

Implementasi Sistem Penghitungan Volume Kendaraan Menggunakan YoloV8

Wihan Perkasa Nugraha Putra¹, Afu Ichsan Pradana², Nurchim³

¹Teknik Informatika, Fakultas Ilmu Komputer, Universitas Duta Bangsa

²Teknik Informatika, Fakultas Ilmu Komputer, Universitas Duta Bangsa

³Teknik Informatika, Fakultas Ilmu Komputer, Universitas Duta Bangsa

¹202030304@mhs.udb.ac.id *, ²afu_ichsan@udb.ac.id, ³nurchim@udb.ac.id

Abstract

With the increasing number of vehicles on the road, it is important to monitor traffic to ensure traffic safety and management are well-maintained. Manual data collection of vehicle counts requires a lot of resources and costs. YOLOv8 (You Only Look Once) is a computer vision technology that enables automatic object detection with high accuracy and speed. This research aims to automate the counting of passing vehicles using YOLOv8, which can detect various types of vehicles in various weather conditions. The study consists of several stages, namely data collection, preprocessing, modeling, and evaluation. The dataset was obtained from the Kaggle dataset and traffic video recordings in Indonesia. The dataset is divided into training and validation data and processed with bounding boxes. Modeling uses YOLOv8 Nano with 50 epochs, and evaluation is performed using a confusion matrix. Implementation is done using OpenCV to count the number of vehicles from video recordings. This system successfully counted the number of vehicles based on different classifications, although there were constraints during frame drops. This research has developed an accurate system and tested it with CCTV traffic recordings in Surakarta, although the vehicle type classification is not yet in accordance with the transportation department standards, and the video frame rate quality still affects the system's accuracy. Real-time testing and the addition of unique identities for the detected vehicles should be conducted.

Keywords: YOLOv8, OpenCV, Dataset, Traffic, Object detection, Classification, Vehicle

Abstrak

Dengan banyaknya kendaraan di jalan, penting untuk memantau lalu lintas agar keamanan dan manajemen lalu lintas dapat terjaga dengan baik. Pengumpulan data jumlah kendaraan secara manual memerlukan banyak sumber daya dan biaya. YOLOv8 (You Only Look Once) adalah teknologi visi komputer yang memungkinkan deteksi objek secara otomatis dengan akurasi dan kecepatan tinggi. Penelitian ini bertujuan untuk mengotomatisasi perhitungan jumlah kendaraan yang melintas dengan YOLOv8, yang dapat mendeteksi berbagai jenis kendaraan dalam berbagai kondisi cuaca. Studi ini terdiri dari beberapa tahap, yaitu pengumpulan data, preprocessing, pemodelan, dan evaluasi. Dataset diperoleh dari dataset Kaggle dan rekaman video lalu lintas di Indonesia. Dataset dibagi menjadi data latih dan validasi dan diproses dengan bounding box. Pemodelan menggunakan YOLOv8 Nano dengan 50 epoch, dan evaluasi dilakukan menggunakan confusion matrix. Implementasi dilakukan menggunakan OpenCV untuk menghitung jumlah kendaraan dari rekaman video. Sistem ini berhasil menghitung jumlah kendaraan berdasarkan klasifikasi yang berbeda dengan baik, meski ada kendala saat frame drop. Penelitian ini berhasil mengembangkan sebuah sistem yang akurat dan diuji dengan rekaman CCTV lalu lintas di Surakarta, meskipun klasifikasi jenis kendaraan belum sesuai dengan standar dinas perhubungan dan kualitas frame rate video masih memengaruhi akurasi sistem. Sebaiknya dilakukan pengujian waktu nyata dan penambahan identitas unik pada kendaraan yang dideteksi.

Kata kunci: YOLOv8, OpenCV, Dataset, Lalu lintas, Deteksi objek, Klasifikasi, Kendaraan

©This work is licensed under a Creative Commons Attribution - ShareAlike 4.0 International

1. Pendahuluan

Dengan meningkatnya populasi kendaraan, pentingnya pemantauan lalu lintas semakin meningkat untuk menjaga keselamatan jalan dan mengoptimalkan manajemen lalu lintas. Salah satu caranya ialah dengan memantau volume kendaraan guna untuk pengelolaan infrastruktur lintas yang baik. Dengan adanya data volume kendaraan dapat digunakan untuk perhitungan kepadatan suatu jalan, prediksi kemacetan serta juga bisa untuk perencanaan pembangunan di masa depan dan peningkatan infrastruktur transportasi lainnya. Umumnya proses pengumpulan data volume kendaraan masih dilakukan secara manual dengan menugaskan beberapa orang untuk berada dilapangan dan menghitung setiap kendaraan yang lewat. Dalam

perhitungan manual ini terdapat beberapa kelemahan seperti banyaknya jumlah sumber daya manusia yang dibutuhkan serta besarnya biaya yang dibutuhkan untuk melakukan sekali pengumpulan data di beberapa titik jalan raya.

