JIKA (Jurnal Informatika) Universitas Muhammadiyah Tangerang *P ISSN : 2549-0710*
Tangerang, April 2024, Vol 8 No.2, pp 204-212 *E ISSN : 2722-2713*

PENGENALAN RAMBU LALU LINTAS MENGGUNAKAN METODE *YOLOV8*

Muh. Ikbal[1)], Rizal Adi Saputra[2)]

[1,2] Program Studi Teknik Informatika, Universitas Halu Oleo, Kampus Hijau Bumi Tridharma Anduonohu, Kec. Kambu, Kota Kendari, Sulawesi Tenggara 93232

Co Responden Email: ikbalahbab2004@gmail.com

Article history
Received 11 Jan 2024
Revised 12 Mar 2024
Accepted 18 Apr 2024
Available online 30 Apr 2024

Keywords
Autonomous vehicle,
Object Recognition,
Sign ,
Traffic,
YOLOv8

Abstract

Autonomous vehicle technology has seen rapid development in recent years. The primary goal of this technology is to reduce traffic accidents and create a safer driving environment. Autonomous vehicles are expected to fully understand the traffic environment and operate according to the rules. One of the artificial intelligence methods often used in this technology is deep learning YOLO. This research develops a method for detecting types of traffic signs using YOLO V8, which is a development of the CNN method. The data used in this research includes 30 labels of traffic sign types with a total of 4650 images. The dataset is divided in a 70:30 ratio, where 70% is used for training and 30% for testing. After the division, the training dataset consists of 3232 images, while the testing dataset includes 1418 images. The test results show that this model performs well, with a Precision value of 0.993, Recall of 0.999, mAP50 of 0.995, and mAP50-95 of 0.984. These values indicate that this model is capable of identifying and classifying traffic signs with high accuracy.

Riwayat
Diterima 11 Jan 2024
Revisi 12 Mar 2024
Disetujui 18 Apr 2024
Terbit Online 30 Apr 2024

Kata Kunci
Kendaraan Otonom,
Lalu-Lintas,
Pengenalan Objek,
Rambu,
YOLOv8

Abstrak

Teknologi kendaraan otonom telah mengalami perkembangan pesat dalam beberapa tahun terakhir. Tujuan utama dari teknologi ini adalah untuk mengurangi kecelakaan lalu lintas dan menciptakan lingkungan berkendara yang lebih aman. Kendaraan otonom diharapkan dapat sepenuhnya memahami lingkungan lalu lintas dan beroperasi sesuai aturan. Salah satu metode kecerdasan buatan yang sering digunakan dalam teknologi ini adalah deep learning YOLO. Penelitian ini mengembangkan metode deteksi jenis rambu lalu lintas menggunakan YOLO V8, yang merupakan pengembangan dari metode CNN. Data yang digunakan dalam penelitian ini mencakup 30 label jenis rambu lalu lintas dengan total 4650 citra. Dataset dibagi dengan proporsi 70:30, di mana 70% digunakan untuk pelatihan dan 30% untuk pengujian. Setelah pembagian, dataset pelatihan terdiri dari 3232 gambar, sementara dataset pengujian mencakup 1418 gambar. Hasil pengujian menunjukkan bahwa model ini memiliki kinerja yang baik, dengan nilai Precision sebesar 0,993, Recall sebesar 0,999, mAP50 sebesar 0,995, dan mAP50-95 sebesar 0,984. Nilai-nilai ini menunjukkan bahwa model ini mampu mengidentifikasi dan mengklasifikasikan rambu lalu lintas dengan akurasi yang tinggi.

