

PERFORMANCE EVALUATION OF YOLOV8 IN REAL-TIME VEHICLE DETECTION IN VARIOUS ENVIRONMENTAL CONDITIONS

Derit Junio Marcelleno^{*1}, Muhammad Pajar Kharisma Putra²

^{1,2}Informatics, Faculty of Engineering and Computer Science, Universitas Teknokrat Indonesia, Indonesia
Email: derit_junio_marcelleno@teknokrat.ac.id, pajarkharisma@teknokrat.ac.id

(Article received: October 09, 2024; Revision: November 03, 2024; published: February 20, 2025)

Abstract

This research focuses on assessing and developing a real-time detection system using the YOLOv8 algorithm. Accurate and fast vehicle detection is a big challenge in modern traffic management, especially in various environmental conditions such as bad weather, low lighting, and high traffic density. The aim of this study was to evaluate the performance of YOLOv8 under these conditions and identify potential improvements. The dataset used consists of 16,990 vehicle images with various variations and environmental conditions. After being trained, the model is evaluated using metrics such as precision, recall, and F1-score, as well as Intersection over Union (IoU) with a threshold of 0.8 on IoU. The results show that YOLOv8 is superior with a fairly high detection accuracy of 78%, with precision of 82% and recall above 90%, and is able to detect vehicles in real-time conditions. However, the challenge of detecting small objects or irregularly shaped vehicles such as tractors still needs to be optimized. This research also compared the performance of YOLOv8 with the SSD (Single Shot Detector) algorithm, where YOLOv8 was proven to be superior in terms of accuracy, precision, recall and F1-score. The research results obtained provide valuable insights for the development of traffic management systems based on deep learning technology. The main contribution of this research is to provide a more efficient and effective vehicle detection solution, which can be applied in modern traffic management systems. Thus, it is hoped that the results of this research can increase the efficiency of traffic management and have a positive impact on the development of intelligent transportation systems in the future.

Keywords: Deep Learning, Intersection over Union (IoU), Real-Time Detection, Vehicle Detection, YOLO.

EVALUASI PERFORMA YOLOV8 DALAM DETEKSI KENDARAAN REAL-TIME PADA BERBAGAI KONDISI LINGKUNGAN

Abstrak

Penelitian ini berfokus pada penilaian dan pengembangan sistem deteksi real-time dengan menggunakan algoritma YOLOv8. Deteksi kendaraan secara akurat dan cepat merupakan tantangan besar dalam manajemen lalu lintas modern, terutama di berbagai kondisi lingkungan seperti cuaca buruk, pencahayaan rendah, dan kepadatan lalu lintas tinggi. Tujuan dari penelitian ini adalah untuk mengevaluasi performa YOLOv8 dalam kondisi tersebut dan mengidentifikasi potensi peningkatannya. Dataset yang digunakan terdiri dari 16.990 gambar kendaraan dengan berbagai variasi dan kondisi lingkungan. Setelah dilatih, model dievaluasi menggunakan metrik seperti presisi, recall, dan F1-score, serta Intersection over Union (IoU) dengan ambang batas threshold 0.8 pada IoU. Hasil menunjukkan bahwa YOLOv8 unggul dengan akurasi deteksi yang cukup tinggi sebesar 78%, dengan presisi 82% dan recall di atas 90%, serta mampu mendeteksi kendaraan dalam kondisi real-time. Meskipun demikian, tantangan dalam mendeteksi objek kecil atau bentuk kendaraan yang tidak teratur seperti traktor masih perlu dioptimalkan. Penelitian ini juga membandingkan performa YOLOv8 dengan algoritma SSD (*Single Shot Detector*), di mana YOLOv8 terbukti lebih unggul dalam hal akurasi, presisi, recall, dan F1-score. Hasil penelitian yang diperoleh memberikan wawasan yang berharga untuk pengembangan sistem manajemen lalu lintas yang didasarkan pada teknologi deep learning. Kontribusi utama dari penelitian ini adalah menyediakan solusi deteksi kendaraan yang lebih efisien dan efektif, yang dapat diaplikasikan dalam sistem manajemen lalu lintas modern. Dengan demikian, hasil penelitian ini diharapkan dapat meningkatkan efisiensi pengelolaan lalu lintas dan memberikan dampak positif terhadap pengembangan sistem transportasi cerdas di masa depan.

Kata kunci: Deteksi Kendaraan, Deep Learning, Intersection over Union (IoU), Real-Time Detection, YOLO.

1. PENDAHULUAN

Kendaraan adalah bagian penting dari kehidupan modern dan memainkan peran penting dalam pergerakan orang dan transportasi barang di seluruh dunia[1]. Mereka memungkinkan mobilitas dan konektivitas yang belum pernah ada sebelumnya, yang mendorong perkembangan ekonomi, sosial, dan budaya. Aktivitas sehari-hari sering kali melibatkan kendaraan, berapapun jumlah kendaraannya, dan bertambahnya jumlah kendaraan di jalan telah menimbulkan permasalahan seperti kemacetan lalu lintas[2]. Semakin meningkatnya volume lalu lintas di berbagai kota besar, kebutuhan akan sistem pemantauan dan manajemen lalu lintas yang efisien menjadi sangat penting. Penerapan sistem pengawasan *Closed-Circuit Television* (CCTV) di sepanjang jalan raya tertentu memungkinkan petugas untuk mencegah tindak pelanggaran untuk mengawasi pergerakan kendaraan, dan menilai kepadatan pengguna kendaraan[3].

Masalah pada subjek ini telah mendapat banyak perhatian dari para spesialis di bidangnya. Serta beberapa peneliti akademis dalam upaya mengatasi komplikasi yang terkait dengan kemacetan lalu lintas. Kategorisasi dan kuantifikasi jenis kendaraan berfungsi sebagai pendekatan metodologis untuk menjelaskan fenomena lalu lintas dan mengumpulkan data terkait lalu lintas serta mengevaluasi kinerja lalu lintas secara keseluruhan[4]. Meski demikian, penerapan sistem kamera *Closed-Circuit Television* (CCTV) hanya berfungsi sebagai mekanisme pengawasan pasif saja. Salah satu tantangan utama dalam manajemen lalu lintas adalah kemampuan untuk mengidentifikasi dan mengukur kendaraan secara real-time dalam beragam kondisi lingkungan. Faktor-faktor seperti variasi pencahayaan yang rumit, dampak cuaca, keragaman jenis dan ukuran kendaraan, serta pemandangan lalu lintas dengan kepadatan tinggi dapat secara signifikan mengurangi efektivitas deteksi dan pelacakan[5]. Adanya penghalang, seperti kendaraan lain, bangunan, atau pepohonan, dapat mengaburkan sebagian atau seluruh kendaraan[6]. Untuk mengatasi masalah ini, berbagai teknologi modern mulai digunakan, khususnya yang berbasis kecerdasan buatan dan computer vision. Teknologi-teknologi memberikan solusi yang layak, karena menunjukkan kinerja yang unggul dan konsistensi yang lebih besar dibandingkan dengan kemampuan manusia[7]. Menanggapi kebutuhan ini, banyak sistem Manajemen Lalu Lintas Cerdas telah dirancang. Sistem ini meningkatkan kemampuan fungsi CCTV saat ini melalui metodologi deteksi objek kendaraan yang canggih untuk memfasilitasi pengamatan dan pengendalian arus kendaraan secara real-time[8]. Deteksi kendaraan dan perhitungan jumlahnya merupakan komponen penting untuk menganalisis kemacetan lalu lintas di jalan raya. Sistem yang disajikan dalam wacana ini bertujuan untuk

memberikan data jumlah mengenai volume kendaraan yang ada di simpang jalan raya[9]. Dalam konteks ini, YOLO (*You Only Look Once*) telah menjadi salah satu algoritma yang paling populer untuk mendeteksi objek dalam gambar atau video[10]. YOLO merupakan algoritma deep learning yang bertujuan untuk melakukan klasifikasi dan deteksi kendaraan guna mencapai hasil model yang optimal saat menjalankan sistem pengawasan di jalan raya[11]. YOLO memiliki banyak keunggulan yang menjadikannya pilihan favorit dalam bidang deteksi kendaraan, terutama kecepatan eksekusi yang tinggi memfasilitasi identifikasi kendaraan secara real-time, menjadikannya sesuai untuk aplikasi yang memerlukan respons cepat[12]. Walaupun metode YOLO memiliki banyak keunggulan, ada beberapa kelemahan yang perlu diperhatikan dalam penerapannya. Kelemahan yang dimiliki oleh model ini yaitu keterbatasan dalam deteksi objek yang berukuran kecil[13].

