ANALISIS KINERJA YOLO UNTUK DETEKSI MANUSIA MENGGUNAKAN OBJECT TRACKING PADA SISTEM KEAMANAN GARASI BERBASIS MONITORING VIDEO

Davin Bramasta, Afu Ichsan Pradana, Nibras Faiq Muhammad
Teknik Informatika, Universitas Duta Bangsa
Jalan Bhayangkara No 55-57 Tipes, Serengan, Surakarta, Jawa Tengah, Indonesia
davinbramasta2606@gmail.com

ABSTRAK

Keamanan garasi sebagai area privat sangat krusial untuk mencegah tindak kriminalitas. Namun, sistem pemantauan konvensional saat ini masih bersifat pasif, dan menemukan keseimbangan algoritma deteksi objek yang cepat sekaligus akurat untuk sistem cerdas masih menjadi tantangan utama. Penelitian ini bertujuan untuk menganalisis dan membandingkan kinerja model YOLOv8s, YOLOv11s, dan YOLOv26s yang diintegrasikan dengan *object tracking* berbasis garis *Region of Interest* (ROI) guna mendeteksi intrusi manusia secara efisien. Metode yang digunakan adalah eksperimen kuantitatif melalui pelatihan ketiga model menggunakan dataset citra publik pada platform Google Colab ber-GPU T4. Evaluasi kinerja diukur secara komprehensif berdasarkan parameter *Precision*, *Recall*, *mAP*, *FPS*, dan *Latency*. Hasil penelitian menunjukkan bahwa YOLOv8s merupakan model paling optimal untuk *monitoring real-time* dengan kecepatan 54,48 FPS dan latensi 18,35 ms. Di sisi lain, YOLOv26s memberikan tingkat presisi deteksi tertinggi sebesar 0,8710 meskipun mencatatkan kecepatan proses yang lebih rendah, yaitu 14,29 FPS.

Kata kunci : *YOLO, Deteksi Manusia, Sistem Keamanan, Garasi, Monitoring Video*

1. PENDAHULUAN

Keamanan lingkungan perumahan, khususnya pada area garasi, merupakan aspek krusial mengingat garasi sering menjadi lokasi penyimpanan aset berharga seperti kendaraan bermotor. Sistem keamanan konvensional saat ini umumnya masih mengandalkan rekaman *Closed-Circuit Television* (CCTV) yang bersifat pasif. Keterbatasan utama dari sistem pemantauan pasif ini adalah sangat bergantung pada pengawasan manusia secara manual dan terus-menerus, sehingga fungsinya seringkali hanya sebatas menjadi alat bukti rekaman pasca-kejadian kriminal [1].

Selain itu, kondisi pencahayaan yang dinamis dan bervariasi pada siang maupun malam hari seringkali menurunkan efektivitas pengawasan visual. Permasalahan utama muncul ketika dibutuhkan sebuah sistem keamanan cerdas yang tidak hanya mampu merekam, tetapi juga mendeteksi keberadaan intrusi manusia secara otomatis secara *real-time*. Perkembangan teknologi *Computer Vision* dan *Deep Learning* sebenarnya telah memberikan solusi signifikan dalam otomatisasi sistem keamanan melalui pendekatan deteksi objek (*object detection*) [2].

Namun, dalam skenario keamanan garasi yang kompleks, sekadar mendeteksi objek manusia tidaklah cukup. Sistem berisiko menghasilkan peringatan palsu (*false alarm*) tinggi jika tidak bisa melacak manuver manusia atau menangani masalah oklusi. Selain itu, menemukan algoritma arsitektur yang memberikan keseimbangan terbaik antara kecepatan inferensi (*Frames Per Second*) dan ketepatan (*mean Average Precision*) masih menjadi tantangan yang memerlukan evaluasi lebih lanjut.

Sebagai rencana penyelesaian masalah tersebut, sistem perlu diintegrasikan menggunakan algoritma *state-of-the-art* seperti *You Only Look Once* (YOLO) yang digabungkan dengan algoritma pelacakan objek (*Object Tracking*) dan *Region of Interest* (ROI). Integrasi deteksi berbasis YOLO dengan metode *tracking* terutama penggunaan garis batas virtual (ROI) telah terbukti mampu meningkatkan keandalan sistem pemantauan. Hal ini sejalan dengan penelitian oleh Adri Surya Kusuma yang menyatakan bahwa performa model yang digunakan dapat mencapai 90,85% jika digabungkan dengan ROI [3].

Oleh karena itu, penelitian ini akan menyelesaikan kesenjangan tersebut dengan merancang sistem menggunakan tiga iterasi YOLO, yakni YOLOv8s, YOLOv11s, dan YOLOv26s, yang dikombinasikan dengan metode *object tracking* berbasis *line crossing* pada area garasi, memanfaatkan dataset citra publik berskala besar dari Kaggle untuk pelatihannya. Berdasarkan permasalahan dan rencana penyelesaian di atas, penelitian ini bertujuan untuk menganalisis dan membandingkan efektivitas kinerja model YOLOv8s, YOLOv11s, dan YOLOv26s yang diintegrasikan dengan *object tracking* dalam mendeteksi penyusup pada sistem keamanan garasi. Hasil analisis komparatif ini diharapkan dapat memberikan evaluasi objektif terkait tingkat akurasi dan efisiensi komputasi dari masing-masing arsitektur, serta menjadi landasan kuat bagi pengembangan sistem peringatan dini yang responsif dan minim peringatan palsu.

