Puspita, T., Swedia, E. R., Cahyanti, M. and Septian, M. R. D. (2025) "A Real-Time Helmet Detection System Based on YOLOv8 to Support Traffic Law Enforcement", Sebatik, 29(1).
Submitted: 2025-05-18 Accepted: 2025-05-30 Published: 2025-06-30

Sebatik Vol. 29 No. 1 June 2025
ISSN: 1410-3737(p) 2621-069X(e)

DOI:10.46984/sebatik.v29i1.2585

A Real-Time Helmet Detection System Based on YOLOv8 to Support Traffic Law Enforcement

Tiara Puspita[1)], Ericks Rachmat Swedia[2)], Margi Cahyanti[3)], dan **M Ridwan Dwi Septian[4)]**

[1,2,4]Informatika, Universitas Gunadarma
[3]Sistem Informasi, Universitas Gunadarma
[1,2,3,4] Jalan Margonda Raya No.100, Pondok Cina, Depok
E-mail : tiarapuspita@unmeka.ac.id[1)], ericks_rs@staff.gunadarma.ac.id[2)], margi@staff.gunadarma.ac.id[3)], dan ridwandwiseptian@staff.gunadarma.ac.id[4)]

ABSTRACT

Helmet use is a critical safety measure for motorcycle riders, yet non-compliance remains high in Indonesia. This study introduces a real-time helmet detection system using the YOLOv8 architecture, deployed on Android devices with the Kotlin programming language. A dataset of 1,197 digital images was collected and annotated using Roboflow Annotate, containing two classes: helmet users (True) and non-users (False). To improve model generalization, data augmentation techniques such as rotation and shear were applied. The model was trained using the pretrained yolov8n.pt weights and evaluated based on mAP and Intersection over Union (IoU). During training, the model achieved a mAP50 of 98% and a mAP50–95 of 59.6%. In testing, the mAP50 reached 98.3% and mAP50–95 reached 61%, with an average IoU of 0.73. The trained model was then converted into TensorFlow Lite format and integrated into an Android application. Real-time testing showed a detection accuracy of 93.3%. These results demonstrate that YOLOv8 is effective for mobile-based real-time helmet detection and has strong potential to support traffic law enforcement systems, especially in urban environments where manual monitoring is inefficient. The system contributes to enhancing public safety through smart technology integration.

Keywords: *Helmet, Roboflow, Traffic, , Vehicle, YOLO*

Sistem Deteksi Penggunaan Helm Secara Real-Time Berbasis YOLOv8 untuk Mendukung Penegakan Hukum Lalu Lintas

ABSTRAK

Permasalahan rendahnya tingkat kepatuhan penggunaan helm oleh pengendara sepeda motor di Indonesia mendorong perlunya inovasi teknologi untuk mendukung sistem penegakan hukum lalu lintas yang lebih efektif. Penelitian ini mengusulkan implementasi arsitektur YOLOv8 untuk mendeteksi penggunaan helm secara *real-time* pada perangkat Android menggunakan bahasa pemrograman Kotlin. Peneliti mengembangkan sistem deteksi otomatis berbasis *deep learning* dengan pendekatan *object detection* menggunakan model YOLOv8 yang telah terbukti memiliki performa unggul dalam hal akurasi dan kecepatan inferensi. *Dataset* yang digunakan terdiri dari 1.197 citra digital dengan dua kelas, yaitu pengguna helm (*True*) dan non-pengguna helm (*False*). Proses anotasi dilakukan melalui *platform Roboflow Annotate*, serta dilengkapi dengan augmentasi data seperti rotasi dan shear untuk meningkatkan keragaman *dataset*. Model dilatih menggunakan bobot awal yolov8n.pt dan dievaluasi dengan metrik mAP dan IoU. Hasil pelatihan menunjukkan nilai mAP50 sebesar 98% dan mAP50–95 sebesar 59,6%, sedangkan hasil pengujian menghasilkan mAP50 sebesar 98,3% dan mAP50–95 sebesar 61%, dengan nilai rata-rata IoU sebesar 0,73. Model yang telah dilatih dikonversi ke format *TensorFlow Lite* dan berhasil diintegrasikan ke dalam aplikasi Android. Uji coba menunjukkan tingkat akurasi deteksi sebesar 93,3%. Hasil ini membuktikan bahwa arsitektur YOLOv8 layak digunakan untuk sistem deteksi helm secara *real-time* pada *platform mobile*

Kata Kunci: Helm, Kendaraan, Lalu Lintas, Roboflow, YOLO

1. PENDAHULUAN

Tingginya angka kecelakaan lalu lintas di Indonesia menjadi perhatian serius, khususnya yang melibatkan kendaraan roda dua. Berdasarkan data yang diterbitkan oleh Databoks Katadata, tercatat sebanyak 116 ribu kasus kecelakaan lalu lintas terjadi pada tahun 2023, meningkat sebesar 6,8% dibandingkan tahun sebelumnya (Cahyani & Junaidy, 2025).

Salah satu faktor utama yang menyebabkan tingginya angka kecelakaan tersebut adalah rendahnya kesadaran masyarakat dalam mematuhi standar keselamatan, terutama dalam penggunaan helm (Nobrihas et al., 2023). Padahal, kewajiban penggunaan helm telah diatur secara eksplisit dalam Undang-Undang Nomor 22 Tahun 2009 tentang Lalu Lintas dan Angkutan Jalan, khususnya pada Pasal 291 ayat (1) dan (2), yang menetapkan sanksi pidana atau denda bagi pengendara dan penumpang sepeda motor yang tidak menggunakan helm berstandar nasional (Indonesia, 2009).

Upaya penegakan hukum yang dilakukan oleh pihak kepolisian melalui razia stasioner atau pengawasan manual masih memiliki keterbatasan dalam hal efisiensi dan skala pengawasan. Bahkan dengan bantuan kamera pengawas *Closed Circuit Television* (CCTV), proses pendeteksian pelanggaran masih membutuhkan campur tangan manusia, yang menimbulkan beban operasional tambahan dan keterbatasan dalam waktu serta tempat (Ely, 2023).

Untuk mengatasi keterbatasan tersebut, diperlukan sistem Transportasi Cerdas berbasis visi komputer yang mampu mendeteksi pelanggaran, seperti ketidakpatuhan terhadap penggunaan helm, secara otomatis dan *real-time* (Pratiwi, 2024). Penerapan teknologi *Electronic Traffic Law Enforcement* (E-TLE) oleh Polri merupakan langkah awal dalam digitalisasi penegakan hukum lalu lintas di Indonesia (Pardede et al., 2022). Hingga Mei 2024, tercatat lebih dari 1.600 kamera ETLE telah terpasang, termasuk kamera statis dan *handheld*. Meskipun demikian, teknologi yang digunakan ini masih belum mampu untuk melakukan klasifikasi spesifik terhadap objek keselamatan seperti helm. Oleh karena itu, dibutuhkan teknologi yang lebih adaptif dan presisi seperti *deep learning* berbasis deteksi objek (Pardede et al., 2022).

