e-ISSN 2828-0490
JURNAL MEDIA COMPUTER SCIENCE

Deteksi Objek Secara *Real-Time* Berbasis YOLOv8 dan Algoritma DeepSORT

Daniel Satria Mahardhika [1)]; Magdalena A. Ineke Pakereng [2)]
[1,2] Program Studi S1 Teknik Informatika, Universitas Kristen Satya Wacana
Email: [1)] danielsatriamahardhika@gmail.com , [2)] ineke.pakereng@uksw.edu

How to Cite :
Mahardika. D. S., Pakereng. M. A. I. (2026) . Deteksi Objek Secara Real-Time Berbasis YOLOv8 dan Algoritma DeepSORT. Jurnal Media Computer Science, 5(1)

ARTICLE HISTORY
Received [15 November 2025]
Revised [25 Januari 2026]
Accepted [28 Januari 2026]

KEYWORDS
Object Detection, Object Tracking, YOLOv8l, DeepSORT, Traffic, Pedestrians.

ABSTRAK

Pemantauan kepadatan lalu lintas dan pergerakan pejalan kaki secara manual sering kali tidak efisien serta rentan terhadap kesalahan. Penelitian ini bertujuan untuk mengembangkan dan mengevaluasi sebuah sistem deteksi dan pelacakan objek otomatis secara real-time sebagai solusi atas permasalahan tersebut, dengan fokus pada empat kelas utama, yaitu mobil, motor, truk, dan orang. Metode yang digunakan meliputi penerapan arsitektur deep learning YOLOv8l untuk deteksi objek dengan presisi tinggi, yang dikombinasikan dengan algoritma DeepSORT untuk pelacakan dan *re-identification* guna mempertahankan ID unik pada setiap objek yang diamati. Data pelatihan dikumpulkan dari berbagai sumber rekaman CCTV dan diperkaya melalui teknik augmentasi, kemudian model dilatih menggunakan platform Google Colab. Untuk pengujian fungsional, sistem diimplementasikan dalam sebuah aplikasi web lokal berbasis Flask yang dapat memproses input dari video, webcam, serta siaran langsung YouTube. Hasil evaluasi menunjukkan bahwa model memiliki performa yang sangat tinggi dan seimbang, dengan nilai *accuracy*, *precision*, *recall*, dan F1-*score* secara keseluruhan mencapai 0,98. Meskipun terdapat tantangan terkait stabilitas *Frame Per Second* (FPS) akibat tingginya beban komputasi, sistem ini secara keseluruhan terbukti fungsional, andal, serta efektif sebagai solusi otomatisasi pemantauan objek di lingkungan publik.

ABSTRACT

Manual monitoring of traffic density and pedestrian movement is often inefficient and prone to errors. This study aims to develop and evaluate an automated real-time object detection and tracking system as a solution to these issues, focusing on four main classes: cars, motorcycles, trucks, and people. The method employed involves the implementation of the YOLOv8l deep learning architecture for high-precision object detection, combined with the DeepSORT algorithm for tracking and re-identification to maintain unique IDs for each observed object. The training data were collected from various CCTV recordings and enriched through augmentation techniques, after which the model was trained using the Google Colab platform. For functional testing, the system was implemented in a local Flask-based web application capable of processing input from videos, webcams, and YouTube live streams. The evaluation results indicate that the model achieved very high and balanced performance, with overall accuracy, precision, recall, and F1-score values reaching 0.98. Although challenges related to Frame Per Second (FPS) stability were identified due to heavy computational loads, the system as a whole proved to be functional, reliable, and effective as an automated solution for object monitoring in public environments.

PENDAHULUAN

Perkembangan teknologi informasi dan kecerdasan buatan (*Artificial Intelligence*/AI) telah memicu lahirnya berbagai inovasi dalam bidang pengolahan citra digital (*computer vision*), termasuk dalam deteksi dan pelacakan objek secara otomatis (Pakpahan, 2021; Purnama et al., 2024). Salah satu aspek yang semakin banyak diteliti dan dikembangkan adalah kemampuan sistem untuk mengenali serta mengikuti pergerakan objek dalam video secara waktu nyata (*real-time*). Deteksi dan pelacakan objek kini menjadi komponen penting dalam berbagai sistem cerdas, seperti pengawasan berbasis kamera (*video surveillance*), navigasi kendaraan otonom, penghitungan jumlah orang dan kendaraan, hingga interaksi manusia–komputer. Dengan meningkatnya kebutuhan akan sistem yang cepat dan akurat, diperlukan pendekatan teknologi yang mampu menjawab tantangan tersebut secara efisien (Fahrezi & Widiyanto, 2024).

Dalam konteks deteksi objek, metode berbasis *deep learning* telah menjadi pendekatan utama dalam satu dekade terakhir. Salah satu model deteksi yang memperoleh banyak perhatian adalah Y*ou Only Look Once (YOLO)*, sebuah algoritma deteksi objek *single-shot* yang terkenal karena kecepatan pemrosesannya (Herdianto et al., 2024). YOLO memungkinkan sistem mendeteksi berbagai jenis objek hanya dalam satu kali proses inferensi tanpa perlu membagi alur deteksi menjadi beberapa tahap seperti metode konvensional. Versi terbaru dari keluarga YOLO, yaitu YOLOv8, menghadirkan berbagai penyempurnaan baik dari sisi arsitektur jaringan maupun efisiensi pelatihan. YOLOv8 menggunakan desain modular dengan *backbone* dan *head* yang lebih ringan serta mendukung berbagai ukuran model, sehingga ideal diterapkan pada perangkat dengan sumber daya komputasi terbatas (Hayati et al., 2023a).

Namun, deteksi objek saja tidak cukup untuk banyak aplikasi dunia nyata. Dalam sistem pengawasan atau penghitungan objek, keberlanjutan identitas objek dari waktu ke waktu merupakan aspek yang sangat krusial. Oleh karena itu, diperlukan algoritma pelacakan yang mampu mempertahankan konsistensi identitas objek yang sama dari satu frame ke frame berikutnya. DeepSORT (*Simple Online and Realtime Tracking with a Deep Association Metric*) hadir sebagai solusi pelacakan objek real-time dengan mempertimbangkan baik posisi maupun fitur visual dari objek yang terdeteksi (Oise et al., 2025). DeepSORT mampu menjaga stabilitas identitas objek meskipun terjadi perubahan arah gerakan, variasi kecepatan, atau ketika objek tertutup sebagian oleh objek lain (Pereira et al., 2022).

Kombinasi antara YOLOv8 sebagai sistem deteksi dan DeepSORT sebagai algoritma pelacakan memberikan potensi besar untuk membangun sistem yang tidak hanya cepat dalam mendeteksi objek, tetapi juga andal dalam mengikuti pergerakannya secara berkesinambungan. Integrasi kedua pendekatan ini memungkinkan penerapan pada berbagai skenario dunia nyata yang menuntut ketepatan tinggi dan respons cepat, seperti pengawasan lalu lintas, *monitoring* kerumunan, hingga sistem keamanan cerdas (Sheng et al., 2024).

Penelitian ini bertujuan untuk merancang, mengimplementasikan, dan menguji sistem deteksi serta pelacakan objek berbasis kombinasi YOLOv8 dan DeepSORT dalam lingkungan video *real-time*. Evaluasi dilakukan terhadap berbagai aspek, seperti akurasi deteksi, stabilitas pelacakan, kecepatan pemrosesan, dan ketahanan sistem terhadap perubahan kondisi lingkungan. Diharapkan hasil penelitian ini dapat memberikan kontribusi nyata dalam pengembangan sistem pengolahan citra cerdas yang responsif, akurat, dan efisien guna mendukung berbagai kebutuhan industri maupun masyarakat luas (Sheng et al., 2024).

