**Jurnal Aplikasi Teknik Sipil**
*Volume 23, Nomor 3, Agustus 2025*

Journal homepage: http://iptek.its.ac.id/index.php/jats

# Sistem Klasifikasi Dan Deteksi Kendaraan Otomatis Dengan Custom Dataset YOLOv8 di Kota Balikpapan

**Muhammad Hadid[1,*], Muhammad Reza Hermawan[1], Ardiansyah Fauzi[1], Arief Hidayat[1]**
Jurusan Teknik Sipil dan Perencanaan, Institut Teknologi Kalimantan, Balikpapan[1]
Koresponden*, Email: *hadid@lecturer.itk.ac.id*

Info Artikel
Diajukan 28 Desember 2023
Diperbaiki 08 Agustus 2025
Disetujui 15 Agustus 2025

*Keywords: deep learning, detection, YOLOv8*

Kata kunci: *deep learning*, deteksi, YOLOv8

***Abstract***

***Vehicle counting surveys are still conducted manually by deploying surveyors in the field. This approach faces several challenges, including the need for high concentration, physically demanding nature of task, and the requirement for many surveyors, which are inherent limitations of manual data collection. A viable alternative is to fully adopt artificial intelligence. This study employs one branch of machine learning, namely deep learning, to design an automatic vehicle detection system utilizing the YOLOv8 algorithm. The dataset was developed from camera footage at an intersection by capturing images of each passing vehicle. From these images, 80% were used for training and the remaining 20% for testing. The analysis results indicate the system's performance achieved accuracy rates ranging from 96.92% in the morning to 100% during the day, and from 91.43% to 100% at night. Furthermore, the F1-Score values ranged from 67% to 100% in daytime, and from 80% to 100% at night.***

**Abstrak**

**Saat ini survey perhitungan kendaraan masih dilakukan secara manual dengan penempatan surveyor di lapangan. Tantangannya meliputi kebutuhan akan konsentrasi tinggi, pekerjaan yang menguras energi, dan kebutuhan akan surveyor dalam jumlah besar yang merupakan kelemahan dalam pengumpulan data manual. Pendekatan dapat diambil dengan sepenuhnya dengan pemanfaatan kecerdasan buatan. Penelitian ini menggunaan salah satu cabang *machine learning* yakni *deep learning* untuk merancang sistem pendeteksi kendaraan otomatis dengan memanfaatkan algoritma YOLOv8. Dataset dibuat berdasarkan data kamera yang terdapat pada persimpangan dengan mengambil gambar setiap kendaraan. Dari data gambar setiap kendaraan kemudian dilakukan training dengan menggunakan 80% dari data gambar, dan sisanya digunakan untuk pengujian. Hasil dari analisis menunjukkan bahwa performa sistem dengan tingkat akurasi yang berkisar antara 96,92% (pagi) hingga 100% (siang) dan 91,43%–100% (malam). Hasil lain menunjukkan bahwa nilai score F1 pada pagi atau siang hari dan malam hari secara berurutan adalah 67%-100% dan 80%-100%.**

## 1. Pendahuluan

Peningkatan volume kendaraan yang beroperasi di tiap tahunnya di Kota Balikpapan yang jika tidak diiringi dengan manajemen rekayasa lalu lintas yang memadai maka akan menyebabkan kemacetan di kemudian hari. Saat ini survei lalu lintas masih dilakukan manual menggunakan tenaga surveyor yang mana memerlukan tingkat konsentrasi yang tinggi [1]–[3]. Pendekatan yang dapat dilakukan dalam penyelesaian permasalahan terhadap pekerjaan dari surveyor di lapangan dapat digantikan sepenuhnya dengan pemanfaatan dari kecerdasan buatan [1], [2], [4]. *Deep learning* yang merupakan salah satu cabang dari kecerdasan buatan dapat dimanfaatkan sebagai deteksi kendaraan otomatis dengan mengimplementasikan teknik computer vision yaitu dengan pendekatan algoritma *You Only Look Once* versi 8 (YOLOv8) sebagai deteksi kendaraan berbasis *real-time* [5].

YOLOv8 merupakan versi terbaru YOLO yang digagas Ulyralytics yang memiliki arsitektur yang serupa dengan pendahulunya namun juga memperkenalkan banyak peningkaran seperti arsitektur *neural network* baru yang menggunakan *Feature Pyramid Network* (FPN) dan *Path Aggregation Network* (PAN) [5]–[7] yang telah menyediakan *Command Line Interface* (CLI) dan Python SDK untuk melaksanakan proses latih data validasi data, dan interferensi secara sekaligus [7]. YOLOv8 sendiri saat ini memiliki model sebanyak lima buah yakni *nano*, *small*, *medium*, *large*, dan *xtra large*. Ada beberapa fungsi perintah pada tugas yang dapat dijalankan, yakni *classify*, *detect*, dan *segment*. Adapun untuk mode yang dapat dijalankan terdapat 4 jenis yaitu *predict*, *train*, *val*, dan *export* [7]. Fungsi *predict* digunakan

untuk melakukan prediksi *frame* yang dituju, baik berupa gambar maupun video berdasarkan modul yang ditentukan, train digunakan untuk melatih *dataset* dengan *pre-trained module configuration* yang ditentukan, *val* yaitu untuk memvalidasi akurasi serta mengevaluasi performa dari model yang telah dilatih, dan terakhir yaitu *export* yaitu untuk mengekstrasi modul dengan format yang ditentukan [7].