YOLO (You Only Look Once) merupakan arsitektur deteksi objek yang dirancang untuk melakukan deteksi dalam satu proses inferensi, menjadikannya salah satu metode yang paling efisien dalam tugas-tugas *real-time*. Versi terbarunya, YOLOv8, diperkenalkan oleh Ultralytics pada tahun 2023 dan menawarkan sejumlah peningkatan seperti penggunaan arsitektur Cross Stage Partial (CSP), dukungan ekspor ke format *TFLite* untuk perangkat *mobile*, serta peningkatan akurasi dalam berbagai skenario deteksi. YOLOv8 juga mendukung berbagai tugas visi komputer seperti segmentasi, klasifikasi, dan pelacakan, menjadikannya solusi ideal untuk aplikasi lapangan (Satya et al., 2023).

Selain arsitektur dan akurasi model, kualitas deteksi juga sangat ditentukan oleh kemampuan model dalam memprediksi lokasi objek secara presisi, yang diukur menggunakan metrik *Intersection over Union* (IoU) (Septian et al., n.d.). IoU merupakan rasio antara area tumpang tindih (overlap) dan area gabungan (union) antara *bounding box* prediksi dengan *bounding box* kebenaran dasar (*ground truth*) (Septian et al., n.d.). Nilai IoU yang tinggi menunjukkan bahwa model tidak hanya mengenali keberadaan objek, tetapi juga berhasil melokalisasi posisinya dengan tepat. Oleh karena itu, dalam penelitian ini, IoU dijadikan salah satu metrik utama selain *mean Average Precision* (mAP) untuk mengevaluasi performa sistem deteksi helm (Meidyan & Yustanti, 2024).

Berbagai studi telah membuktikan efektivitas arsitektur YOLO dalam mendeteksi penggunaan helm. (Yunyun & JIANG, 2021) mengombinasikan YOLOv4 dengan MobileNet dan mencapai nilai mAP sebesar 94,47%, sementara (Febriana, 2023) menerapkan YOLOv4 untuk deteksi pelanggaran helm secara *real-time* dan memperoleh akurasi yang sangat tinggi, yakni mAP 99,69%.

YOLOv5 juga telah banyak digunakan untuk aplikasi serupa. Penelitian oleh (Jia et al., 2021) melaporkan mAP sebesar 98,5% dalam skenario lalu lintas perkotaan. Akurasi ini meningkat menjadi 95,9% dalam penelitian (Shan et al., 2024) setelah menambahkan struktur BiFPN dan mekanisme channel attention.

Pada versi terbaru, YOLOv8 menunjukkan kinerja yang kompetitif. (Reis et al., 2023) melaporkan mAP sebesar 99,1% dalam deteksi objek terbang setelah penerapan transfer learning, sementara (Desai et al., 2024) mencatat mAP sebesar 85% untuk deteksi helm dan plat nomor kendaraan secara simultan.

Selain YOLO, pendekatan lain seperti Faster R-CNN juga telah digunakan. (Azhari & Wahyono, n.d.) menerapkan metode ini untuk mendeteksi penggunaan helm dengan tingkat akurasi hingga 87,5%. Metode transfer learning berbasis Inception V3 juga menunjukkan performa tinggi, dengan akurasi mencapai 97,24% dalam studi oleh (Mercado Reyna et al., 2023).

Meskipun berbagai penelitian telah membuktikan efektivitas YOLO dalam deteksi objek, penerapan YOLOv8 pada aplikasi *mobile* berbasis Android, khususnya untuk kasus penggunaan helm di Indonesia, masih jarang dijumpai. Integrasi model ke dalam sistem *real-time* di perangkat bergerak juga menghadirkan tantangan tersendiri, baik dari sisi konversi model, efisiensi inferensi, maupun akurasi di kondisi lapangan. Oleh karena itu, penelitian ini bertujuan untuk mengimplementasikan YOLOv8 dalam deteksi penggunaan helm secara *real-time* pada *platform* Android dengan pemrograman Kotlin, melalui konversi model ke format *TensorFlow Lite* (*TFLite*), serta mengevaluasi performanya sebagai bagian dari upaya peningkatan keselamatan lalu lintas berbasis teknologi (Pandey & Asati, 2023).

Dengan demikian, penelitian ini diharapkan dapat memberikan kontribusi dalam pengembangan sistem deteksi keselamatan lalu lintas yang efisien, adaptif, dan dapat diimplementasikan langsung pada perangkat *mobile* untuk mendukung upaya digitalisasi penegakan hukum di Indonesia.

2. RUANG LINGKUP

Penelitian ini difokuskan pada pengembangan dan implementasi sistem deteksi penggunaan helm secara *real-time* dengan memanfaatkan arsitektur YOLOv8 yang diintegrasikan ke dalam aplikasi *mobile* berbasis Android menggunakan bahasa pemrograman Kotlin. Cakupan permasalahan meliputi proses pelatihan model deteksi helm menggunakan *dataset* citra digital dua kelas (memakai helm dan tidak memakai helm), konversi model ke format *TensorFlow Lite* (*TFLite*) agar kompatibel dengan sistem Android, serta pengujian performa model dalam kondisi nyata melalui kamera perangkat *mobile*.

Adapun batasan-batasan yang ditetapkan dalam penelitian ini meliputi:

1. Data yang digunakan bersumber dari *dataset* primer hasil pengumpulan manual melalui internet dengan total 1.197 gambar
2. Hanya mencakup dua kelas objek, yaitu pengguna helm (*True*) dan non-pengguna helm (*False*)
3. Deteksi dilakukan secara *real-time* tanpa melibatkan pelacakan objek berkelanjutan (*tracking*)
4. Model hanya diimplementasikan pada perangkat Android dengan spesifikasi minimal sistem operasi Android 12 dan RAM 2 GB.
5. Penelitian ini juga tidak membahas aspek integrasi dengan sistem hukum atau ETLE secara langsung

Melalui proses pelatihan dan pengujian model, penelitian ini diharapkan menghasilkan sistem deteksi helm yang memiliki tingkat akurasi tinggi, dengan nilai *mean Average Precision* (mAP) dan Intersection over Union (IoU) yang representatif terhadap performa model. Implementasi dalam aplikasi Android juga ditargetkan mampu mencapai respons inferensi *real-time* dengan akurasi minimum di atas 90%, sehingga sistem dapat digunakan sebagai prototipe pendukung keselamatan lalu lintas berbasis teknologi *mobile*.

3. BAHAN DAN METODE

Penelitian ini menggunakan pendekatan eksperimen terapan untuk mengembangkan sistem deteksi penggunaan helm secara *real-time* berbasis YOLOv8 yang diintegrasikan ke dalam aplikasi Android menggunakan Kotlin. Tahapan dilakukan secara berurutan mulai dari pengumpulan data, anotasi, pra-pemrosesan, pelatihan model, hingga implementasi dan evaluasi performa deteksi. Berikut bagian bahan dan metode yang digunakan dalam mendukung penelitian ini adalah sebagai berikut:

3.1 Aturan Memakai Helm di Indonesia

Penggunaan helm bagi pengendara dan penumpang sepeda motor di Indonesia merupakan kewajiban hukum yang tertuang dalam Undang-Undang Republik Indonesia Nomor 22 Tahun 2009 tentang Lalu Lintas dan Angkutan Jalan. Pada Pasal 106 ayat (8) disebutkan bahwa setiap pengendara sepeda motor dan penumpangnya wajib mengenakan helm yang memenuhi standar nasional Indonesia (SNI) (Bayunegara et al., 2025). Lebih lanjut, Pasal 291 menetapkan sanksi pidana berupa kurungan maksimal satu bulan atau denda paling banyak Rp250.000,00 bagi siapa pun yang tidak mematuhi ketentuan tersebut (Aprian, 2024).