Memposisikan diri sebagai kelanjutan sekaligus perluasan dari penelitian-penelitian relevan sebelumnya, studi ini mengambil inspirasi dari keberhasilan penerapan YOLOv8 untuk penghitungan objek, seperti yang dilakukan oleh Hayati, Singasatia, dan Muttaqin (2023) yang berfokus pada penghitungan kendaraan (Hayati et al., 2023b). Namun, penelitian ini melampaui batasan tersebut dengan memperluas cakupan objek tidak hanya pada kendaraan, tetapi juga mencakup analisis pergerakan pejalan kaki dalam dua konteks lingkungan dinamis: lalu lintas di jalan umum dan arus keluar-masuk pengunjung di pusat perbelanjaan.

e-ISSN 2828-0490
JURNAL MEDIA COMPUTER SCIENCE

Perbedaan utama penelitian ini dibandingkan studi sebelumnya terletak pada penggunaan dataset yang sepenuhnya baru dan dirancang khusus untuk merepresentasikan skenario yang lebih kompleks. Selain itu, inovasi signifikan dari studi ini adalah implementasi sistem ke dalam sebuah aplikasi *website* lokal. Pendekatan ini tidak hanya ditujukan untuk memvalidasi algoritma, tetapi juga untuk membangun sebuah purwarupa fungsional yang mampu menampilkan hasil deteksi dan pelacakan secara interaktif, sehingga lebih mudah diakses, dianalisis, dan dimanfaatkan oleh pengguna akhir.

LANDASAN TEORI

Penelitian berjudul "Aplikasi Penghitung Jarak dan Jumlah Orang Berbasis YOLO sebagai Protokol Kesehatan Covid-19" menggunakan YOLOv3 untuk menghitung jumlah orang dengan akurasi 90,04%. Studi tersebut memvalidasi relevansi YOLO untuk mendeteksi objek berupa manusia. Namun, penelitian ini melangkah lebih jauh dengan mengadopsi YOLOv8l dan mengintegrasikan DeepSORT. Penambahan ini sangat penting untuk meningkatkan fungsionalitas dari sekadar penghitungan menjadi sistem pelacakan (tracking) dengan kemampuan re-identification pada objek *orang* maupun *kendaraan* (mobil, motor, truk) (Indaryanto et al., 2021). Studi lain berjudul "Implementasi *Convolutional Neural Network* untuk Deteksi Nyeri Bayi melalui Citra Wajah dengan YOLO" menggunakan YOLO untuk mendeteksi ekspresi nyeri pada bayi dengan nilai mAP@0.5 yang sangat tinggi (96,9%–99,2%) dan akurasi model 70%. Temuan tersebut menjadi dasar keyakinan akan keandalan YOLO dalam tugas deteksi presisi tinggi. Penelitian ini kemudian mengembangkan pendekatan tersebut dengan mengadopsi YOLOv8l yang lebih modern dan mengintegrasikan DeepSORT, sehingga fungsionalitas sistem meningkat dari sekadar deteksi menjadi pelacakan *real-time* dengan *re-identification* untuk objek orang dan kendaraan (Abuzairi et al., 2021).

Penelitian "Pemanfaatan Google Colab untuk Aplikasi Pendeteksian Masker Wajah Menggunakan Algoritma *Deep Learning* YOLOv7" memanfaatkan Google Colab dan YOLOv7 untuk mendeteksi masker wajah. Studi ini relevan karena menunjukkan efektivitas Google Colab sebagai *platform* komputasi untuk melatih model-model dalam keluarga YOLO. Pendekatan metodologis tersebut diadopsi dalam penelitian ini untuk melatih model YOLOv8l dengan memanfaatkan akselerasi GPU T4 guna meningkatkan efisiensi proses pelatihan (Gelar Guntara, 2023). Studi berjudul "Deteksi Tumpukan Sampah dengan Metode *You Only Look Once* (YOLO)" menerapkan YOLOv5 untuk mendeteksi tumpukan sampah dan berhasil mencapai akurasi *mAP@0.5* sebesar 100%. Keberhasilan tersebut membuktikan potensi besar keluarga YOLO dalam menghasilkan akurasi sangat tinggi. Hal ini menjadi landasan ekspektasi performa dalam penelitian ini, di mana arsitektur YOLOv8l dipilih untuk mencapai tingkat akurasi serupa pada skenario yang lebih kompleks, yaitu objek bergerak seperti kendaraan dan pejalan kaki (Maulidiansyah & Yaqin, 2023).

Penelitian "Deteksi Gerak pada Video Gerak Lansia Berbasis YOLO-V5 dan YOLO-V7" membandingkan YOLOv5 dan YOLOv7, di mana YOLOv5 unggul dalam kecepatan, sedangkan YOLOv7 lebih baik dalam ketelitian. Studi ini menyoroti adanya *trade-off* antara kecepatan dan akurasi. Temuan tersebut memperkuat alasan pemilihan YOLOv8l dalam penelitian ini, karena arsitekturnya menawarkan keseimbangan optimal antara akurasi dan kecepatan inferensi yang sangat penting untuk aplikasi *real-time* (Susanto et al., 2024). Studi "Deteksi Objek Menggunakan Metode YOLO dan Implementasinya pada Robot Bawah Air" mengevaluasi berbagai arsitektur YOLOv5 dan menemukan bahwa YOLOv5x (varian besar) memiliki nilai *mAP@[0.5:0.95]* tertinggi (0,881) dibanding YOLOv5s (varian kecil) (0,872). Temuan ini memberikan wawasan bahwa varian model yang lebih besar cenderung menghasilkan akurasi lebih tinggi. Pemahaman ini sejalan dengan keputusan penelitian ini menggunakan YOLOv8l (*large*) untuk memaksimalkan akurasi deteksi pada empat kelas target (Husnan et al., 2023). Penelitian "Pengaruh Jarak Objek Citra pada Model Deteksi dan Klasifikasi Botol Plastik Menggunakan YOLO" menggunakan YOLOv8 dan menemukan adanya penurunan akurasi ketika jarak objek tidak sesuai. Studi ini relevan karena

menggunakan arsitektur yang sama dan menyoroti tantangan variasi jarak objek. Temuan tersebut menjadi pertimbangan penting mengingat data CCTV pada penelitian ini memiliki variasi skala dan jarak objek yang cukup tinggi (Rosanti et al., 2024).Studi "Implementasi Deteksi Drone Menggunakan YOLO (*You Only Look Once*)" berfokus pada deteksi drone menggunakan YOLOv8 dan berhasil meraih *mAP* sebesar 0,994, melampaui performa YOLOv5. Temuan ini secara langsung memvalidasi keunggulan arsitektur YOLOv8. Performa superior tersebut menjadi landasan kuat bagi penelitian ini untuk mengadopsi YOLOv8l, dengan ekspektasi mencapai tingkat presisi dan akurasi yang sama tingginya (Wijanarko et al., 2024). Studi berjudul "Implementasi YOLO dalam Deteksi Jumlah Kendaraan" menggunakan YOLOv8m (*medium*) untuk mendeteksi dan menghitung kendaraan dengan akurasi 93%. Penelitian ini memiliki kedekatan subjek dan menjadi dasar pengembangan lanjutan. Penelitian ini memperluas fungsionalitas tersebut dengan:
(1) menggunakan varian YOLOv8l (*large*) untuk akurasi lebih tinggi,
(2) menambahkan kelas orang, dan
(3) mengintegrasikan DeepSORT untuk bertransformasi dari sekadar sistem penghitungan menjadi sistem pelacakan (*tracking*) multiobjek (Mukhlish et al., 2024).

