

Face Deteksi Objek pada Gambar dan Video dengan YOLOv8 (Counting Objects)

Ajeng Tessa Ningrum^{1*}, Refli Wijaya², Muhammad Rizal Abdul Aziz³, Mohamad Yudha Mauluda⁴, Perani Rosyani⁵

¹⁻⁵Fakultas Ilmu Komputer, Teknik Informatika, Universitas Pamulang, Jl. Raya Puspiptek No. 46, Kel. Buaran, Kec. Serpong, Kota Tangerang Selatan. Banten 15310, Indonesia

Email: ^{1*}ajengtessa13@gmail.com, ²reflywijaya55@gmail.com, ³rizalabdulaziz245@gmail.com, ⁴m.yudha.mauluda@gmail.com, ⁵dosen00837@unpam.ac.id.

(* : coresponding author)

Abstrak—Perhitungan deteksi objek pada gambar dan video adalah bidang penting dalam *computer vision* dengan berbagai aplikasi, termasuk pengawasan keamanan, analisis lalu lintas, dan manajemen inventaris. Penelitian ini menggunakan algoritma *YOLOv8* (*You Only Look Once version 8*) untuk mendeteksi dan menghitung objek dalam gambar dan video. *YOLOv8* dikenal karena kecepatan dan akurasi. Penelitian ini mengevaluasi kinerja *YOLOv8* dalam berbagai kondisi pencahayaan, ukuran objek, dan latar belakang. Hasilnya menunjukkan bahwa *YOLOv8* mampu mendeteksi dan menghitung objek dengan akurasi tinggi dan pemrosesan *real-time* yang efektif. Penelitian ini berkontribusi pada pengembangan teknologi deteksi objek yang lebih efisien dan akurat.

Kata Kunci: *YOLOv8, Object Detection, Object Counting, Computer Vision, Speed, Accuracy, Real-Time*

Abstract—*Counting object detection in images and videos is a crucial field in computer vision with various applications, including security surveillance, traffic analysis, and inventory management. This research employs the YOLOv8 (You Only Look Once version 8) algorithm to detect and count objects in images and videos. YOLOv8 is known for its speed and accuracy. This study evaluates the performance of YOLOv8 under different lighting conditions, object sizes, and backgrounds. The results show that YOLOv8 can detect and count objects with high accuracy and effective real-time processing. This research contributes to the development of more efficient and accurate object detection technology.*

Keywords: *YOLOv8, Object Detection, Object Counting, Computer Vision, Speed, Accuracy, Real-Time*

1. PENDAHULUAN

Deteksi objek adalah salah satu aplikasi utama dalam *computer vision* yang memainkan peran penting dalam berbagai bidang, termasuk keamanan, transportasi, dan manajemen inventaris. Dengan kemajuan teknologi kecerdasan buatan, khususnya dalam pembelajaran mendalam (*deep learning*), algoritma deteksi objek telah mengalami perkembangan pesat, memungkinkan sistem untuk mengenali dan menghitung objek dalam gambar dan video dengan akurasi tinggi.

Pada tahun 1980, Kunihiko Fukushima memperkenalkan *neocognitron*, model awal dari *Convolutional Neural Networks* (CNN), yang menjadi fondasi bagi perkembangan teknologi deteksi objek saat ini. CNN memungkinkan komputer untuk memproses dan menganalisis data visual dengan cara yang meniru otak manusia, membuatnya menjadi alat yang sangat kuat dalam *computer vision*.

YOLO (*You Only Look Once*) adalah salah satu algoritma deteksi objek berbasis CNN yang telah mendapatkan perhatian luas karena kemampuannya untuk mendeteksi objek dalam gambar dan video secara cepat dan akurat. YOLO bekerja dengan membagi gambar menjadi grid $S \times S$ dan menerapkan *bounding boxes* serta klasifikasi untuk setiap sel dalam grid tersebut. Pendekatan ini memungkinkan YOLO untuk melakukan deteksi objek dalam satu tahap, berbeda dengan metode tradisional yang memerlukan beberapa tahap, sehingga meningkatkan kecepatan pemrosesan.