Acuan terhadap peraturan ini menjadi penting dalam konteks penelitian karena membuktikan bahwa penggunaan helm bukan sekadar praktik keselamatan individual, melainkan dari regulasi nasional yang dapat ditegakkan secara hukum. Oleh karena itu, pengembangan sistem deteksi otomatis untuk memantau penggunaan helm secara *real-time* melalui teknologi berbasis *deep learning* dapat diposisikan sebagai upaya yang mendukung penegakan hukum, serta mendukung digitalisasi sistem lalu lintas seperti *Electronic Traffic Law Enforcement* (E-TLE). Pendekatan ini tidak hanya bersifat teknis, tetapi juga memiliki implikasi sosial dan hukum dalam meningkatkan disiplin berkendara dan keselamatan publik (Bayunegara et al., 2025).

3.2 *Dataset* dan Anotasi

Dataset yang digunakan dalam penelitian ini merupakan data primer yang dikumpulkan secara manual dari berbagai sumber publik di internet. Proses pengumpulan dilakukan dengan menelusuri gambar-gambar pengendara sepeda motor dari situs terbuka dan media daring, kemudian dilakukan proses seleksi untuk memastikan keberadaan objek target yang relevan, yaitu manusia yang menggunakan atau tidak menggunakan helm. Jumlah total citra yang terkumpul sebanyak 1.197 gambar, yang diklasifikasikan ke dalam dua kelas utama, yaitu:

1. *True*: objek manusia menggunakan helm
2. *False*: objek manusia tidak menggunakan helm

Kedua kelas ini difungsikan untuk mendeteksi tingkat kepatuhan terhadap penggunaan alat keselamatan berkendara. Proses anotasi ini dilakukan secara manual menggunakan *platform Roboflow Annotate*, yang menyediakan antarmuka untuk memberi label dan menggambar *bounding box* secara presisi di sekitar objek yang diamati. Kualitas anotasi yang baik sangat krusial karena secara langsung memengaruhi performa dari pelatihan model untuk mendeteksi objek tersebut (Tzutalin, 2015; Wen et al., 2022).

Format anotasi mengikuti standar kompatibel dengan YOLOv8, yaitu dalam bentuk *file* .txt terpisah untuk setiap gambar, di mana setiap baris berisi lima elemen yang menunjukkan kelas objek, serta posisi dan ukuran *bounding box* dalam format ter-normalisasi terhadap lebar dan tinggi gambar.

1. class_id: ID numerik dari kelas objek (misal: 0 (*False*) untuk helm, 1 (*True*) untuk tanpa helm),
2. *x_center, y_center*: koordinat pusat *bounding box* yang telah dinormalisasi terhadap lebar dan tinggi gambar.
3. *width, height*: lebar dan tinggi *bounding box*. Semua nilai koordinat telah dinormalisasi terhadap

lebar dan tinggi gambar, sehingga seluruh nilai berada dalam rentang 0 hingga 1. Format anotasi yang digunakan mengikuti standar sebagai berikut pada rumus (1) dan (2) (Tzutalin, 2015; Wen et al., 2022) :

$$\text{class}_{\text{id}} = \text{x}_{\text{center}}\ \text{y}_{\text{center}}\ \text{w}\ \text{h} \quad (1)$$

dengan:

$$\text{x}_{\text{center}} = \frac{\text{x}_{\text{min}} + \text{x}_{\text{max}}}{2\text{W}}, \text{y}_{\text{center}} = \frac{\text{y}_{\text{min}} + \text{y}_{\text{max}}}{2\text{H}}, \quad \text{w} = \frac{\text{x}_{\text{min}} + \text{x}_{\text{max}}}{\text{W}}, \text{h} = \frac{\text{y}_{\text{min}} + \text{y}_{\text{max}}}{\text{H}} \quad (2)$$

Di mana W dan H masing-masing merupakan lebar dan tinggi dari gambar asli, yang digunakan sebagai acuan untuk menormalisasi nilai koordinat agar berada dalam skala 0 hingga 1, sehingga model dapat memproses gambar dengan berbagai resolusi secara konsisten.

Proses anotasi ini merupakan tahap krusial karena menentukan kualitas data pelatihan yang akan digunakan oleh model YOLOv8 untuk mengenali fitur visual objek dengan akurasi tinggi. Tidak seperti *dataset* umum seperti COCO, data ini dikustomisasi secara khusus untuk mendeteksi penggunaan helm pada pengendara sepeda motor, sehingga relevan dan kontekstual dengan kebutuhan penerapan di lapangan (Jain et al., 2022).

3.3 Pembagian dan Augmentasi Data

Setelah proses anotasi selesai dilakukan pada *dataset*, data kemudian dibagi menjadi tiga subset utama, yaitu data latih, data validasi, dan data uji. Pembagian ini dilakukan dengan tujuan untuk memisahkan proses pelatihan model dari proses evaluasi performa, guna menghindari *overfitting* dan memastikan kemampuan generalisasi model terhadap data yang belum pernah dilihat sebelumnya(Jain et al., 2022). Proporsi pembagian yang digunakan dalam penelitian ini adalah 70% untuk data latih, 20% untuk data validasi, dan 10% untuk data uji, menghasilkan masing-masing sebanyak 838, 239, dan 120 citra dari total 1.197 gambar.

Pada subset data latih, dilakukan proses augmentasi data untuk meningkatkan keberagaman visual citra dan memperluas distribusi representasi objek yang tersedia. Augmentasi merupakan salah satu teknik umum dalam pelatihan model *deep learning* untuk mengurangi risiko *overfitting*, terutama saat jumlah data relatif terbatas (Shorten et al., 2021). Augmentasi dalam penelitian ini diterapkan menggunakan *platform Roboflow Annotate*, yang secara otomatis menghasilkan variasi citra baru dari data asli berdasarkan beberapa transformasi.

Transformasi augmentasi yang digunakan dalam penelitian ini terdiri dari empat jenis utama, yang dipilih secara spesifik untuk mensimulasikan variasi kondisi visual yang umum terjadi dalam lingkungan nyata. Keempat transformasi ini tidak hanya memperkaya data pelatihan dari sisi posisi dan orientasi objek, tetapi juga bertujuan untuk meningkatkan ketahanan model terhadap perbedaan perspektif dan distribusi spasial objek dalam citra. Adapun jenis transformasi yang diterapkan meliputi:

1. Rotasi citra sebesar -15° dan +15°,
2. Shear (distorsi perspektif) sebesar ± 10° Horizontal, ± 10° Vertikal,
3. Rotasi *bounding box* -15° dan +15°,
4. Shear *bounding box* ± 10° Horizontal, ± 10° Vertikal.