Pada masa saat ini teknologi visi komputer telah berkembang pesat hingga dapat secara otomatis mendeteksi objek tertentu melalui citra digital [1]. Hal tersebut bisa diwujudkan dengan beberapa metode salah satunya menggunakan metode yang bernama YOLOv8 (You Only Look One), yang merupakan sebuah model deteksi objek yang teruji dan terbukti handal dalam mengenali objek dengan akurasi dan kecepatan tinggi [2]. Mengapa menggunakan YOLOv8? Hal ini dikarenakan YOLOv8 adalah versi

terbaru dari algoritma deteksi objek You Only Look Once (YOLO) yang populer, yang pertama kali diperkenalkan oleh Joseph Redmon dkk. pada tahun 2016. Sejak itu, algoritme YOLO telah menjadi pilihan utama untuk pendeteksian objek dan telah mengalami banyak iterasi, dengan YOLOv5 menjadi yang paling dikenal luas. Namun, Ultralytics YOLOv8 memberikan kemampuan paling canggih dan mengungguli versi sebelumnya baik dibidang keakuratan dan kecepatan deteksinya [3]. YOLOv8 juga mendukung model, tugas, dan aplikasi kontribusi pengguna, yang semakin meningkatkan fleksibilitas platformnya. Fitur ini memungkinkan pengguna untuk berbagi model dan tugas mereka, yang dapat membantu mempercepat pengembangan proyek pendeteksi objek.

Dalam upaya mengatasi permasalahan dalam efisiensi pengumpulan data volume kendaraan. Maka penelitian ini kami lakukan guna untuk mengotomatisasi perhitungan jumlah kendaraan yang melintas dengan menggunakan pendekatan citra visual di suatu sudut jalan. Penggunaan pendekatan citra visual ini diharapkan mampu mendeteksi kendaraan dalam berbagai macam kondisi cuaca lingkungan di suatu jalan. Dengan menggunakan metode YOLOv8 sistem ini akan dapat mengenali berbagai jenis kendaraan mulai dari sepeda motor, mobil, hingga kendaraan berat lainnya.

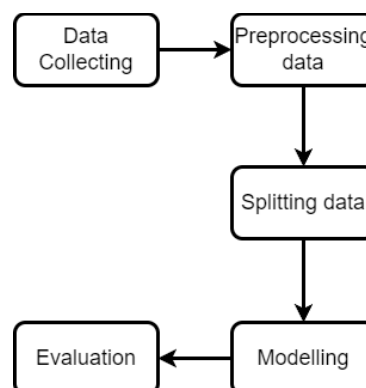
Dengan YOLOv8 diharapkan dapat menghasilkan keakurasian deteksi yang lebih baik dibandingkan versi-versi sebelumnya. Sebagai contoh pada penelian yang dilakukan oleh M. Rizky Pramana dkk pada tahun 2024 dengan judul “Sistem Deteksi Jenis Kendaraan Metode YOLOv4 Untuk Mendukung Transportasi Cerdas Kota Medan”. Dalam penelitian tersebut menghasilkan bahwa Model YOLOv4 berhasil mencapai nilai Mean Average Precision (maP) sebesar 77.88% dalam mendeteksi berbagai objek jenis kendaraan pada masing-masing kelas [4].

Penelitian sistem penghitungan jumlah kendaraan ini memiliki serangkaian tujuan yang jelas. Salah satu fokus utamanya adalah meningkatkan akurasi deteksi kendaraan dengan mengoptimalkan algoritma deteksi. Ini penting untuk meminimalkan kesalahan dalam penghitungan jumlah kendaraan, yang menjadi dasar kehandalan sistem. Selanjutnya penulis berharap penelitian ini dapat berguna untuk mendapatkan data yang berkaitan dengan kepadatan jalan. Sehingga data yang terkumpul dapat menjadi acuan dalam pengaturan lalu lintas serta dapat membantu dalam perencanaan pembangunan di masa yang akan datang.

2. Metode Penelitian

Dalam penelitian ini menggunakan metode research and development (R&D) menyediakan kerangka kerja yang terstruktur, yang mencakup tahap-tahap seperti *data collecting*, *preprocessing data*, *splitting data*, *modelling*, dan *evaluation*, untuk memastikan bahwa pengembangan sistem dilakukan secara teratur dan

terencana [5]. Dimana data yang kami dapatkan akan diolah menggunakan Roboflow serta Notebook Google Colab. Adapun tahap-tahap yang digunakan dapat dilihat pada gambar dibawah ini.



Gambar 1. Metode Penelitian

2.1. Data Collecting

Langkah pertama dalam penelitian ini adalah tahap awal yang penting, di mana kami mengumpulkan data awal dengan objektif dan telah diverifikasi [6]. Pada tahap ini kami melakukan pencarian data-data foto citra lalu lintas yang akan digunakan untuk merancang model *machine learning* yang akan kami buat. Penelitian ini memanfaatkan dataset yang kami ambil dari platfrom Kaggle serta beberapa data yang kami kumpulkan secara mandiri yang disesuaikan dengan citra lalu lintas yang terdapat di Indonesia.