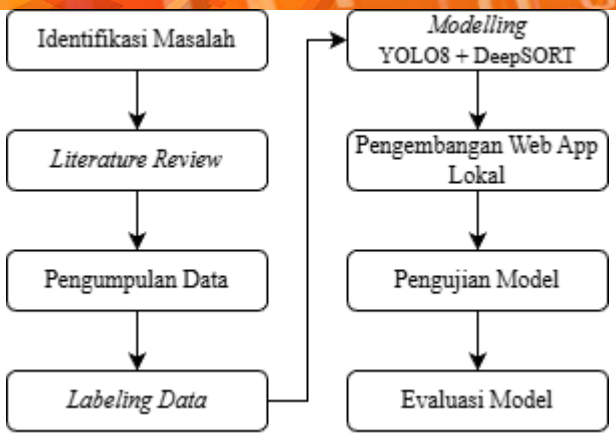
Terakhir, penelitian "YOLOv8-DeepSORT: *A High-Performance Framework for Real-Time Multi-Object Tracking with Attention and Adaptive Optimization"* membahas integrasi YOLOv8 dan DeepSORT untuk pelacakan multiobjek secara *real-time*. Sistem tersebut berhasil mencapai *Multiple Object Tracking Accuracy (MOTA)* sebesar 78,2% dan kecepatan pemrosesan *real-time* sebesar 28,6 FPS pada *benchmark* MOT20. Studi ini menjadi pijakan metodologis utama bagi penelitian yang sedang dilakukan, karena secara langsung memvalidasi efektivitas kombinasi teknologi yang sama (YOLOv8 dan DeepSORT) dalam menyelesaikan permasalahan pemantauan lalu lintas dan pergerakan pejalan kaki (Oise et al., 2025). Penelitian ini mengadopsi kombinasi YOLOv8l dan algoritma DeepSORT. Pemilihan YOLOv8l dilatarbelakangi oleh kemampuannya dalam menyeimbangkan akurasi dan kecepatan, menjadikannya model deteksi objek berkinerja tinggi yang efisien serta mampu melakukan pelokalan objek secara cepat dalam satu kali proses inferensi. Versi ini merupakan peningkatan dari generasi YOLO sebelumnya, dengan perbaikan pada akurasi, kecepatan, dan kemampuan mendeteksi berbagai jenis objek dalam kondisi lingkungan yang menantang.

Integrasi dengan DeepSORT dipilih karena algoritma ini merupakan pelacak objek tingkat lanjut yang menggabungkan pendekatan pelacakan tradisional dengan fitur penampakan berbasis pembelajaran mendalam. DeepSORT memanfaatkan fitur visual (*appearance features*) yang diekstraksi oleh model ReID serta informasi pergerakan untuk melacak objek antarframe, bahkan dalam skenario rumit seperti oklusi atau kerumunan. Selain itu, DeepSORT menggunakan metrik jarak kosinus untuk mencocokkan objek berdasarkan kemiripan penampakan, sehingga memastikan konsistensi identitas secara akurat. Karakteristik ini menjadikannya sangat ideal untuk aplikasi *real-time* seperti pemantauan lalu lintas, sistem pengawasan, dan kendaraan otonom karena kecepatan serta ketepatannya dalam mendeteksi dan melacak objek bergerak.

METODE PENELITIAN

Penelitian ini menggunakan pendekatan sistematis yang diawali dengan identifikasi masalah, kemudian dilanjutkan dengan *literature review*, pengumpulan dan pelabelan data. Tahap berikutnya meliputi proses *modelling*, pengembangan situs web lokal, pengujian model, serta evaluasi model, sebagaimana ditunjukkan pada Gambar 1.

e-ISSN 2828-0490
JURNAL
MEDIA COMPUTER SCIENCE



Gambar 1. Tahapan Penelitian

Tahap pertama, identifikasi masalah dilakukan dengan berfokus pada kebutuhan akan sistem deteksi objek otomatis yang akurat. Fokus utama penelitian ini adalah pendeteksian kendaraan di area lalu lintas serta pendeteksian orang atau pejalan kaki. Kemampuan deteksi tersebut diharapkan dapat dimanfaatkan untuk berbagai aplikasi praktis, seperti penghitungan jumlah kendaraan pada suatu ruas jalan untuk analisis kepadatan lalu lintas, serta penghitungan jumlah pejalan kaki atau pengunjung yang memasuki area tertentu, misalnya pusat perbelanjaan atau mall. Sistem ini ditujukan untuk menyediakan data kuantitatif yang akurat secara efisien, sebagai alternatif terhadap metode manual yang sering kali kurang efektif dan rentan terhadap kesalahan.

Tahap kedua, *literature review* mencakup telaah terhadap penelitian-penelitian terdahulu mengenai deteksi dan pelacakan objek, khususnya penelitian yang memanfaatkan algoritma *deep learning* seperti keluarga YOLO. Studi-studi tersebut menjadi landasan dalam pengembangan sistem deteksi kendaraan dan pejalan kaki untuk kebutuhan penghitungan lalu lintas maupun pengunjung.

Tahap ketiga, proses pengumpulan data merupakan langkah penting dalam melatih model deteksi objek. Data diperoleh dari berbagai rekaman video CCTV yang berada di wilayah Demak, Sukoharjo, Yogyakarta, dan Gunung Kidul. Selain itu, terdapat pula rekaman pejalan kaki yang direkam secara mandiri, serta cuplikan gambar yang diekstraksi dari platform YouTube. Koleksi data ini dilengkapi dengan gambar-gambar statis yang relevan, mencakup berbagai jenis kendaraan seperti mobil, sepeda motor, dan truk, serta objek manusia. Secara keseluruhan, terkumpul 414 gambar yang kemudian diproses lebih lanjut melalui tahapan pelabelan menggunakan platform Roboflow untuk memastikan anotasi objek yang presisi.

Tahap keempat, *labeling* data dilakukan dengan memanfaatkan platform Roboflow. Pada tahap ini, setiap objek terkait diidentifikasi dan diberikan anotasi berupa *bounding box* berdasarkan empat kelas yang telah ditetapkan. Dataset kemudian dibagi menjadi tiga bagian, yaitu *train*, *validation*, dan *test*. Seluruh gambar diubah ukurannya menjadi resolusi 640×640 piksel dan disesuaikan orientasinya (*auto-orient*). Untuk meningkatkan variasi serta memperkuat kemampuan generalisasi model, digunakan beberapa teknik augmentasi, seperti rotasi, penambahan *noise*, dan penyesuaian kecerahan. Setelah melalui proses ini, dataset siap digunakan dalam tahap pemodelan dengan memanfaatkan API Roboflow.

Tahap kelima, *modelling* dilakukan dengan melatih dataset beranotasi menggunakan arsitektur YOLOv8l. Pemilihan YOLOv8l didasarkan pada pertimbangan keseimbangan antara akurasi deteksi dan kecepatan inferensi, sehingga sangat cocok digunakan untuk aplikasi *real-time* seperti penghitungan kendaraan dan pejalan kaki. Proses pelatihan diimplementasikan menggunakan bahasa pemrograman Python dengan memanfaatkan sumber daya komputasi Google Colaboratory, khususnya akselerasi GPU T4, guna mempercepat proses pelatihan model secara efisien.

Tahap keenam, pengembangan Web App lokal, dilakukan dengan mengimplementasikan model deteksi objek yang telah dilatih ke dalam sebuah aplikasi web yang dapat dijalankan secara

lokal. Pengembangan aplikasi ini memanfaatkan *framework* Flask pada Python untuk membangun sisi *backend* serta mengelola logika pemrosesan data. Pada sisi antarmuka pengguna, digunakan teknologi standar web berupa HTML, CSS, dan JavaScript untuk menciptakan tampilan yang interaktif dan mudah digunakan, sehingga pengguna dapat menguji model secara langsung melalui aplikasi tersebut.

Tahap ketujuh, pengujian model, bertujuan untuk mengevaluasi kinerja deteksi dan pelacakan objek pada skenario nyata. Video rekaman CCTV dari Demak dan Yogyakarta digunakan sebagai data uji. Untuk memastikan pelacakan objek secara berkelanjutan dan mempertahankan identitas objek antar-*frame*, algoritma DeepSORT diintegrasikan dalam sistem. Dengan kemampuan *Re-Identification (Re-ID)*, DeepSORT memungkinkan pelacakan objek yang sama meskipun terjadi oklusi atau objek sempat keluar dari bidang pandang, sehingga penghitungan kendaraan dan pejalan kaki dapat dilakukan secara lebih akurat.

Tahap kedelapan, evaluasi model, merupakan tahap penting untuk menilai efektivitas keseluruhan sistem deteksi objek yang dikembangkan. Pada tahap ini, *confusion matrix* digunakan sebagai alat utama untuk menganalisis performa model secara mendalam. Berdasarkan *confusion matrix* tersebut, berbagai metrik evaluasi dihitung, termasuk akurasi (*accuracy*), presisi (*precision*), *recall*, dan skor-F1 (*F1-score*). Metrik-metrik ini memberikan gambaran komprehensif mengenai kemampuan model dalam mengidentifikasi objek secara benar, mengurangi kesalahan deteksi, serta menjaga keseimbangan antara deteksi positif yang tepat dan kemampuan menangkap objek yang seharusnya terdeteksi.