YOLOv8 diharapkan dapat mengatasi sebagian besar masalah ini. Agar hasilnya dapat diandalkan di dunia nyata, implementasi ini harus diuji secara menyeluruh dengan dataset kendaraan yang beragam[14]. Dalam implementasi, YOLOv8 akan melakukan deteksi dengan menggunakan bounding box untuk menentukan lokasi kendaraan pada gambar. Kemudian, untuk menentukan jenis kendaraan yang terdeteksi. Metode ini memungkinkan YOLOv8 untuk mengidentifikasi keberadaan dan konteks serta kategori kendaraan. Selain itu, model ini dibuat untuk menangani seperti variasi pencahayaan, ukuran kendaraan, dan berbagai kondisi lingkungan, yang semuanya sering memengaruhi kinerja deteksi[15].

Penelitian ini berupaya untuk menilai seberapa efektif penggunaan algoritma YOLOv8 untuk mendeteksi dan menghitung kendaraan di berbagai kondisi lingkungan[16]. Dengan memanfaatkan teknologi seperti YOLOv8, untuk identifikasi dan deteksi kendaraan yang banyak digunakan dalam tugas-tugas deteksi objek karena kemampuannya untuk melakukan deteksi dengan cepat dan akurat. Selain itu, YOLOv8 diharapkan dapat mengatasi sebagian besar masalah dalam sistem manajemen lalu lintas. Agar hasilnya dapat diandalkan di dunia nyata, implementasi ini harus diuji secara menyeluruh dengan dataset kendaraan yang beragam[17]. YOLOv8 diproyeksikan dapat mencapai kinerja terbaik dalam skenario lalu lintas nyata dengan memanfaatkan keunggulan dalam kecepatan dan akurasi. Ini akan membantu mengembangkan sistem manajemen lalu lintas yang lebih aman dan andal[18].

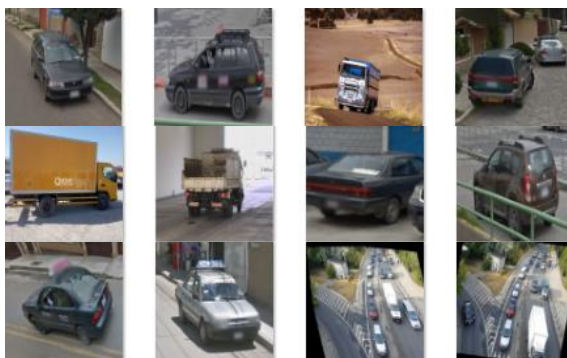
2. METODE PENELITIAN

Penggunaan model deteksi kendaraan yang dikembangkan secara cermat dengan mengimplementasikan algoritma YOLO v8, yang dikenal efektif dalam tugas deteksi objek secara real-

time. Kumpulan data yang digunakan dalam penelitian ini terdiri dari gambar kendaraan yang dikumpulkan melalui platform Roboflow.

2.1. Pengumpulan Data

Dalam penelitian ini dataset yang digunakan berisi 16990 gambar dengan anotasi terpadu untuk klasifikasi gambar. Dataset ini mencakup beragam kondisi seperti lokasi, skala, warna kendaraan, serta sudut pandang. Variasi kondisi lingkungan dalam dataset berpengaruh besar terhadap kinerja akhir model, karena model perlu belajar mengenali kendaraan dalam berbagai situasi agar mampu menggeneralisasi dengan baik saat diterapkan pada data baru. Beberapa gambar visual tertentu mungkin gagal menggambarkan komponen mobil secara keseluruhan karena jeda waktu dalam proses fotografi. Selain itu, variasi sudut pandang (misalnya tampak depan, samping, atau belakang kendaraan) juga penting. Jika model hanya dilatih pada satu atau dua sudut pandang, model mungkin tidak akan mendeteksi kendaraan dengan baik dari perspektif yang berbeda. Begitu juga dengan variasi ukuran kendaraan dalam gambar model harus belajar mengenali kendaraan baik yang dekat maupun jauh dari kamera. Misalnya, kendaraan yang lebih besar mungkin hanya terlihat sebagian, atau kendaraan yang bergerak cepat mungkin tertangkap dalam posisi yang tidak ideal dan perbedaan ukuran kendaraan serta posisi dan lokasi yang berbeda-beda. Setiap gambar dapat menampilkan satu atau dua kendaraan, lengkap dengan lokasi masing-masing kendaraan. Dengan dataset yang mencakup berbagai kondisi lingkungan, model memiliki peluang lebih besar untuk beradaptasi dan mengenali kendaraan secara konsisten dalam situasi yang beragam, yang meningkatkan keandalan hasil deteksi kendaraan di lapangan. Gambar 1 adalah contoh dari dataset yang kami gunakan.



Gambar 1. Dataset Kendaraan

2.2. Proses Pelabelan

Pelabelan data adalah langkah esensial dalam membangun model deteksi kendaraan yang akurat. Proses ini melibatkan penandaan atau anotasi pada setiap gambar dalam dataset, sehingga model dapat belajar mengenali objek tertentu berdasarkan label

yang diberikan. Dalam deteksi kendaraan, pelabelan data mencakup identifikasi lokasi kendaraan dalam gambar dan sering kali menyertakan informasi tambahan, seperti tipe kendaraan, orientasi, dan kondisi khusus. Proses ini melibatkan pemberian label pada gambar dengan cara menetapkan koordinat objek yang terdeteksi[19]. Koordinat ini terdiri dari titik-titik yang membentuk bounding box, yaitu kotak yang mengelilingi objek tertentu dalam gambar. Setiap objek di dalam gambar akan diberi bounding box, yang kemudian dihubungkan dengan kelas atau kategori objek yang sesuai dengan jenis kendaraan seperti mobil, motor, atau truk[20]. Selain itu, konsistensi dalam pelabelan sangat penting untuk menghindari kesalahan interpretasi oleh model. Jika beberapa gambar diberi label yang tidak konsisten, seperti posisi kotak pembatas yang tidak sejajar dengan objek, model bisa menjadi tidak stabil dalam performanya. Gambar 2 merupakan contoh dari hasil proses pelabelan data yang digunakan.