2.2. Preprocessing Data

Pemrosesan awal data dalam pembelajaran mesin, khususnya dalam model YOLO, merupakan langkah penting karena berpengaruh dalam kualitas data yang akan diproses nantinya [6]. Melibatkan beberapa langkah untuk mempersiapkan data mentah sebelum digunakan untuk pelatihan model. Pada tahap ini tujuannya adalah untuk meningkatkan kualitas data yang diolah untuk mendapatkan model pembelajaran mesin yang valid dan lebih akurat serta mencegah terjadinya *overfitting* pada hasil model pelatihannya [7].

Di YOLO, pemrosesan data biasanya melibatkan konversi rasio aspek, seperti mengubah ukuran gambar agar sesuai dengan resolusi yang dibutuhkan oleh model YOLO. Setelah itu, tandai atau beri label pada gambar tersebut untuk menunjukkan lokasi dan kategori barang tersebut. Penandaan atau anotasi objek dilakukan dengan menandai objek yang ingin dideteksi pada gambar, biasanya dalam bentuk koordinat kotak pembatas, dan di YOLO, file hasil anotasi gambar biasanya dalam format file teks. Lalu ada teknik augmentasi data seperti membalik, memutar, dan mengubah kecerahan gambar. Dengan menambahkan lebih banyak informasi, kumpulan data yang lebih serbaguna dapat dihasilkan, sehingga model dapat memiliki contoh yang lebih serbaguna. Hal ini dapat membantu model mempelajari fitur yang lebih canggih

dan meningkatkan keumumannya. Dengan mengikuti langkah-langkah di atas, model YOLO diharapkan dapat belajar dengan baik dan memberikan prediksi deteksi objek yang akurat.

2.3. Splitting Data

Merupakan proses pembagaaian dataset menjadi beberapa bagian, umumnya dataset YOLO dibagi menjadi tiga bagian utama: data train, data valid, dan data test [8]. Proses ini penting untuk dilakukan, karena hal ini dapat menentukan keakuratan hasil model yang akan kita latih. Untuk itu harus benar-benar memperhatikan rasio perbandingan antara data train, data valid, dan data test untuk menghasilkan model yang memuaskan [6]. Data latih digunakan untuk melatih model YOLO, sehingga model dapat belajar mendeteksi objek berdasarkan data ini. Ukuran data latih biasanya cukup besar, sekitar 60-80% dari total dataset. Data validasi digunakan untuk memantau kinerja model selama pelatihan, serta membantu mencegah overfitting. Ukuran data validasi biasanya sekitar 10-20% dari total dataset. Sementara itu, data uji digunakan untuk mengevaluasi kinerja akhir model, mengukur seberapa baik model dapat mendeteksi objek pada data baru yang belum pernah dilihat sebelumnya. Ukuran data uji juga biasanya sekitar 10-20% dari total dataset.

2.4. Modelling

Proses modelling pada YOLO (You Only Look Once) melibatkan perancangan, pelatihan, dan optimisasi model YOLO agar mampu melakukan deteksi objek secara efisien. YOLO menggunakan arsitektur konvolusi neural network (CNN) yang telah dioptimasi untuk deteksi objek yang cepat dan akurat [9]. Dalam pelatihan, model akan belajar mengekstraksi fitur yang relevan dari data pelatihan untuk mendeteksi objek, dengan menggunakan teknik seperti backpropagation, optimisasi fungsi kerugian, dan regularisasi. Terdapat beberapa versi arsitektur YOLO yang tersedia, seperti YOLOv3, YOLOv4, dan YOLOv5, masing-masing dengan perbedaan dalam kompleksitas, kecepatan, dan akurasi. Selain itu, optimisasi hyperparameter, seperti laju pembelajaran, ukuran batch, dan jumlah epoch, juga dilakukan untuk meningkatkan kinerja model dalam mendeteksi objek dengan akurasi dan efisiensi yang tinggi. Evaluasi dan penyetelan model juga dilakukan menggunakan data validasi dan uji untuk memastikan model yang dihasilkan mampu mengatasi variasi kondisi dalam dataset.

2.5. Evaluation

Tahapan terakhir adalah evaluasi model yang dihasilkan menggunakan *confusion matrix* yang digunakan sebagai alat ukur untuk mendapatkan jumlah ketepatan klasifikasi terhadap kelas dengan algoritme. Konsep-konsep seperti True Positive, True Negative, False Positive, dan False Negative dijelaskan dalam analogi deteksi kendaraan. Formula untuk menghitung akurasi, presisi, dan recall

juga disajikan [10]. Nilai Confusion Matrix biasanya ditunjukkan dalam satuan persen (%). Ketika model dirasa sudah memenuhi kriteria, selanjutnya melakukan pengkodean untuk penghitungan kendaraan dengan library opencv pada python.