HASIL DAN PEMBAHASAN

Hasil

Masalah utama yang diangkat dalam penelitian ini adalah keterbatasan metode konvensional dalam memantau objek bergerak di ruang publik. Penghitungan jumlah kendaraan untuk analisis kepadatan lalu lintas di jalan raya, serta pemantauan pergerakan pejalan kaki di area komersial, hingga kini masih banyak bergantung pada pengamatan manual. Pendekatan tersebut pada hakikatnya tidak efisien, memerlukan waktu yang cukup lama, dan memiliki tingkat akurasi rendah karena sangat rentan terhadap kesalahan manusia (*human error*). Selain sekadar menghitung objek, tantangan yang lebih kompleks muncul ketika diperlukan kemampuan untuk melacak setiap kendaraan atau individu secara unik, terutama ketika terjadi tumpang tindih atau objek sempat menghilang dari bidang pandang (oklusi). Untuk menjawab permasalahan tersebut, penelitian ini menawarkan solusi otomatis melalui integrasi arsitektur YOLOv8l, yang dikenal unggul dalam mendeteksi objek, dengan algoritma DeepSORT yang mampu melakukan pelacakan (*tracking*) sekaligus *re-identification* sehingga setiap objek dapat mempertahankan ID yang konsisten selama proses pemantauan.

Pengembangan sistem ini didasari oleh landasan teoretis dan teknis yang diperoleh dari berbagai penelitian terdahulu. Kajian literatur yang telah dipaparkan pada bagian tinjauan pustaka berfungsi sebagai fondasi utama yang memberikan justifikasi kuat terhadap pemilihan arsitektur YOLOv8l dan algoritma DeepSORT. Beragam jurnal yang menjadi acuan tidak hanya menyediakan validasi metodologis, tetapi juga menjadi sumber inspirasi dalam merancang dan mengembangkan sistem yang mampu mengatasi tantangan spesifik dalam deteksi dan pelacakan kendaraan maupun pejalan kaki secara efektif.

Dalam penelitian ini, dataset primer dikumpulkan dengan memanfaatkan rekaman video CCTV publik yang tersedia melalui platform YouTube, mencakup pemantauan lalu lintas di beberapa wilayah seperti Yogyakarta, Demak, dan Sukoharjo. Untuk meningkatkan keragaman visual, dataset ini juga diperkaya dengan sejumlah gambar yang diambil secara mandiri. Video-video tersebut kemudian diekstraksi menjadi *frame* gambar tunggal yang menangkap objek target, yaitu truk,

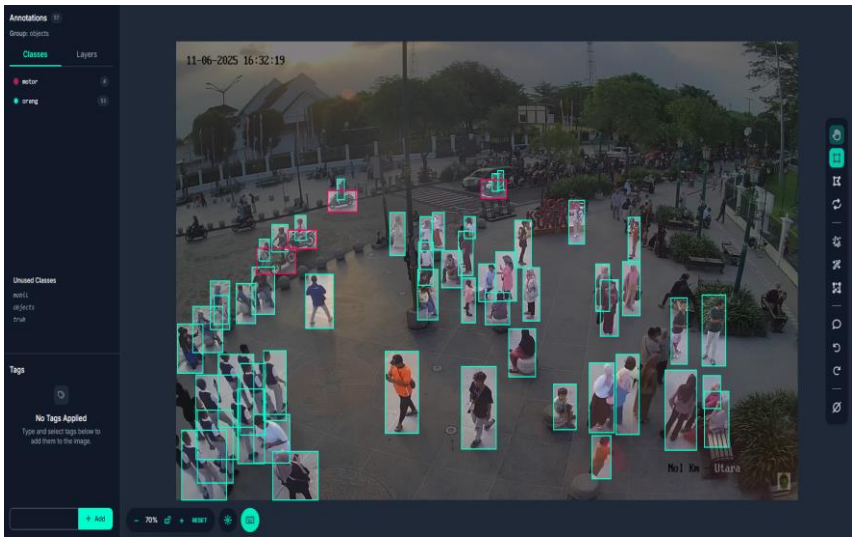
e-ISSN 2828-0490
JURNAL
MEDIA COMPUTER SCIENCE

mobil, motor, dan orang. Melalui proses ini, berhasil dikumpulkan sebanyak 414 gambar yang siap diproses lebih lanjut pada tahap pelabelan, dengan contoh dataset ditampilkan pada Gambar 2.

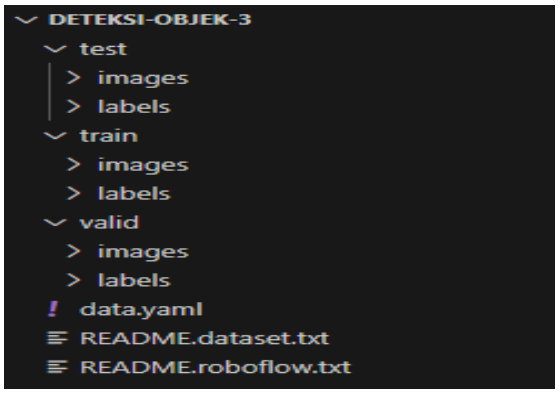
Gambar 2. Dataset Mentah

Setelah data berhasil dikumpulkan, tahap berikutnya adalah proses pelabelan yang dilakukan sepenuhnya menggunakan platform Roboflow. Setiap gambar dalam dataset dianotasi secara manual menggunakan *bounding box* untuk mengidentifikasi objek sesuai empat kelas yang telah ditetapkan, yaitu mobil, motor, truk, dan orang. Setelah proses pelabelan selesai, dataset secara otomatis dibagi menjadi tiga bagian dengan proporsi 70% untuk data latih (*train*), 20% untuk data validasi (*validation*), dan 10% untuk data uji (*test*).

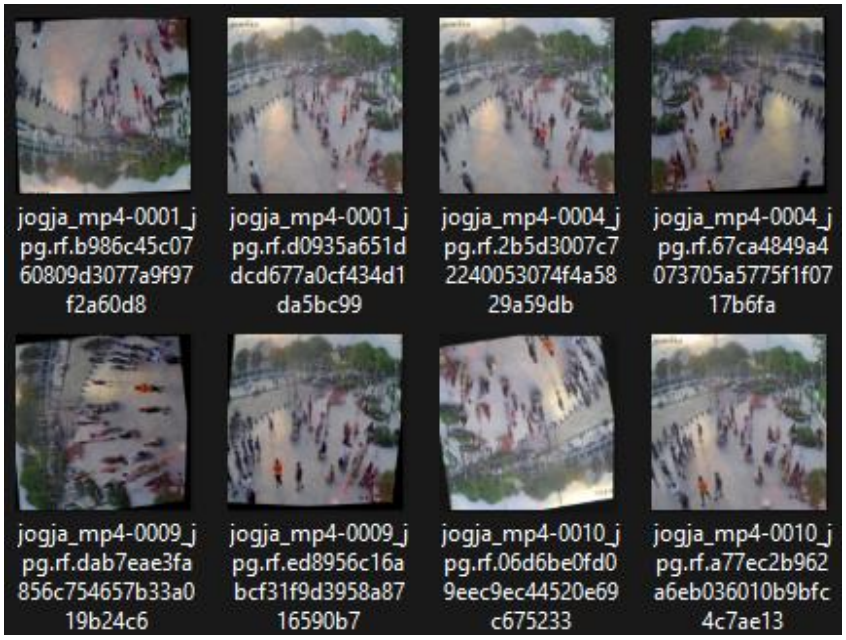
Sebelum dataset siap digunakan pada tahap pemodelan, dilakukan proses pra-pemrosesan yang meliputi pengaktifan fitur *auto-orient* dan penyesuaian resolusi seluruh gambar menjadi 640×640 piksel. Untuk meningkatkan keragaman data serta memperkuat kemampuan generalisasi model, diterapkan berbagai teknik augmentasi seperti *flip*, rotasi, *crop*, rotasi 90 derajat, *shear*, penyesuaian *hue*, kecerahan (*brightness*), *exposure*, serta penambahan efek *blur* dan *noise*. Melalui rangkaian proses tersebut, jumlah dataset meningkat hingga mencapai 994 gambar yang siap digunakan dalam tahap pemodelan. Proses pelabelan, struktur folder pembagian dataset, dan hasil akhir augmentasi dataset masing-masing ditunjukkan pada Gambar 3, Gambar 4, dan Gambar 5.