Selain untuk dimanfaatkan di bidang transportasi [1], [4], YOLO juga dapat membantu dalam proses evakuasi korban bencana alam dengan akurasi yang tinggi [8]

YOLOv8 merupakan pengembangan dari *ultralytics* yang telah dirilis bulan Januari 2023 oleh Glenn Jocher setelah kesuksesannya terhadap YOLOv5 pada tahun 2020. YOLOv8 terdiri dari dua bagian: *Backbone* yang bertanggung jawab untuk menghasilkan piramida fitur setelah ekstraksi fitur dan *Head* yang bertanggung jawab untuk mengidentifikasi objek dan menampilkan kotak pembatas bersama dengan nilai *objekness* [9]. YOLOv8 dapat dijalankan melalui antarmuka baris perintah (CLI) juga dapat diinstal menggunakan PIP. YOLOv8 menggunakan augmentasi *mosaic* selama pelatihan namun telah ditemukan bahwa augmentasi ini dapat mengurangi presisi jika digunakan selama seluruh proses pelatihan, yang mana augmentasi ini tidak benar-benar dimatikan selama 10 *epoch* terakhir, namun augmentasinya dikurangi untuk menghindari *overfitting* data [10]. Augmentasi *mosaic* adalah teknik dimana 4 gambar pelatihan digabungkan menjadi satu gambar baru. Teknik augmentasi ini membantu model untuk melatih dan belajar dengan lebih efisien [11]. YOLOv8 lebih efisien daripada versi sebelumnya karena menggunakan peta fitur yang lebih besar dan jaringan konvolusi yang lebih efisien [12].

Dengan demikian, studi ini bertujuan untuk merancang arsitektur sistem deteksi kendaraan melalui YOLOv8 sebagai dasar model sistem deteksi kendaraan yang menggunakan *mechine learning* dalam. Studi ini menggunakan lokasi penelitian pada persimpangan bersinyal di jalan Jenderal Ahmad Yani, Muara Rapak (Simpang Rapak), Kota Balikpapan. Pengambilan lokasi ini didasari oleh lokasi kamera dengan sudut kamera tidak terlihat menimpa satu sama lain atau mengalami *overlapping* yang begitu signifikan untuk dilakukan pengambilan data video [1], [2], [4].

## 2. Metode

Proses awal yang akan dilakukan ialah pengumpulan data yang merupakan data sekunder yaitu dataset video lalu lintas kendaraan diperoleh dari Dinas Perhubungan Kota Balikpapan pada Simpang Rapak dari berbagai sudut.

Objektif utama penelitian ini adalah untuk merancang model yang dapat mendeteksi dan mengklasifikasikan kendaraan secara otomatis dengan bantuan YOLOv8 melalui rekaman video lalu lintas, dengan keterangan 6 kelas kendaraan yang dipisah antara kelas kendaraan umum dan kelas kendaraan spesifik kemudian diukur metrik performa dari sistem. Analisis data dilakukan untuk mendapatkan persentase akurasi, presisi, *recall*, dan F1-Score dari sistem deteksi dan klasifikasi kendaraan yang menggunakan custom dataset terlatih. Dalam penelitian ini, nilai *F1-Score* minimum yang diharapkan adalah 80%, yang mengindikasikan model dengan baik dapat memprediksi masing-masing kelas dengan tingkat klasifikasi yang baik. Beberapa faktor yang mempengaruhi akurasi dari pendeteksian dalam klasifikasi kendaraan diantaranya terhadap proses penganotasian atau labelling pada tiap objek [1], [8] serta penggunaan dari konfigurasi hiperparameter juga berpengaruh secara signifikan terhadap performa dari data hasil training [13].

Durasi video uji yang digunakan yaitu 1 menit per video serta banyak video penelitian sebanyak total 9 video pada 3 hari berbeda. Kemudian video diekstraksi menjadi kumpulan frame gambar. Dilakukan proses anotasi data dengan pemberian label nama kelas pada setiap objek dengan menggunakan labelImg. Proses *training dataset* yang akan ditinjau terdiri atas gambar klasifikasi kendaraan umum terdiri atas kelas MC (sepeda motor), LV (kendaraan ringan roda empat), dan HV (kendaraan berat) dan klasifikasi kendaraan spesifik LV untuk kendaraan angkutan LV(ANGKUTAN), LV(BOX) untuk mobil box, dan HV(BUS) untuk bus. Kemudian dilakukan training data pada kumpulan dataset menggunakan *pre-trained* model YOLOv8 Nano dengan Google Colaboratory dan akan terbentuk karakteristik berupa file weights terakhir yang merupakan konfigurasi custom model YOLOv8 terlatih dari dataset. Proses latih modul dataset sudah cukup untuk dapat dijalankan menggunakan bantuan Google Colab untuk memperoleh *output custom weights model* dari YOLOv8 selain dengan menggunakan PC bawaan [14]. Alasan digunakannya *Google Colab* ialah dikarenakan *Colab* dapat dengan mudah dijalankan secara *cloud*, serta tersedianya *graphic processor unit* (GPU) bawaan dari *Colab* untuk menjalankan proses training.

Setelah file hasil *training* diperoleh kemudian merancang sistem deteksi dan klasifikasi kendaraan dengan bantuan *library* dari Ultralytics YOLOv8, OpenCV, dan numpy. Sistem ini dijalankan menggunakan bahasa pemrograman Python dan dengan IDE Pycharm yang mana YOLO hanya digunakan sebatas untuk mendeteksi kendaraan yang

terdapat pada gambar namun tidak difokuskan sampai proses untuk menghitung jumlah kendaraan. Sistem secara garis besar terdiri atas bagian *import library*, *path label*, *module configuration weights*, *model network*, lokasi penyimpanan video, fungsi kelas dari objek, penentuan kelas yang akan dideteksi, pembuatan bounding box. Implementasi dilakukan dari hasil desain menggunakan bahasa python. Setelah video selesai diproses kemudian secara otomatis pula *output* video yang telah melalui proses *rendering* tersimpan kedalam direktori *folder* yang telah ditentukan.