3. Hasil dan Pembahasan

3.1. Data Collecting

Dalam penelitian ini kami mengumpulkan data dari dua sumber yang berbeda. Pertama, kami berhasil mengumpulkan 8219 gambar citra lalu lintas dari Kaggle dataset library. Selanjutnya kami juga merekam citra video lalu lintas dengan memanfaatkan kamera pengawas lalu mengolah data tersebut hingga terkumpul sebanyak 1577 gambar yang bisa kami gunakan sebagai dataset nantinya.

3.2. Preprocessing Data

Dari data yang telah terkumpul, terdapat sejumlah data yang dinilai kurang sesuai dengan realitas citra lalu lintas yang terdapat di Indonesia serta ada beberapa data yang memiliki citra yang hampir sama. Maka dari itu, dilakukan seleksi gambar terlebih dahulu sebelum melakukan proses pengolahan menjadi dataset. Dari seleksi gambar yang kami lakukan menghasilkan sebanyak 7224 gambar yang dinilai sesuai dengan kriteria citra lalu lintas yang ada di Indonesia.

Selanjutnya dilakukan proses pelabelan atau anotasi pada gambar, yang dimana langkah dilakukan pada website pengelolaan dataset Roboflow. Dari data yang sudah terkumpul kami pilah menjadi 8 kelas, yang dimana tiap-tiap kelas menunjukkan jenis model kendaraan yang dapat dilihat pada gambar 2.

```
# Classes
nc: 8 # number of classes
names: [
  "auto",
  "bus",
  "car",
  "lcv",
  "motorcycle",
  "multiagle",
  "tractor",
  "truck",
] # class names
```

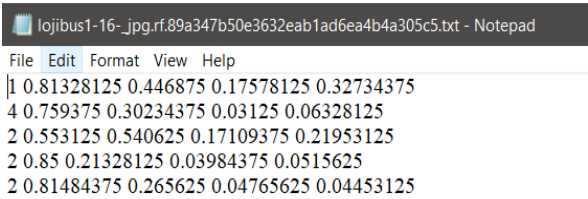
Gambar 2. Classes Object

Dengan mengacu pada 8 kelas tersebut maka anotasi gambar akan disesuaikan dengan citra objek apakah yang terdapat pada gambar. Objek yang dianotasi akan ditandai dengan bounding box yang disesuaikan dengan keterangan kelas objek tersebut seperti pada gambar 3.



Gambar 3. Proses Anotasi

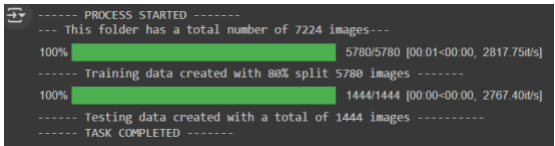
Dari proses anotasi ini akan menghasilkan file berformat txt, yang dimana dalam file tersebut terdapat keterangan dimana koordinat dari objek tersebut berada serta pada kelas apakah objek tersebut akan dikenali. Adapun hasil dari penganotasian gambar tersebut dapat dilihat pada gambar 4.



Gambar 4. Hasil Anotasi

3.3. Splitting Data

Setelah data melalui preprocessing, selanjutnya dataset akan dipisah menjadi dua bagian yaitu data train dan data valid. Dalam tahap ini data dipisah secara acak dalam runtime google colab. Perbandingan data train dan data valid dapat dilihat pada gambar 5.



Gambar 5. Splitting Dataset

Dari gambar 5 terlihat bahwa data sebanyak 7224, dipisah antara data train dan data valid dengan persentase 80% berbanding 20%. Sehingga menghasilkan data train sebanyak 5780 gambar dan data valid sebanyak 1444 gambar.

3.4. Training Model

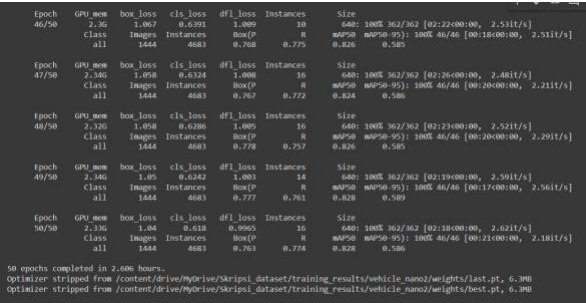
Tahapan training model pada penelitian ini menggunakan metode YOLOv8 dan dilakukan pada runtime Google Colab. Google Colab merupakan suatu platform berbasis cloud yang memfasilitasi pengguna untuk menjalankan dan berbagi notebook Jupyter. Platform ini memberikan kesempatan bagi pengguna untuk memanfaatkan GPU dan TPU secara gratis, yang merupakan aspek penting dalam melatih model pembelajaran yang kompleks [11]. Dataset akan diproses dengan pengaturan sebanyak 50 epoch, yang berarti data yang diproses akan diulang sebanyak 50 kali pelatihan. Hal ini diharapkan agar memperoleh hasil model machine learning yang cukup akurat dalam

mengklasifikasikan objek sesuai dengan kelas yang diharapkan.