2. TINJAUAN PUSTAKA

2.1. Object Tracking pada Pemantauan Video

Object tracking merupakan proses estimasi lintasan target secara berkesinambungan di sepanjang bingkai video yang mampu menangkap pergerakan objek dari berbagai macam arah termasuk kecepatan dan perubahan arah [4].

Dalam konteks sistem keamanan garasi, integrasi antara *object detection* dan *object tracking* memungkinkan sistem tidak hanya mendeteksi keberadaan manusia, tetapi juga mempertahankan identitas unik target (*ID assignment*) meskipun objek tersebut bergerak, berubah postur, atau mengalami oklusi (terhalang oleh kendaraan).

2.2. Region of Interest dan Deteksi Intrusi

Dalam pengolahan citra digital untuk sistem keamanan, *Region of Interest* (ROI) merujuk pada area spesifik di dalam bingkai video yang menjadi fokus utama pemantauan supaya mendapatkan hasil ekstraksi lebih efektif [5].

Pada aplikasi keamanan garasi, ROI sering diimplementasikan dalam bentuk garis batas virtual (*virtual line crossing*). Logika intrusi bekerja dengan menganalisis lintasan dari *object tracking*; jika titik pusat (*centroid*) atau *bounding box* dari target manusia terdeteksi memotong garis virtual tersebut, sistem akan mengklasifikasikannya sebagai penyusup. Pendekatan ini sangat efektif untuk menyaring aktivitas normal di luar area sensitif dan menekan angka peringatan palsu [6].

2.3. Perkembangan Arsitektur YOLO

YOLO (*You Only Look Once*) adalah algoritma *state-of-the-art* berbasis *Convolutional Neural Network* (CNN) yang memproses citra secara *end-to-end* dalam satu tahap inferensi. Evolusi arsitektur YOLO terus dikembangkan untuk mencapai keseimbangan antara akurasi dan kecepatan.

YOLOv8s hadir dengan pendekatan *anchorfree* dan optimasi *backbone* yang secara signifikan meningkatkan kecepatan inferensi (*Frames Per Second*) dan akurasi (mAP). YOLOv11s merupakan generasi lanjutan yang berfokus pada efisiensi jumlah parameter dan *Floating Point Operations Per Second* (FLOPs), menjadikannya sangat ringan untuk komputasi terbatas. Sementara itu, YOLOv26s dirancang dengan kemampuan ekstraksi fitur yang lebih sensitif terhadap detail visual kompleks, yang secara teoritis berpotensi menghasilkan tingkat presisi tinggi pada kondisi lingkungan yang dinamis [7].

2.4. Google Colaboratory

Google Colaboratory digunakan untuk melatih tiap versi model YOLO menggunakan dataset public dari Kaggle dengan ketentuan tertentu seperti ukuran citra 640x640, epoch 25, dan batch 16. Penggunaan GPU T4 yang ada disediakan pada Google Colaboratory dalam pelatihan model YOLO memungkinkan proses komputasi yang jauh lebih cepat dibandingkan CPU konvensional, terutama saat memproses dataset citra berskala besar untuk ekstraksi fitur yang kompleks [8].

2.5. Metrik Evaluasi Kinerja

F1-score dihitung sebagai rata-rata harmonik dari precision dan recall dengan persamaan sebagai berikut:

$$F1 = 2 \times \frac{precision \times recall}{precision + recall} \quad (1)$$

Nilai F1-score berada pada rentang 0 hingga 1, di mana nilai yang semakin mendekati 1 menunjukkan performa model yang semakin baik [9].

Untuk membandingkan kinerja dari ketiga model YOLO secara objektif, evaluasi dilakukan menggunakan metrik standar visi komputer, yang meliputi: *Precision*, *Recall*, dan *F1-Score* untuk mengukur ketepatan dan keseimbangan prediksi kemudian *Mean Average Precision* (mAP@50) sebagai metrik utama akurasi deteksi, dan kecepatan inferensi (*Latency* dan FPS) yang krusial untuk pemantauan langsung serta kompleksitas komputasi (Jumlah Parameter dan GFLOPS) untuk mengukur beban komputasi model [10], [11].

2.6. Penggunaan Dataset

Kualitas dan keragaman dataset merupakan faktor fundamental dalam pelatihan model *Deep Learning*. Dataset yang terlalu homogen dapat menyebabkan *overfitting* dan data yang diolah menjadi tidak akurat hal ini sejalan dengan penelitian oleh [11] yaitu jumlah dan kualitas data dapat berpengaruh terhadap performa model. Oleh karena itu, penelitian ini memanfaatkan dataset publik yang dikumpulkan dari platform Kaggle berupa dataset *person*. Penggunaan dataset publik dari Kaggle memberikan variasi kondisi visual yang lebih representatif seperti perbedaan intensitas cahaya, variasi sudut pandang, dan latar belakang garasi yang beragam guna meningkatkan kemampuan generalisasi model saat diimplementasikan pada kondisi nyata.