Pada penelitian ini ditinjau kemampuan sistem untuk mendeteksi dan mengklasifikasikan kendaraan berdasarkan kelasnya, serta mengukur metrik performa dari sistem. *Confusion matrix* merupakan salah satu metode pengukuran kinerja dari suatu sistem klasifikasi untuk mengevaluasi model deteksi yang terdiri atas *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN). Pengukuran kinerja dari *confusion matrix* dapat diolah melalui representasi hasil proses klasifikasi (nilai TP, TN, FP, dan FN) untuk dievaluasi nilai akurasi, presisi, recall, dan skor F1 dari kinerja sistem [12].

Dalam *confusion matrix*, "*actual*" atau "*ground truth*" merujuk pada nilai kelas sebenarnya atau label yang benar untuk setiap sampel data. Di lain sisi, "*predicted value*" atau "prediksi" merujuk pada nilai yang diprediksi oleh model atau sistem. Kedua nilai ini digunakan untuk membandingkan dan mengukur performa model dalam mengklasifikasikan atau memprediksi kelas objek dengan benar [12]. TP adalah saat sistem mendeteksi salah satu dari kelas objek kendaraan yang melewati area bayang yang telah ditentukan, misal pada MC, TP bertambah 1 saat kendaraan *Motorcycle* terdeteksi benar sebagai MC saat melewati garis bayang yang ditentukan. TN adalah saat sistem mendeteksi secara benar kelas kendaraan selain MC yang melewati garis bayang. FP adalah saat sistem mendeteksi kelas selain *Motorcycle* sebagai MC. FN adalah saat sistem mendeteksi *Motorcycle* sebagai kelas selain MC. Nilai akurasi menggambarkan sejauh mana model dapat memprediksi dengan benar secara keseluruhan. Rumus dari nilai akurasi (A) dapat dilihat pada Persamaan (1).

$$\mathrm{A} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}} \times 100\% \qquad (1)$$

Tingkat presisi mengukur seberapa akurat model dalam mengidentifikasi kelas yang spesifik. Rumus dari nilai presisi (P) dapat dilihat pada Persamaan (2).

$$\mathrm{P} = \frac{\mathrm{TP}}{\mathrm{FP} + \mathrm{TP}} \times 100\% \qquad (2)$$

*Recall* adalah matrik evaluasi yang mengukur sejauh mana model dapat mendeteksi atau mengklasifikasikan dengan benar objek dari kelas yang tepat secara keseluruhan. Rumus untuk memperoleh nilai recall dapat dilihat pada Persamaan (3).

$$\mathrm{Recall} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \times 100\% \qquad (3)$$

*F1-Score* menggabungkan presisi dan recall menjadi satu nilai yang mencerminkan keseluruhan performa model dalam mengklasifikasikan objek kendaraan. Dengan demikian, *F1-Score* memberikan gambaran komprehensif tentang keseimbangan antara presisi dan *recall* dalam sistem deteksi kendaraan yang mana pada matrik akurasi tidak dapat digambarkan. Rumus untuk memperoleh nilai F1-Score dapat dilihat pada Persamaan (4).

$$\mathrm{F1Score} = 2 \times \frac{(\mathrm{Presisi} \times \mathrm{Recall})}{(\mathrm{Presisi} + \mathrm{Recall})} \times 100\% \qquad (4)$$

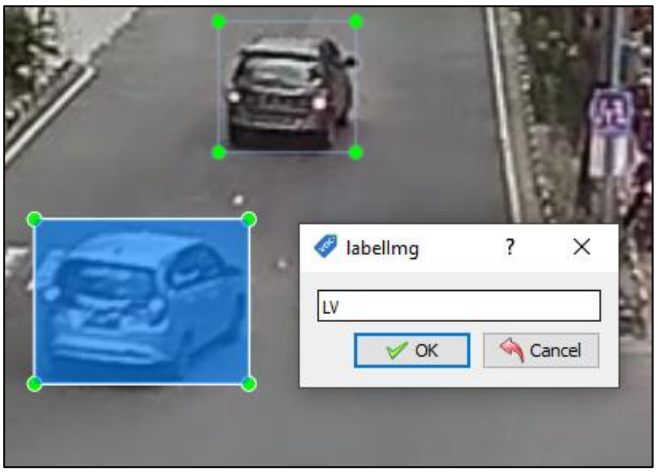
Proses anotasi data dengan keterangan banyak dataset akhir yang digunakan diambil dari 3 sudut CCTV berbeda yakni pada file video 1 (RAPAK FIX 1) sebanyak 147 himpunan data, pada file video 2 (SP. RAPAK) sebanyak total 159 himpunan data, kemudian terakhir pada file video 3 (RAPAK FIX 2) sebanyak 150 himpunan data. Sehingga, Total dataset akhir yang digunakan adalah 456 dataset yang dapat dilihat pada **Tabel 1**.

**Tabel 1**. Jumlah dataset yang digunakan

| Waktu pengambilan data CCTV | Banyak himpunan dataset |
|---|---|
| Pukul 23:53:16 – 00:13:20 | 147 |
| Pukul 17:49:48 – 17:52:11 | 83 |
| Pukul 00:08:17 – 00:23:32 | 76 |
| Pukul 11:48:20 – 12:08:18 | 150 |
| **Total** | **456** |

LabelImg yakni software yang digunakan untuk anotasi/pelabelan objek pada frame (**Gambar 1**). Setelah anotasi data selesai dilakukan, dilanjutkan proses training data yang dilaksanakan melalui *Google Colaboratory* bertujuan untuk melatih sistem dalam pengolahan data yang telah diberi anotasi agar terbentuk suatu karakteristik sebagai pertimbangan dalam mencapai suatu prediksi. *Google Colaboratory* digunakan dalam proses latih dataset untuk memperoleh model weights YOLOv8 dari custom dataset yang telah dianotasi sebelumnya. Diperlukan terlebih dahulu persiapan terkait konfigurasi yang diinginkan dan model *pre-*

*trained* dari YOLOv8. Pelatihan dilakukan selama 10 epoch dengan *batch size 32* dan ukuran gambar 640.