Gambar 3. Proses Pelabelan Gambar



Gambar 4. Struktur Folder Pembagian Dataset



Gambar 5. Dataset Akhir

Tahap pemodelan difokuskan pada proses pelatihan model yang dijalankan sepenuhnya di lingkungan Google Colaboratory untuk memanfaatkan kemampuan komputasi GPU, khususnya GPU NVIDIA T4. Langkah pertama adalah menyiapkan lingkungan kerja dengan menginstal berbagai library esensial, seperti Roboflow beserta Pillow versi 9.5.0, Ultralytics versi 8.0.3, serta PyTorch versi 2.3.1 yang telah dioptimalkan untuk akselerasi CUDA. Setelah seluruh dependensi terpasang, dataset yang telah diproses sebelumnya diimpor secara langsung melalui API Roboflow.

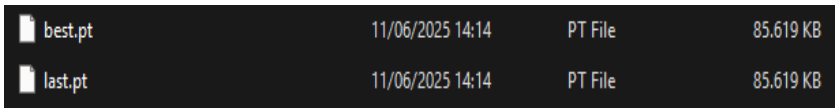
Proses pelatihan model dilakukan menggunakan bobot dasar *yolov8l.pt*, dengan pengaturan 100 *epochs* dan ukuran input gambar sebesar 640 piksel. Penentuan jumlah *epochs* tersebut dipilih sebagai titik awal yang memadai, mengingat ukuran dataset yang relatif terbatas. Keseluruhan skrip pelatihan yang digunakan dalam tahap pemodelan ditampilkan pada Kode Program 1.

Kode Program 1 Pelatihan Model

```
1. BEGIN ModelTrainingProcess
2.
3.   // 1. Inisialisasi Lingkungan
4.   Initialize Environment
5.   Install required libraries:
6.     - Roboflow (untuk akuisisi dataset)
7.     - Ultralytics (untuk framework YOLOv8)
```

```
8.    - PyTorch (sebagai backend deep learning)
9.
10.  // 2. Akuisisi Dataset dari Roboflow
11.  Import Roboflow tool
12.
13.  Authenticate with Roboflow service
14.    Set api_key = "lcaM99QwklYLBHEZNYL1"
15.
16.  Access Roboflow project
17.    Set workspace_name = "daniel-mahardika"
18.    Set project_name = "deteksi-objek-crcu3"
19.    Set project_version = 3
20.
21.  Download dataset from project version
22.    Set download_format = "yolov8"
23.
24.  // 3. Persiapan Pelatihan
25.  Change current directory to the main working directory (HOME)
26.
27.  // 4. Eksekusi Pelatihan Model
28.  Execute training script (train.py) with parameters:
29.    Set base_model = "yolov8l.pt"
30.    Set data_config_file = "Deteksi-Objek-3/data.yaml"
31.    Set number_of_epochs = 100
32.    Set image_input_size = 640
33.
34. END ModelTrainingProcess
```

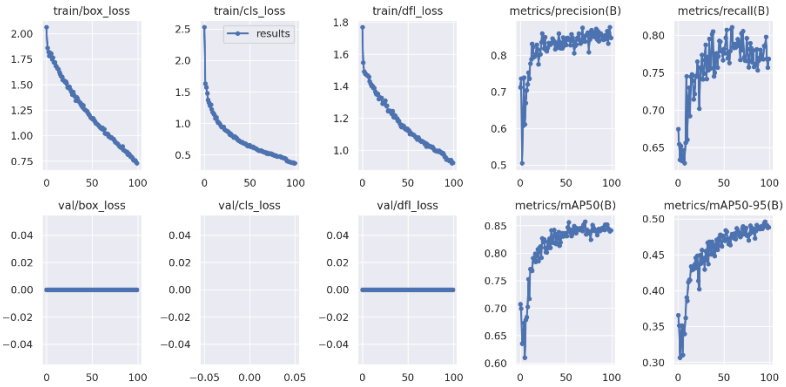
Sebagai hasil dari proses pelatihan yang dilakukan, sistem menghasilkan file bobot model (*model weights*) yang secara otomatis disimpan dalam direktori "weights". Dari proses tersebut diperoleh dua keluaran utama, yakni "last.pt", yang merupakan bobot pada *epoch* terakhir, serta "best.pt", yang mewakili bobot dengan performa terbaik berdasarkan evaluasi pada data validasi. Dalam penelitian ini, file "best.pt" dipilih sebagai model utama untuk implementasi selanjutnya karena memiliki kinerja paling optimal dalam mengenali empat kelas objek, yaitu orang, truk, motor, dan mobil. Kedua hasil bobot model tersebut ditampilkan pada Gambar 6.



Gambar 6 Model Hasil Pelatihan

Analisis terhadap hasil pelatihan model selama 100 *epochs* menunjukkan perkembangan yang sangat positif dan konvergen. Hal ini terlihat dari kurva kerugian (*loss*), di mana nilai val/box_loss (kerugian lokalisasi) dan val/cls_loss (kerugian klasifikasi) menurun secara konsisten, menandakan bahwa model mampu mempelajari pola data dengan baik. Pada akhir pelatihan, val/box_loss stabil di kisaran 0,656, sedangkan val/cls_loss turun hingga 0,458. Penurunan kerugian ini sejalan dengan peningkatan berbagai metrik performa.

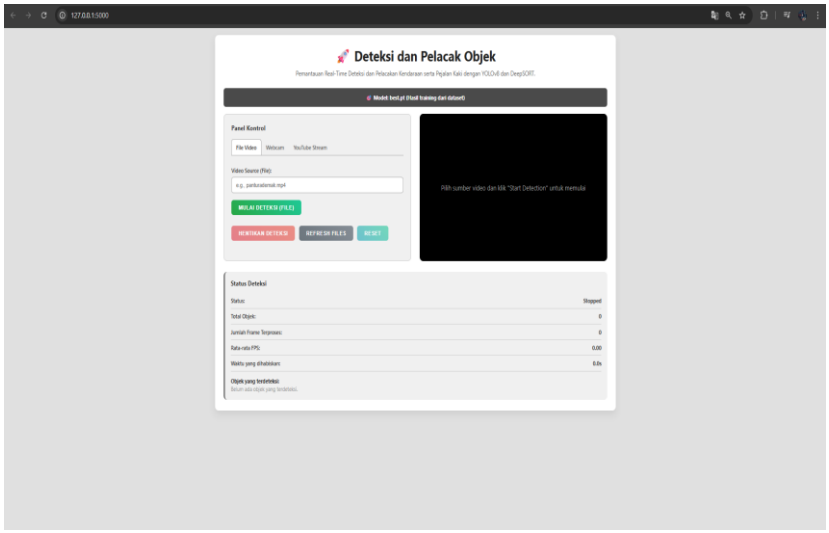
Metrik *mean Average Precision* pada ambang batas 0.5 *(mAP50(B))* mencapai nilai puncak 0,933, menunjukkan tingkat akurasi deteksi yang sangat baik. Untuk standar evaluasi yang lebih ketat *(mAP50–95(B))*, model memperoleh skor 0,735. Kinerja tersebut turut diperkuat oleh nilai precision sebesar 0,932 dan recall sebesar 0,882 pada akhir pelatihan. Konsistensi peningkatan pada seluruh metrik ini mengonfirmasi bahwa proses pelatihan telah berjalan secara efektif dan stabil. Grafik hasil pelatihan ditunjukkan pada Gambar 7.



Gambar 7 Grafik *Training*

Sebagai tahap implementasi akhir, sebuah aplikasi web lokal dikembangkan dengan memanfaatkan *framework* Flask (Python) sebagai backend, serta antarmuka interaktif berbasis HTML, CSS, dan JavaScript. Fokus utama antarmuka ini adalah sebuah kotak penampil video yang memungkinkan pengguna memantau proses deteksi objek secara langsung. Untuk memberikan kendali penuh kepada pengguna, aplikasi dilengkapi berbagai tombol fungsional, seperti *start detection*, *stop detection*, *refresh files*, dan *reset*. Selain itu, mekanisme *error handling* turut disertakan untuk memberikan notifikasi apabila terjadi kendala, seperti input tidak valid atau kamera yang tidak terdeteksi.