2.7. Penelitian Terdahulu

Beberapa penelitian sebelumnya telah mengevaluasi kinerja YOLO dalam berbagai kasus *computer vision*. Penelitian yang dilakukan oleh Muhammad Sobirin dan Panji Wijonarko [12]

Penelitian menggunakan YOLOv12 untuk deteksi kerusakan pakaian bekas mencapai nilai mAP50 sebesar 0,426 [13].

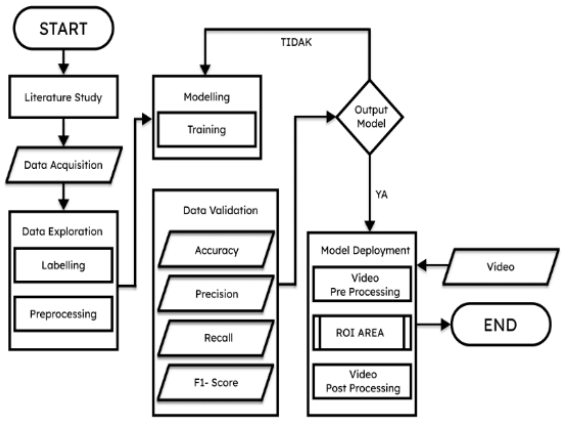
Studi lain mengimplementasikan YOLOv11-Pose untuk pengenalan gerakan sandi semafor dengan nilai mAP@0.5 mencapai 0,984 [14].

Selain itu, penerapan YOLOv8 pada deteksi penyakit daun mangga menunjukkan performa sangat tinggi dengan mAP50 mencapai 0,99 [9]. Hasil ini membuktikan bahwa peforma YOLO dalam *computer vision* sangat andal dalam berbagai bidang. Meskipun algoritma YOLO telah terbukti andal dalam berbagai klasifikasi citra statis, studi komparatif yang membandingkan secara langsung efisiensi komputasi

dan akurasi antara YOLOv8s, YOLOv11s, dan YOLOv26s ketika diintegrasikan dengan *object tracking* pada lingkungan pengawasan garasi masih belum banyak dieksplorasi.

3. METODE PENELITIAN

Dalam membuat metode, terdapat serangkaian kerangka kerja yang diikuti untuk memastikan alur penelitian tersampaikan dengan baik dan benar secara keseluruhan. Dalam penelitian ini flowchart pada Gambar 1 digunakan sebagai representasi visual dari tahapan metode yang menggambarkan penerapan model algoritma YOLO untuk mengembangkan sistem pemantauan garasi menggunakan YOLO dengan tambahan garis ROI.



Gambar 1. Tahapan pengembangan sistem deteksi manusia menggunakan YOLO

Pada gambar 1 dijelaskan bahwa tahapan pengembangan sistem dimulai dari proses *literature study* kemudian dilanjutkan dengan data *acquisition* setelah itu berlanjut ke proses *data exploration* yang mencakup proses *labelling* dan *preprocessing*. Data yang sudah diolah kemudian akan di *modelling* di latih sesuai dengan label yang dibutuhkan dengan evaluasi *accuracy, precision, recall, F1-score* yang akan menghasilkan output model yang nantinya dapat diproses lebih lanjut untuk menentukan ROI dan mendapatkan hasil dari proses pelatihan model.

Dataset yang digunakan dalam penelitian ini bersumber dari platform publik Kaggle, yang secara spesifik memuat citra objek manusia dalam berbagai variasi kondisi lingkungan, pencahayaan, dan postur yang relevan dengan skenario pengawasan keamanan. Dataset tersebut dipersiapkan dan dibagi menjadi tiga subset utama, yaitu data pelatihan, data validasi, dan data pengujian.

3.1. Pengumpulan Dataset

Data citra yang digunakan dalam penelitian ini difokuskan pada pengenalan objek manusia. Dataset sekunder diunduh dari repositori publik Kaggle yang telah memiliki anotasi *bounding box* format YOLO dengan jumlah 1.215 gambar dan 1 *class* yaitu *person*. Untuk memastikan model memiliki kemampuan generalisasi yang baik dan terhindar dari kondisi *overfitting*, keseluruhan dataset dibagi menjadi tiga proporsi utama: pelatihan data untuk membangun bobot model, data validasi untuk mengukur performa selama proses pelatihan berlangsung, dan pengujian data berupa video garasi untuk pengujian implementasi akhir, dengan membagi menjadi 3 bagian yaitu *training* 75%, *validation* 20%, dan *testing* 5%. Seluruh citra distandarisasi ke dalam dimensi resolusi 640x640 piksel sebelum diproses.

3.2. Tahapan Pelatihan Data

Pada tahapan pertama yaitu pelatihan data model dilatih menggunakan data latih dengan parameter yang telah ditentukan (seperti jumlah *epoch*, *batch size*, dan *learning rate*). Pada tahap ini, model melakukan proses ekstraksi fitur secara iteratif untuk mengenali pola visual dari objek manusia.