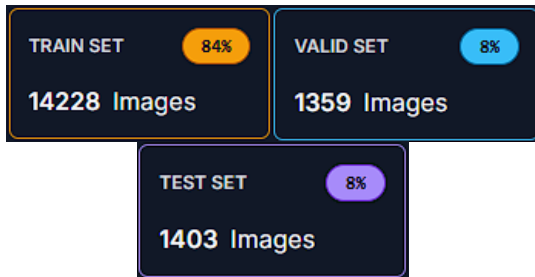
Gambar 2. Proses Pelabelan

Objek yang akan dideteksi kemudian diberi label sesuai nama kelas kendaraan dengan format file.txt[21]. File data tersebut berisi informasi koordinat bounding box dan kelas objek. Informasi ini kemudian digunakan dalam pelatihan model deteksi objek, di mana model belajar untuk mengenali objek berdasarkan pola yang ada dalam dataset terlabel. Dengan demikian, kualitas labelling sangat mempengaruhi kinerja model, karena data yang terlabel dengan baik akan membantu model dalam melakukan prediksi yang lebih akurat dan proses labelling dilakukan pada platform Roboflow.

2.3. Pembagian Dataset

Setelah melakukan proses Labelling Data, data yang berjumlah 16990 kemudian akan dibagi kedalam data split seperti 14228 data training, 1359 data validation dan 1403 data testing. Penggunaan nilai rasio dalam pembagian dataset, seperti train, validation, dan test set, sangat penting untuk memastikan model memperoleh pembelajaran yang optimal sekaligus memberikan gambaran akurat tentang performanya. Rasio yang digunakan, 84:8:8 bertujuan untuk memaksimalkan data train 84% sehingga model dapat belajar lebih dalam, sementara sisa data dialokasikan untuk validation dan test masing-masing 8% untuk menguji kemampuan

model pada data baru yang tidak terlihat selama pelatihan[22]. Pembagian ini penting untuk memastikan bahwa model dilatih, dievaluasi, dan diuji dengan cara yang efektif dan tidak overfitting. Gambar 3 merupakan hasil pembagian dataset.



Gambar 3. Pembagian Dataset

Alasan penggunaan train set yang lebih besar adalah agar model dapat mempelajari pola yang lebih kompleks dan menghindari masalah underfitting, di mana model gagal mengenali pola yang diperlukan untuk memprediksi data baru. Train set dapat membantu model mengenali pola dalam data dan mengoptimalkan parameter model agar bisa mendeteksi objek dengan baik di gambar yang belum pernah dilihat, data validation digunakan untuk mengevaluasi performa model selama pelatihan, sehingga kita bisa memonitor kapan model mulai overfitting atau underfitting dan test set digunakan setelah pelatihan selesai, untuk mengevaluasi generalisasi model pada data baru dan menguji kemampuan model untuk generalisasi terhadap data data baru yang belum pernah dilihat selama proses pelatihan.

2.4. Pelatihan Model

Tahap ini sangat penting karena dataset yang digunakan harus sangat detail agar deteksi objek dapat berjalan stabil dan mencapai tingkat akurasi yang tinggi[23]. Data yang telah melalui tahap sebelumnya kemudian dilatih dengan memanfaatkan GPU pada Google Colab. Google Colab juga menyediakan library dan tools, dengan tujuan mempercepat pelatihan dan meningkatkan efisiensi dalam pelatihan model[24]. Dalam melakukan pelatihan model perlu memperhatikan parameter-parameter yang dapat menunjang keberhasilan dalam melakukan pelatihan. Parameter yang digunakan dalam pelatihan bisa dilihat pada Tabel 1 di bawah ini.

Tabel 1. Parameter Pelatihan Model

Parameter	Nilai
Epoch	500
Image Size	320
Batch Size	64

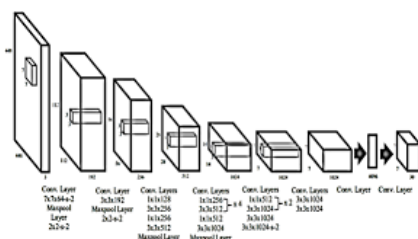
Fase selanjutnya mencakup pelaksanaan pelatihan kumpulan data melalui modifikasi parameter seperti yang digambarkan dalam Tabel 1. Parameter pelatihan, seperti Epoch, menentukan jumlah iterasi pelatihan yang dilakukan oleh model,

yang dalam kasus ini bisa mencapai 500 epoch. Setiap epoch mewakili satu siklus penuh di mana model memproses seluruh dataset. Tujuan penggunaan epoch sebanyak ini adalah untuk memberikan kesempatan kepada model memperbaiki parameter secara bertahap berdasarkan kesalahan (error) yang dihasilkan setelah memproses dataset. Semakin banyak epoch, semakin lama model belajar dari data, sehingga dapat menemukan pola yang lebih baik dan mendetail. Penggunaan 500 epoch diharapkan dapat meningkatkan performa model secara bertahap, mengurangi kesalahan, dan memungkinkan model untuk menyesuaikan bobot bobotnya sehingga lebih optimal dalam mengenali pola gestur yang kompleks. Selanjutnya, *Image Size*, juga disebut sebagai dimensi gambar masukan, digunakan selama proses pelatihan, di mana gambar akan diubah ukurannya menjadi 320x320 piksel sebelum diproses oleh model. Pemilihan ukuran gambar ini bertujuan untuk mempercepat komputasi selama proses pelatihan. Gambar yang lebih kecil memerlukan daya komputasi yang lebih sedikit, sehingga memungkinkan model melakukan lebih banyak iterasi dalam waktu yang sama, meningkatkan kecepatan pelatihan. Selain itu, dalam aplikasi real time, seperti deteksi gestur langsung, ukuran gambar yang lebih kecil memungkinkan sistem untuk bekerja lebih cepat tanpa mengorbankan akurasi secara signifikan. Ukuran gambar 320x320 sering dianggap sebagai keseimbangan ideal antara kecepatan dan ketepatan deteksi, karena meskipun gambar diperkecil, informasi penting yang relevan untuk pengenalan gestur tetap dapat dipertahankan. Selanjutnya, *Batch Size* mengacu pada jumlah data yang digunakan dalam setiap iterasi selama pelatihan, khususnya yang terdiri dari 64 gambar per batch. Artinya, pada setiap iterasi, model akan memproses 64 gambar secara bersamaan sebelum melakukan backpropagation, yaitu proses memperbarui bobot model berdasarkan kesalahan yang dihitung dari hasil prediksi. Menggunakan ukuran batch yang lebih besar memungkinkan pelatihan yang lebih cepat karena lebih banyak data yang diproses dalam satu waktu. Selain itu, ukuran batch yang besar membuat model menjadi lebih stabil karena gradien rata-rata yang dihitung lebih akurat. GPU juga dapat digunakan secara lebih efisien dengan ukuran batch yang besar, karena GPU didesain untuk melakukan komputasi paralel yang optimal saat menangani data dalam jumlah besar sekaligus. Hal ini mengurangi waktu pelatihan secara keseluruhan, yang sangat penting untuk model dengan jumlah epoch yang besar seperti dalam penelitian ini. Model yang telah menjalani pelatihan melanjutkan ke tahap pengujian, pada tahap pengujian model, model yang telah melalui proses pelatihan akan dibandingkan dengan algoritma SSD (*Single Shot Detector*) dan akan diuji dengan menggunakan dataset yang sama yaitu berjumlah 16.990. Dalam perbandingan antara algoritma YOLOv8 dan SSD untuk mendeteksi

objek, terutama kendaraan, YOLOv8 dan SSD diuji menggunakan dataset sebanyak 16.990 gambar. Untuk mencapai akurasi terbaik, YOLOv8 dilatih selama 500 epoch, sementara SSD dilatih dalam 50.000 langkah (steps). Pengujian ini dilakukan dalam skenario yang sama guna memastikan konsistensi performa kedua model dalam kondisi lingkungan yang bervariasi, termasuk pencahayaan rendah, cuaca buruk, dan sudut pandang berbeda. Hasil pengujian menunjukkan bahwa YOLOv8 memiliki keunggulan nyata dalam hal kecepatan dan akurasi dibandingkan SSD. YOLOv8 dirancang dengan arsitektur yang sangat efisien, memungkinkan deteksi objek secara cepat dengan frame rate tinggi, sehingga sangat sesuai untuk aplikasi real-time. Dari segi akurasi, YOLOv8 juga unggul, terutama dalam mengenali objek-objek kecil dan yang sebagian tertutup, yang sering kali menjadi kendala bagi SSD. Selain itu, dengan penggunaan epoch yang lebih sedikit, YOLOv8 dapat mencapai tingkat akurasi yang sama atau bahkan lebih tinggi dibandingkan SSD, menjadikannya pilihan optimal untuk aplikasi yang membutuhkan deteksi cepat dan andal tanpa mengorbankan kualitas deteksi.