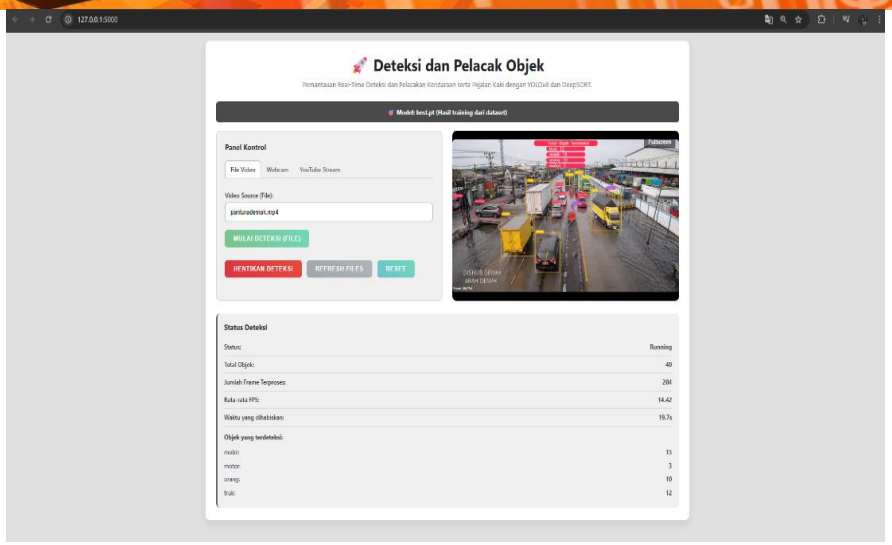
Selama proses deteksi berlangsung, aplikasi menghadirkan panel status yang menyajikan data telemetri secara real-time, meliputi status deteksi, total objek terhitung, jumlah frame yang telah diproses, rata-rata *frame per second* (FPS), waktu proses, serta daftar objek yang berhasil terdeteksi. Aplikasi yang didukung oleh model best.pt ini mampu melakukan deteksi dari berbagai sumber mulai dari file video, webcam, hingga tautan YouTube berkat integrasi FFmpeg. Dengan fitur yang komprehensif tersebut, aplikasi web ini menjadi alat pengujian yang efektif untuk tahap evaluasi model berikutnya. Tampilan antarmuka aplikasi web ditunjukkan pada Gambar 8.



Gambar 8. Tampilan Antarmuka Web Lokal

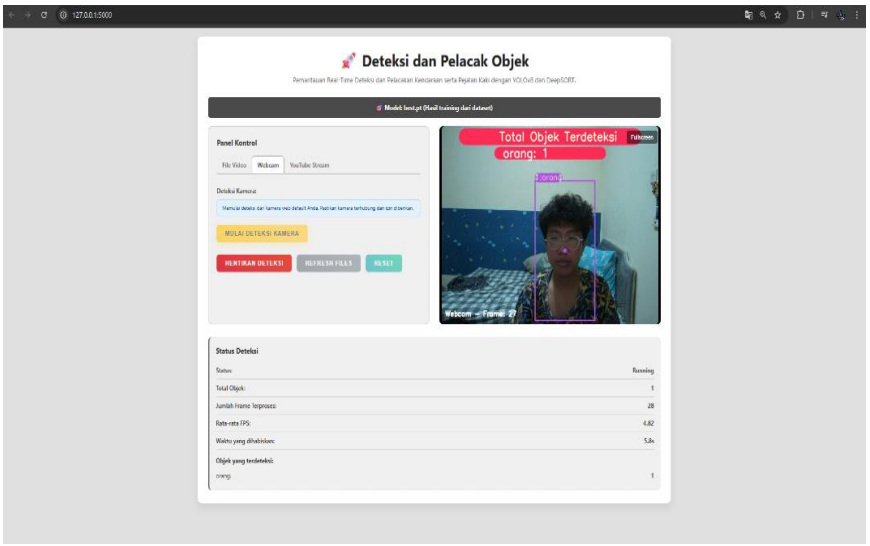
Pengujian awal model dilakukan menggunakan aplikasi web lokal yang telah dikembangkan. Cuplikan video CCTV dari DISHUB arah Demak digunakan sebagai sumber input pertama. Hasil pengujian menunjukkan performa deteksi yang cukup baik; model mampu mengidentifikasi berbagai kelas objek yang relevan, termasuk orang, mobil, motor, dan truk yang muncul dalam rekaman tersebut. Tampilan hasil deteksi pada aplikasi web dapat dilihat pada Gambar 9.

e-ISSN 2828-0490
JURNAL
MEDIA COMPUTER SCIENCE



Gambar 9. Pengujian Model dengan File Video

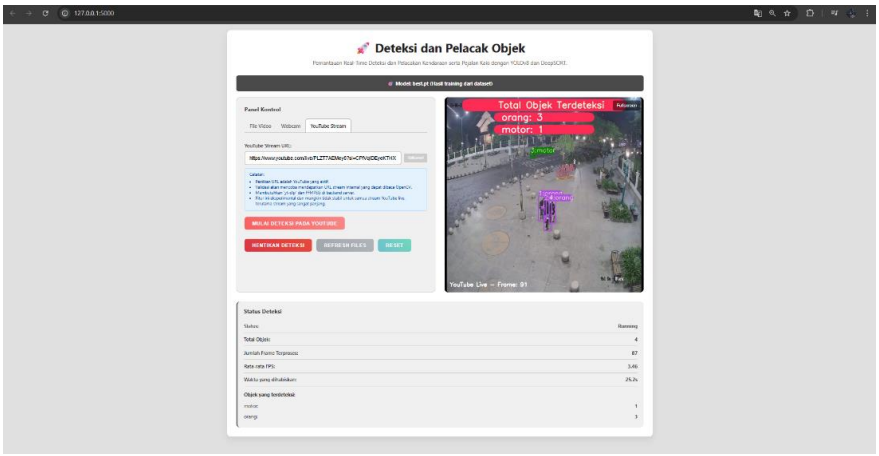
Pengujian selanjutnya dilakukan untuk memvalidasi kemampuan deteksi secara *real-time* menggunakan input langsung dari kamera. Pada skenario ini, aplikasi web mampu mengakses dan memproses siaran dari webcam dengan lancar dan stabil. Model juga berhasil mendeteksi serta memberikan label pada objek orang yang tertangkap di depan kamera secara akurat. Hasil deteksi menggunakan input kamera ditampilkan pada Gambar 10.



Gambar 10. Pengujian Model dengan Kamera

Skenario pengujian ketiga dilakukan untuk memvalidasi fungsionalitas yang lebih kompleks, yaitu kemampuan deteksi dari siaran langsung (*live stream*) YouTube. Pada uji coba ini, digunakan tautan CCTV Nol Km – Utara yang disediakan oleh Pemkot Yogyakarta. Dengan dukungan FFmpeg yang terintegrasi pada backend, aplikasi dapat menarik dan memproses aliran video tersebut secara stabil.

Hasil pengujian menunjukkan bahwa model mampu bekerja secara efektif pada lingkungan yang ramai dan dinamis, terlihat dari keberhasilannya mendeteksi kerumunan orang serta beberapa unit motor yang melintas. Pengujian ini sekaligus membuktikan bahwa konsep deteksi real-time dalam penelitian ini dapat diterapkan dengan baik. Hasil pengujian dari siaran langsung YouTube ditampilkan pada Gambar 11.



Gambar 11. Pengujian Model dengan *Platform* YouTube

Berdasarkan seluruh rangkaian pengujian melalui aplikasi web, dapat disimpulkan bahwa sistem deteksi dan pelacakan objek telah berfungsi dengan baik dan sesuai ekspektasi. Deteksi objek pada berbagai sumber input berjalan lancar, sementara algoritma DeepSORT juga berhasil diimplementasikan secara fungsional. Keberhasilan ini terlihat dari kemampuannya memberikan ID unik (*re-identification*) yang konsisten pada setiap objek yang terdeteksi, baik kendaraan maupun pejalan kaki. Seluruh pengujian dijalankan pada perangkat laptop dengan spesifikasi CPU Ryzen 7 6800H, GPU RTX 3050 Laptop (VRAM 4GB), dan RAM 24GB. Selama pengujian, kendala utama yang ditemukan adalah performa *Frame Per Second* (FPS) yang kurang stabil. Fluktuasi ini sangat dipengaruhi oleh kemampuan perangkat keras, mengingat model YOLOv8l memiliki kompleksitas tinggi, ditambah beban komputasi dari algoritma DeepSORT yang memerlukan proses pelacakan secara kontinu. Meskipun demikian, sistem secara keseluruhan tetap berjalan cukup optimal dan stabil, dengan catatan khusus pada tingkat FPS.