PENDAHULUAN

Teknologi kendaraan otonom (*autonomous vehicle)* telah berkembang pesat dalam beberapa tahun terakhir. Alasan utama munculnya teknologi kendaraan tanpa pengemudi adalah untuk mengurangi kecelakaan lalu lintas dengan menciptakan lingkungan berkendara yang aman. Dengan teknologi ini, kendaraan diharapkan dapat mengenali sepenuhnya lingkungan lalu lintas yang dilaluinya dan bergerak sesuai aturan. Dalam konteks ini, pengenalan rambu lalu lintas merupakan bidang yang sangat penting untuk teknologi ini. (Xuan-Ha Nguyen, 2022)

Penelitian pengenalan rambu lalu lintas sudah banyak dilakukan menggunakan metode berdasarkan warna dan bentuk. Meskipun metode berdasarkan warna dan bentuk dapat mengenali bagian-bagian penting dari rambu seperti berhenti dan

peringatan, namun hanya beberapa bagian saja yang dapat dikenali dan keakuratannya tidak tinggi .(Zhang et al., 2023). Keterbatasan itu menunjukkan bahwa pendekatan yang lebih canggih diperlukan.

Metode seperti YOLO dapat membantu mengatasi masalah ini. YOLO, singkatan dari "*You Only Look Once*," adalah algoritma yang membantu mendeteksi objek secara *real-time* menggunakan arsitektur *Convolutional Neural Network* (CNN).

YOLO pada awalnya dikembangkan oleh Joseph Redmon dan Ali Farhadi pada 2015. YOLO terinspirasi dari *GoogleNet* yang merupakan model yang digunakan untuk klasifikasi gambar. (Fahim, Abdal, Rasel, Sarker, and Chowdhury, n.d.)

Dengan memanfaatkan *deep convolutional neural networks*, YOLO dapat belajar mengenali berbagai macam objek secara akurat dan melokalisasi objek ke dalam gambar(Redmon, Divvala, Girshick, and Farhadi, 2016). YOLO dapat mendeteksi beberapa objek dari kelas yang berbeda secara bersamaan, sehingga sangat berguna untuk aplikasi yang membutuhkan pemrosesan waktu nyata dan akurasi deteksi yang tinggi, seperti pengemudian otonom, mengenali rambu lalu lintas dengan akurat, pengawasan video, dan robotika. (Sinha, Hadassa, Krishna, and Ravi Kumar, 2020)

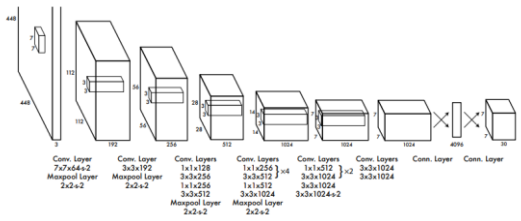
Terdapat penelitian-penelitian tentang pengenalan rambu lalu lintas menggunakan metode YOLO seperti implementasi object recognition pada rambu-rambu dan lampu lalu lintas dengan raspberry pi dengan algoritma yolov5(Nugroho and Cahyono, 2022), Development of Real-Time Traffic-Object and Traffic-Sign Detection Models Applied for Autonomous Intelligent Vehicles(Shen, Lang, and Song, 2023), Pengenalan Rambu Lalu Lintas di Indonesia Secara Real-time Menggunakan YOLOv4-tiny(Gregorius, Goenawan, and Tjondrowiguno, 2022) dan lain-lain.

METODE PENELITIAN

Dalam penelitian ini, digunakan pendekatan *deep learning* dengan menggunakan algoritma *You Only Look Once* (YOLO) untuk memecahkan masalah deteksi rambu lalu lintas.

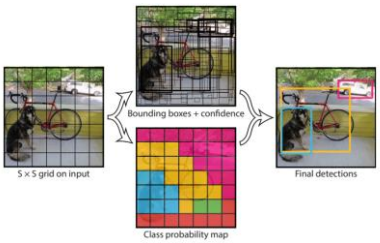
You Only Look Once (YOLO) memiliki arsitektur yang terdiri dari 24 lapisan konvolusional. Lapisan-lapisan ini berfungsi untuk mengekstrak fitur dari gambar, yang kemudian digunakan untuk memprediksi probabilitas dan koordinat objek. Setelah ekstraksi fitur, metode YOLO melanjutkan dengan dua lapisan terhubung, yang bertanggung jawab untuk memprediksi probabilitas dan koordinat.