Gambar 6. YOLOv8 Command Line

Seperti yang bisa dilihat dari gambar 6, bahwa terdapat beberapa macam command line untuk penggunaan YOLOv8 dalam melakukan training, validasi, dan penyimpulan model dalam beberapa format. Disini kami menggunakan jenis model YOLOv8 Nano, model ini kami pilih dikarenakan tidak terlalu banyak menggunakan sumber daya komputer dalam pemrosesannya.

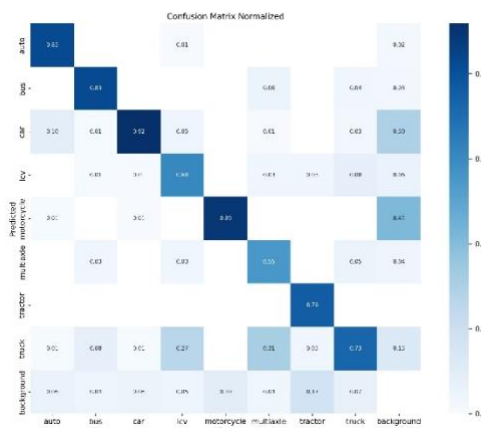


Gambar 7. Proses Training Data

Pada gambar 7 dapat kita lihat bahwa proses training sebanyak 50 epoch telah selesai dan memakan waktu dalam 2,606 jam. Proses ini menghasilkan file berformat PT serta beberapa file evaluasi dari hasil training yang dilakukan.

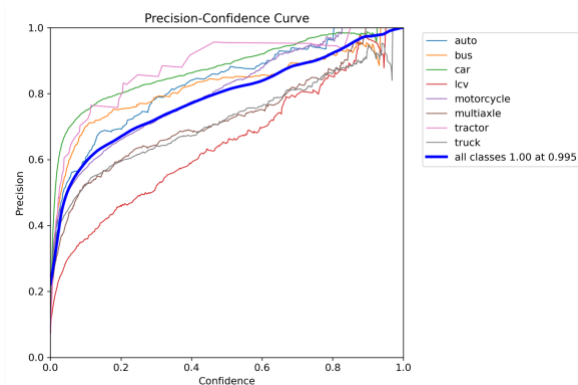
3.5. Evaluasi Model

Dalam tahap ini, evaluasi model dilakukan dengan melihat keluaran hasil dari training yang telah kami lakukan pada tahap sebelumnya. Untuk proses evaluasi ini akan menggunakan confusion matrix yang merupakan sebuah alat yang berguna untuk mengukur performa dari model pelatihan yang kerap digunakan pada masalah klasifikasi objek yang akan menghasilkan output yang terdiri dari dua kelas atau lebih [12]. Berikut merupakan gambar nilai confusion matrix normalize yang menampilkan nilai-nilai secara probabilitas atau persentase sehingga memudahkan perbandingan performa model klasifikasi antar kelas.



Gambar 8. Kurva Confusion Matrix Normalized

Akan ada empat atribut yang akan dijadikan dasar dalam perhitungan beberapa metrik evaluasi.

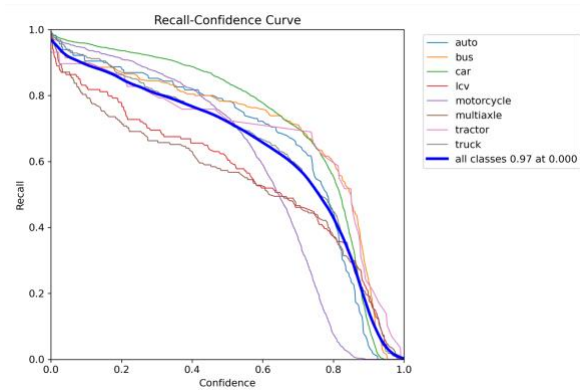


Gambar 9. Kurva Precision-Confidence

Pada gambar 9 merupakan nilai kurva dari atribut *precision-confidence*. *Precision* adalah perbandingan antara *True Positive* (TP) dengan total data yang diprediksi positif. Dengan demikian, *precision* bertujuan untuk mengurangi kemungkinan *False Positive* (FP). Nilai *precision* dapat dihitung menggunakan persamaan (3.18).

$$precision = \frac{TP}{TP + FP} \quad (3.18)$$

Sumbu x menunjukkan level *Confidence* atau tingkat kepercayaan model dalam melakukan klasifikasi. Semakin ke kanan, semakin tinggi tingkat kepercayaan modelnya. Di sisi lain, sumbu y menunjukkan nilai *Precision* atau akurasi prediksi model. Semakin ke atas, semakin tinggi nilai prediksi modelnya. Dari kurva diatas didapat bahwa nilai *precision all classes* mencapai nilai 1.00 pada saat nilai *confidence* pada 0.995.