2.5. You Only Look Once (YOLO)

YOLO, atau You Only Look Once, adalah salah satu metode yang digunakan untuk memprediksi jenis dan lokasi objek dalam gambar secara *real-time* menggunakan jaringan konvolusi (*convolutional network*)[25]. YOLO menggunakan arsitektur jaringan saraf terpadu untuk menjalankan deteksi objek dalam gambar. Arsitektur ini memanfaatkan karakteristik semua gambar yang tersedia untuk memperkirakan setiap bounding box, sehingga memungkinkan prediksi kotak pembatas dan probabilitas terkait dilakukan dalam satu evaluasi. Arsitektur deteksi YOLO terdiri dari 24 lapisan konvolusional, diikuti oleh 2 lapisan yang terhubung sepenuhnya, mengakomodasi gambar masukan berukuran 224×224 piksel. Lapisan konvolusional tertentu memanfaatkan lapisan reduksi dimensi 1×1 sebagai pendekatan tambahan untuk mengurangi kedalaman peta fitur, yang selanjutnya diikuti oleh lapisan konvolusional 3×3 . Arsitektur tersebut dapat dilihat pada Gambar 4.



Gambar 4. Arsitektur YOLO[26]

Model ini bekerja dengan menerima input berupa gambar berukuran 32×32 piksel dengan satu saluran, misalnya gambar grayscale. Arsitektur ini cocok untuk tugas klasifikasi gambar sederhana,

namun untuk dataset yang lebih kompleks seperti kendaraan, model ini mungkin perlu disesuaikan dengan ukuran input yang lebih besar dan penambahan filter untuk menangkap fitur yang lebih detail. Model ini diklaim dapat beroperasi dengan kecepatan 45 frame per detik (fps) akibatnya, pendekatan ini sangat cocok untuk diterapkan dalam skenario yang memerlukan pemrosesan waktu nyata, seperti analisis video. Sejalan dengan meningkatnya kebutuhan akan algoritma deteksi objek yang lebih canggih dan efisien[26]. Metode Deep Learning memerlukan sumber daya komputasi yang besar. Ini menyiratkan bahwa seseorang juga harus memiliki perangkat keras komputasi berkinerja tinggi.

2.6. Evaluasi Matrix

Model kemudian di evaluasi menggunakan confusion matrix sebagai metrik utama untuk mengevaluasi efektivitas hasil pelatihan. Confusion matrix biasanya mencakup empat komponen mendasar. True Positif (TP) merupakan jumlah data yang sebenarnya positif dan juga diidentifikasi sebagai positif, sedangkan True Negatif (TN) adalah jumlah data yang benar-benar negatif dan diprediksi sebagai negatif. False Positive (FP) menunjukkan jumlah data yang seharusnya negatif tetapi salah diidentifikasi sebagai positif (biasanya disebut sebagai kesalahan Tipe I), sedangkan False Negatif (FN) adalah jumlah data yang sebenarnya positif tetapi diprediksi sebagai negatif (juga dikenal sebagai kesalahan Tipe II). Confusion matriks ini dapat dihitung dari beberapa kinerja matriks, seperti:

A. Accuracy

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{1}$$

Akurasi berfungsi sebagai kriteria penilaian kuantitatif yang mengkuantifikasi frekuensi model prediksi menghasilkan perkiraan yang benar dalam kaitannya dengan agregat dari semua prediksi yang dihasilkan[27].

B. Precision

$$Precision = \frac{TP}{TP+FP} \tag{2}$$

Presi mencerminkan kemampuan model dalam mengidentifikasi objek yang relevan, diukur sebagai persentase prediksi yang akurat[28].

C. Recall

$$Recall = \frac{TP}{TP+FN} \tag{3}$$

Recall merupakan metrik yang menilai kuantitas informasi terkait yang diperoleh oleh suatu sistem melalui penghitungan rasionya[29].

D. F1 Score

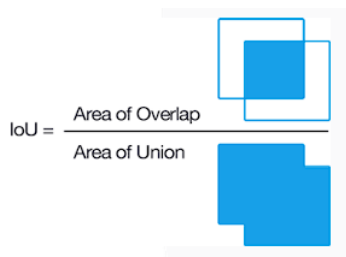
$$F1\ Score = 2x \frac{Precision \ x \ recall}{Precision + Recall} \tag{4}$$

F1 Score merupakan rata-rata harmonis antara nilai presisi dan recall[30]. Hal ini digunakan untuk mencapai keseimbangan antara presisi dan perolehan, khususnya di hadapan ketidak seimbangan kelas.

E. Intersection over Union (IoU)

$$IoU = \frac{A \cap B}{A \cup B} \tag{5}$$

IoU mewakili metrik yang didasarkan pada korelasi statistik dan variabilitas yang melekat dalam kumpulan data sampel, yang bertujuan menilai sejauh mana interseksionalitas antara dua bounding box yang ditentukan, terutama bounding box yang diprediksi dan bounding box ground truth[31]. IoU dapat ditemukan menggunakan persamaan berikut:



Gambar 5. Ilustrasi IOU[32]

Gambar 5 menampilkan diagram yang menggambarkan konsep Intersection over Union (IoU), menunjukkan area tumpang tindih antara bounding box hasil prediksi dan ground truth. Visualisasi ini membantu menilai akurasi deteksi objek melalui perbandingan nilai IoU.

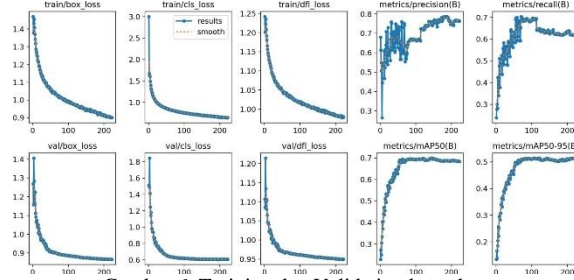
3. HASIL DAN PEMBAHASAN

3.1. Hasil Proses Training

Berdasarkan hasil pelatihan model menggunakan YOLO v8 pada proses sebelumnya didapatkan performa yang cukup beragam baik dalam hal kecepatan dan akurasi deteksi. Model mampu mengenali kendaraan dengan tingkat akurasi yang bervariasi, di mana nilai precision dan recall dari keseluruhan nilai mendapatkan 0.7 dan 0.6, model dilatih untuk mendeteksi 11 kelas kendaraan. Hasil ini menunjukkan bahwa model cukup efektif dalam mendeteksi berbagai kendaraan yang ditunjukkan pada Tabel II dibawah ini.

