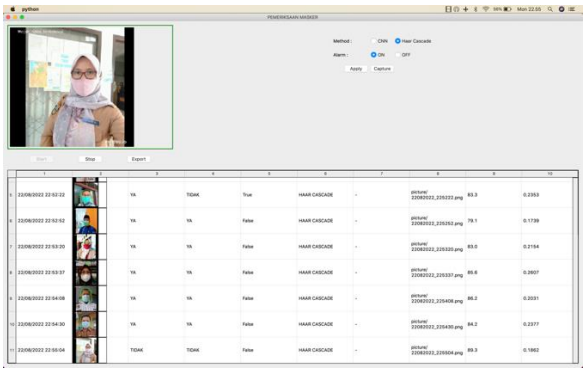
Gambar 9. Program Mampu Membedakan yang Memakai Masker dan yang Tidak Memakai Masker

h. Menampilkan *output* atau menyalakan alarm

Tapah terakhir menampilkan hasil keputusan dari deteksi program, jika program tidak mendeteksi masker pada wajah maka akan menyalakan suara alarm dan jika masker terdeteksi tidak menyalakan alaram, hanya menampikan pesan Memakai Masker di layar.

3.3 Pengujian Sistem

Hasil analisis dan perancangan yang telah dilakukan dapat diuji menggunakan aplikasi yang hasilnya disajikan pada Gambar 10.



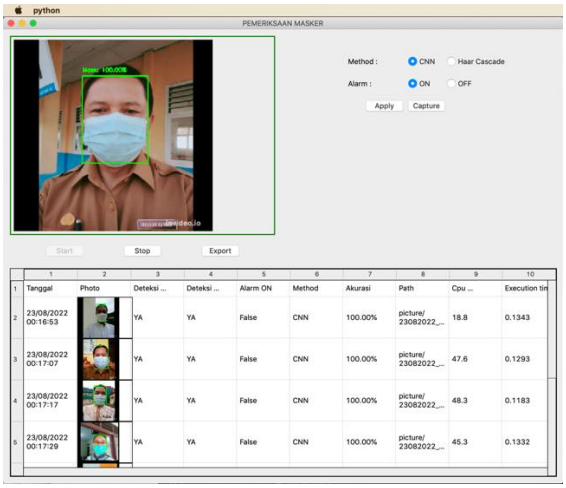
Gambar 10. Tangkapan Layar Hasil Saat Pengujian Metode *Haar Cascade*

Tabel 1. Terdeteksi Menggunakan Metode *Haar Cascade*

10 Orang Pegawai	Wajah	Masker	Keberhasilan
Memakai Masker	8	6	6
Tanpa Masker	10	0	10
Akurasi			80%

Berdasarkan tingkat keberhasilan deteksi masker pada wajah dengan menggunaan metode *Haar Cascade* mendapatkan hasil 80% untuk akurasi kerberhasilan.

Dilanjutkan untuk pengujian menggunakan metode CNN dengan data dari 10 orang memakai masker dan 10 orang yang tidak memakai masker. Hasil pengujian deteksi masker metode CNN ini terdapat pada Gambar 11 dan Tabel 2.



Gambar 11. Tangkapan Layar Hasil Saat Pengujian Metode CNN

Tabel 2. Terdeteksi Menggunakan Metode CNN

10 Orang Pegawai	Wajah	Masker	Keberhasilan
Memakai Masker	10	10	10
Tanpa Masker	10	0	10
Akurasi			100%

Berdasarkan tingkat keberhasilan deteksi masker pada wajah dengan menggunaan metode CNN mendapatkan hasil 100% untuk akurasi kerberhasilan.

4. Kesimpulan

Berdasarkan penelitian yang telah dilakukan dapat ditarik kesimpulan bahwa identifikasi masker pada *face detection* menggunakan metode CNN mendapatkan nilai lebih baik yaitu dengan nilai akurasi 100%, sedangkan menggunakan metode *Haar Cascade* nilai akurasi yang diperoleh adalah 80%. Aplikasi yang dibangun menggunakan Bahasa pemograman python dapat berjalan dengan baik untuk implementasi kedua metode yang dipakai yaitu *Haar Cascade* dan CNN.