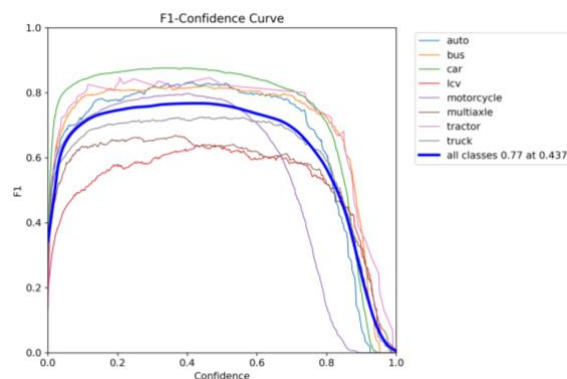


Gambar 10. Kurva Recall-Confidence

Pada gambar 10 merupakan nilai kurva dari *recall-confidence*. *Recall* adalah perbandingan antara *True Positive* (TP) dengan total data yang sebenarnya bernilai positif. Dengan demikian, *recall* bertujuan untuk mengurangi kemungkinan *False Negative* (FN). Nilai *recall* dapat dihitung menggunakan persamaan (3.19).

$$recall = \frac{TP}{TP + FN} \quad (3.19)$$

Sumbu x menampilkan *Confidence* atau tingkat Keyakinan model saat melakukan klasifikasi. Semakin ke kanan, semakin tinggi tingkat keyakinan modelnya. Sumbu y menunjukkan nilai *Recall* atau seberapa banyak objek yang diklasifikasikan dengan benar oleh model. Semakin ke atas, semakin tinggi nilai *Recall* tersebut. Dari kurva diatas maka didapat bahwa Grafik recall-confidence menunjukkan keseluruhan kelas objek pada rentang tingkat kepercayaan dari 0.97 pada 0.000.



Gambar 11. Kurva F1-confidence

Pada gambar 11 merupakan kurva dari atribut F-1 score. F1-score adalah rata-rata harmonis antara *precision* dan *recall*. Nilai F1-score dapat dihitung dengan menggunakan persamaan (3.20).

$$F1\ score = \frac{2 * precision * recall}{precision + recall} + \frac{2TP}{2TP + FP + FN} \quad (3.20)$$

Sumbu x menggambarkan tingkat *Confidence* atau keyakinan model dalam melakukan klasifikasi, dimana semakin ke kanan, tingkat kepercayaan modelnya semakin tinggi. Pada sumbu y, menunjukkan nilai F1-

Score yang merupakan ukuran keseimbangan antara *Precision* dan *Recall* di mana semakin ke atas, semakin tinggi nilai *F1-Score* tersebut. Dari kurva diatas maka didapat bahwa Grafik *F1-Confidence* menampilkan seluruh kelas objek pada rentang tingkat kepercayaan antara 0.77 pada 0.437.



Gambar 12. Tingkat Akurasi Model

Pada gambar 12 merupakan nilai rasio *accuracy* dari deteksi objek yang dilakukan dengan data test. *Accuracy* adalah proporsi prediksi yang benar (baik positif maupun negatif) dari seluruh data. Metrik ini sering digunakan karena kemudahannya perhitungannya. Namun, kelemahannya terletak pada ketidakakuratan dalam menangani data yang tidak seimbang. Rumus untuk menghitung nilai *accuracy* adalah (3.17).

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.17}$$

Nilai dari *accuracy* akan semakin baik jika mendekati angka 1.

Berikut kami paparkan hasil dari 5 epoch terakhir proses training model dari total 50 epoch yang dijalankan.

epoch	metrics/ precision(B)	metrics/ recall(B)	metrics/ mAP50(B)
46	0.76759	0.77501	0.82563
47	0.76674	0.77229	0.8237
48	0.77845	0.7574	0.82614
49	0.77674	0.76116	0.82824
50	0.76306	0.77389	0.82773

Table 1. Hasil Training

Dari data table diatas maka dapat disimpulkan bahwa nilai *Mean Average Precision* (maP) model terbilang cukup baik dimana mencapai 82.77% pada masing-

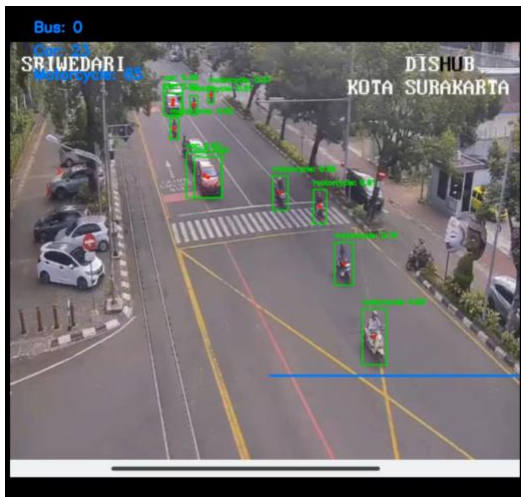
masing kelas. Jika kita bandingkan dengan penelitian yang dilakukan oleh M. Rizky Pramana dkk pada tahun 2024 dengan menggunakan YOLOv4 yang hanya menghasilkan nilai *Mean Average Precision* (maP) sebesar 77.88%. Maka dapat disimpulkan bahwa hasil model yang didapat menggunakan YOLOv8 mendapat score akurasi yang lebih baik.