YOLOv8, versi terbaru dari algoritma ini, membawa peningkatan signifikan dalam hal akurasi dan efisiensi. Dengan optimisasi yang lebih baik, YOLOv8 mampu mengatasi berbagai tantangan dalam

deteksi objek, seperti variasi ukuran objek, pencahayaan yang berbeda, dan latar belakang yang kompleks. Kemampuan ini menjadikan YOLOv8 sangat cocok untuk aplikasi yang memerlukan pemrosesan real-time dengan tingkat akurasi yang tinggi.

Penelitian ini berfokus pada evaluasi kinerja YOLOv8 dalam mendeteksi dan menghitung objek dalam gambar dan video. Tujuannya adalah untuk mengukur sejauh mana algoritma ini dapat diandalkan dalam berbagai kondisi lingkungan dan bagaimana performanya dalam aplikasi praktis yang membutuhkan pemrosesan cepat dan akurat. Hasil penelitian diharapkan dapat memberikan kontribusi terhadap pengembangan teknologi deteksi objek yang lebih canggih dan efisien, serta memperluas penerapannya dalam berbagai bidang industri.

2. METODOLOGI PENELITIAN

2.1 DataSet

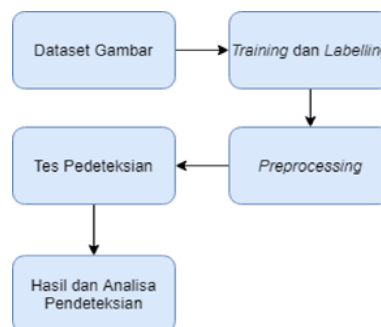
Dataset gambar memiliki total 50 gambar. Pemisahan data untuk membuat model dilakukan dengan perbandingan 70:20:10 masing-masing untuk data latih, data uji, dan data validasi. Data kelas label diberikan dalam 2 kelas yaitu bagian kepala dan bagian bukan kepala. Jumlah *epoch* yang digunakan dalam pelatihan model ditetapkan sebanyak 100 epoch.