Teknik-teknik ini tidak hanya mengubah tampilan global gambar, tetapi juga memperluas kemungkinan konfigurasi posisi objek dalam citra yang akan dipelajari oleh model. Transformasi *bounding box* juga dijaga agar tetap selaras dengan posisi objek yang diubah, untuk mempertahankan validitas anotasi.

Hasil dari proses augmentasi tersebut menyebabkan peningkatan jumlah data pelatihan dari 838 menjadi total 2.873 citra secara keseluruhan. Setelah augmentasi, proporsi pembagian data juga menyesuaikan, yakni menjadi 88% untuk pelatihan, 8% untuk validasi, dan 4% untuk pengujian. Dengan adanya augmentasi ini, model YOLOv8 diharapkan mampu mengenali pola visual objek dengan latar belakang, posisi, dan orientasi yang lebih bervariasi, sehingga mendukung performa deteksi pada situasi nyata yang dinamis.

Berikut adalah Tabel 1. Jenis dan Parameter Transformasi Augmentasi yang sesuai dengan proses augmentasi

Tabel 1. Jenis dan Parameter Transformasi Augmentasi

Table 1. Types and Parameters of Augmentation Transformations

No	Transformation Type	Parameter	Description
1	Image Rotation	Angle ±15°	Rotates the image clockwise or counterclockwise to add variation in object orientation.
2	Horizontal and Vertical Shear	Angle ±10°	Shears the image diagonally to simulate perspective changes.
3	Bounding Box Rotation	Follows image rotation	Rotates the bounding box position to remain aligned with object orientation after rotation.
5	Bounding Box Shear	Follows image shear	Adjusts bounding box shape to accurately follow object deformation caused by shear.

Transformasi-transformasi augmentasi yang ditampilkan pada Tabel 1 bertujuan untuk menambah variasi visual objek dalam *dataset* tanpa mengubah makna

semantiknya. Rotasi citra berguna untuk melatih model agar mampu mengenali objek dengan orientasi kepala yang berbeda, sedangkan shear diaplikasikan untuk mensimulasikan kondisi sudut pandang kamera yang miring atau perspektif yang berubah. Agar tetap akurat, *bounding box* juga mengalami transformasi yang disesuaikan, baik dalam bentuk rotasi maupun shear, agar tetap mengikuti bentuk dan posisi objek yang telah dimodifikasi. Dengan demikian, proses augmentasi tidak hanya memperkaya jumlah data, tetapi juga meningkatkan daya generalisasi model dalam menghadapi variasi objek di dunia nyata, termasuk dalam situasi *real-time* yang tidak selalu ideal atau terstandar.

3.4 Arsitektur YOLOv8

YOLOv8 (You Only Look Once *version* 8) merupakan pengembangan terbaru dari keluarga algoritma YOLO yang dirilis oleh *Ultralytics* pada tahun 2023. Arsitektur ini dikembangkan sebagai detektor satu tahap (*one-stage detector*) yang menggabungkan kecepatan inferensi tinggi dan akurasi deteksi yang kompetitif. Dalam penelitian ini, digunakan varian YOLOv8n (nano), yakni versi paling ringan dari model YOLOv8, yang dirancang khusus untuk kebutuhan komputasi terbatas seperti pada perangkat *mobile* Android (Satya et al., 2023).

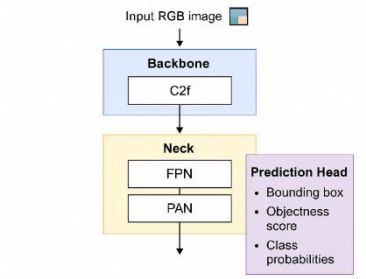
Secara umum, arsitektur YOLOv8 terdiri dari tiga bagian utama, yaitu: *Backbone, Neck*, dan *Head*. Komponen *Backbone* menggunakan struktur C2f (*Cross-Stage Partial Fusion*) untuk mengekstraksi fitur spasial dan semantik dari citra *input* dengan efisien. Bagian *Neck* mengadopsi kombinasi *Feature Pyramid Network* (FPN) dan *Path Aggregation Network* (PAN) untuk menggabungkan fitur dari berbagai skala, sehingga memperkuat kemampuan deteksi terhadap objek berukuran kecil hingga besar (Satya et al., 2023). Pada bagian Head, YOLOv8 menggunakan pendekatan anchor-free detection dan secara langsung menghasilkan output berupa *bounding box*, dan label kelas dalam satu tahap inferensi (Balla, 2024).

Untuk mengevaluasi kualitas prediksi *bounding box* yang dihasilkan model, digunakan metrik Intersection over Union (IoU). IoU mengukur tingkat kesesuaian antara *bounding box* hasil prediksi dan ground truth dengan membandingkan luas area tumpang tindih (intersection) terhadap luas gabungan area kedua *bounding box* tersebut (union). Nilai IoU dinyatakan dalam rentang 0 hingga 1, di mana nilai yang mendekati 1 menunjukkan bahwa model berhasil melokalisasi objek dengan sangat akurat (Septian et al., n.d.). Dalam konteks YOLOv8, IoU digunakan sebagai salah satu parameter evaluasi utama yang berkontribusi dalam perhitungan metrik mAP (*mean Average Precision*) . IoU mengukur tingkat tumpang tindih antara prediksi model dengan anotasi kebenaran dasar (ground truth), yang didefinisikan pada rumus (3) sebagai berikut (Septian et al., n.d.):

$$\mathrm{IoU} = \frac{\mathrm{A_{pres}} \cap \mathrm{A_{true}}}{\mathrm{A_{pres}} \cup \mathrm{A_{true}}} \quad (3)$$

Di mana A_{pres} adalah area dari *bounding box* hasil prediksi, dan A_{true} adalah area dari *bounding box* ground truth. Nilai IoU berada dalam rentang 0 hingga 1, dengan nilai yang lebih tinggi menunjukkan kualitas deteksi lokasi yang lebih akurat. IoU juga menjadi dasar dalam perhitungan metrik *mean Average Precision* (mAP) yang digunakan untuk mengevaluasi performa model secara keseluruhan (Septian et al., n.d.).

Keunggulan YOLOv8 tidak hanya terletak pada akurasi dan kecepatan, tetapi juga pada fleksibilitas integrasinya ke berbagai *platform*. Model ini mendukung ekspor ke format *TensorFlow Lite* (*TFLite*), ONNX, dan *TorchScript*, sehingga sangat ideal untuk aplikasi lintas *platform*, termasuk *deployment* ke Android. Dengan kombinasi efisiensi, akurasi deteksi, dan kemudahan integrasi, YOLOv8 menjadi pilihan arsitektur yang tepat untuk penelitian ini dalam mengembangkan sistem deteksi helm secara *real-time* pada perangkat *mobile (Satya et al., 2023)*.



Gambar 1. Arsitektur YOLOv8
Figure 1. YOLOv8 Architecture

Gambar 1 menunjukkan arsitektur umum YOLOv8 yang terdiri dari tiga komponen utama: *Backbone*, *Neck*, dan *Prediction* Head. Proses deteksi objek dimulai dari *Input* RGB *image* (gambar berwarna tiga saluran) yang masuk ke dalam model.