**Gambar 1**. Proses anotasi data pada CCTV

Hasil pelatihan disimpan pada direktori "/content/drive/MyDrive/yolov8/training_results" dengan nama "kendaraan". Optimizer yang digunakan adalah SGD dengan beberapa parameter seperti learning rate (lr), momentum, dan weight decay. Evaluasi model dilakukan menggunakan metrik box loss, class loss, dan DFL loss. Setelah pelatihan selesai, dilakukan evaluasi pada data validasi dengan menggunakan metrik precision (P), recall (R), dan *mean average precision* (mAP) dengan threshold 0.5 dan 0.5-0.95.

Selanjutnya dilakukan implementasi algoritma YOLOv8 untuk melakukan pendeteksian serta pengklafisikasian kendaraan berdasarkan kelasnya masing-masing. Proses ini menghasilkan luaran berupa file weight dengan format ".pt" yang terlah di modifikasi berdasarkan banyak kelas kendaraan dan dataset latih yang telah diproses (**Gambar 2**). *File custom weight* selanjutnya akan diunduh untuk selanjutnya dapat diimplementasikan kepada sistem untuk diuji dan dijalankan pada bahasa pemrograman python dan PyCharm sebagai IDE-nya. Selanjutnya dibuat baris kode untuk dapat melakukan prediksi kendaraan otomatis.

Baris pertama pada **Gambar 2** adalah untuk mengimpor package dari Ultralytics untuk dapat menjalankan model YOLOv8. Baris kedua dan ketiga untuk load pre-trained module dan mendefinisikan lokasi targetnya dari YOLO yang mana menginisialisasi model YOLO yang digunakan. Baris terakhir ialah saat ingin menggunakan fungsi prediksi *objek predict()* dengan *pre-trained module* yang tersedia oleh YOLOv8. Pada bagian yang didalam kurung terdapat 3 parameter yaitu yang pertama *source* yang mana angka 0 berarti merujuk pada kamera yang ada pada komputer, jika ingin merujuk pada video yang lainnya, dapat dengan mengganti "*0*" dengan nama file dan disertakan direktori dari file tersebut jika file tidak berada pada direktori folder yang sama. Parameter show disetting untuk *True*, untuk menampilkan hasil deteksi di window terpisah. Parameter conf menetapkan *confidence score* untuk deteksi objek menjadi 0,60 yang artinya hanya objek dengan skor di atas 0,60 yang akan terdeteksi. Secara keseluruhan, kode ini memuat model YOLO, melakukan deteksi objek pada file video, dan menampilkan hasil deteksi menggunakan *library* Ultralytics YOLO. Pada proses yang akan dijalankan pada PyCharm IDE dibentuk rangkaian code yang berfungsi untuk menampilkan gambar/video pada *dekstop*, *load custom* model YOLO, memproses *bounding box* beserta *confidence score*-nya, *resize* gambar/video, dan ekstraksi video untuk selanjutnya diolah pada proses pengujian sistem.

```
from ultralytics import YOLO

YOLO_MODEL_PATH = r"C:\Users\yolov8.pt"

model = YOLO(YOLO_MODEL_PATH)

model.predict(source="0", show = True,
conf = 0.80)
```

**Gambar 2.** Tampilan Baris Kode Prediksi YOLOv8

Selanjutnya proses pengujian sistem dilakukan untuk mengetahui hasil dari model deteksi dan klasifikasi kendaraan. Pengujian dilakukan dengan memasukkan input video CCTV Dishub Kota Balikpapan kepada baris kode yang telah dirancang.

Tahap selanjutnya dilakukan implementasi algoritma YOLO untuk melakukan pendeteksian serta pengklafisikasian kendaraan berdasarkan kelasnya masingmasing. Implementasi dilakukan kepada hasil desain sistem dengan menggunakan bahasa pemrograman python pada komputer dan ditambah library yang diperlukan sebagai penunjang.

Selanjutnya pada algoritma yang telah dirancang (**Gambar 3**) kemudian dijalankan untuk mendeteksi kendaraan berdasarkan kelasnya masing-masing. Dilakukan pengujian terhadap konsistensi model deteksi dengan melakukan pengulangan running sebanyak total 30 kali untuk mengetahui ada atau tidaknya deviasi yang signifikan untuk menentukan bahwa model deteksi berdasarkan custom weight yang dibangun sudah konsisten dalam memprediksi banyaknya objek kendaraan yang lalu-lalang. Kemudian setelah dinyatakan bahwa model deteksi konsisten dilanjutkan dengan melakukan perhitungan secara manual

dan diperoleh hasil jumlah kendaraan yang teprediksi di tiap kelasnya masing-masing dengan keterangan terkait TP, TN, FP, dan FN dari tiap kelas kendaraan. Hasil dari perhitungan kemudian dilanjutkan kepada proses analisis data untuk melihat performa sistem terhadap tingkat akurasi, presisi, recall, dan terakhir F1-Score dari sistem yang akan menjadi salah satu dasar dalam menyatakan model deteksi untuk dapat digunakan sebagai detektor kendaraan *multi-class*.

```
import cv2
from ultralytics import YOLO

# Initialize the YOLO model
model = YOLO("best may.pt")

# Set the source image or directory
source = "*.mp4"

# Perform prediction on the source
predictions =
model.predict(source=source, show=True)
# conf=0.55
# Save the images/videos predictions
results = model(source=source,
save=True, stream=True)
for r in results:
    boxes = r.boxes  # Boxes object for
bbox outputs
    masks = r.masks  # Masks object for
segment masks outputs
    probs = r.probs  # Class
probabilities for classification outputs

    if cv2.waitKey(0) & 0xFF==27:
        break

# Release the video capture and close
windows
predictions.release()
cv2.destroyAllWindows()
```

**Gambar 3.** Baris kode dengan custom model YOLOv8 yang digunakan dalam proses tiap video uji

## 3. Hasil dan Pembahasan

Setelah model yang dirancang telah berhasil mendeteksi kendaraan motor (MC), mobil ringan (LV), mobil berat (HV), dan klasifikasi angkutan umum (LV (ANGKUTAN)), mobil box (LV (BOX)), dan bus besar (HV (BUS)), dilanjutkan pada proses pengujian performa sistem. Pengujian sistem dilakukan dengan menghitung arus lalu lintas yang berada di area terkait sesuai pada **Gambar 4** yang mana penganotasian data dilakukan di area terkait pada tiap sudut CCTV.