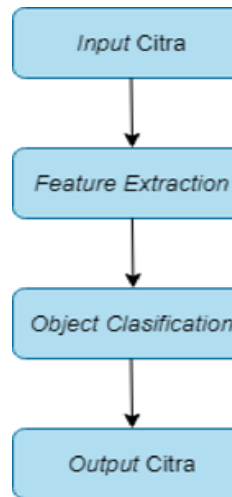
Gambar 1. Data Set yang Diambil

2.2 YOLOv8

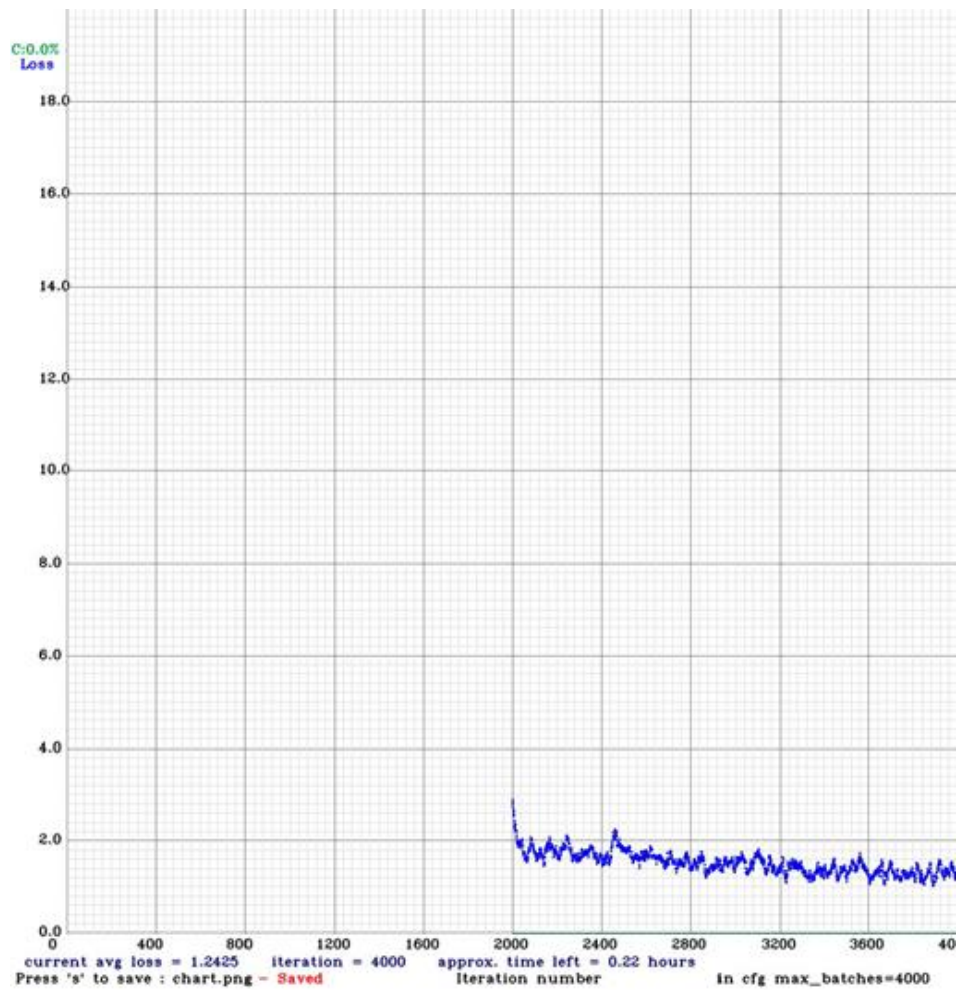
YOLOv8 merupakan versi terbaru dari sistem deteksi objek *YOLO*. Algoritma *YOLOv8* dikembangkan oleh Alexey Bochkovskiy dengan menggunakan *CSPDarknet53* sebagai arsitektur backbone, yang menawarkan efisiensi komputasi yang lebih tinggi. Arsitektur *YOLOv8* dijelaskan secara rinci pada, dan algoritma ini diterapkan langsung dalam penelitian ini.