Bagian *backbone* bertanggung jawab untuk mengekstraksi fitur awal dari citra *input*. YOLOv8 menggunakan modul C2f (*Cross-Stage Partial Fusion*) yang menggabungkan efisiensi dari CSPNet dengan teknik fusi dua konvolusi, yang bertujuan mengurangi kompleksitas komputasi tanpa mengorbankan kualitas fitur. Hasil dari *backbone* adalah representasi fitur beresolusi lebih rendah yang kaya secara semantik (Desai et al., 2024).

Fitur yang dihasilkan dari *backbone* diteruskan ke bagian neck yang terdiri dari dua modul: FPN (*Feature Pyramid Network*) dan PAN (*Path Aggregation Network*). FPN memungkinkan kombinasi fitur dari berbagai tingkat kedalaman untuk mempertahankan detail objek kecil, sementara PAN menyempurnakan aliran informasi dari level bawah ke atas agar fitur spasial tetap terjaga. Kombinasi ini membantu YOLOv8 menangani deteksi objek dalam berbagai skala secara lebih efektif (Desai et al., 2024).

Output dari *neck* dikirim ke prediction head, yaitu bagian yang menghasilkan hasil akhir deteksi objek (Desai et al., 2024).

Ketiga komponen tersebut dievaluasi selama pelatihan model menggunakan fungsi loss yang mempertimbangkan kesalahan lokasi (IoU), klasifikasi. Hasil dari prediction head ini menjadi basis untuk menentukan posisi dan label dari objek yang terdeteksi dalam gambar *input*.

3.5 Pelatihan dan Evaluasi Model

Proses pelatihan model dilakukan menggunakan pustaka Ultralytics YOLOv8 melalui API Python, dengan model awal berupa YOLOv8n (nano) yang telah dilatih sebelumnya pada *dataset* COCO. Model ini kemudian dilakukan fine-tuning menggunakan *dataset* deteksi helm yang telah disiapkan dan dianotasi sebelumnya. Tujuan dari pelatihan ini adalah untuk menyesuaikan bobot model terhadap karakteristik visual pada domain spesifik helm.

Parameter pelatihan yang digunakan dalam eksperimen ini meliputi:

1. Jumlah *epoch*: 100
2. Ukuran *batch*: 16
3. *Learning rate* awal: 0.01
4. *Optimizer*: SGD (*Stochastic Gradient Descent*)
5. Ukuran *input*: 640 × 640 piksel
6. *Early stopping*: *patience* = 10

Pada proses pelatihan dilakukan dengan strategi *early-stopping*, di mana pelatihan akan dihentikan otomatis jika tidak terjadi peningkatan pada nilai validasi selama 10 *epoch* berturut-turut. Dalam eksperimen ini, model berhenti secara otomatis pada *epoch* ke-29 (*epoch* terbaik), yaiut dengan performa terbaik tercapai pada *epoch* ke-19.

Untuk mengevaluasi performa model deteksi objek, digunakan dua metrik utama, yaitu *mean Average Precision* (mAP) dan Intersection over Union (IoU).

IoU merupakan metrik yang mengukur sejauh mana kesesuaian prediksi model terhadap *ground truth*, yang didefinisikan pada rumus (4) sebagai berikut ini (Septian et al., n.d.):

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}} = \frac{\text{A}_{\text{pres}} \cap \text{A}_{\text{true}}}{\text{A}_{\text{pres}} \cup \text{A}_{\text{true}}} \qquad (4)$$

Nilai IoU digunakan untuk menentukan apakah suatu prediksi dianggap benar (true positive), berdasarkan ambang batas tertentu.

Berdasarkan nilai IoU, dilakukan perhitungan Average Precision (AP) untuk masing-masing kelas, dan selanjutnya dihitung *mean Average Precision* (mAP) sebagai rata-rata dari seluruh kelas yang di jelaskan pada rumus (5) (Septian et al., n.d.) :

$$\text{mAP} = \frac{1}{\text{N}} \sum_{\text{i}=1}^{\text{N}} \text{AP}_{\text{i}} \qquad (5)$$

Dalam implementasi YOLOv8, digunakan dua varian mAP, yaitu:

1. mAP50 (IoU = 0.5): hanya mengukur presisi rata-rata pada satu ambang IoU tertentu.
2. mAP50–95: rata-rata dari 10 nilai AP dengan IoU mulai dari 0.50 hingga 0.95 dengan interval 0.05, memberikan evaluasi yang lebih komprehensif.

Hasil pelatihan menunjukkan bahwa model YOLOv8 berhasil mencapai nilai mAP50 sebesar 98%, mAP50–95 sebesar 82%, dan nilai IoU rata-rata sebesar 0.73 pada data validasi. Hasil ini mengindikasikan bahwa model mampu mengenali objek helm dengan akurasi tinggi dan posisi *bounding box* yang cukup presisi.

3.6 Konversi Model dan Implementasi Android

Setelah model YOLOv8 selesai dilatih dan menghasilkan bobot akhir yang optimal, langkah selanjutnya adalah melakukan konversi model ke format *TensorFlow Lite* (*TFLite*) agar dapat diintegrasikan ke dalam aplikasi Android. Proses ini dilakukan untuk menyesuaikan model dengan keterbatasan sumber daya pada perangkat *mobile*, seperti memori, prosesor, dan kebutuhan inferensi *real-time* (Pandey & Asati, 2023).

Konversi dilakukan menggunakan fitur ekspor bawaan dari pustaka Ultralytics dengan perintah berikut: model.export(format="tflite").

Perintah ini akan menghasilkan *file* model dengan ekstensi .tflite, yang dapat langsung digunakan dalam lingkungan Android Studio dengan bantuan library *TensorFlow Lite*. Format *TFLite* memungkinkan model untuk dijalankan dengan cepat dan efisien karena ukuran *file* lebih kecil dan telah melalui proses optimasi seperti *quantization* dan *operator simplification*.

Implementasi aplikasi Android dilakukan menggunakan bahasa pemrograman Kotlin dengan antarmuka pengembangan berbasis Android Studio. Untuk pengambilan *input* citra secara *real-time*, digunakan *CameraX* API yang menyediakan akses langsung ke kamera perangkat dengan latensi rendah dan integrasi mudah. Proses inferensi dilakukan menggunakan *TensorFlow Lite Interpreter*, yang memuat model .tflite dan menerima *input* berupa citra beresolusi 640×640 piksel dalam format RGB.

Alur inferensi dalam aplikasi mencakup tahapan berikut:

1. Kamera menangkap citra secara *real-time* menggunakan *CameraX*.
2. Citra diubah ukurannya dan dipra-proses agar sesuai dengan *input* model (normalisasi piksel).
3. *TensorFlow Lite* menjalankan inferensi menggunakan model YOLOv8.tflite.
4. Hasil prediksi berupa koordinat *bounding box*, dan label kelas.
5. Aplikasi menampilkan *bounding box* dan label secara visual langsung di layar perangkat.

Untuk menjaga responsivitas aplikasi, seluruh proses inferensi dijalankan di thread terpisah (*background*

thread) agar tidak mengganggu performa *UI*. Proses implementasi ini memungkinkan pendeteksian objek helm dilakukan secara *real-time* langsung di perangkat Android tanpa memerlukan koneksi internet atau *server* eksternal.