Pada CCTV SP. RAPAK dilakukan pengekstraksian data lalu lintas menuju arah Jl. Ahmad Yani, Jl. Soekarno Hatta, dan terakhir menuju arah Jl. Klamono dan Jl. Ahmad Yani dalam selama 1 menit video berlangsung yang untuk kemudian dilakukan pendeteksian di tiap sampel videonya. Video uji CCTV SP. RAPAK selanjutnya diproses menggunakan model YOLOv8. Secara default hasil prediksi akan tersimpan pada folder yang sama dengan lokasi kode dijalankan yang kemudian berada didalam 'runs/predict'.

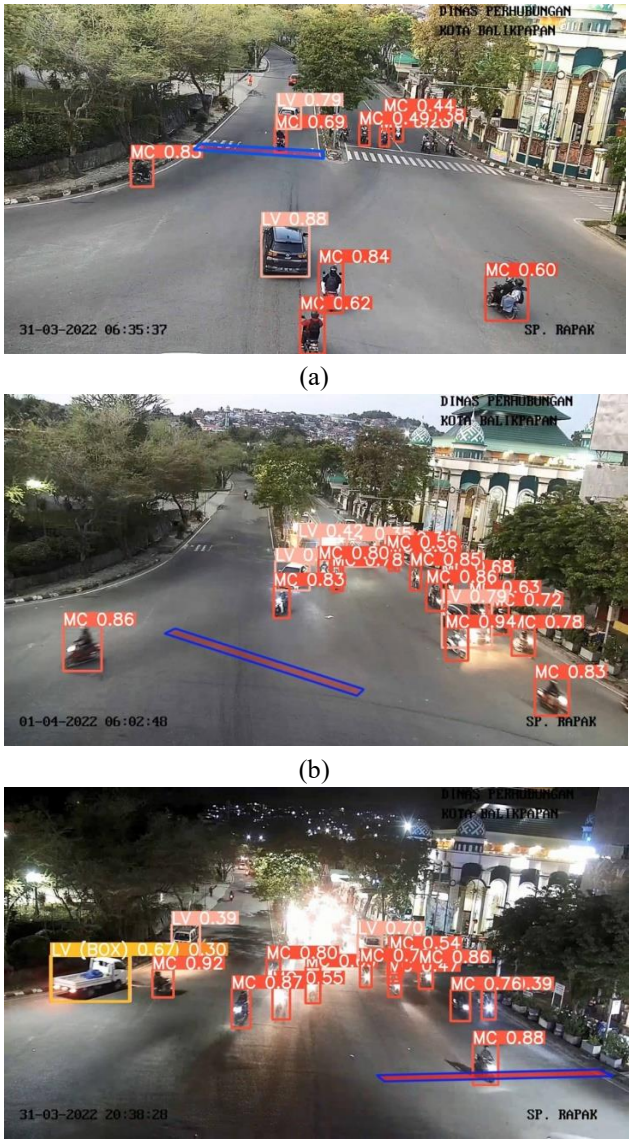

**Gambar 4**. Contoh area anotasi pada Simpang Rapak

Berikutnya pengujian dilakukan dengan video yang telah disebutkan pada **Gambar 4** dengan lama durasi per-video yakni 1 menit. Pertama dilakukan pengujian terhadap data hasil eksekusi sistem terhadap video dengan interval waktu pada pagi hari yakni pukul 6:35:32, siang pada pukul 13:09:54, dan malam pada pukul 23:23:50. Kemudian performa dari sistem dalam mendeteksi kendaraan ditinjau terhadap arah Jl. Ahmad Yani yang garis bayangnya dapat dilihat pada **Gambar 5a**. Selanjutnya, diperoleh hasil perhitungan terhadap pengujian performa sistem pada CCTV SP. RAPAK menuju arah Jl. Klamono dan Ahmad Yani untuk tiap kelasnya yang dapat dilihat pada **Tabel 2**, **Tabel 3**, **Tabel 4** dengan rekapitulasi terhadap tingkat akurasi dan nilai F1-Score akhir (**Tabel 5**).

Merujuk pada penentuan tingkat nilai diagnosa [15] untuk model yang telah dilatih ialah sebesar 80% dengan diagnose yang bernilai 0,80–0,90 yang sudah terklasifikasi sebagai *fair - good classification*. Adapun beberapa nilai kosong yang terdapat pada **Tabel 5** dikarenakan pada perhitungan tingkat presisi dan nilai *recall* dari sistem tidak ditemukannya kelas kendaraan yang dituju dalam interval waktu yang telah ditentukan, namun sebagian besar nilai sudah cukup dalam merepresentasikan hasil kinerja dan performa dari sistem deteksi.

Secara garis besar, merujuk pada tingkat pendeteksian model pada kelas secara umum (**Tabel 2**, **Tabel 3**, dan **Tabel 4**) menunjukkan bahwa model telah memiliki akurasi yang cukup tinggi dalam mendeteksi tiap objek kendaraan yang mana model memiliki kinerja yang lebih baik dalam

mengklasifikasikan sampel dari kelas kendaraan umum yakni MC, HV, dan LV.