Setelah pengujian fungsional, penelitian dilanjutkan ke tahap evaluasi terhadap hasil pelatihan dan pengujian model YOLOv8l tahap krusial untuk menilai efektivitas sistem secara kuantitatif. Evaluasi performa dilakukan menggunakan *confusion matrix* sebagai alat utama untuk menganalisis kualitas prediksi model. *Confusion matrix* memetakan hasil prediksi terhadap label sebenarnya, sehingga memberikan gambaran jelas mengenai keberhasilan maupun kesalahan klasifikasi yang terjadi. Berdasarkan data ini, dihitung beberapa metrik evaluasi penting seperti *precision*, *recall*, *F1-score*, dan *accuracy*. Keempat metrik tersebut memberikan perspektif komprehensif mengenai kemampuan model dalam mengidentifikasi objek secara tepat, meminimalkan kesalahan, serta menyeimbangkan antara deteksi yang benar dan kelengkapan identifikasi objek. *Confusion matrix* yang menjadi dasar analisis ditampilkan pada Tabel 1.

Pembahasan

Tabel 1. *Confusion Matrix*

Diprediksi (*Predicted*)	mobil (Aktual)	motor (Aktual)	orang (Aktual)	truk (Aktual)
mobil	86	1	0	1
motor	0	84	0	0
orang	0	0	72	0
truk	2	0	0	84

Untuk menginterpretasikan kinerja yang ditampilkan dalam *confusion matrix*, analisis dilakukan berdasarkan empat komponen dasar hasil klasifikasi. Pertama, *True Positive* (TP), yaitu kondisi ideal ketika sistem berhasil mendeteksi objek dan mengklasifikasikannya sesuai dengan label aslinya. Kedua, *True Negative* (TN), yaitu ketika sistem secara tepat mengenali area non-objek

(misalnya latar belakang) dan tidak memberikan klasifikasi yang keliru. Komponen ketiga adalah *False Positive* (FP) atau kesalahan Tipe I, yakni situasi ketika sistem mendeteksi suatu objek padahal objek tersebut tidak ada pada kenyataannya. Terakhir, *False Negative* (FN) atau kesalahan Tipe II, yaitu kondisi ketika sistem gagal mendeteksi objek yang sebenarnya muncul dalam gambar. Keempat komponen inilah yang menjadi dasar penghitungan berbagai metrik kinerja pada tahap evaluasi.

Berdasarkan analisis kuantitatif terhadap confusion matrix, performa model YOLOv8l untuk masing-masing kelas menunjukkan hasil yang sangat kuat. Pada kelas 'orang', model berhasil mencapai hasil yang paling ideal dengan 72 deteksi benar (*True Positive*) tanpa satu pun kesalahan prediksi, ditandai oleh nilai 0 pada *False Positive* maupun *False Negative*, serta 258 prediksi *True Negative*. Kelas 'motor' juga menunjukkan performa yang sangat baik dengan 84 *True Positive* dan 245 *True Negative*, tanpa adanya kasus *False Negative* dan hanya 1 kasus *False Positive*. Kelas 'truk' mencatatkan 84 *True Positive* dan 243 *True Negative*, dengan kesalahan minimal berupa 1 *False Positive* dan 2 *False Negative*. Sementara itu, kelas 'mobil' menunjukkan kinerja yang solid dengan 86 *True Positive* dan 240 *True Negative*, serta tingkat kesalahan yang rendah, yaitu hanya 2 kasus *False Positive* dan 2 kasus *False Negative*.

Selanjutnya, metrik *accuracy* atau akurasi digunakan sebagai salah satu indikator utama untuk menilai performa model secara keseluruhan. Metrik ini merepresentasikan proporsi prediksi yang benar dibandingkan dengan seluruh prediksi yang dihasilkan model. Dengan kata lain, *accuracy* menggambarkan seberapa sering model membuat klasifikasi yang tepat pada seluruh dataset. Oleh karena itu, nilai accuracy yang tinggi mencerminkan bahwa sistem bekerja secara efektif dan andal dalam mengenali objek. Rumus matematis untuk menghitung nilai akurasi ditunjukkan pada Persamaan (1).

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} = \frac{86 + 84 + 72 + 84}{86 + 84 + 72 + 84 + 1 + 0 + 0 + 2} = \frac{326}{330} = 0.988 \quad (1)$$

Berbeda dengan accuracy yang menilai kinerja model secara keseluruhan, metrik *precision* atau presisi memberikan penilaian yang lebih terfokus pada kualitas setiap deteksi positif yang dihasilkan. Secara prinsip, *precision* mengukur tingkat keandalan model dalam menyatakan bahwa suatu objek benar-benar terdeteksi. Metrik ini menghitung proporsi prediksi positif yang memang tepat sesuai dengan kondisi sebenarnya. Nilai *precision* yang tinggi menunjukkan bahwa model memiliki jumlah *False Positive* yang rendah, sehingga dapat dipercaya ketika mengidentifikasi keberadaan suatu objek. Rumus matematis untuk menghitung nilai presisi ditunjukkan pada Persamaan (2).

$$Precision = \frac{TP}{TP + FP} = \frac{86}{86+2} + \frac{84}{84+1} + \frac{72}{72+0} + \frac{84}{84+1} = \frac{86}{88} + \frac{84}{85} + \frac{72}{72} + \frac{84}{85} = 0.988 \quad (2)$$

Metrik recall berfungsi sebagai pelengkap bagi precision dengan menilai tingkat keberhasilan atau sensitivitas model dalam menemukan seluruh objek yang relevan. Secara lebih spesifik, *recall* mengukur proporsi deteksi benar dibandingkan dengan jumlah total objek yang seharusnya terdeteksi. Nilai *recall* yang tinggi menunjukkan rendahnya tingkat kegagalan deteksi (*False Negative*), yang berarti model mampu menjangkau targetnya secara lebih menyeluruh. Rumus perhitungan *recall* ditunjukkan pada Persamaan (3).

$$Recall = \frac{TP}{TP + FN} = \frac{86}{86+2} + \frac{84}{84+0} + \frac{72}{72+0} + \frac{84}{84+2} = \frac{86}{88} + \frac{84}{84} + \frac{72}{72} + \frac{84}{86} = 0.988 \quad (3)$$

F1-Score merupakan metrik terpadu yang menyeimbangkan *precision* dan *recall* melalui perhitungan rata-rata harmonik dari keduanya. Metrik ini menghasilkan satu nilai komprehensif yang memberikan bobot setara pada upaya meminimalkan kesalahan identifikasi (*False Positive*) maupun kegagalan deteksi (*False Negative*). Dengan demikian, *F1-Score* sangat efektif digunakan

untuk menilai kinerja model secara menyeluruh dalam satu ukuran yang ringkas. Rumus perhitungannya ditunjukkan pada Persamaan (4).

$$F1\ \text{Score} = 2 \times \frac{\text{precision} \times \text{Recall}}{\text{precision} + \text{Recall}} = 2 \times \frac{0.988 \times 0.988}{0.988 + 0.988} = 2 \times \frac{0.976}{1.976} = 0.988 \quad (4)$$

KESIMPULAN DAN SARAN

Kesimpulan

Berdasarkan rangkaian penelitian yang telah dilakukan, dapat disimpulkan bahwa sistem deteksi dan pelacakan kendaraan berbasis arsitektur YOLOv8l dan algoritma DeepSORT berhasil dikembangkan dengan performa yang sangat kuat dan andal. Hasil evaluasi kuantitatif menunjukkan bahwa model mencapai nilai *accuracy*, *precision*, *recall*, dan *F1-score* sebesar 0.98, yang menegaskan konsistensi dan efektivitas sistem dalam mengidentifikasi serta melacak objek secara tepat. Keberhasilan ini semakin diperkuat melalui pengujian pada berbagai jenis sumber input, meliputi *file* video, kamera webcam, dan siaran langsung YouTube.