Daftar Rujukan

[1] WHO. (2020). Advice on the use of masks in the context of covid-19: interim guidance, 5 june 2020 (Tech. Rep.). World Health Organization.

[2] Hussain, S., Yu, Y., Ayoub, M., Khan, A., Rehman, R., Wahid, J. A., & Hou, W. (2021). IoT and deep learning based approach for rapid screening and face mask detection for infection spread control of COVID-19. *Applied Sciences*, *11*(8), 3495. https://doi.org/10.3390/app11083495

[3] Du, J. (2018). Understanding of Object Detection Based on CNN Family and YOLO. *Journal of Physics: Conference Series*,1004. DOI: https://doi.org/10.1088/1742-6596/1004/1/012029

[4] Effendi, M., Fitriyah, F., & Effendi, U. (2017). Identifikasi Jenis dan Mutu Teh Menggunakan Pengolahan Citra Digital dengan Metode Jaringan Syaraf Tiruan. *Jurnal Teknotan*, 11(2). DOI: https://doi.org/10.24198/jt.vol11n2.7

[5] Liu, L., Ouyang, W., Wang, X., Fieguth, P., Chen, J., Liu, X., & Pietikäinen, M. (2019). Deep Learning for Generic Object Detection: A Survey. *International Journal of Computer Vision*, 128(2), 261–318. DOI: https://doi.org/10.1007/s11263-019-01247-4

[6] Santoso, A., &Ariyanto, G. (2018). Implementasi Deep Learning Berbasis Keras Untuk Pengenalan Wajah. *Emitor: Jurnal Teknik Elektro*, 18(1), 15–21.DOI: https://doi.org/10.23917/emitor.v18i01.6235

[7] Danukusumo, K.P., Pranowo.,& Maslim, M. (2017). Indonesia Ancient Temple Classification Using Convolutional Neural Network. *International Conference on Control, Electronics, Renewable Energy and Communications* (ICCREC).DOI: http://dx.doi.org/10.1109/iccerec.2017.8226709

[8] Asmara, R. A., Ridwan, M., & Budiprasetyo, G. (2021, September). Haar Cascade and Convolutional Neural Network Face Detection in Client-Side for Cloud Computing Face Recognition. In *2021 International Conference on Electrical and Information Technology (IEIT)* (pp. 1-5). IEEE, 2021. https://doi.org/10.1109/IEIT53149.2021.9587388

[9] Hussain, N., Khan, M. A., Kadry, S., Tariq, U., Mostafa, R. R., Choi, J. I., & Nam, Y. (2021). Intelligent deep learning and improved whale optimization algorithm based framework for object recognition. *Hum. Cent. Comput. Inf. Sci*, *11*, 34, 2021. https://doi.org/10.1109/HCIS.2021.11.034

[10] Choi, C. H., Kim, J., Hyun, J., Kim, Y., & Moon, B. (2022). Face Detection Using Haar Cascade Classifiers Based on Vertical Component Calibration. *Human-Centric Computing And Information Sciences*, *12*, 2022. https://doi.org/10.22967/HCIS.2022.12.011

[11] Hadiprakoso, R. B., & Qomariasih, N. (2022). Deteksi Masker Wajah Menggunakan Deep Transfer Learning Dan Augmentasi Gambar. *Jiko (Jurnal Informatika dan Komputer)*, *5*(1), 12-18, 2022. https://doi.org/10.33387/jiko.v5i1.3591

[12] Labib, R. P. M. D., Hadi, S., & Widayaka, P. D. (2021). Low Cost System for Face Mask Detection Based Haar Cascade Classifier Method. *MATRIK: Jurnal Manajemen, Teknik Informatika dan Rekayasa Komputer*, *21*(1), 21-30, 2021. https://doi.org/https://doi.org/10.30812/matrik.v21i1.11