**Gambar 5**. Garis bayangan perhitungan (a) lengan Ahmad Yani; (b) lengan menuju Jalan Soekarno-Hatta; (c) jalan menuju Jalan Klamono dan Ahmad Yani

Terkait kelas kendaraan lainnya merujuk pada hasil perolehan tingkat akurasi sistem dapat disimpulkan bahwa model juga dapat mendeteksi objek kendaraan tersebut namun kebanyakan terdeteksi dengan tingkat kepercayaan yang rendah serta kelas kendaraan tersebut juga dapat terprediksi sebagai 2–3 kelas kendaraan lainnya. Hasil ini sejalan dengan penelitian terdahulu dimana tingkat akurasi penggunaan YOLO untuk deteksi kendaraan mencapai nilai akurasi melebihi 80% [1], [4]. Beberapa kelas kendaraan yang kurang terepresentasi nilainya dapat selanjutnya dijelaskan pada hasil luaran *confusion matrix normalized* yang ditunjukkan pada **Gambar 6**.

**Tabel 2**. Rekapitulasi nilai akurasi pada CCTV SP. RAPAK ke arah Jl. Ahmad Yani

| Classes | Pagi | Siang | Malam |
|---|---|---|---|
| MC | 100% | 100% | 100% |
| LV | 95.8% | 97.3% | 91.4% |
| LV (ANG) | 97.2% | 97.3% | 97.0% |
| LV (BOX) | 98.6% | 100% | 94.1% |
| HV | 100% | 100% | 100% |
| HV (Bus) | 100% | 100% | 100% |

**Tabel 3**. Rekapitulasi nilai akurasi pada CCTV SP. RAPAK ke arah Jl. Soekarno-Hatta

| Classes | Pagi | Siang | Malam |
|---|---|---|---|
| MC | 100% | 100% | 100% |
| LV | 90% | 97.0% | 100% |
| LV (ANG) | 100% | 97.0% | 100% |
| LV (BOX) | 100% | 100% | 100% |
| HV | 88.9% | 100% | 100% |
| HV (Bus) | 100% | 100% | 100% |

**Tabel 4**. Rekapitulasi nilai akurasi pada CCTV SP. RAPAK ke arah Jl. Klamono dan Ahmad Yani

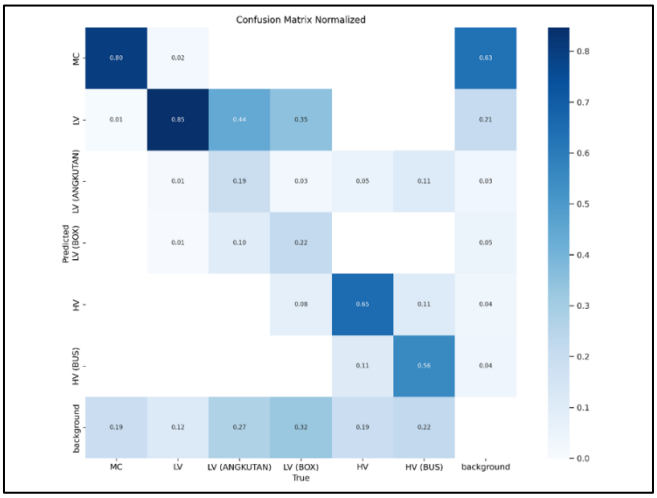
| Classes | Pagi | Siang | Malam |
|---|---|---|---|
| MC | 98.2% | 100% | 98.3% |
| LV | 98.3% | 96.9% | 98.3% |
| LV (ANG) | 100% | 98.4% | 98.3% |
| LV (BOX) | 100% | 100% | 100% |
| HV | 88.9% | 100% | 100% |
| HV (Bus) | 100% | 100% | 100% |

**Tabel 5**. Rekapitulasi nilai F1-score pada CCTV SP. RAPAK

| Classes | Pagi | Siang | Malam | $\bar{x}$ |
|---|---|---|---|---|
| MC | 99.5 | 100 | 99.6 | 99.7 |
| LV | 82.7 | 94.4 | 93.0 | 90.0 |
| LV (ANG) | 100 | 69.4 | 80.0 | 83.2 |
| LV (BOX) | 66.7 | 55.6 | - | 40.7 |
| HV | 66.7 | 100 | - | 55.6 |
| HV (Bus) | - | - | - | - |

Dengan melakukan normalisasi pada *confusion matrix*, pemahaman mengenai kinerja relatif dari klasifikasi di berbagai kelas menjadi terlihat lebih jelas. Hal ini membantu mengidentifikasi kelas-kelas yang mungkin memiliki akurasi yang lebih tinggi atau lebih rendah, serta memungkinkan

perbandingan dan evaluasi yang lebih baik terhadap kinerja model. Kelas yang condong berhasil dan stabil terdeteksi sesuai dengan kelasnya yang terlihat pada grafik *confusion matrix normalized* ini ialah kelas MC yakni 80% dan LV 85%, kemudian diikuti dengan kelas HV 65% dan HV BUS 56% yang masih perlu sedikit lagi untuk memperkaya *dataset* terkait objek kendaraannya. Selanjutnya kelas yang menjadi prioritas utama untuk diperkaya *dataset*-nya ialah kelas LV(BOX) yakni 22% dan LV(ANGKUTAN) 19% yang mana kedua kelas ini untuk dapat dideteksi secara spesifik masih sering salah terdeteksi sebagai LV yang merupakan kelas kendaraan ringan secara general dan masih sangat bervariasi.