Pengujian tersebut membuktikan bahwa sistem dapat berjalan dengan baik dalam kondisi yang beragam, mendeteksi objek secara *real-time*, dan mempertahankan identitas objek secara konsisten berkat integrasi DeepSORT.Kendati demikian, penelitian ini juga mengungkap adanya keterbatasan pada stabilitas *Frame Per Second* (FPS), yang cenderung fluktuatif selama proses deteksi berlangsung.

Ketidakstabilan ini terutama disebabkan oleh beban komputasi model YOLOv8l yang cukup besar serta proses pelacakan tambahan dari algoritma DeepSORT. Sebagai tindak lanjut untuk penyempurnaan sistem, terdapat beberapa rekomendasi pengembangan. Pertama, perlu dilakukan penambahan jumlah dan variasi data pada dataset untuk meningkatkan kemampuan generalisasi model. Kedua, stabilitas FPS perlu dioptimalkan melalui penyetelan model, pengurangan kompleksitas komputasi, atau dengan memanfaatkan perangkat keras yang lebih mumpuni.

Ketiga, sistem perlu di-*deploy* ke lingkungan produksi agar performanya dapat diuji secara langsung dalam kondisi nyata. Dengan berbagai temuan dan rekomendasi tersebut, penelitian ini diharapkan dapat menjadi dasar pengembangan sistem deteksi dan pelacakan objek yang lebih efektif, efisien, dan aplikatif di masa mendatang.

Saran

Untuk pengembangan selanjutnya, disarankan untuk menambah jumlah dan keragaman data guna meningkatkan kemampuan generalisasi model. Stabilitas FPS juga perlu dioptimalkan melalui penyetelan model, pengurangan beban komputasi, atau penggunaan perangkat keras yang lebih mumpuni. Selain itu, sistem perlu diuji pada lingkungan nyata agar performanya dapat divalidasi secara praktis. Pengembangan fitur analitik lanjutan, seperti perhitungan kepadatan lalu lintas atau analisis lintasan, juga dapat dipertimbangkan untuk meningkatkan fungsi sistem.

DAFTAR PUSTAKA

Abuzairi, T., Widanti, N., Kusumaningrum, A., & Rustina, Y. (2021). Implementasi Convolutional Neural Network Untuk Deteksi Nyeri Bayi Melalui Citra Wajah Dengan YOLO. J. RESTI (Rekayasa Sist. Dan Teknol. Inf.), 5(4), 624–630. https://doi.org/10.29207/resti.v5i4.3184

Fahrezi, M. A., & Widiyanto, E. P. (2024). Implementasi YOLOv8 Dalam Penghitung Masuk Dan Keluar Manusia Pada Gedung. Jurnal Teknik Informatika Dan Sistem Informasi, 11(3). https://doi.org/10.35957/jatisi.v11i3.9050

Gelar Guntara, R. (2023). Pemanfaatan Google Colab Untuk Aplikasi Pendeteksian Masker Wajah Menggunakan Algoritma Deep Learning YOLOv7. Jurnal Teknologi Dan Sistem Informasi Bisnis, 5(1), 55–60. https://doi.org/10.47233/jteksis.v5i1.750

Hayati, N. J., Singasatia, D., & Muttaqin, M. R. (2023a). Object Tracking Menggunakan Algoritma You Only Look Once (YOLO)v8 untuk Menghitung Kendaraan. Komputa : Jurnal Ilmiah Komputer Dan Informatika, 12(2), 91–99. https://doi.org/10.34010/komputa.v12i2.10654

Hayati, N. J., Singasatia, D., & Muttaqin, M. R. (2023b). Object Tracking Menggunakan Algoritma You Only Look Once (YOLO)v8 untuk Menghitung Kendaraan. Komputa : Jurnal Ilmiah Komputer Dan Informatika, 12(2), 91–99.

Herdianto, H., Hafni, H., Nasution, D., & Ramadhan, S. (2024). Implementasi Metode Yolo pada Deteksi Objek Manusia. Jurnal Manajemen Informatika & Komputerisasi Akuntansi, 8(2), 234–240. https://doi.org/10.46880/jmika.Vol8No2.pp234-240

Husnan, H., Fatichah, C., & Dikairono, R. (2023). Deteksi Objek Menggunakan Metode YOLO dan Implementasinya pada Robot Bawah Air. J. Tek. ITS, 12(3). https://doi.org/10.12962/j23373539.v12i3.122326

Indaryanto, F., Nugroho, A., & Suni, A. F. (2021). Aplikasi Penghitung Jarak dan Jumlah Orang Berbasis YOLO Sebagai Protokol Kesehatan Covid-19. Edu Komputika J., 8(1), 31–38. https://doi.org/10.15294/edukomputika.v8i1.47837

Maulidiansyah, M., & Yaqin, M. A. (2023). Deteksi Tumpukan Sampah dengan Metode You Only Look Once (YOLO). TRILOGI: Jurnal Ilmu Teknologi, Kesehatan, Dan Humaniora, 4(2), 76–79. https://doi.org/10.33650/trilogi.v4i2.6185

Mukhlish, N. S. A., Wulanningrum, R., & Sanjaya, A. (2024). Implementasi YOLO Dalam Deteksi Jumlah Kendaraan. Prosiding SEMNAS INOTEK (Seminar Nasional Inovasi Teknologi), 8(3), 1274–1281. https://doi.org/10.29407/r5e52h49

Oise, G. P., Unuigbokhai, N. B., Onwuzo, C. J., Nwabuokei, O. C., Ejenarhome, P. O., Atake, O. M., & Bakare, S. K. (2025). YOLOv8-DeepSORT: A high-performance framework for real-time multi-object tracking with attention and adaptive optimization. Journal of Science Research and Reviews, 2(2), 92–100. https://doi.org/10.70882/josrar.2025.v2i2.50

Pakpahan, R. (2021). ANALISA PENGARUH IMPLEMENTASI ARTIFICIAL INTELLIGENCE DALAM KEHIDUPAN MANUSIA. Journal of Information System, Informatics and Computing Issue Period, 5(2), 506–513. https://doi.org/10.52362/jisicom.v5i2.616

Pereira, R., Carvalho, G., & Garrote Lu\'\is and Nunes, U. J. (2022). Sort and Deep-SORT based multi-Object Tracking for mobile robotics: Evaluation with new data association metrics. Appl. Sci. (Basel), 12(3), 1319. https://doi.org/10.3390/app12031319

Purnama, B., Setyorini, S., Insanudin, E., Ismail, I., Labib, F., & Furqoon, N. S. (2024). Implementasi Computer Vision untuk Deteksi Truk. Jurnal Pengembangan Dan Pengabdian Masyarakat Multikultural, 2(1), 17–23. https://doi.org/10.57152/batik.v2i1.1223

Rosanti, N., Latifah, R., Munir, S., & Maududi, I. A. Q. (2024). Pengaruh Jarak Objek Citra pada Model Deteksi dan Klasifikasi Botol Plastik menggunakan YOLO. J. Teknologi Terpadu, 10(1), 63–69. https://doi.org/10.54914/jtt.v10i1.1247

Sheng, W., Shen, J., Huang, Q., Liu, Z., & Ding, Z. (2024). Multi-objective pedestrian tracking method based on YOLOv8 and improved DeepSORT. Math. Biosci. Eng., 21(2), 1791–1805. https://doi.org/10.3934/mbe.2024077

Susanto, A., Mulyono, I. U. W., & Sudaryanto, S. (2024). DETEKSI GERAK PADA VIDEO GERAK

LANSIA BERBASIS YOLO-V5 DAN YOLO-V7. Semnasristek, 8(01). https://doi.org/10.30998/semnasristek.v8i01.7142

Wijanarko, R. G., Pradana, A. I., & Hartanti, D. (2024). IMPLEMENTASI DETEKSI DRONE MENGGUNAKAN YOLO (You Only Look Once). JURNAL FASILKOM, 14(2), 437–442. https://doi.org/10.37859/jf.v14i2.7374