Tabel 2. Confusion Matrix

Class	Precision	Recall	mAP50
Multi Truck	0.891	0.667	0.668
Bike	0.811	0.867	0.894
Bus	0.767	0.731	0.784
Car	0.877	0.86	0.892
Cycle	0.634	0.441	0.616
Mini Van	0.691	0.775	0.797
Pick Up	0.617	0.543	0.601
Bemo	0.816	0.822	0.866
Tractor	0.607	0.4	0.47
Tractor Trailer	0.65	0.25	0.417
Truck	0.745	0.699	0.765
All	0.722	0.646	0.698



Gambar 6. Training dan Validation loss plots







Tabel II menunjukkan nilai presisi tertinggi terdapat pada kelas Multi-Axle Truck (0.891), Car (0.877), dan Bemo (0.816), yang berarti model jarang salah dalam memprediksi kendaraan-kendaraan ini. Namun, presisi terendah ada pada kelas Tractor (0.607), menunjukkan model kesulitan membedakan kendaraan ini dari yang lain. Beberapa kelas seperti Bike, Bemo, dan Car memiliki nilai recall yang cukup tinggi dengan mendapat nilai 0.8 keatas, menandakan model mampu mendeteksi semua kendaraan tersebut. Sedangkan, kelas Tractor (0.4), Tractor With Trailer (0.25) dan Pick Up (0.543) memiliki nilai recall yang rendah, yang berarti beberapa kendaraan mungkin tidak terdeteksi. Untuk mAP50, nilai tertinggi ada pada kelas Bike, Car dan Bemo (0.8), menunjukkan akurasi tinggi dalam mendeteksi posisi dan bentuk kendaraan. Namun, kelas Tractor With Trailer memiliki mAP50 terendah (0.417), yang menunjukkan kekurangan dalam keakuratan lokasi dan bentuk kendaraan, memerlukan penyempurnaan lebih lanjut pada model. Gambar 6 menunjukkan kurva hasil dari pelatihan yang telah dilakukan, menggunakan 500 epoch bertujuan untuk memberi model cukup kesempatan belajar pola-pola dalam data secara mendalam. Pada awalnya, model biasanya mengalami peningkatan akurasi yang signifikan. Namun, setelah beberapa epoch, peningkatannya akan melambat hingga mencapai titik jenuh atau bahkan mulai overfitting. Dengan jumlah epoch yang tinggi, kita bisa mengamati kapan model mencapai akurasi terbaik dan kapan model mulai overfitting, sehingga bisa menentukan epoch optimal untuk pelatihan di masa depan.

3.2. Pengujian Model

Proses evaluasi model dirancang agar dapat menilai kemandirian model yang telah dilatih dengan meneliti ketepatan hasil prediktifnya. Setelah melalui tahap pelatihan, model diuji dengan menggunakan tiga gambar kendaraan mobil untuk mengevaluasi kinerjanya. Setiap gambar diuji dengan tujuan untuk melihat apakah model dapat mendeteksi dan mengidentifikasi kendaraan secara akurat, termasuk lokasi dan bentuknya. Pada hasil pengujian ini, model berhasil mendeteksi seluruh mobil yang ada di setiap gambar. Skor prediksi yang dihasilkan oleh model bervariasi, dengan nilai yang berkisar antara 0.86 hingga 0.94. Nilai ini menunjukkan tingkat keyakinan model dalam melakukan prediksi. Semakin tinggi nilai

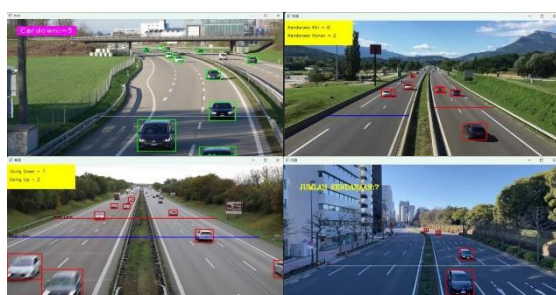
hasil prediksi semakin yakin model bahwa deteksi tersebut benar. Setiap hasil prediksi juga diberi dengan keterangan seperti yang terdapat pada Tabel III. Dengan demikian, dapat disimpulkan model yang telah mengalami pelatihan sebelumnya menunjukkan performa yang memuaskan, terutama dalam mengidentifikasi dan mendeteksi objek kendaraan.

Tabel 3. Hasil Pengujian Model

No	Gambar Asli	Prediksi	Keterangan
1			Benar
2			Benar
3			Benar

3.3. Evaluasi Model YOLO

Pada proses deteksi kendaraan mobil secara realtime menggunakan video, performa model diuji untuk menilai ketepatan kotak prediksi yang dihasilkan. Model yang telah dilatih sebelumnya menunjukkan kemampuan yang signifikan dalam mengidentifikasi kategori kendaraan, meskipun tingkat akurasi bervariasi. Dalam berbagai frame video, model dapat mengenali kendaraan secara akurat, namun tingkat akurasi ini dipengaruhi oleh beberapa faktor, seperti kondisi lingkungan, pencahayaan yang kurang optimal, dan kualitas video yang digunakan. Beberapa deteksi yang kurang akurat terjadi ketika kendaraan terlihat sebagian atau terhalang objek lain. Hal ini menunjukkan bahwa model YOLOv8 sensitif terhadap kondisi visual, sehingga faktor-faktor eksternal dapat memengaruhi performanya. Dengan demikian, performa model YOLOv8 dalam melakukan deteksi kendaraan mobil secara realtime dapat dianggap baik meskipun ada beberapa yang memerlukan optimasi lebih lanjut. Gambar 7 menunjukkan hasil deteksi kendaraan dengan memanfaatkan video.



Gambar 7. Deteksi Kendaraan Mobil menggunakan Video

Pada gambar 8, menampilkan beberapa deteksi yang kurang akurat seperti kendaraan yang seharusnya truck tetapi terdeteksi sebagai mobil kemudian kendaraan yang seharusnya pick up terdeteksi oleh model sebagai mobil. Hal ini menunjukkan bahwa model memiliki kesulitan dalam membedakan kendaraan yang mirip. Kegagalan model dalam mendeteksi kendaraan bisa disebabkan oleh kurangnya data latih yang representatif untuk kendaraan tertentu atau kebutuhan untuk menyesuaikan parameter dalam algoritma pembelajaran mesin dan faktor-faktor lainnya, seperti pencahayaan yang kurang baik dan rentang jarak kendaraan dengan kamera. Dengan demikian, performa model YOLO v8 dalam mendeteksi kendaraan secara real-time masih dapat dianggap baik, meskipun ada beberapa kendaraan yang membutuhkan optimisasi lebih lanjut.