3.5. Implementasi

Setelah melewati tahap evaluasi dan telah mendapatkan nilai yang memuaskan maka selanjutnya akan dilakukan proses implementasi pada sistem perhitungan volume kendaraan. Pada proses ini akan menggunakan *python library* yang bernama OpenCV. OpenCV yang dimana sebuah perpustakaan sumber terbuka yang menyediakan lebih dari 500 fungsi untuk pengolahan citra dan visi komputer, dalam kasus ini akan digunakan untuk menciptakan alat penghitungan kendaraan yang telah terdeteksi oleh model *machine learning* [13]. Penelitian ini ditujukan untuk memodifikasi model agar dapat mendeteksi serta mengklasifikasi objek gambar dalam frame video beserta menampilkan nama kelas kendaraan. Untuk kelas yang ditampilkan untuk melakukan perhitungan hanya sebatas mobil, motor, dan bus. Hal ini dikarenakan lokasi tempat pengambilan gambar hanya diperuntukan untuk jenis kendaraan tersebut.

Dalam pengujian kinerja sistem, kami menggunakan rekaman video lalu lintas yang diambil dari CCTV ATCS Indonesia di wilayah Sriwedari Kota Surakarta. Durasi setiap video sekitar 120 detik dan diambil pada dua waktu berbeda, siang dan malam. Proses pengolahan video dilakukan dengan membaca input video dalam format mp4 menggunakan perintah VideoCapture dari library OpenCV. Kemudian, model akan melakukan analisis frame-by-frame dari video input untuk menentukan bounding box, kelas objek, dan tingkat kepercayaan deteksi.

Untuk alur perhitungan jumlah kendaraan yang melintas adalah dengan menentukan koordinat titik tengah dari bounding box yang terbentuk ketika model mendeteksi objek. Setelah itu membuat sebuah garis dengan menggunakan ROI yang dimana nantinya garis itu akan menjadi pemicu dalam perhitungan kendaraan berdasarkan kelas yang terdeteksi. Dalam penempatan garis ROI harus memerhatikan dimana arus lalu lintas tersebut mengarah serta pada bagian jalan mana yang ingin kita hitung banyaknya jumlah kendaraan yang melintas. Sistem akan mulai menghitung ketika koordinat titik tengah objek bertemu dengan koordinat garis ROI yang telah dibuat serta akan mengurutkan berapa jumlah objek yang melintas berdasarkan kelas yang terdeteksi. Hasil pengurutan tersebut nantinya akan disimpan dalam sebuah variabel yang telah disesuaikan dengan nama kelas objek tersebut. Untuk hasil perhitungannya dapat dilihat dalam gambar dibawah ini.



Gambar 13. Hasil Deteksi Siang

Pada gambar 13, hasil pengetesan yang dilakukan pada siang hari menunjukkan hasil dimana dalam video yang berdurasi sepanjang 120 detik sistem telah mendeteksi sebanyak 65 motor dan 23 mobil yang telah melintas dalam jalan tersebut serta tidak mendeteksi adanya bus yang melintas.



Gambar 14. Hasil Deteksi Malam

Sedangkan pada gambar 14, hasil pengetesan dilakukan pada malam hari menunjukkan hasil dimana dalam video yang berdurasi 134 detik sistem berhasil mendeteksi terdapat sebanyak 90 motor, 25 mobil, dan 1 bus telah melintas dalam jalan tersebut. Dari pengetesan tersebut membuktikan bahwa sistem telah berjalan dengan baik pada kondisi malam maupun siang hari berfungsi sesuai dengan tujuan yang kami harapkan. Akan tetapi sistem ini memiliki permasalahan ketika video yang diproses terdapat sebuah frame drop maka sistem akan terkendala dalam perhitungan objek tersebut. Hal ini dikarenakan ketika terjadi frame drop terkadang koordinat titik tengah tidak bertemu dengan koordinat garis ROI sehingga sistem tidak dapat melakukan perhitungan objek. Maka dari itu kualitas frame rate pada video yang akan diproses menjadi penentu keakuratan sistem dalam melakukan perhitungan kendaraan yang melintas.

4. Kesimpulan

Studi ini telah mengembangkan implementasi deteksi objek menggunakan metode YOLOv8 untuk mengenali kendaraan serta menghitung volume jumlah kendaraan yang melintas. Pada penelitian ini didapatkan nilai performa deteksi *Mean Average Precision* (maP) mencapai 82.77%. Dimana hasil performanya melebihi nilai *Mean Average Precision* (maP) pada penelitian menggunakan model YOLOv4 yang dilakukan oleh M. Rizky Pramana dkk pada tahun 2024 yang hanya mencapai 77.88%. Dalam penelitian ini, disarankan untuk melakukan penghitungan kendaraan dengan menetapkan identitas unik untuk setiap kendaraan yang terdeteksi. Kemampuan untuk menghitung jumlah kendaraan, seperti mobil, sepeda motor, dan bus sudah berhasil diimplementasikan. Akurasi deteksi dipengaruhi oleh analisis video dari perekaman kamera yang ditempatkan pada posisi dan sudut yang berbeda serta kualitas dari frame rate video tersebut, baik pada kondisi siang maupun malam. Salah satu kelemahan studi ini adalah dimana ketika video yang diproses terdapat frame drop maka sistem terkadang tidak mendeteksi adanya objek yang telah melintas karena titik tengah objek tidak bertemu dengan koordinat garis ROI, sehingga objek yang terdeteksi tidak terhitung. Pengujian model juga belum dilakukan dalam waktu nyata. Selain itu, klasifikasi jenis kendaraan belum didasarkan pada orientasi kendaraan sesuai dengan standar klasifikasi yang ditetapkan oleh dinas perhubungan.