Dengan demikian, YOLO tidak hanya mengidentifikasi objek dalam gambar, tetapi juga menentukan lokasi dan ukuran objek tersebut. Metode ini terdiri dari beberapa tahap dan berfokus pada penentuan tingkat akurasi objek secara real-time. Ini menjadikan YOLO sebagai algoritma yang efisien dan efektif dalam mendeteksi objek pada gambar.



Gambar 1. Arsitektur YOLO

YOLO dapat menggunakan neural network untuk memprediksi bounding box dan kelas objek yang ada di gambar. Kemudian, neural network akan membagi gambar menjadi grid S x S, dan jika bagian tengah objek menyentuh sel grid, sel grid akan mengenali objek tersebut. (Gong, Li, Song, Xu, and He, 2022)



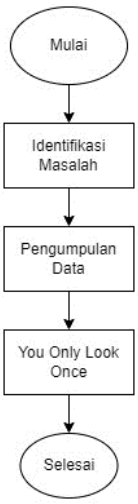
Gambar 2. Hasil dari penerapan YOLO

YOLOv8 merupakan versi terbaru dari YOLO yang dikembangkan pada tahun 2023 dan telah menunjukkan peningkatan yang signifikan dalam kecepatan dan akurasi dibandingkan dengan versi-versi sebelumnya. YOLOv8 dibangun di atas kesuksesan versi sebelumnya, memperkenalkan fitur-fitur baru dan peningkatan untuk meningkatkan kinerja, fleksibilitas, dan efisiensi. (Raza M, 2023).

YOLOv8 memakai versi yang lebih baik dari arsitektur CSPDarknet53, yang telah

terbukti efektif dalam mencatat data lokasi dengan presisi (Shahab, 2023). Ini menjadikan YOLOv8 sebagai algoritma yang efisien dan efektif dalam mendeteksi objek pada gambar.(Janapriya, 2023)

Adapun tahapan dalam penelitian ini yaitu identifikasi masalah, pengumpulan dataset, dan implementasi *You Only Look Once* (YOLO) versi 8. Berikut adalah gambar alur penelitian ini.



Gambar 3. Alur Penelitian

1. Identifikasi Masalah

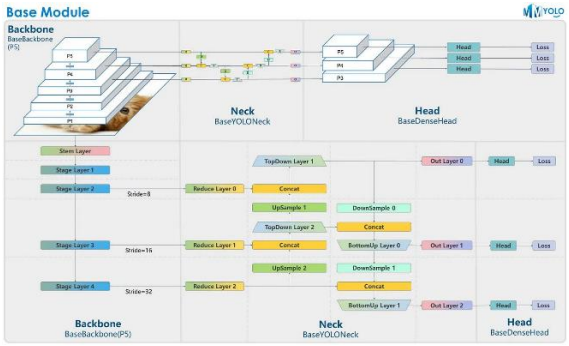
Setelah memilih topik penelitian, langkah pertama adalah menentukan masalah yang akan diteliti. Untuk menjaga penelitian tetap fokus pada tujuannya, identifikasi ini dilakukan untuk memperjelas batasan masalah. Studi ini menggunakan algoritma *You Only Look Once* (YOLO) versi 8 untuk mengklasifikasikan rambu jalan dalam konteks ini.

2. Pengumpulan Data

Proses pengumpulan data dilakukan dengan memanfaatkan data yang telah di-upload oleh Putri Mawaring Wening. Data tersebut berisi gambar rambu-rambu lalu lintas dalam format JPG. Resolusi gambar telah distandarisasi menjadi 640 x 640 piksel. Gambar tersebut akan diklasifikasikan menjadi 30 jenis rambu lalu lintas di platform *Roboflow*. Total gambar yang digunakan adalah 4650 gambar.

3. Pengembangan Model *YOLOv8*

Metode yang digunakan dalam penelitian ini adalah *YOLOv8* dengan ukuran input 640 x 640. Proses training model dilakukan dengan menggunakan arsitektur *YOLOV8*. Terdapat 2 bagian utama pada model *YOLOV8* yaitu *Backbone* dan *Head*/*Prediction* .