**Gambar 6**. Grafik *cunfusion matrix normalized*

Kelas MC termasuk salah satu kelas yang sangat konsisten untuk dapat terdeteksi oleh sistem dan sangat kecil kemungkinannya untuk salah terdeteksi sebagai kelas lainnya. Di sisi lain, MC merupakan satu kelas yang tidak ada kelas lain yang bentuknya serupa dengannya. Adapun beberapa hal yang menyebabkannya terdeteksi sebagai kelas lain yakni terdeteksi sebagai LV sebesar 1% dan background sebesar 19% ialah dikarenakan sangat beragamnya bentuk, arah, intensitas cahaya, dan ukuran kelas MC yang teranotasi yang mana pada dataset malam hari banyak kelas kendaraan MC yang bentuknya tertutup oleh terangnya cahaya dari lampu kendaraan tersebut sehingga pada beberapa kasus terutama pada malam hari beberapa bagian background terdeteksi sebagai MC. Hal ini bisa menjadi salah satu dasar untuk evaluasi dalam melakukan anotasi data terkhusus kelas MC. Penurunan tingkat akurasi deteksi pada kondisi malam dialami pula pada penelitian sebelumnya dimana terjadi penurunan akurasi sekitar 37% dibanding pada saat siang hari [4].

Terkait apakah model lebih baik untuk dibagi menjadi 3 segmen besar intensitas waktu yang berbeda mengingat pada siang hari dan malam hari perbedaannya pada tiap objek sangat kontras, dengan pencahayaan yang kurang pada lokasi yang dituju dan kebutuhan agar objek kendaraan tersebut tetap dapat untuk terdeteksi menyebabkan kelas ini merupakan salah satu kelas yang cukup besar kemungkinannya terdeteksi sebagai background pada video uji malam hari.

Kelas LV merupakan kelas yang sangat stabil dapat terdeteksi sesuai dengan kelasnya oleh sistem, yang mana kelas ini merupakan kelas kendaraan kedua yang terbanyak teranotasi melalui proses anotasi *dataset* sebelumnya. Kelas ini sangat kecil kemungkinannya untuk mendeteksi kendaraan ringan sebagai kelas kendaraan lainnya, yang mana persentasenya hanya di kisaran 2–1% saja. Terlepas dari hal tersebut kelas LV masih sangat sering terdeteksi oleh kelas kendaraan lainnya, yakni yang terbesar ialah kelas LV(ANGKUTAN) sebesar 44% dan LV(BOX) sebesar 35% yang diikuti oleh background sebesar 12% dan terakhir sebagai MC yaitu 2% yang sangat kecil persentasenya untuk dapat dianggap sebagai 0. Merujuk pada *confusion matrix normalized* menunjukkan kelas kendaraan HV dan HV(BUS) tidak akan terdeteksi sebagai kelas LV dengan *custom model* YOLOv8 yang terbaru ini. Yang meningkatkan tingkat deteksinya terhadap kelas lainnya juga dikarenakan terdapat 2 kelas LV dengan bentuk kendaraan yang serupa namun dengan spesifikasi spesifik yakni pada LV(ANGKUTAN) terhadap perbedaan warna, dan LV(BOX) terhadap adanya barang angkutan dan box yang memiliki kapasitas angkutan yang bentuknya dapat beragam namun terpisah dengan didefinisikan sebagai kelas lain selain daripada LV. Hal ini bisa menjadikan dasar untuk evaluasi dalam bagaimana baiknya untuk menganotasi data pada kelas kendaraan LV dan kelas LV(ANGKUTAN) dan LV(BOX). Terkait perlunya untuk lebih memperbanyak kelas kendaraan untuk angkutan dan mobil box mengingat dari segi bentuk kendaraan dan ukurannya yang serupa satu sama lainnya, dengan aspek yang membedakannya hanya terletak pada perbedaan konsistensi warna dan bentuk dari barang bawaan dibelakang box yang ada pada kendaraan ringan kelas mobil box agar objek kendaraan tersebut tetap dapat untuk terdeteksi sesuai dengan kelasnya masing-masing.

Kelas HV dengan bentuk yang beragam dan *general* dan *dataset* kendaraan yang tidak sebanyak kelas MC dan LV pada saat penganotasian data sudah cukup dapat untuk terprediksi sesuai kelasnya, yakni dengan skor 65%. Kelas kendaran HV juga untuk dapat terdeteksi sebagai kelas lainnya skor tertingginya hanya sebesar 11% yakni dengan

kelas HV(BUS) 11% yang memiliki ukuran tinggi yang serupa pula, diikuti dengan kelas LV(ANGKUTAN) sebesar 5%, dan tidak jarang juga terdeteksi sebagai background dengan skor sebesar 19% saja. Kelas kendaraan lain juga dapat secara salah terdeteksi oleh kelas kendaraan ini namun hanya dengan persentase yang lebih kecil lagi yakni HV(BUS) dapat terdeteksi sebagai HV dengan persentase sebesar 11% diikuti oleh LV(BOX) dengan persentase sebesar 8% dan *background* dengan persentase sebesar 4% saja. Terkait ketiga kelas spesifik lainnya dengan itu dapat dilihat untuk menjadi dasar evaluasi dalam bagaimana baiknya untuk menganotasi data pada kelas khusus yang lebih spesifik yakni pada kelas LV(ANGKUTAN), LV(BOX), dan HV(BUS).

## 4. Simpulan

Studi ini berhasil mengembangkan sistem deteksi dan klasifikasi kendaraan berbasis YOLOv8 dengan custom dataset yang diperoleh dari rekaman CCTV Dinas Perhubungan Kota Balikpapan. Model yang dilatih mampu mengidentifikasi enam kelas kendaraan, yaitu MC, LV, HV, LV(ANGKUTAN), LV(BOX), dan HV(BUS), dengan tingkat akurasi yang konsisten tinggi. Secara spesific, Kelas umum seperti MC, LV, dan HV menunjukkan akurasi lebih 90% di berbagai kondisi waktu, dengan F1-Score yang kompetitif, khususnya pada malam hari untuk kelas tertentu. Namun, variasi intensitas cahaya, terutama efek silau dari lampu kendaraan pada malam hari, masih memengaruhi kinerja model. Kelas spesifik seperti LV(BOX) dan LV(ANGKUTAN) memiliki tingkat salah klasifikasi yang cukup tinggi karena kesamaan bentuk dengan LV umum, sehingga memerlukan peningkatan jumlah dan keragaman data latih. Lebih lanjut, hasil normalisasi *confusion matrix* menegaskan bahwa penambahan *dataset*, khususnya untuk kelas minoritas, akan meningkatkan performa sistem. Dengan demikian, sistem ini menunjukkan potensi signifikan sebagai solusi otomatisasi survei lalu lintas yang efisien, real-time, dan mengurangi ketergantungan terhadap survei manual. Temuan ini memperkuat bukti bahwa integrasi *deep learning* dan *computer vision* dapat diadaptasi secara efektif untuk konteks lalu lintas perkotaan di Indonesia, bahkan pada lingkungan dengan variabilitas pencahayaan yang tinggi.