3.3. Tahapan Validasi Data

Dilakukan secara bersamaan selama proses pelatihan. Model mengevaluasi data validasi yang tidak pernah dilihat sebelumnya untuk memastikan bahwa metrik akurasi (*Precision, Recall, mAP*) mengalami peningkatan yang stabil.

3.4. Tahapan Pengujian Data

Tahap akhir di mana model yang telah selesai dilatih akan menghasilkan file bobot yaitu best.pt diuji menggunakan dataset video. Pada tahap ini, metrik kinerja spesifik seperti *Frames Per Second* (FPS) dan latensi inferensi diukur secara langsung.

3.5. Skenario Implementasi Pagar Virtual

Untuk mengaplikasikan model YOLO ke dalam sistem keamanan garasi, penelitian ini merancang skenario pengawasan berbasis Object Tracking dan Pagar Virtual. Pagar virtual dibangun dengan mendefinisikan ROI dalam bentuk poligon menggunakan koordinat dua dimensi yang menutupi area akses masuk garasi. Saat algoritma YOLO mendeteksi dan melacak objek manusia, sistem mengekstrak nilai titik tengah dari bounding box penyusup tersebut. Evaluasi spasial kemudian dilakukan menggunakan fungsi kalkulasi geometri; apabila centroid objek melewati atau berada di dalam area poligon ROI, maka sistem secara otomatis menandainya sebagai ancaman penyusupan yang ditandai dengan perubahan warna indikator visual menjadi merah dan mencatat ID pelacakan objek tersebut.

4. HASIL DAN PEMBAHASAN

Dataset yang menjadi sumber data primer dalam penelitian ini diambil dari repositori publik Kaggle, yang secara khusus dikurasi untuk tugas deteksi manusia. Dataset ini terdiri dari 1.215 citra yang sudah dilengkapi dengan pelabelan dengan klasifikasi objek berfokus pada satu kelas objek tunggal yaitu *person* serta pembagian data untuk menghindari overfitting

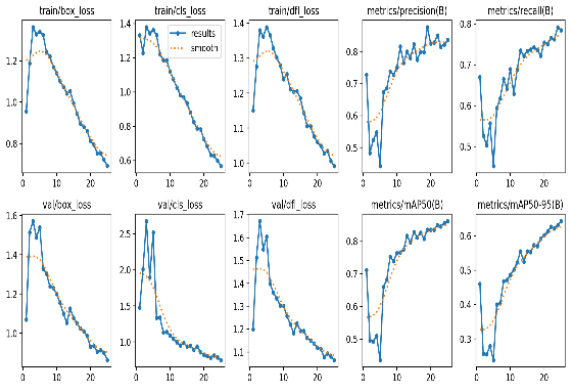
dilakukan menjadi 3 bagian 1.105 citra sebagai pelatihan, 70 citra untuk validasi, dan 40 citra untuk data uji.

4.1. Skenario Pengujian

Pengujian dilakukan dengan membandingkan ketiga varian model yakni YOLOv8s, YOLOv11s, YOLOv26s dilatih dengan konfigurasi parameter yang identik. Penggunaan batch size dan jumlah epoch disamakan untuk memastikan bahwa perbedaan performa yang dihasilkan murni merupakan representasi dari keunggulan tiap varian masing-masing.

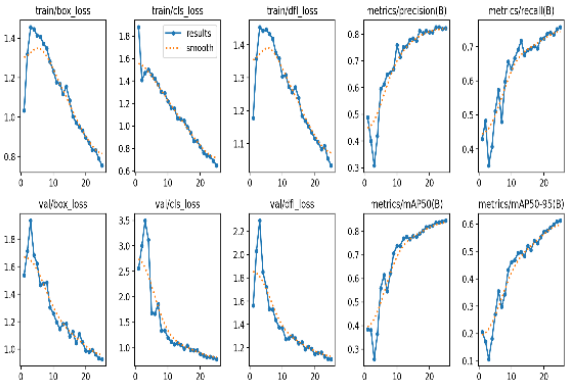
4.2. Analisis Proses Pelatihan dan Penurunan Grafik Losss

Sebelum membandingkan metrik akurasi akhir, evaluasi terhadap riwayat proses pelatihan sangat penting untuk dilakukan. Hal ini bertujuan untuk memastikan bahwa setiap arsitektur model telah mencapai titik konvergensi yang optimal dan tidak mengalami *overfitting* selama mempelajari dataset manusia di area garasi. Berikut adalah pemaparan hasil pelatihan untuk ketiga versi YOLO yang dievaluasi.