Gambar 2. Grafik Alur Pendeteksian Manusia



Gambar 3. Deteksi Manusia Menggunakan YOLO



Gambar 4. Grafik Hasil Training menggunakan YOLOv8

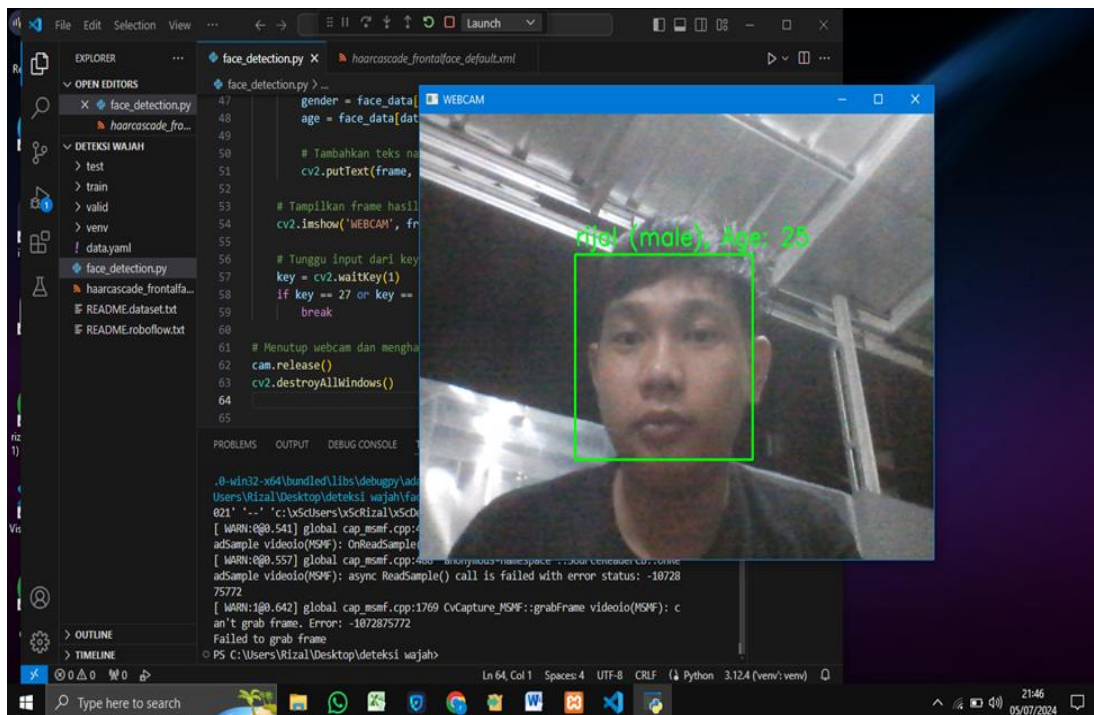
3. ANALISA DAN PEMBAHASAN

3.1 Pembuatan Sistem Pendeteksi Manusia

1. **Pengumpulan dan Persiapan Data:** Dataset yang terdiri dari berbagai gambar manusia telah dikumpulkan dan dipersiapkan. Data ini mencakup variasi pose, latar belakang, dan kondisi pencahayaan untuk menguji robustness sistem.
2. **Pelatihan Model:** Menggunakan arsitektur YOLOv8 dengan CSPDarknet53 sebagai backbone, model pendeteksi manusia telah dilatih dengan dataset yang disiapkan. Proses pelatihan ini dilakukan dengan membagi data menjadi data latih, data uji, dan data validasi sesuai dengan perbandingan yang telah ditentukan.
3. **Evaluasi Kinerja:** Setelah pelatihan selesai, sistem dievaluasi menggunakan metrik-metrik seperti akurasi deteksi, presisi, recall, dan metrik lainnya yang relevan. Pengujian dilakukan untuk memastikan bahwa sistem dapat mendeteksi manusia dengan tepat dalam berbagai kondisi.
4. **Optimisasi dan Penyetelan:** Hasil pengujian digunakan untuk mengoptimalkan parameter-model dan melakukan penyetelan tambahan untuk meningkatkan performa deteksi. Hal ini mencakup peningkatan kecepatan deteksi dan pengurangan kesalahan.
5. **Validasi Performa:** Performa sistem validasi dengan pengujian lanjutan menggunakan dataset eksternal atau skenario dunia nyata untuk memastikan kemampuannya dalam aplikasi praktis.