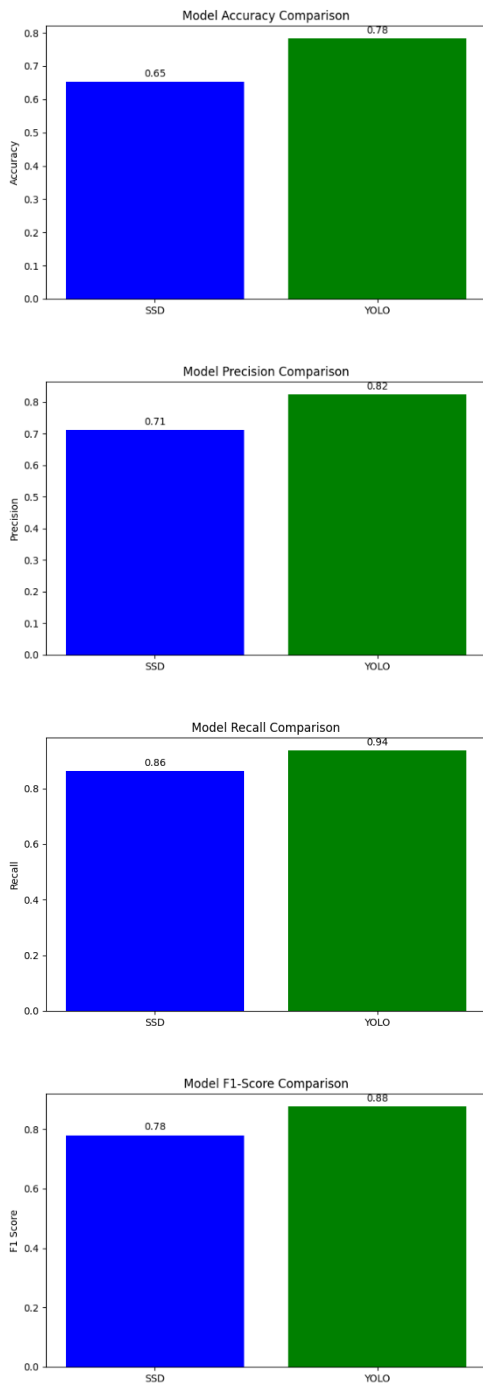


Gambar 8. Deteksi Error

Setelah mengevaluasi performa model YOLOv8 dalam mendeteksi dan menghitung jumlah kendaraan, langkah selanjutnya adalah membandingkannya dengan algoritma *Single Shot Detector* (SSD) adalah algoritma deep learning yang umum digunakan untuk mendeteksi objek dalam gambar. SSD dirancang untuk memberikan keseimbangan yang baik antara kecepatan dan akurasi. Perbandingan ini akan dilakukan dengan mengukur beberapa metrik evaluasi seperti akurasi, presisi, recall, dan F1-score, serta kecepatan pemrosesan untuk setiap algoritma. Tujuannya adalah untuk memahami algoritma mana yang memiliki performa lebih unggul dalam situasi nyata. Hasil dari perbandingan ini akan memberikan pemahaman yang sangat berharga dalam memilih model yang paling sesuai dengan kebutuhan untuk pengelolaan lalu lintas yang efisien dan responsif, apakah SSD atau YOLOv8 lebih cocok digunakan tergantung pada kebutuhan spesifik yang dihadapi.

Tabel 4. Kinerja Model

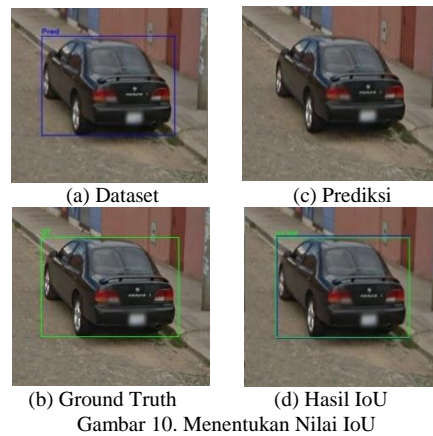
No	Models	Steps	Accuracy (%)	Precisi on (%)	Reca ll (%)	F-1 Sco re (%)
1	YOLOv 8	500	0.78	0.82	0.94	0.88
	SSD	5000				
2	Mobile Net	0	0.65	0.71	0.86	0.78



Gambar 9. Diagram Perbandingan Matrix Kinerja Model

Pada Tabel IV, menampilkan hasil perbandingan antara algoritma YOLOv8 dan SSD, bertujuan untuk dapat memberikan pemahaman yang jelas tentang seberapa efektif dan efisien dari setiap model. Pada Gambar 9 memperlihatkan bahwa penggunaan algoritma model YOLOv8 lebih unggul dibandingkan dengan algoritma SSD. Kemampuan model ini tidak hanya mampu mengenali kelas kendaraan saja tetapi juga dapat membuat prediksi yang konsisten pada gambar, video dan rekaman secara realtime, bahkan dalam situasi yang rumit dan bervariasi. Dalam menentukan kinerja model yang

terdapat dalam Tabel IV, dengan menggunakan IOU (*Intersection over Union*) adalah metrik yang penting dalam evaluasi performa model deteksi objek, terutama dalam menilai seberapa akurat model dalam mengenali dan menentukan lokasi objek dalam sebuah gambar. IoU mengukur tingkat kesesuaian antara bounding box prediksi yang dihasilkan oleh model dan bounding box sebenarnya (*ground truth*) yang sudah dilabeli pada data. Nilai IoU dihitung sebagai rasio luas area perpotongan (*intersection*) antara kotak prediksi dan kotak sebenarnya. IoU memberikan cara untuk mengevaluasi seberapa akurat posisi dan ukuran kotak prediksi. Misalnya, jika IoU tinggi ini berarti kotak prediksi dan kotak sebenarnya memiliki kesesuaian yang baik, menunjukkan bahwa model berhasil dalam mendeteksi objek secara akurat. Gambar 10 merupakan contoh dalam menentukan nilai IoU.



Gambar 10. Menentukan Nilai IoU

Selanjutnya, digunakan threshold sebesar 0.8 untuk menentukan kualitas deteksi objek. Threshold ini dipilih karena memberikan keseimbangan yang baik antara mendeteksi objek yang relevan dan meminimalkan deteksi palsu. Dengan menetapkan ambang batas IoU ini, prediksi akan dikategorikan sebagai *True Positive* (TP) jika nilai IoU mencapai 0.8 atau lebih, menunjukkan bahwa prediksi memiliki kesesuaian yang tinggi dengan objek sebenarnya. Penggunaan threshold ini bertujuan untuk memastikan bahwa model yang digunakan cukup akurat dalam mendeteksi objek, terutama dalam konteks deteksi kendaraan di jalan raya. Semakin tinggi nilai IoU, semakin baik tingkat kesesuaian antara prediksi dan ground truth, yang menandakan deteksi yang lebih akurat. Jika nilai IoU berada di bawah 0.8, prediksi akan dianggap sebagai *False Positive* (FP) atau *False Negative* (FN), yang dapat mempengaruhi metrik evaluasi seperti precision dan recall[31]. Dalam proses evaluasi model, IoU threshold berfungsi sebagai parameter penting untuk menilai performa model secara spesifik. Nilai ambang batas ini sangat penting untuk menilai seberapa baik model tersebut mampu mendeteksi objek dengan benar tanpa memunculkan terlalu banyak kesalahan deteksi, sehingga mempengaruhi hasil akhir dari evaluasi kinerja model.

4. DISKUSI

Berdasarkan hasil penelitian yang telah dilakukan menunjukkan bahwa YOLOv8 lebih handal dalam mendeteksi kendaraan, khususnya dalam berbagai kondisi lingkungan, sehingga sangat potensial untuk diaplikasikan dalam sistem manajemen lalu lintas. YOLOv8 memiliki tingkat kecepatan dan akurasi yang tinggi dalam mendeteksi kendaraan, terutama untuk kelas-kelas kendaraan seperti Multi-Axle Truck, Car, dan Bemo, dengan nilai presisi di atas 80%. Namun, tantangan terdeteksi pada kendaraan kecil atau yang memiliki bentuk tidak beraturan seperti traktor, yang memberikan nilai presisi dan recall lebih rendah. Meskipun model ini memiliki keunggulan dalam kecepatan dan kinerja deteksi objek secara real-time, masih diperlukan optimasi untuk meningkatkan performa dalam mendeteksi kendaraan kecil dan kondisi lingkungan yang lebih kompleks. Tantangan ini harus diatasi melalui optimasi lanjutan pada dataset dan arsitektur model agar deteksi kendaraan dapat dilakukan secara lebih akurat dalam berbagai kondisi lingkungan.