Daftar Rujukan

- [1] R. F. Putra and D. I. Mulyana, "Optimasi Deteksi Objek Dengan Segmentasi dan Data Augmentasi Pada Hewan Siput Beracun Menggunakan Algoritma You Only Look Once (YOLO)," *J. JTik (Jurnal Teknol. Inf. dan Komunikasi)*, vol. 8, no. 1, pp. 93–103, 2024, doi: 10.35870/jtik.v8i1.1391.
- [2] P. Y. Putra, A. S. Arifianto, Z. E. Fitri, and T. D. Puspitasari, "Deteksi Kendaraan Truk pada Video Menggunakan Metode Tiny-YOLO v4," *J. Inform. Polinema*, vol. 9, no. 2, pp. 215–222, 2023, doi: 10.33795/jip.v9i2.1243.
- [3] B. Yolov, Y. O. U. Only, and L. Once, "Abstrak – Tindak kekerasan dan perundungan pada anak merupakan masalah serius yang memerlukan perhatian khusus, terutama di lingkungan sekolah. Deteksi kekerasan dan perundungan masih banyak dilakukan secara manual, yang rentan terhadap kesalahan dan mema," vol. 3, no. 9, 2024.
- [4] M. City, "Sistem Deteksi Jenis Kendaraan Metode YOLOv4 Untuk Mendukung Transportasi Cerdas Kota Medan," pp. 146–161, 2024.
- [5] L. Hulu and A. Bawamenewi, "Pengembangan Media Pembelajaran Bahasa Indonesia Berbasis Komik di Sekolah Menengah Pertama (SMP)," vol. 7, no. 20, pp. 15820–15827, 2023.
- [6] A. I. Pradana et al., "PERBANDINGAN DATA UNTUK MEMPREDIKSI KETEPATAN STUDI Keberhasilan merupakan tujuan yang menempuh Salah satu keberhasilan mahasiswa tersebut dapat dilihat dari ketepatan waktu dalam menyelesaikan studinya (Hasibuan and Mahdiana 2023). Keberhasilan studi m," vol. 8, no. 2, pp. 221–228, 2024.
- [7] Rachmat Santoso, "Augmentasi Data pada Prasasti Logam untuk Deteksi Aksara Kawi," *J. Fasilkom*, vol. 14, no. 1, pp. 234–241, 2024, doi: 10.37859/jf.v14i1.6952.

-
- [8] F. A. Artanto, "Support Vector Machine Berbasis Particle Swarm Optimization Pada Analisis Sentimen Anggota KPPS," *J. Fasilkom*, vol. 14, no. 1, pp. 75–79, 2024, doi: 10.37859/jf.v14i1.6795.
- [9] C. Fatichah and R. Dikairono, "122326-242135-1-Pb," vol. 12, no. 3, 2023.
- [10] M. K. Suryadewiansyah and T. E. E. Tju, "Naïve Bayes dan Confusion Matrix untuk Efisiensi Analisa Intrusion Detection System Alert," *J. Nas. Teknol. dan Sist. Inf.*, vol. 8, no. 2, pp. 81–88, 2022, doi: 10.25077/teknosi.v8i2.2022.81-88.
- [11] R. Gelar Guntara, "Deteksi Atap Bangunan Berbasis Citra Udara Menggunakan Google Colab dan Algoritma Deep Learning YOLOv7," *J. Manaj. Sist. Inf.*, vol. 2, no. 1, pp. 9–18, 2023, doi: 10.59431/jmasif.v2i1.156.
- [12] T. I. Z. M. Putra, S. Suprpto, and A. F. Bukhori, "Model Klasifikasi Berbasis Multiclass Classification dengan Kombinasi Indobert Embedding dan Long Short-Term Memory untuk Tweet Berbahasa Indonesia," *J. Ilmu Siber dan Teknol. Digit.*, vol. 1, no. 1, pp. 1–28, 2022, doi: 10.35912/jisted.v1i1.1509.
- [13] Normalisa, A. Rachmaniar, D. Diana, M. Saefudin, and R. Parulian, "Application Of Computer Vision Detection Of Apples And Oranges Using Python Language," *J. Inf. Syst. Informatics Comput.*, vol. 6, no. 2, pp. 455–466, 2022, doi: 10.52362/jisicom.v6i2.946.