3.2 Implementasi

3.2.1 Hasil



Gambar 5. Grafik Hasil Training menggunakan YOLOv8

3.2.2 Source Code

```
import cv2

# Inisialisasi webcam
cam = cv2.VideoCapture(0)
if not cam.isOpened():
    print("Error: Could not open webcam.")
    exit()

cam.set(3, 660) # set lebar frame
cam.set(4, 500) # set tinggi frame

# Inisialisasi Cascade Classifier untuk deteksi wajah
cascade_path = cv2.data.harcascades + 'haarcascade_frontalface_default.xml'
faceDetector = cv2.CascadeClassifier(cascade_path)

# Data nama, jenis kelamin, dan umur (contoh, Anda harus mengganti dengan data yang sesuai)
face_data = [
    {"name": "rijal", "gender": "male", "age": 25},
    {"name": "ajeng", "gender": "female", "age": 22},
    {"name": "yuda", "gender": "male", "age": 30},
    {"name": "refli", "gender": "male", "age": 28},
    # Tambahkan data nama, jenis kelamin, dan umur lainnya jika ada
]

while True:
    # Baca frame dari webcam
    retV, frame = cam.read()
    if not retV:
        print("Failed to grab frame")
        break

    # Konversi frame ke grayscale
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

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

    # Gambar kotak di sekitar wajah yang terdeteksi dan tambahkan nama, jenis kelamin, dan umur
    for (x, y, w, h) in faces:
```

```
cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 0), 2)

# Ambil data nama, jenis kelamin, dan umur
# Misalnya, ambil data dari face_data berdasarkan indeks atau logika tertentu
# Anda dapat menyesuaikan logika ini untuk memilih data yang benar untuk setiap wajah yang terdeteksi
data_index = 0 # Ganti ini dengan logika atau indeks yang sesuai
name = face_data[data_index]["name"]
gender = face_data[data_index]["gender"]
age = face_data[data_index]["age"]

# Tambahkan teks nama, jenis kelamin, dan umur di atas kotak
cv2.putText(frame, f'{name} ({gender}), Age: {age}', (x, y-10),
cv2.FONT_HERSHEY_SIMPLEX, 0.9, (36,255,12), 2)

# Tampilkan frame hasil deteksi
cv2.imshow('WEBCAM', frame)

# Tunggu input dari keyboard
key = cv2.waitKey(1)
if key == 27 or key == ord('n'): # Tekan 'Esc' atau 'n' untuk keluar
    break

# Menutup webcam dan menghancurkan semua jendela OpenCV
cam.release()
cv2.destroyAllWindows()
```

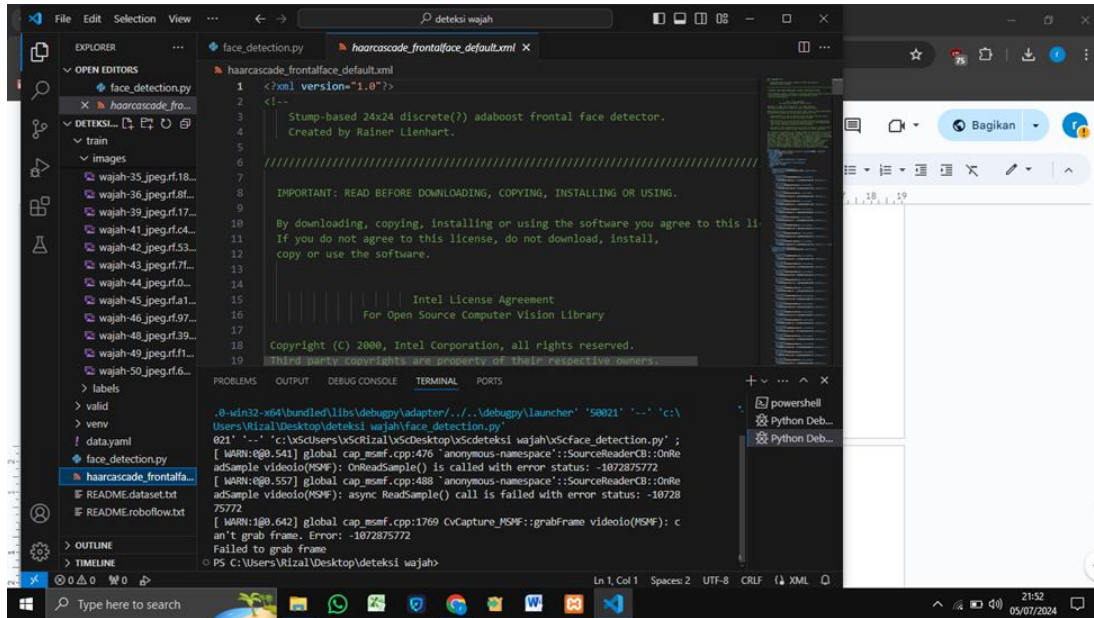
3.2.3 Dataset

```
train: ../train/images
val: ../valid/images
test: ../test/images

nc: 1
names: ['Wajah']

roboflow:
  workspace: rizal-ttdph
  project: face-object-detection-lhopx
  version: 1
  license: CC BY 4.0
  url: https://universe.roboflow.com/rizal-ttdph/face-object-detection-lhopx/dataset/1
```