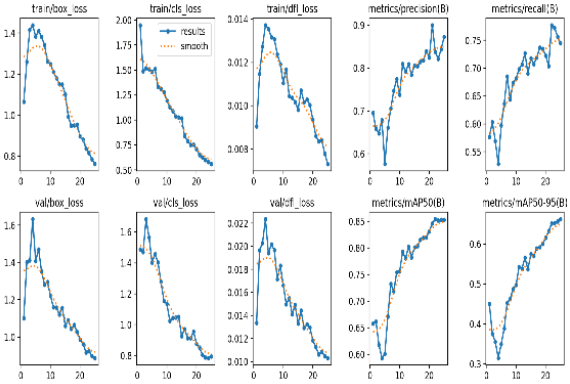
Gambar 2. Kurva peforma pelatihan dan penurunan *loss* yolov8s

Berdasarkan Gambar 2, dapat diamati pergerakan kurva pelatihan dari model YOLOv8s selama 25 *epoch*. Kurva *Box Loss* dan *Class Loss* pada set pelatihan maupun validasi menunjukkan tren penurunan yang sangat konsisten sejak *epoch* awal. Penurunan nilai *loss* yang terjadi secara simultan dan berdekatan antara data latih dan data validasi ini membuktikan bahwa model YOLOv8s mampu mempelajari ekstraksi fitur visual manusia dengan sangat baik tanpa mengalami *overfitting*. Sebagai hasilnya, kurva metrik evaluasi seperti rata-rata presisi (*mAP@50*) langsung menanjak tajam dan mulai mencapai titik konvergensi (kestabilan) yang solid pada kisaran *epoch* ke-15 hingga ke-25.



Gambar 3. Kurva peforma pelatihan dan penurunan *loss* yolov11s

Pada Gambar 3, ditampilkan riwayat pelatihan untuk arsitektur YOLOv11s. Mengingat model ini memiliki jumlah parameter yang paling ringan yaitu 9,43 Juta parameter dibandingkan versi lainnya, kurva pembelajarannya terlihat sangat mulus dan landai. Penurunan *Validation Loss* terjadi secara perlahan namun pasti, mengindikasikan proses adaptasi bobot jaringan yang stabil terhadap kondisi pencahayaan dan latar belakang dataset garasi. Konsistensi ini juga tercermin pada kurva *precision* dan *recall* yang merangkak naik secara proporsional. Tidak adanya lonjakan fluktuasi yang drastis pada akhir *epoch* menandakan bahwa YOLOv11s sangat efisien dalam memproses dataset berukuran sedang dengan jumlah iterasi yang relatif singkat.



Gambar 4. Kurva peforma pelatihan dan penurunan *loss* yolov26s

Terakhir, Gambar 4 menyajikan visualisasi grafik pelatihan dari arsitektur terbaru, yakni YOLOv26s. Meskipun secara struktur model ini memiliki mekanisme *NMS-Free* yang diklaim lebih canggih, grafik menunjukkan bahwa model ini membutuhkan usaha komputasi yang sedikit lebih intens pada *epoch-epoch* awal untuk menemukan pola objek. Hal ini terlihat dari penurunan *loss* yang tajam di awal, sebelum akhirnya mulai melandai secara perlahan mendekati *epoch* ke-20. Menariknya, terlepas dari dinamika *loss* tersebut, kurva *mAP@50* dan metrik Presisi pada YOLOv26s berhasil meroket dan

menyentuh angka tertinggi di antara ketiga model pada akhir sesi pelatihan. Grafik ini secara empiris memvalidasi keunggulan YOLOv26s dalam meminimalkan kesalahan deteksi (*False Positive*).

4.3. Evaluasi Kinerja Model Berdasarkan Metrik Akurasi

Kapasitas setiap model dalam mengidentifikasi objek penyusup secara tepat sasaran dievaluasi menggunakan empat matriks, yaitu *Precision*, *Recall*, *Mean Average Precision* (MAP), dan *F1-Score*. Pengukuran ini sangat krusial untuk memastikan sistem keamanan garasi tidak hanya peka terhadap ancaman, tetapi juga minim kesalahan. Rekapitulasi hasil pengujian akurasi dari ketiga arsitektur tersebut disajikan secara lengkap pada Tabel 1

Tabel 1. Perbandingan matriks tiap varian model

Model	Precision	Recall	MAP	F1-Score
Yolov8s	0.841	0.781	862	810
Yolov11s	0.828	0.755	847	790
Yolov26s	0.871	0.743	851	802

Merujuk pada Tabel 1, terlihat adanya variasi karakteristik performa yang cukup mencolok antargenerasi model. YOLOv26s mencatatkan keunggulan absolut pada aspek precision dengan nilai 0.871. Hal ini mengindikasikan bahwa YOLOv26s memiliki tingkat keandalan terbaik dalam menekan rasio *false alarm* atau sinyal palsu. Dalam konteks pengawasan area garasi model ini mampu meminimalisir kesalahan seperti salah mengidentifikasi objek misalnya mendeteksi tumpukan barang, bayangan, atau kendaraan sebagai sosok manusia yang dapat menyebabkan sinyal palsu.

Di sisi lain, saat evaluasi ditinjau dari perspektif kelengkapan deteksi, YOLOv8s justru menunjukkan dominasinya. Model ini meraih nilai *Recall* tertinggi di angka 0,781, yang bermakna bahwa YOLOv8s paling sedikit melewatkan objek manusia yang sebenarnya ada di dalam jangkauan kamera. Keunggulan YOLOv8s juga tervalidasi pada metrik MAP sebesar 0,862. Nilai MAP yang superior ini membuktikan bahwa penempatan kotak pembatas pada target manusia yang bergerak relatif lebih presisi dan akurat secara geometris dibandingkan dua versi lainnya.