3.7 Pengujian *Real-time* dan Visualisasi

Setelah proses konversi model ke format *TensorFlow Lite* dan integrasi ke dalam aplikasi Android selesai dilakukan, langkah selanjutnya adalah melakukan pengujian sistem secara *real-time* untuk mengevaluasi kinerja deteksi helm dalam kondisi lapangan sesungguhnya. Tujuan dari pengujian ini adalah untuk menilai kemampuan model dalam melakukan inferensi langsung dari kamera perangkat, serta menampilkan hasil deteksi secara instan pada antarmuka aplikasi.

Pengujian dilakukan menggunakan perangkat Android dengan spesifikasi minimum sistem operasi Android 12, RAM 2 GB, serta dukungan *CameraX* API. Aplikasi diuji pada berbagai kondisi lingkungan, termasuk pencahayaan terang, pencahayaan redup, latar belakang kompleks, dan posisi objek yang bervariasi. Citra ditangkap secara langsung melalui kamera, kemudian diproses oleh model yang dijalankan secara lokal menggunakan *TensorFlow Lite* Interpreter.

Hasil deteksi divisualisasikan dalam bentuk *bounding box* berwarna yang mengelilingi objek terdeteksi, disertai label kelas ("*True*" atau "*False*"). Visualisasi ini ditampilkan secara overlay pada layar perangkat Android melalui antarmuka pengguna yang dirancang sederhana dan intuitif. Seluruh proses inferensi dijalankan pada thread terpisah untuk memastikan antarmuka tetap responsif. Pengamatan selama pengujian menunjukkan bahwa, Model mampu mendeteksi objek secara akurat dengan tingkat akurasi *real-time* sebesar 93,3%, berdasarkan pengujian terhadap 30 citra uji, di mana 28 citra diklasifikasikan dengan benar dan 2 citra salah klasifikasi.

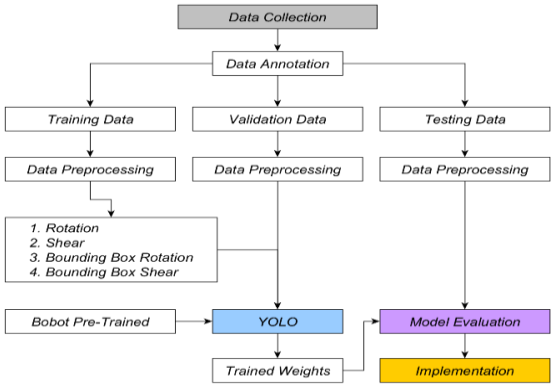
Visualisasi *real-time* ini menjadi komponen penting dalam membuktikan keandalan sistem deteksi helm secara langsung di perangkat *mobile*. Implementasi penuh tanpa koneksi ke *server* eksternal menunjukkan bahwa sistem ini bersifat standalone, efisien, dan praktis diterapkan di lapangan sebagai bagian dari solusi Transportasi Cerdas berbasis visi komputer.

3.8 Diagram Alur Penelitian

Tahapan-tahapan dalam proses penelitian ini, disusun sebuah diagram alur yang menggambarkan keseluruhan proses mulai dari pengumpulan data hingga implementasi aplikasi. Diagram ini berfungsi sebagai representasi visual dari metode yang telah dijelaskan pada subbagian sebelumnya.

Pada data latih, dilakukan augmentasi untuk menambah variasi citra melalui transformasi rotasi, shear, serta manipulasi *bounding box*. Setelah proses augmentasi, data dilanjutkan ke tahap pelatihan model dengan memanfaatkan bobot awal (pre-trained weights) dari YOLOv8. Model yang telah dilatih menghasilkan bobot terlatih (*trained model*) yang siap digunakan.

Model YOLOv8 kemudian dikonversi ke format *TensorFlow Lite* (.tflite) dan digunakan dalam tahap pengujian model. Hasil model kemudian diimplementasikan ke dalam aplikasi Android, di mana sistem melakukan deteksi helm secara *real-time* melalui kamera perangkat. Proses inferensi ini menampilkan output berupa *bounding box* dan label kelas secara langsung di layar. Seluruh tahapan tersebut divisualisasikan dalam Gambar 2 berikut.



Gambar 2. Alur Proses Penelitian
Figure 2. Research Process Flow

Diagram alur Gambar 2 memberikan gambaran menyeluruh mengenai tahapan metodologis dalam penelitian ini, mulai dari tahap persiapan data hingga implementasi sistem deteksi helm pada perangkat Android. Setiap langkah dirancang secara sistematis untuk memastikan model yang dibangun dapat beroperasi secara optimal dalam kondisi nyata. Dengan alur yang terstruktur ini, diharapkan sistem yang dihasilkan tidak hanya akurat dari sisi teknis, tetapi juga adaptif dan efisien untuk diterapkan secara langsung dalam mendukung keselamatan berkendara. Tahapan selanjutnya akan membahas hasil implementasi serta analisis performa sistem secara lebih rinci dalam pembahasan berikutnya.

4. PEMBAHASAN

Berdasarkan pengujian terhadap 30 citra uji dalam lingkungan nyata, sistem mencatat tingkat akurasi sebesar 93,3%, di mana 28 prediksi benar dan 2 prediksi salah. Kesalahan prediksi sebagian besar terjadi pada kondisi pencahayaan rendah atau ketika posisi helm tidak sepenuhnya terlihat akibat sudut kamera. Hal ini menunjukkan bahwa meskipun model cukup andal dalam situasi umum, performanya tetap dipengaruhi oleh faktor pencahayaan, orientasi objek, dan kejernihan citra.

Selain akurasi, waktu inferensi rata-rata tercatat berada pada kisaran 40 sampai dengan 329 milidetik per *frame*, yang menunjukkan bahwa sistem mampu beroperasi secara lancar dalam skenario *real-time*. Tidak ditemukan lag atau gangguan performa selama pengujian

berlangsung, berkat penggunaan *thread* terpisah dalam proses inferensi.

Secara fungsional, sistem menunjukkan kemampuan deteksi multi-objek, mampu mengenali lebih dari satu kepala pengendara dalam satu frame tanpa penurunan kinerja. Hal ini menunjukkan bahwa arsitektur YOLOv8 memiliki kapasitas deteksi spasial yang memadai dan cocok diterapkan untuk skenario dunia nyata, termasuk pemantauan lalu lintas secara langsung.

Temuan ini mendukung bahwa kombinasi antara YOLOv8, *TensorFlow Lite*, dan *platform* Android dapat menghasilkan sistem deteksi helm yang akurasi tinggi, ringan, dan portable. Namun demikian, untuk penerapan skala besar seperti integrasi ke sistem E-TLE nasional, diperlukan pengujian lebih lanjut dalam skala data yang lebih besar, beragam, serta dalam kondisi ekstrem seperti malam hari, hujan, dan latar belakang yang padat.