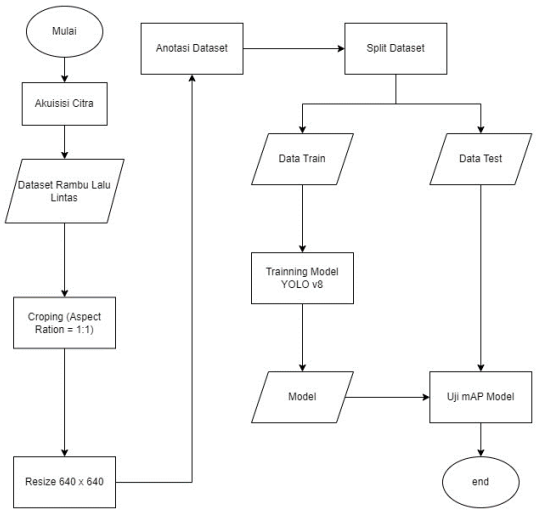


Gambar 4. Arsitektur *YOLOv8*

Backbone Dikenal juga sebagai ekstraktor fitur, bertanggung jawab untuk mengekstrak fitur penting dari input. Bagian ini menangkap pola sederhana di lapisan awal, seperti tepi dan tekstur(Munawar, 2023). *Neck* bertindak sebagai jembatan antara backbone dan head, melakukan operasi fusi fitur dan mengintegrasikan informasi kontekstual. *Head* merupakan bagian akhir dari jaringan dan bertanggung jawab untuk menghasilkan output jaringan, seperti kotak pembatas dan skor kepercayaan untuk deteksi objek. *Head* berfungsi untuk menghasilkan kotak pembatas yang terkait dengan objek yang mungkin ada dalam gambar, menetapkan skor kepercayaan untuk setiap kotak pembatas untuk menunjukkan seberapa mungkin suatu objek ada, mengurutkan objek dalam kotak pembatas menurut kategori.(Fahim, Abdal, Rasel, Sarker, and Chowdhury, 2024)

Dalam proses pembuatan model YOLOv8 untuk mengenali objek lampu lalu lintas, beberapa langkah penting perlu dilakukan. Pertama, pengumpulan dataset yang mencakup gambar lampu lalu lintas dari berbagai kondisi dan sudut. Kedua, melakukan pemotongan (*crop*) gambar sehingga rasio aspeknya menjadi 1:1, ini penting untuk memastikan konsistensi dalam data. Ketiga, melakukan anotasi objek pada gambar, yaitu menandai dan mengidentifikasi objek lampu lalu lintas dalam setiap gambar. Terakhir, membagi dataset menjadi dua kategori, yaitu data latihan (*train*) dan data uji (*test*), untuk melatih dan kemudian menguji model. Dengan demikian, model YOLOv8

dapat dilatih dan diuji untuk mengenali objek lampu lalu lintas dengan efektif.(Munawar, 2023)



Gambar 5. Alur pengembangan YOLOV8

a Persiapan Dataset Rambu Lalu Lintas

Dataset yang digunakan dalam penelitian ini adalah kumpulan gambar rambu lalu lintas. Dataset ini mencakup 30 kategori rambu lalu lintas yang telah diklasifikasikan dengan hati-hati. Setiap kategori mewakili jenis rambu lalu lintas yang berbeda, memberikan variasi yang luas dalam data. Dataset ini kemudian dibagi menjadi dua bagian data pelatihan dan data pengujian. Pembagian ini memungkinkan model untuk belajar dari data pelatihan dan kemudian diuji pada data pengujian untuk mengevaluasi kinerjanya. Untuk penelitian ini, gambar dalam dataset akan diproses dengan pemotongan berdasarkan rasio 1:1. Secara spesifik, setiap gambar dalam dataset akan dipotong menjadi ukuran 640 x 640 piksel.

b Anotasi Dataset

Anotasi objek adalah pemberian tanda dengan menentukan bounding box yang berisi nama kelas sesuai dengan masing-masing objek(Takarob R, 2022). *Bounding box* ini berisi label atau nama kelas yang sesuai dengan objek tersebut. Proses ini penting dalam pelatihan model deteksi objek, karena memberikan informasi kepada model tentang lokasi dan jenis objek dalam gambar.