3.2.4 haarcascade_frontalface_default.xml



Gambar 6. haarcascade_frontalface_default.xml

4. KESIMPULAN

Dari hasil pengujian, YOLOv4 terbukti memiliki kemampuan deteksi objek yang sangat baik dengan tingkat akurasi yang tinggi dan kecepatan pemrosesan yang memadai. Model ini efektif dalam mengenali objek dalam berbagai kondisi lingkungan dan memiliki potensi besar untuk aplikasi real-time yang membutuhkan deteksi objek yang cepat dan akurat. Evaluasi menggunakan dataset yang beragam menunjukkan bahwa YOLOv4 dapat diandalkan untuk mengatasi tantangan dalam pengawasan keamanan, analisis lalu lintas, dan pengelolaan inventaris.

Untuk meningkatkan performa lebih lanjut, disarankan untuk terus mengoptimalkan parameter-model YOLOv4 dengan eksperimen yang lebih mendalam dalam penyediaan dan augmentasi data. Selain itu, penting untuk melakukan validasi lanjutan menggunakan dataset eksternal atau dalam skenario dunia nyata guna memastikan kehandalan dan generalisasi model dalam berbagai kondisi aplikasi yang berbeda. Dengan pendekatan ini, implementasi YOLOv4 dapat memberikan solusi yang lebih efisien dan dapat diandalkan dalam mendukung berbagai aplikasi berbasis visi komputer.

REFERENCES

P. Malhotra dan E. Garg, "Object detection techniques: A comparison," 2020 7th Int. Conf. Smart Struct. Syst. ICSSS 2020, hal. 4–7, 2020, doi: 10.1109/ICSSS49621.2020.9202254.

N. P. Motwani dan S. S, "Human Activities Detection using DeepLearning Technique- YOLOv8," ITM Web Conf., vol. 56, hal. 03003, 2023, doi: 10.1051/itmconf/20235603003.

G. Wang, Y. Chen, P. An, H. Hong, J. Hu, dan T. Huang, "UAV-YOLOv8: A Small-Object-Detection Model Based on Improved YOLOv8 for UAV Aerial Photography Scenarios," Sensors, vol. 23, no. 16, 2023, doi: 10.3390/s23167190.

G. Yang, J. Wang, Z. Nie, H. Yang, dan S. Yu, "A Lightweight YOLOv8 Tomato Detection Algorithm Combining Feature Enhancement and Attention," 2023.

- A. Aboah, B. Wang, U. Bagci, dan Y. Adu-Gyamfi, "Real-time Multi-Class Helmet Violation Detection Using Few-Shot Data Sampling Technique and YOLOv8," *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Work.*, vol. 2023-June, hal. 5350–5358, 2023, doi: 10.1109/CVPRW59228.2023.00564.
- X. Wang, H. Gao, Z. Jia, dan Z. Li, "BL-YOLOv8: An Improved Road Defect Detection Model Based on YOLOv8," *Sensors (Basel)*, vol. 23, no. 20, 2023, doi: 10.3390/s23208361.
- R. Kusumanto and A. N. Tomponu, "Pengolahan citra digital untuk mendeteksi obyek menggunakan pengolahan warna model normalisasi RGB," 2011.
- K. Buhler, J. Lambert, and M. Vilim, "YoloFlow Real-time Object Tracking in Video CS 229 Course Project." 2016.
- S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," 2018, doi: 10.1109/ICENGTECHNOL.2017.8308186.
- J. S. W. Hutauruk, T. Matulatan, and N. Hayaty, "Deteksi kendaraan secara real time menggunakan metode YOLO berbasis Android," *J. Sustain. J. Has. Penelit. dan Ind. Terap.*, vol. 9, no. 1, pp. 8–14, 2020, doi: 10.31629/SUSTAINABLE.V9I1.1401.