Jika dibandingkan dengan penelitian terdahulu yang menggunakan model YOLOv5[33], terlihat bahwa YOLOv8 menunjukkan peningkatan yang signifikan dalam hal kecepatan dan akurasi dibandingkan YOLOv5. Hal ini membuktikan bahwa perbaikan dalam arsitektur YOLO khususnya pada versi 8, berkontribusi pada peningkatan performa yang signifikan dalam mendeteksi objek di lingkungan yang lebih beragam. Penelitian ini menegaskan pentingnya penerapan algoritma deteksi objek yang cepat dan akurat seperti YOLOv8. Dampak praktis dari penelitian ini sangat relevan untuk implementasi nyata, terutama dalam sistem manajemen lalu lintas modern. Hasil penelitian ini menunjukkan bahwa YOLOv8 dapat menjadi solusi yang efisien dan efektif untuk deteksi kendaraan real-time, yang dapat diintegrasikan ke dalam sistem manajemen lalu lintas yang ada. Dengan demikian, hasil penelitian ini diharapkan dapat meningkatkan efisiensi pengelolaan lalu lintas dan memberikan dampak positif terhadap pengembangan sistem transportasi cerdas di masa depan.

5. KESIMPULAN

Hasil penelitian dalam dokumen, disimpulkan bahwa algoritma YOLOv8 menunjukkan keunggulan signifikan dalam deteksi kendaraan secara real-time dibandingkan dengan algoritma SSD (*Single Shot Detector*). Model YOLOv8 mencapai akurasi deteksi 78%, dengan precision 82% dan recall di atas 90%, mengungguli SSD yang memiliki akurasi 65%. Keunggulan utama YOLOv8 terletak pada kemampuannya mendeteksi objek secara cepat dan konsisten, bahkan dalam kondisi lingkungan yang beragam. Evaluasi yang dilakukan dengan menggunakan threshold Intersection over Union (IoU) 0,8 membantu menjaga keseimbangan antara

akurasi dan pengurangan deteksi palsu, sehingga dapat mempertahankan tingkat presisi dan recall yang tinggi. isi dan recall yang tinggi.

Dalam konteks yang lebih luas, penelitian ini berkontribusi signifikan terhadap pengembangan sistem manajemen lalu lintas yang pintar. Implementasi YOLOv8 membantu dalam memberikan data yang lebih akurat dan real-time untuk analisis lalu lintas, yang dapat digunakan untuk meningkatkan efisiensi lalu lintas, mengurangi kemacetan, dan meningkatkan keselamatan jalan. Dengan demikian, penelitian ini diharapkan dapat memberikan solusi yang lebih efisien dan efektif untuk manajemen lalu lintas yang lebih baik.

Namun, ada tantangan dalam mendeteksi kendaraan kecil atau yang bentuknya tidak teratur seperti Tractor. Penelitian selanjutnya disarankan untuk fokus pada beberapa aspek seperti optimasi deteksi kendaraan kecil dengan mengembangkan metode yang lebih baik untuk mendeteksi objek kecil dan bentuk tidak beraturan seperti traktor. Ini bisa dilakukan dengan memperbaiki arsitektur model atau menggunakan dataset yang lebih representatif. Kemudian evaluasi lebih lanjut pada kondisi lingkungan yang beragam dengan melakukan pengujian pada kondisi lingkungan yang lebih luas dan beragam, termasuk pencahayaan rendah dan cuaca ekstrem, untuk memastikan model tetap mampu untuk menangani berbagai pengaruh yang dapat memengaruhi operasinya.

Pada penelitian ini dapat disimpulkan bahwa Model YOLOv8 sudah cukup optimal untuk mendeteksi berbagai jenis kendaraan seperti mobil, truk, dan motor dalam berbagai kondisi lingkungan. karena model memiliki performa tertinggi dibandingkan dengan model SSD, terutama dalam hal akurasi dan keseimbangan antara precision dan recall. Selain itu, performa YOLOv8 juga tetap konsisten meskipun pada kondisi lingkungan yang bervariasi, menunjukkan bahwa model ini mampu mendeteksi kendaraan dengan baik dalam berbagai kondisi.

DAFTAR PUSTAKA

- [1] T. Sutisna, A. Rachmat Raharja, E. Hariyadi, and V. Hafizh Cahaya Putra, "Penggunaan Computer Vision untuk Menghitung Jumlah Kendaraan dengan Menggunakan Metode SSD (Single Shoot Detector)," *Innov. J. Soc. Sci. Res.*, vol. 4, pp. 6060–6067, 2024, [Online]. Available: <https://j-innovative.org/index.php/Innovative/article/view/10071/6958>
- [2] 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.
- [3] R. G. Fajri, I. Santoso, and Y. A. Adi

- Soetrisno, "Perancangan Program Pendeteksi Dan Pengklasifikasi Jenis Kendaraan Dengan Metode Convolutional Neural Network (Cnn) Deep Learning," *Transient J. Ilm. Tek. Elektro*, vol. 9, no. 1, pp. 97–106, 2020, doi: 10.14710/transient.v9i1.97-106.
- [4] N. J. Hayati, D. Singasatia, and M. R. Muttaqin, "Object Tracking Menggunakan Algoritma You Only Look Once (YOLO)v8 untuk Menghitung Kendaraan," *Komputa J. Ilm. Komput. dan Inform.*, vol. 12, no. 2, pp. 91–99, 2023, doi: 10.34010/komputa.v12i2.10654.
- [5] L. You, Y. Chen, C. Xiao, C. Sun, and R. Li, "Multi-Object Vehicle Detection and Tracking Algorithm Based on Improved YOLOv8 and ByteTrack," *Electron.*, vol. 13, no. 15, 2024, doi: 10.3390/electronics13153033.
- [6] A. Sundaresan Geetha, M. A. R. Alif, M. Hussain, and P. Allen, "Comparative Analysis of YOLOv8 and YOLOv10 in Vehicle Detection: Performance Metrics and Model Efficacy," *Vehicles*, vol. 6, no. 3, pp. 1364–1382, 2024, doi: 10.3390/vehicles6030065.
- [7] A. Rezky, A. Bagir, D. Pamerean, and F. Makhrus, "Deteksi Kecelakaan Lalu Lintas Otomatis Pada Rekaman CCTV Indonesia Menggunakan Deep Learning," *Bul. Pagelaran Mhs. Nas. Bid. Teknol. Inf. dan Komun.*, vol. 1, no. 1, pp. 1–5, 2023, [Online]. Available: <https://buletingemastik.id/index.php/bg/article/view/2/1>
- [8] I. Irawanto, A. Sunyoto, and K. Kusnawi, "Peningkatan Akurasi Deteksi Kendaraan Menggunakan Kombinasi Haar Cascade Classifier dan Convolutional Neural Networks (CNN)," *J. Electr. Eng. Comput.*, vol. 6, no. 1, pp. 47–57, 2024, doi: 10.33650/jeeecom.v6i1.8242.
- [9] A. Meirza and N. R. Puteri, "Implementasi Metode YOLOV5 dan Tesseract OCR untuk Deteksi Plat Nomor Kendaraan," vol. 9, pp. 424–435, 2024, [Online]. Available: <https://journal.unusida.ac.id/index.php/jik/article/view/1288/887>
- [10] M. G. A. Binuri, T. Haryanti, and M. A. Haq, "Penerapan Algoritma YOLO v7 Sebagai Deteksi Kecelakaan Kendaraan Pada Lalu Lintas," *Comput. Insight J. Comput. Sci.*, vol. 4, no. 2, pp. 7–14, 2022, [Online]. Available: <https://journal.um-surabaya.ac.id/CI/article/view/22524/7607>
- [11] A. F. Oklilas, "MODEL YOLO VERSI 4 PADA PENGENALAN KENDARAAN DI JALAN RAYA KOTA PALEMBANG," no. 3, pp. 136–139, 2023, [Online]. Available: <https://ejournal.undip.ac.id/index.php/transmission/article/download/52732/pdf>
- [12] amwin Aldhiyatika, "Algoritma You Only Look Once (Yolo) Tugas Akhir," pp. 1–60, 2021, [Online]. Available: https://dspace.uui.ac.id/bitstream/handle/123456789/34154/17523176_Aldhiyatika_Amwin.pdf?sequence=1&isAllowed=y
- [13] A. F. Hidayah *et al.*, "MENGUNAKAN ALGORITMA COUNTER DALAM," vol. 11, no. 2, 2024, [Online]. Available: <https://ejournal.lppmunsera.org/index.php/PROSISO/article/view/8918/3220>
- [14] R. G. Wijanarko, A. I. Pradana, and D. Hartanti, "IMPLEMENTASI DETEKSI DRONE MENGGUNAKAN YOLO (You Only Look Once)," vol. 14, no. 2, pp. 437–442, 2024, [Online]. Available: <https://ejournal.umri.ac.id/index.php/JIK/article/view/7374/3067>
- [15] A. P. H. Telaumbanua, T. P. Larosa, P. D. Pratama, R. H. Fauza, and A. M. Husein, "Vehicle Detection and Identification Using Computer Vision Technology with the Utilization of the YOLOv8 Deep Learning Method," *Sinkron*, vol. 8, no. 4, pp. 2150–2157, 2023, doi: 10.33395/sinkron.v8i4.12787.
- [16] M. IKBAL and R. A. Saputra, "Pengenalan Rambu Lalu Lintas Menggunakan Metode Yolov8," *JIKA (Jurnal Inform.*, vol. 8, no. 2, p. 204, 2024, doi: 10.31000/jika.v8i2.10609.
- [17] Y. Zhang, Z. Guo, J. Wu, Y. Tian, H. Tang, and X. Guo, "Real-Time Vehicle Detection Based on Improved YOLO v5," *Sustain.*, vol. 14, no. 19, 2022, doi: 10.3390/su141912274.
- [18] N. H. Harani, C. Prianto, and M. Hasanah, "Deteksi Objek Dan Pengenalan Karakter Plat Nomor Kendaraan Indonesia Menggunakan Metode Convolutional Neural Network (CNN) Berbasis Python," *J. Tek. Inform.*, vol. 11, no. 3, pp. 47–53, 2019, [Online]. Available: <https://ejournal.ulbi.ac.id/index.php/informatika/article/view/658>
- [19] R. Wulanningrum and A. Sanjaya, "Implementasi YOLO Dalam Deteksi Jumlah Kendaraan," vol. 8, pp. 1274–1281, 2024, [Online]. Available: <https://proceeding.unpkediri.ac.id/index.php/inotek/article/view/5064/3498>
- [20] M. A. Fahrezi *et al.*, "Implementasi YOLOv8 Dalam Penghitung Masuk Dan Keluar Manusia Pada Gedung," vol. 11, no. 3, 2024, [Online]. Available: <https://jurnal.mdp.ac.id/index.php/jatisi/article/view/9050/2055>