4.1 HASIL UJI REAL-TIME

Temuan ini mendukung bahwa kombinasi antara YOLOv8, *TensorFlow Lite*, dan *platform* Android dapat menghasilkan sistem deteksi helm yang akurasi tinggi, ringan, dan portable. Sistem ini menunjukkan potensi untuk diterapkan dalam skenario nyata, seperti pemantauan lalu lintas atau edukasi keselamatan berkendara. Namun demikian, untuk penerapan skala besar seperti integrasi ke dalam sistem E-TLE nasional, diperlukan pengujian lebih lanjut dengan cakupan data yang lebih luas, kondisi lingkungan yang bervariasi, serta validasi performa secara berkelanjutan guna memastikan keandalan sistem dalam berbagai situasi di lapangan.

Berdasarkan hasil pengujian terhadap 30 sampel citra uji, diperoleh hasil sebagai berikut, 28 data diklasifikasikan dengan benar dan 2 data diklasifikasikan secara keliru.

yang menghasilkan nilai akurasi *real-time* sebesar 93,3%. Perhitungan akurasi dilakukan menggunakan rumus (6) (Febriana, 2023):

$$\begin{aligned} Accuracy &= \frac{\text{Number of Correct Predictions}}{\text{PTotal Predictions}} \\ &= \frac{28}{30} \\ &= 0{,}933 = 93{,}3\% \end{aligned} \quad (6)$$

Hasil prediksi yang dikumpulkan menunjukkan bahwa model dapat secara konsisten mendeteksi objek helm dan non-helm pada citra uji yang ditangkap secara langsung. Visualisasi hasil deteksi ditampilkan melalui *bounding box* berwarna dengan label kelas ("*True*" atau "*False*") terhadap prediksinya.

Dalam uji coba ini, waktu inferensi rata-rata tercatat sebesar 111.4 milidetik per *frame*, tergantung pada kompleksitas citra, latar belakang, dan pencahayaan. Dengan waktu pemrosesan tersebut, sistem mampu melakukan deteksi dan pelabelan objek secara lancar tanpa menimbulkan *lag* atau gangguan performa antarmuka pengguna. Inferensi dilakukan pada thread terpisah, memastikan bahwa respons antarmuka tetap stabil meskipun proses deteksi berjalan secara terus-menerus.

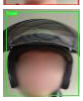
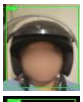
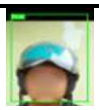
Secara keseluruhan, hasil uji real-time menunjukkan bahwa sistem deteksi helm yang dikembangkan mampu beroperasi secara efektif dan efisien dalam kondisi lapangan nyata. Kemampuan model dalam mengenali objek helm secara langsung dari kamera perangkat Android, serta menampilkan hasil deteksi secara instan melalui bounding box dan label kelas, membuktikan bahwa YOLOv8 dapat dioptimalkan untuk implementasi di perangkat mobile tanpa mengorbankan akurasi deteksi maupun kecepatan inferensi.

Pengujian dilakukan terhadap 30 sampel citra uji yang merepresentasikan variasi kondisi visual, termasuk pencahayaan dan sudut pandang. Dari hasil pengujian tersebut, sistem mencatatkan 28 prediksi benar dan 2 prediksi salah, menghasilkan tingkat akurasi sebesar 93,3%. Nilai ini menunjukkan tingkat keandalan yang cukup tinggi dalam konteks deteksi objek real-time di lingkungan terbatas sumber daya.

Untuk memberikan gambaran lebih rinci, Tabel 2 menyajikan distribusi hasil klasifikasi model terhadap data uji yang digunakan.

Tabel 2. Hasil Pengujian
Table 2. Test Results

No	Input	Prediction Results	Result	Images	Time (ms)
1	Not wearing a helmet	Not wearing a helmet	Yes		95
2	Wearing a helmet	Wearing a helmet	Yes		94
3	Wearing a helmet	Wearing a helmet	Yes		98
4	Not wearing a helmet	Not wearing a helmet	Yes		98
5	Not wearing a helmet	Not wearing a helmet	Yes		93
6	Wearing a helmet	Wearing a helmet	Yes		99
7	Wearing a helmet	Wearing a helmet	Yes		96
8	Not wearing a helmet	Not wearing a helmet	Yes		88
9	Not wearing a helmet	Not wearing a helmet	Yes		175
10	Wearing a helmet	Wearing a helmet	Yes		169

No	Input	Prediction Results	Result	Images	Time (ms)
11	Wearing a helmet	Wearing a helmet	Yes		44
12	Not wearing a helmet	Not wearing a helmet	Yes		47
13	Not wearing a helmet	Not wearing a helmet	Yes		102
14	Wearing a helmet	Wearing a helmet	Yes		41
15	Not wearing a helmet	Not wearing a helmet	Yes		94
16	Wearing a helmet	Wearing a helmet	Yes		104
17	Not wearing a helmet	Not wearing a helmet	Yes		50
18	Wearing a helmet	Wearing a helmet	Yes		329
19	Wearing a helmet	Wearing a helmet	Yes		135
20	Not wearing a helmet	Not wearing a helmet	Yes		187
21	Not wearing a helmet	Not wearing a helmet	Yes		110
22	Wearing a helmet	Wearing a helmet	Yes		128
23	Not wearing a helmet	Not wearing a helmet	Yes		68
24	Wearing a helmet	Wearing a helmet	Yes		58
25	Not wearing a helmet	Wearing a helmet	No		53
26	Not wearing a helmet	Not wearing a helmet	Yes		229
27	Wearing a helmet	Wearing a helmet	No		228
28	Not wearing a helmet	Wearing a helmet	Yes		130
29	Wearing a helmet	Wearing a helmet	Yes		40
30	Not wearing a helmet	Not wearing a helmet	Yes		60

5. KESIMPULAN

Penelitian ini berhasil mengembangkan dan mengimplementasikan sistem deteksi penggunaan helm secara real-time berbasis Android menggunakan arsitektur YOLOv8. Model dilatih menggunakan 1.197 citra digital dengan dua kelas utama, yaitu pengguna helm dan non-pengguna helm. Hasil pelatihan menunjukkan bahwa model mencapai performa yang sangat baik dengan nilai mAP50 sebesar 0,98 dan mAP50-95 sebesar 0,596. Pengujian lebih lanjut pada *dataset* uji menghasilkan mAP50 sebesar 0,983 dan mAP50-95 sebesar 0,61, serta nilai rata-rata IoU sebesar 0,73. Pada pengujian *real-time* melalui aplikasi Android, sistem mampu mencapai akurasi deteksi sebesar 93,3%. Meskipun demikian, ditemukan indikasi *overfitting* pada model, khususnya terhadap objek dengan latar belakang polos. Selain itu, model masih mengalami kesalahan klasifikasi terhadap objek dengan aksesori kepala lain, seperti topi, yang diklasifikasikan sebagai helm. Hal ini menunjukkan bahwa meskipun performa model tinggi, kemampuan generalisasi terhadap data yang lebih kompleks masih perlu ditingkatkan.