Untuk melihat gambaran performa lainnya, digunakan metrik *F1-Score* yang merupakan nilai rata-rata harmonik antara *Precision* dan *Recall*. Keseimbangan performa komputasi YOLOv8s kembali terbukti dengan capaian *F1-Score* tertinggi (0,810). YOLOv26s menyusul di posisi kedua (0,802) karena meskipun presisinya sangat tajam, model ini memiliki kelemahan pada nilai *Recall* (0,743). Sementara itu, YOLOv11s mencatatkan performa akurasi di posisi terbawah untuk pengujian ini dengan *F1-Score* 0,790. Secara keseluruhan, dari segi akurasi identifikasi murni, YOLOv8s dinilai sebagai arsitektur yang menawarkan keseimbangan paling rasional antara kepekaan mendeteksi objek dan kemampuan menahan peringatan yang tidak valid.

4.4. Analisis Kecepatan Infrensi dan Kompleksitas Komputasi

Selain tingkat akurasi identifikasi, kelayakan sebuah model untuk diimplementasikan pada sistem pengawasan sangat ditentukan oleh kapabilitas pemrosesan bingkai citra atau yang lebih sering disebut FPS (*Frames Per Second*) dan waktu tunda atau *Latency*. Berdasarkan rekapitulasi data pada Tabel 1 sebelumnya, terdapat temuan pengujian yang sangat kontras terkait kecepatan inferensi dari ketiga model yang dievaluasi.

Tabel 2. Tabel perbandingan kecepatan infrensi

Nilai	Yolov8s	Yolov11s	Yolov26s
FPS	54.5	38.0	14.2
Latency	18.3	26.3	70.0
Parameter	11.14M	9.43M	9.95M
GFLOPS	28.6	21.5	22.5

Seperti yang terlihat pada Tabel 2, model YOLOv8s mendominasi pengujian performa kecepatan dengan raihan tertinggi mencapai 54,5 FPS dan latensi inferensi yang sangat singkat, yakni 18,3 ms. Angka ini melampaui batas standar minimal video *real-time* (30 FPS), sehingga menjamin pergerakan objek manusia pada antarmuka dapat dilacak secara mulus tanpa adanya efek patah-patah. Responsivitas yang tinggi ini menjadikan YOLOv8s sangat ideal untuk sistem keamanan yang menuntut aksi atau peringatan seketika saat terjadi pelanggaran batas area.

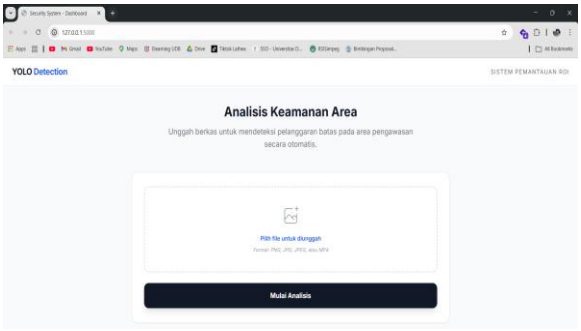
Sebaliknya, meskipun YOLOv26s memiliki tingkat presisi yang superior, arsitektur ini mengalami perlambatan atau *bottleneck* kecepatan yang cukup ekstrem pada skenario pengujian ini. YOLOv26s hanya mampu memproses 14,2 FPS dengan lonjakan waktu tunda mencapai 70,0 ms per bingkai citra. Perlambatan ini mengindikasikan bahwa meskipun model ini diklaim memiliki arsitektur yang disempurnakan, beban komputasi aktual pada saat inferensi video berjalan memakan waktu yang jauh lebih lama, yang berpotensi menyebabkan pelacakan ID objek menjadi kurang konsisten pada target yang bergerak sangat cepat.

Evaluasi kelayakan model juga ditinjau dari sisi kompleksitas ruang memori dan beban operasi matematis atau GFLOPS. Dari perspektif efisiensi sumber daya, YOLOv11s membuktikan dirinya sebagai model yang paling ringan dengan ukuran parameter terkecil yaitu 9,43 Juta parameter dan beban operasi terendah di angka 21,5 GFLOPS. Meskipun secara kecepatan YOLOv11s berada di posisi menengah yaitu 38,0 FPS dan latensi di angka 26,3 ms, efisiensi komputasi yang ditawarkannya menjadikannya varian alternatif yang sangat kuat apabila sistem garasi cerdas ini kedepannya akan dikembangkan pada perangkat *Edge Computing*

seperti Raspberry Pi atau *mini-PC* yang memiliki keterbatasan kapasitas RAM dan daya listrik.

4.5. Implementasi Antarmuka dan Analisis Visual

Tahap akhir dari pengujian ini adalah implementasi model ke dalam antarmuka berbasis *web* menggunakan *framework* Flask. Antarmuka ini dirancang untuk memudahkan pengguna dalam memonitor area garasi serta melihat visualisasi dari pagar virtual yang telah dikonfigurasi. Pada sistem ini, area pengawasan dibatasi oleh beberapa titik poligon yang disesuaikan dengan perspektif kamera pada area garasi.