- [21] A. F. Oklilas, D. Dwinta, G. Shofi, N. P. Mariza, and ..., "Akurasi Pengujian Model Hasil Training menggunakan YOLOv4 untuk Pengenalan Kendaraan di Jalan Raya," ... *J. Penelit. Ilmu* ..., pp. 799–806, 2023, [Online]. Available: <https://jurnal.polsri.ac.id/index.php/jupiter/article/view/6537%0Ahttps://jurnal.polsri.ac.id/index.php/jupiter/article/download/6537/2528>
- [22] M. Sauqi, "Deteksi Kendaraan Menggunakan Algoritma You Only Look Once (YOLO) V3," *Univ. Islam Indones.*, pp. 5–8, 2022, [Online]. Available: <https://dspace.uui.ac.id/bitstream/handle/123456789/38956/17523142.pdf?sequence=1&isAllowed=y>
- [23] D. Iskandar Mulyana and M. A. Rofik, "Implementasi Deteksi Real Time Klasifikasi Jenis Kendaraan Di Indonesia Menggunakan Metode YOLOV5," *J. Pendidik. Tambusai*, vol. 6, no. 3, pp. 13971–13982, 2022, doi: 10.31004/jptam.v6i3.4825.
- [24] V. R. Pratiwi and N. Rijati, "Classification Vehicle Tire Quality using Convolutional Neural Networks," *J. Appl. Informatics Comput.*, vol. 8, no. 1, pp. 213–220, 2024, doi: 10.30871/jaic.v8i1.8074.
- [25] I. W. Adi, A. Wiguna, R. R. Huizen, and G. A. Pradipta, "Model Deteksi Objek Menggunakan Yolov5 untuk Pengendalian Pengaturan Lalu Lintas," pp. 840–844, 2024, [Online]. Available: <https://spinter.stikom-bali.ac.id/index.php/spinter/article/view/244/209>
- [26] R. Gelar Guntara, "Pemanfaatan Google Colab Untuk Aplikasi Pendeteksian Masker Wajah Menggunakan Algoritma Deep Learning YOLOv7," *J. Teknol. Dan Sist. Inf. Bisnis*, vol. 5, no. 1, pp. 55–60, 2023, doi: 10.47233/jteksis.v5i1.750.
- [27] A. Imran, "Deteksi Pelanggaran Pada Bahu Jalan Tol Dengan Intelligent Transportation System Menggunakan Algoritma Yolov5 Violation Detection On The Roadside Of The Toll Roads With Intelligent Transportation System Using Yolov5 Algorithm," vol. 10, no. 5, p. 4498, 2023, [Online]. Available: <https://openlibrarypublications.telkomuniversity.ac.id/index.php/engineering/article/view/17957/17589>
- [28] Hendri Candra Mayana and Desmarita Leni, "Deteksi Kerusakan Ban Mobil Menggunakan Convolutional Neural Network dengan Arsitektur ResNet-34," *J. Surya Tek.*, vol. 10, no. 2, pp. 842–851, 2023, doi: 10.37859/jst.v10i2.6336.
- [29] L. Rahma, H. Syaputra, A. H. Mirza, and S. D. Purnamasari, "Objek Deteksi Makanan Khas Palembang Menggunakan Algoritma YOLO (You Only Look Once)," *J. Nas. Ilmu Komput.*, vol. 2, no. 3, pp. 213–232, 2021, doi: 10.47747/jurnalnik.v2i3.534.
- [30] R. M. Taufiq, Sunanto, Y. Rizki, and M. R. A. Pratama, "Simulasi Deteksi Golongan Kendaraan pada Gerbang Tol Menggunakan YOLOv4," *J. CoSciTech (Computer Sci. Inf. Technol.)*, vol. 3, no. 2, pp. 199–206, 2022, doi: 10.37859/coscitech.v3i2.3928.
- [31] Wahyono, M. E. Wibowo, A. Ashari, and M. P. K. Putra, "Improvement of Deep Learning-based Human Detection using Dynamic Thresholding for Intelligent Surveillance System," *Int. J. Adv. Comput. Sci. Appl.*, vol. 12, no. 10, pp. 472–477, 2021, doi: 10.14569/IJACSA.2021.0121053.
- [32] M. Dio Riza Pratama, B. Priyatna, S. S. Hilabi, and A. L. Hananto, "Deteksi Objek Kecelakaan Pada Kendaraan Roda Empat Menggunakan Algoritma YOLOv5," *J. Ilm. Sist. Informas*, vol. 12, no. 2, pp. 15–26, 2022, [Online]. Available: <https://journal.unipdu.ac.id/index.php/teknologi/article/view/3260/1523>.