c Membuat Model *YOLOV8*

Pada tahap ini peneliti berencana membangun model YOLOV8. Tujuan utama pembuatan model ini adalah untuk mengklasifikasikan berbagai jenis rambu lalu lintas. Model ini dilatih menggunakan dataset yang berisi gambar rambu-rambu jalan dari berbagai jenis dan bentuk. Setelah model dilatih, model dapat mengenali dan mengklasifikasikan rambu-rambu jalan pada gambar baru dengan akurasi tinggi.

d *Learning (Traning)*

Melatih model YOLOv8 adalah proses yang melibatkan beberapa langkah penting. Pertama, model YOLOv8 yang telah diverifikasi sebelumnya atau model terlatih dimuat. Setelah itu, dataset yang digunakan untuk pelatihan dibuat. Kumpulan data ini biasanya dibagi menjadi data latih dan data uji. Setelah itu, model dilatih menggunakan data pelatihan(Liu, Shi, Li, and Zhao, 2022).

Selama proses ini, parameter model disesuaikan untuk meminimalkan kesalahan prediksi. Selain itu, hyperparameter model dapat disesuaikan untuk mengoptimalkan performa model. Metrik pelatihan dipantau secara real time dan pembelajaran dapat divisualisasikan untuk pemahaman yang lebih baik. Jika lebih dari satu GPU tersedia, beban pelatihan dapat dibagi di antara keduanya untuk mempercepat proses. Proses ini mungkin berbeda-beda tergantung pada data spesifik dan tujuan penelitian. (Arief R, 2021)

e *Testing*

Proses pengujian adalah langkah akhir dalam rangkaian penelitian ini. Tahap ini dimulai dengan penyesuaian ukuran gambar input agar cocok dengan model yang ditargetkan. Setelah ukuran gambar disesuaikan, metode *You Only Look Once* (YOLO) akan digunakan untuk mengklasifikasikan gambar input tersebut.

f Matriks Evaluasi

Precision, *Recall*, mAP50, dan mAP50-95 adalah teknik evaluasi yang digunakan. Untuk deteksi objek, *Mean Average Precision* (mAP) adalah metode evaluasi. YOLOV8 menggunakan mAP50 dan mAP50-95 secara default. Matriks *confusion* digunakan untuk menghitung metrik *Precision* dan *Recall* dari setiap label pada gambar dalam penghitungan metrik mAP. Metriks mAP50 menggunakan nilai batas IoU sebesar 0,5, sementara metriks mAP50-95 adalah rata-rata mAP dengan sepuluh nilai batas IoU yang berbeda, yaitu 0,5, 0,55, 0,60, 0,65, 0,70, 0,75, 0,80, 0,85, 0,90, dan 0,95..(Janapriya, 2023)

Untuk menghitung nilai Nilai Positif Benar (TP), Nilai Tidak Benar (FP), dan Nilai Negatif Benar (FN), matriks confusion pada

deteksi objek menggunakan nila IoU. Nilai Negatif Benar (TN) tidak digunakan karena tidak mempengaruhi kinerja metode deteksi objek. Tabel 2 menunjukkan penjelasan untuk setiap nilai metrik dalam matriks kebingungan.

Tabel 1. Matriks *confusion* pada deteksi objek

TP (*True Positive*)	IoU ≥ α (*thresold* IoU).
FP (*False Positive*)	*IoU < α (thresold IoU).*
TN (*True Negative*)	semua bagian gambar yang tidak memiliki objek dan tidak dapat dideteksi sebagai objek.
FN (*False Negative*)	Ground truth yang sama sekali tidak terdeteksi.

HASIL DAN PEMBAHASAN

1. Hasil Pengumpulan Data

Dataset yang digunakan oleh peneliti dalam penelitian ini terdiri dari 4650 gambar yang mencakup 30 jenis rambu lalu lintas.