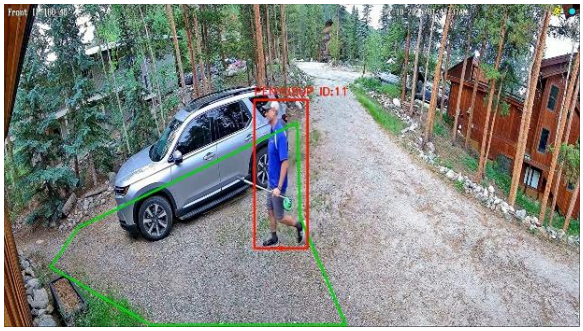


Gambar 5. Antarmuka website analisis yolo

Pada gambar 5 adalah desain antarmuka website yang dibuat untuk menginputkan foto atau video yang ingin di analisis dan akan menghasilkan output berupa hasil analisis seperti pada gambar 6.

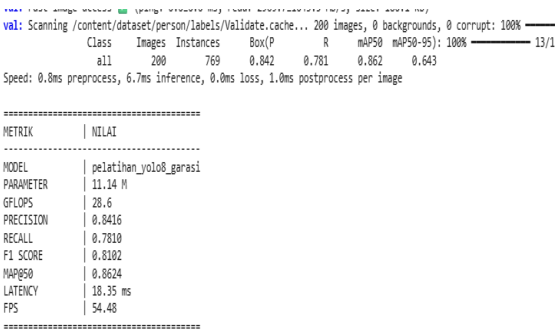
Gambar 6. Objek manusia diluar garis ROI

Visualisasi hasil deteksi pada sistem ini memberikan indikasi warna yang berbeda berdasarkan posisi objek manusia terhadap area ROI. Sebagaimana ditampilkan pada gambar 6, saat sistem mendeteksi keberadaan manusia namun posisi titik tengah objek masih berada di luar koordinat poligon terlarang, maka *bounding box* akan berwarna biru dengan keterangan status "person". Namun, seketika objek melintasi batas pagar virtual, sistem secara otomatis mengubah visualisasi kotak menjadi merah dengan label "penyusup" serta menampilkan *tracking ID* yang unik untuk setiap individu seperti pada gambar 7.



Gambar 7. Objek manusia didalam garis ROI

Penggunaan fitur *object tracking* pada implementasi ini terbukti memberikan stabilitas deteksi yang lebih baik. Meskipun objek sempat mengalami kesulitan dalam mengidentifikasi karena terhalang sebagian, namun sistem pelacakan tetap mampu mempertahankan identitas objek yang sama sehingga meminimalisir kesalahan pendeteksian.



```
val: Scanning /content/dataset/person/labels/Validate.cache... 200 images, 0 backgrounds, 0 corrupt: 100% ━━━━━━━━━━━━ 200/200
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ━━━━━━━━━━━━ 13/13
                   all        200        769      0.842      0.781      0.862      0.643
Speed: 0.8ms preprocess, 6.7ms inference, 0.0ms loss, 1.0ms postprocess per image

=======================================
METRIK          | NILAI
---------------------------------------
MODEL           | pelatihan_yolo8_garasi
PARAMETER       | 11.14 M
GFLOPS          | 28.6
PRECISION       | 0.8416
RECALL          | 0.7810
F1 SCORE        | 0.8102
MAP@50          | 0.8624
LATENCY         | 18.35 ms
FPS             | 54.48
=======================================
```

Gambar 8. Hasil pengujian yolov8s pada google colaboratory

Hasil analisis visual ini mengkonfirmasi bahwa integrasi algoritma YOLOv8s dengan logika pagar virtual mampu memberikan proteksi aktif yang responsif untuk keamanan garasi dengan hasil FPS dan *latency* yang berada diatas model YOLOv11s dan YOLOv26s.

5. KESIMPULAN DAN SARAN

Kesimpulan pada penelitian ini yaitu, model YOLOv8s, YOLOv11s, dan YOLOv26s mampu mendeteksi objek berupa orang serta dapat mengidentifikasi objek jika melewati garis ROI yang telah ditentukan. Namun sistem belum bisa mengenali objek manusia yang bukan termasuk orang asing contohnya anggota. Saran untuk penelitian selanjutnya adalah menambahkan sistem untuk pengenalan wajah dengan cara mendaftar terlebih dahulu agar nantinya jika yang melewati garis ROI adalah anggota keluarga yang terdaftar maka tidak menimbulkan peringatan atau *false alarm.*

DAFTAR PUSTAKA

[1] D. Andreas *et al.*, "Optimasi Sistem Deteksi Pencurian Motor Real-Time Menggunakan YOLO dan TensorRT," *RIGGS: Journal of Artificial Intelligence and Digital Business*, vol. 4, no. 2, pp. 3957–3964, Jun. 2025, doi: 10.31004/riggs.v4i2.1147.