Namun, penelitian ini terbatas pada lokasi tunggal dengan kondisi lalu lintas dan cuaca spesifik, sehingga generalisasi ke area lain masih perlu divalidasi. Selain itu, jumlah dataset untuk kelas minoritas relatif sedikit, sehingga model cenderung bias terhadap kelas mayoritas dan kurang optimal mendeteksi kendaraan dengan bentuk serupa.

Tantangan utama pengembangan sistem ini meliputi peningkatan akurasi deteksi pada kondisi malam hari, khususnya dalam mengatasi *glare* dan *shadow* yang memengaruhi segmentasi objek. Diperlukan strategi augmentasi data yang lebih adaptif serta penambahan variasi sudut pandang kamera. Optimasi *hyperparameter* dan arsitektur model juga dapat memperbaiki keseimbangan antara akurasi dan kecepatan *inferensi*. Integrasi dengan teknologi edge computing diperlukan agar pemrosesan dapat dilakukan dekat dengan sumber data untuk mengurangi latensi. Selain itu, pengujian di berbagai lokasi dengan karakteristik lalu lintas berbeda akan memperluas cakupan penerapan dan memastikan robustnes sistem dalam berbagai kondisi operasional.

Secara akademis, studi ini berkontribusi untuk memperluas literatur terkait penerapan YOLOv8 dengan custom dataset untuk klasifikasi multi-kelas kendaraan di lingkungan urban Indonesia. Temuan mengenai performa per kelas dan pengaruh kondisi pencahayaan memberikan referensi empiris penting bagi pengembangan algoritma deteksi kendaraan berbasis *deep learning* di negara berkembang. Sedangkan kontribusi praktis, sistem ini menawarkan alternatif survei lalu lintas yang hemat biaya, akurat, dan *real-time*. Implementasinya dapat membantu otoritas transportasi dalam perencanaan manajemen lalu lintas, pemantauan volume kendaraan, serta evaluasi efektivitas kebijakan transportasi secara lebih responsif.

## Daftar Pustaka

[1] 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.

[2] A. Amwin, "Deteksi Dan Klasifikasi Kendaraan Berbasis Algoritma You Only Look Once (YOLO)," 2021.

[3] M. Ghifari'Azmi, "Sistem Otomatisasi Penghitungan Kendaraan (Vehicle Counting) Berbasis Convolutional Neural Network." UIN Sunan Kalijaga Yogyakarta, 2020.

[4] L. Rahmawati and K. Adi, "Rancang bangun penghitung dan pengidentifikasi kendaraan menggunakan Multiple Object Tracking," *Youngster Phys. J.*, vol. 6, no. 1, pp. 70–75, 2017.

[5] D. Reis, J. Kupec, J. Hong, and A. Daoudi, "Real-Time Flying Object Detection with YOLOv8," *arXiv Prepr. arXiv2305.09972*, 2023.

[6] R.-Y. Ju and W. Cai, "Fracture Detection in Pediatric Wrist Trauma X-ray Images Using YOLOv8

Algorithm," *arXiv Prepr. arXiv2304.05071*, 2023.

[7] G. Jocher and A. Chaurasia, "Ultralytics YOLOv8," 2023. https://docs.ultralytics.com/

[8] M. Sarosa and N. Muna, "Implementasi Algoritma You Only Look Once (YOLO) untuk Deteksi Korban Bencana Alam," *J. Teknol. Inf. dan Ilmu Komput.*, vol. 8, no. 4, pp. 787–792, 2021.

[9] F. M. Talaat and H. ZainEldin, "An improved fire detection approach based on YOLO-v8 for smart cities," *Neural Comput. Appl.*, vol. 35, no. 28, pp. 20939–20954, Oct. 2023, doi: 10.1007/S00521-023-08809-1/FIGURES/9.

[10] L. Kang, Z. Lu, L. Meng, and Z. Gao, "YOLO-FA: Type-1 fuzzy attention based YOLO detector for vehicle detection," *Expert Syst. Appl.*, vol. 237, p. 121209, Mar. 2023, doi: 10.1016/J.ESWA.2023.121209.

[11] M. H. Hamzenejadi and H. Mohseni, "Fine-tuned YOLOv5 for real-time vehicle detection in UAV imagery: Architectural improvements and performance boost," *Expert Syst. Appl.*, vol. 231, p. 120845, Nov. 2023, doi: 10.1016/J.ESWA.2023.120845.

[12] A. Solichin, Z. Rahman, and U. B. Luhur, "Identifikasi Plat Nomor Kendaraan Berbasis Mobile dengan Metode Learning Vector Quantization," *J. TICom*, vol. 3, no. 3, 2015.

[13] S. R. Dewi, "Deep Learning Object Detection Pada Video Menggunakan Tensorflow Dan Convolutional Neural Network," 2018.

[14] M. S. Khatami, "Deteksi Kendaraan Menggunakan Algoritma You Only Look Once (Yolo) V3," 2022.

[15] F. Jalled and I. Voronkov, "Object detection using image processing," *arXiv Prepr. arXiv1611.07791*, 2016.

**Halaman ini sengaja dikosongkan**