Tabel 2. Rambu dan jumlah kemunculan dalam dataset

No	Nama Rambu	Jumlah Kemunculan
1	Balai Pertolongan Pertama	155
2	Banyak Anak-Anak	155
3	Dilarang Belok Kanan	155
4	Dilarang Berhenti	155
5	Dilarang Berjalan Terus	155
6	Dilarang Masuk	155
7	Dilarang Mendahului	155
8	Dilarang Parkir	155
9	Dilarang Putar Balik	155
10	Gereja	155
11	Hati-Hati	155
12	Jalur Penyebrangan	155
13	Lampu Lalu Lintas	155
14	Larangan Kecepatan - 30km-jam	155
15	Larangan Kecepatan - 40km-jam	155
16	Larangan Kendaraan MST - 10 Ton	155
17	Masjid	155
18	Pemberhentian Bus	155
19	Perintah Ikuti Bundaran	155
20	Perintah Jalur Sepeda	155
21	Perintah Lajur Kiri	155
22	Perintah Pilih Satu Jalur	155
23	Persimpangan 3 Prioritas	155
24	Persimpangan 3 Sisi Kanan Prioritas	155
25	Persimpangan 3 Sisi Kiri Prioritas	155
26	Persimpangan Empat	155
27	Putar Balik	155
28	Rumah Sakit	155
29	SPBU	155
30	Tempat Parkir	155

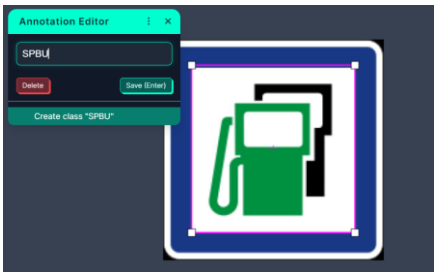
Setelah memproses gambar dalam dataset dengan pemotongan berdasarkan rasio 1:1 dan mengubah ukuran setiap gambar menjadi 640 x 640 piksel, dapat meningkatkan efisiensi dan akurasi model *YOLOV8*

Gambar 6. Dataset yang telah di resize 640 x 640

2. Hasil Anotasi Dataset

Setelah gambar telah dikumpulkan dan disusun dalam dataset, masing-masing gambar diberi anotasi sesuai format YOLO.. Pemberian label pada objek dilakukan dengan menggunakan Label Img dalam format YOLO. Hasil dari proses anotasi ini adalah file data yang berisi detail bounding box dalam bentuk format .txt. Proses anotasi ini

dapat dilakukan secara otomatis menggunakan platform seperti Roboflow, yang ditunjukkan pada Gambar 7.



Gambar 7. Anotasi pada *roboflow*

3. Hasil Split Dataset

Hasil dari split dataset dibagi menjadi dua kategori yaitu kategori pelatihan (*Test*) dan kategori pengujian (*Train*) . Pembagian ini dilakukan secara acak, dengan mempertimbangkan keberadaan tanda-tanda pada gambar. Proporsi pembagian dataset adalah 70:30, dengan 70% untuk pelatihan (*Train*) dan 30% untuk pengujian (*Test*). Setelah proses pembagian, dataset pelatihan (Train) terdiri dari 3232 gambar, sementara dataset pengujian mencakup1.418 gambar.

4. Pembuatan Model YOLOv8 Untuk Mendeteksi Rambu-Rambu Jalan

Dalam tahap pembuatan model ini, peneliti memilih untuk menggunakan arsitektur YOLOv8. Model ini dilatih menggunakan dataset rambu lalu lintas di Indonesia, dengan tujuan untuk mengklasifikasikan berbagai jenis rambu lalu lintas

```
# Build from YAML and transfer weights
Final_model = YOLO('yolov8n.yaml').load('yolov8n.pt')

# Training The Final Model
Result_Final_model = Final_model.train(data="/kaggle/input/traffic-sign-in-indonesia/data.yaml",epoc
hs=100, imgsz = 416, batch = 64 ,lr0=0.0001, dropout= 0.15, device = 0)
```

Gambar 8. Pembuatan model YOLOv8

Kode ini menjelaskan proses pelatihan model YOLOv8. Pertama, model YOLOv8 dimuat dengan menggunakan file konfigurasi ‘yolov8n.yaml’ dan file bobot model yang telah dilatih sebelumnya ‘yolov8n.pt’. Kemudian, model tersebut dilatih menggunakan dataset rambu lalu lintas di Indonesia.