6. SARAN

Variasi kelas yang terbatas, yakni hanya terdiri dari kelas helm (*true*) dan no-helm (*false*), menyebabkan model belum sepenuhnya mampu membedakan berbagai jenis aksesori kepala secara akurat. Kondisi ini mengindikasikan bahwa model masih kesulitan dalam menggeneralisasi fitur visual, terutama ketika dihadapkan pada objek yang menyerupai helm namun sebenarnya bukan bagian dari kelas yang ditargetkan. Untuk meningkatkan performa model, disarankan untuk memperluas cakupan kelas pelatihan, khususnya dengan menambahkan kategori aksesori kepala lain seperti topi, kerudung, atau helm non-standar, serta meningkatkan keragaman latar belakang citra. Penambahan variasi ini akan memberikan model lebih banyak contoh untuk mempelajari perbedaan pola visual yang mirip, sehingga kemampuannya dalam melakukan klasifikasi menjadi lebih baik. Perbaikan ini juga diharapkan dapat mengurangi risiko *overfitting* terhadap pola-pola tertentu dalam data pelatihan, sekaligus meningkatkan kemampuan generalisasi model terhadap data baru. Dengan demikian, potensi kesalahan deteksi pada aplikasi real-time dapat diminimalkan, dan sistem menjadi lebih andal dalam menghadapi kondisi lingkungan yang beragam.

7. REFERENSI

Aprian, A. T. (2024). *Penerapan Pasal 291 Ayat (1) Dan Ayat (2) Undang–Undang Nomor 22 Tahun 2009 Tentang Lalu Lintas Dan Angkutan Jalan Terhadap Bukti Pelanggaran (E–Tilang)(Studi Kasus Surat E–Tilang Nomor B/487129/Xi/Yan. 1.2/2023)*. Fakultas Hukum.

Azhari, A. N., & Wahyono, W. (N.D.). Automatic Detection Of Helmets On Motorcyclists Using Faster-Rcnn. *Ijccs (Indonesian Journal Of Computing And Cybernetics Systems)*, *16*(4), 357–366.

Balla, F. (2024). *Less Is More-Adapting The Yolov8 Network For Multi-Spectral Human Presence Detection*. Oslo Metropolitan University.

Bayunegara, D., Anggraeni, Y. M., Fitriani, E., Setiadi, W., & Triadi, I. (2025). Analisis Yuridis Pelaksanaan Sanksi Denda Tilang Electronic Traffic Lawenforcement (E-Tle) Terhadap Pelaku Pelanggar Lalu Lintas Menurutundang-Undang Nomor 22 Tahun 2009. *Quantum Juris: Jurnal Hukum Modern*, *7*(1).

Cahyani, A. N., & Junaidy, A. B. (2025). Larangan Bermain Smartphone Saat Berkendara Berdasarkan Prespektif Sad-Dhariah. *Court Review: Jurnal Penelitian Hukum (E-Issn: 2776-1916)*, *5*(02), 1–16.

Desai, S., Das, J., Langde, P., & Umate, L. (2024). Helmet And Number Plate Detection Using Yolov8. *2024 Ieee 3rd World Conference On Applied Intelligence And Computing (Aic)*, 1228–1234.

Ely, I. (2023). *Efektivitas Fungsi Kepolisian Dalam Penegakan Hukum Tindak Pidana Kecelakaan Lalu Lintas*. Undaris.

Febriana, N. H. (2023). *Analisis Deteksi Helm Pada Pengendara Bermotor Untuk Mendeteksi Pelanggaran Lalu Lintas Menggunakan Metode You Only Look Once (Yolov4)*. Universitas Pembangunan Nasional" Veteran" Jawa Timur.

Indonesia, R. (2009). *Undang-Undang Republik Indonesia Nomor 22 Tahun 2009 Tentang Lalu Lintas Dan Angkutan Jalan*.

Jain, S., Dash, S., Deorari, R., & Others. (2022). Object Detection Using Coco Dataset. *2022 International Conference On Cyber Resilience (Iccr)*, 1–4.

Jia, W., Xu, S., Liang, Z., Zhao, Y., Min, H., Li, S., & Yu, Y. (2021). Real-Time Automatic Helmet Detection Of Motorcyclists In Urban Traffic Using Improved Yolov5 Detector. *Iet Image Processing*, *15*(14), 3623–3637.

Meidyan, M. A., & Yustanti, W. (2024). Implementasi Metode You Only Look Once (Yolov5) Dalam Deteksi Pelanggaran Helm. *Journal Of Emerging Information System And Business Intelligence (Jeisbi)*, *5*(3), 214–222.

Mercado Reyna, J., Luna-Garcia, H., Espino-Salinas, C. H., Celaya-Padilla, J. M., Gamboa-Rosales, H., Galván-Tejada, J. I., Galván-Tejada, C. E., Sol\'\Is Robles, R., Rondon, D., & Villalba-Condori, K. O. (2023). Detection Of Helmet Use In Motorcycle Drivers Using Convolutional Neural Network. *Applied Sciences*, *13*(10), 5882.

Nobrihas, R. Y. T., Mahoklory, S. S., Mbanga, E., & Sellan, R. N. (2023). Kepatuhan Penggunaan Helm Saat Berkendara Dalam Mencegah Cedera Kepala Pada Pengendara Remaja. *Lasalle Health Journal*, *2*(2), 147–157.

Pandey, J., & Asati, A. R. (2023). Lightweight Convolutional Neural Network Architecture Implementation Using Tensorflow Lite. *International Journal Of Information Technology*, *15*(5), 2489–2498.

Pardede, C. R. V., Nita, S., & Setyabudi, C. M. (2022). Analisis Program Electronic Traffic Law Enforcement (Etle) Dalam Rangka Menciptakan Kamseltibcarlantas (Studi Kasus Kota Serang). *Journal Of Innovation Research And Knowledge*, *1*(8), 533–542.

Pratiwi, H. (2024). *Buku Ajar Kecerdasan Buatan: Disertai Praktik Baik Pemanfaatannya*. Asadel Liamsindo Teknologi.

Reis, D., Kupec, J., Hong, J., & Daoudi, A. (2023). Real-Time Flying Object Detection With Yolov8. *Arxiv Preprint Arxiv:2305.09972*.

Satya, L., Septian, M. R. D., Sarjono, M. W., Cahyanti, M., & Swedia, E. R. (2023). Sistem Pendeteksi Plat Nomor Polisi Kendaraan Dengan Arsitektur Yolov8. *Sebatik*, *27*(2), 753–761.

Septian, M. R. D., Masitoh, A. H., & Sari, I. M. (N.D.). Implementation Of Deep Learning Algorithm For Vehicle Count Monitoring System. *Tepian*, *5*(4), 587833.

Shan, D., Yang, Z., Wang, X., Meng, X., & Zhang, G. (2024). An Aerial Image Detection Algorithm Based On Improved Yolov5. *Sensors*, *24*(8), 2619.

Shorten, C., Khoshgoftaar, T. M., & Furht, B. (2021). Deep Learning Applications For Covid-19. *Journal Of Big Data*, *8*(1), 1–54.

Tzutalin. (2015). *Labelimg: Label Object Bounding Boxes In Images*.

Wen, H., Du, Z., Zhang, Y., & Li, Z. (2022). Annotation Efficiency And Its Impact On Deep Learning Object Detection Performance. *Sensors*, *22*(5), 1801. Https://Doi.Org/10.3390/S22051801

Yunyun, L. I. U., & Jiang, W. (2021). Detection Of Wearing Safety Helmet For Workers Based On Yolov4. *2021 International Conference On Computer Engineering And Artificial Intelligence (Icceai)*, 83–87.