[2] D. Pakiding, A. Selao, and W. Wahyuddin, "Implementasi Computer Vision dalam Mendeteksi Penyakit pada Tanaman Cabai dan Tomat Menggunakan Algoritma Convolutional Neural Networks," *MALCOM: Indonesian Journal of Machine Learning and Computer Science*, vol. 5, no. 3, pp. 841–850, Jun. 2025, doi: 10.57152/MALCOM.V5I3.1989.

[3] S. K. Adri, A. I. Pradana, and B. W. Pamekas, "PENGEMBANGAN SISTEM PERHITUNGAN JUMLAH KENDARAAN BERDASARKAN JENIS KENDARAAN MENGGUNAKAN ALGORITMA YOLO SECARA REALTIME," *SKANIKA: Sistem Komputer dan Teknik Informatika*, vol. 7, no. 2, pp. 166–179, Jul. 2024, doi: 10.36080/SKANIKA.V7I2.3201.

[4] F. Dalimarta, P. N. Andono, Moch. A. Soeleman, and Z. A. Hasibuan, "Towards Automated Motor Impulsivity Monitoring in Real-world Scenarios: A Multiple Object Tracking Approach," *Data Science: Journal of Computing and Applied Informatics*, vol. 9, no. 1, pp. 1–17, Jan. 2025, doi: 10.32734/JOCAI.V9.I1-16686.

[5] S. Hendrizal, "Penentuan Region Of Interest (ROI) Untuk Menghitung Jumlah Kendaraan Pada Jalan Raya Menggunakan Frame Substratcion," *Jurnal Komputer Teknologi Informasi Sistem Komputer (JUKTISI)*, vol. 2, no. 2, pp. 455–460, Sep. 2023, doi: 10.62712/JUKTISI.V2I2.139.

[6] M. S. Hossain *et al.*, "Region of interest (ROI) selection using vision transformer for automatic analysis using whole slide images," *Scientific Reports 2023 13:1*, vol. 13, no. 1, pp. 11314-, Jul. 2023, doi: 10.1038/s41598-023-38109-6.

[7] R. Sapkota and M. Karkee, "Ultralytics YOLO Evolution: An Overview of YOLO26, YOLO11, YOLOv8 and YOLOv5 Object Detectors for Computer Vision and Pattern Recognition," Oct. 2025, Accessed: May 02, 2026.

[8] A. Arif *et al.*, "Object Detection for Safety Attire Using YOLO (You Only Look Once)," *Journal of Advanced Research in Applied Mechanics Journal homepage*, vol. 113, pp. 37–51, 2024, doi: 10.37934/aram.113.1.3751.

[9] N. P. Purnawan, N. W. Utami, and J. M. Suhendro, "DETEKSI PENYAKIT DAUN MANGGA MENGGUNAKAN YOLOV8 BERBASIS DEEP LEARNING," *JATI (Jurnal Mahasiswa Teknik Informatika)*, vol. 10, no. 2, pp. 2385–2392, Mar. 2026, doi: 10.36040/JATI.V10I2.17415.

[10] R. Praseli *et al.*, "Implementasi Perbandingan YOLO v8, v9, dan v11 dalam Penerapan Tata Tertib K3: Deteksi Penggunaan Helm Keselamatan di Lingkungan Konstruksi," *MALCOM: Indonesian Journal of Machine Learning and Computer Science*, vol. 6, no. 2, pp. 513–523, Apr. 2026, doi: 10.57152/MALCOM.V6I2.2554.

[11] C. R. A. Widiawati, "Pengaruh Dataset terhadap Performa Convolutional Neural Network pada Klasifikasi X-Ray Pasien Covid-19," *Jurnal Teknologi Informasi Dan Ilmu Komputer (JTIIK), 2022*. doi: 10.25126/JTIIK.202295645.

[12] M. Sobirin and P. Wijonarko, "Pemantauan Keamanan Real-Time pada Base Transceiver Station (BTS) Menggunakan YOLOv8 dan Integrasi Telegram: Optimasi Model dan Evaluasi Performa," *ILKOMNIKA*, vol. 7, no. 3, pp. 533–547, Dec. 2025, doi: 10.28926/ILKOMNIKA.V7I3.824.

[13] H. U. Adzkia, A. Asriyanik, and W. Apriandari, "IMPLEMENTASI YOLOV12 UNTUK DETEKSI KERUSAKAN VISUAL PADA PAKAIAN BEKAS BERBASIS CITRA DIGITAL," *JATI (Jurnal Mahasiswa Teknik Informatika)*, vol. 10, no. 2, pp. 3586–3593, Mar. 2026, doi: 10.36040/JATI.V10I2.17628.

[14] E. Nasrul, A. Asriyanik, and A. Pambudi, "IMPLEMENTASI YOLOV11 UNTUK DETEKSI GERAKAN SANDI SEMAFOR DENGAN PENDEKATAN POSE ESTIMATION," *JATI (Jurnal Mahasiswa Teknik Informatika)*, vol. 10, no. 2, pp. 3403–3410, Mar. 2026, doi: 10.36040/JATI.V10I2.17672.