5. *Training*

Dalam penelitian ini deteksi rambu lalu lintas dengan YOLOV8 menggunakan nilai *hyperparameter learning rate* (lr0) yang digunakan adalah 0,001, sementara jumlah *epoch* adalah 100. Ini menjadi patokan untuk kombinasi *hyperparameter* yang digunakan dalam penelitian ini. Jumlah *epoch* yang digunakan adalah 100, ukuran *batch* adalah 64, dan *learning rate* adalah 0,001. Bobot yang telah dilatih sebelumnya digunakan sebagai bobot awal pada YOLOV8

Gambar 9. Hasil pelatihan model YOLOv8

Proses pelatihan model dilakukan dengan menggunakan data pelatihan(Train). Selama proses ini, model belajar untuk mengenali dan mengklasifikasikan berbagai jenis rambu lalu lintas. Setelah model dilatih, metrik evaluasi seperti *Precision*, *Recall*, dan *Mean Average Precision* (mAP) dihitung. Metrik-metrik ini diperoleh melalui proses validasi, yang melibatkan pengujian model yang telah dilatih pada data yang belum pernah dilihat sebelumnya. Proses validasi ini memberikan gambaran tentang sejauh mana model dapat menggeneralisasi apa yang telah dipelajari ke data baru. Dengan demikian, metrik evaluasi ini memberikan ukuran kinerja model dalam deteksi dan klasifikasi rambu lalu lintas. Hasil deteksi ditunjukkan pada Tabel 2 yang mencantumkan tiga indikator evaluasi P(*precision*), R (*Recall*)dan mAP(*Mean Average Precision*).

Tabel 3. Hasil dari training YOLOv8

Class	Box(P)	R	mAP50	mAP50 -95
All	0.994	1	0.995	0.984
Balai Pertolongan Pertama	0.993	1	0.995	0.995

Banyak Anak-Anak	1	1	0.995	0.99
Dilarang Belok Kanan	0.997	1	0.995	0.925
Dilarang Berhenti	0.996	1	0.995	0.995
Dilarang Berjalan Terus	0.994	1	0.995	0.995
Dilarang Masuk	0.996	1	0.995	0.986
Dilarang Mendahului	0.994	1	0.995	0.988
Dilarang Parkir	0.997	1	0.995	0.991
Dilarang Putar Balik	1	0.995	0.995	0.985
Gereja	0.988	1	0.995	0.995
Hati-Hati	0.991	1	0.995	0.995
Jalur Penyebrangan	0.994	1	0.995	0.979
Lampu Lalu Lintas	0.992	1	0.995	0.995
Larangan Kecepatan - 30km/jam	0.993	1	0.995	0.995
Larangan Kecepatan - 40km/jam	0.992	1	0.995	0.995
Larangan Kendaraan MST - 10 Ton	0.993	1	0.995	0.995
Masjid	0.998	1	0.995	0.987
Pemberhentian Bus	0.991	1	0.995	0.983
Perintah Ikuti Bundaran	0.994	1	0.995	0.984
Perintah Jalur Sepeda	0.995	1	0.995	0.995
Perintah Lajur Kiri	0.978	1	0.995	0.963
Perintah Pilih Satu Jalur	0.997	1	0.995	0.929
Persimpangan 3 Prioritas	0.994	1	0.995	0.979
Persimpangan 3 Sisi Kanan Prioritas	0.993	1	0.995	0.99
Persimpangan 3 Sisi Kiri Prioritas	0.99	1	0.995	0.995
xPersimpangan Empat	0.993	1	0.995	0.995
Putar Balik	0.994	1	0.995	0.977
Rumah Sakit	0.995	1	0.995	0.981
SPBU	0.997	1	0.995	0.972
Tempat Parkir	0.993	1	0.995	0.995

Hasil dalam tabel tersebut menunjukkan bahwa model deteksi objek telah melakukan pekerjaan yang sangat baik dalam mengidentifikasi dan melokalisasi berbagai jenis tanda dan tempat dalam gambar. Berikut adalah beberapa poin penting:

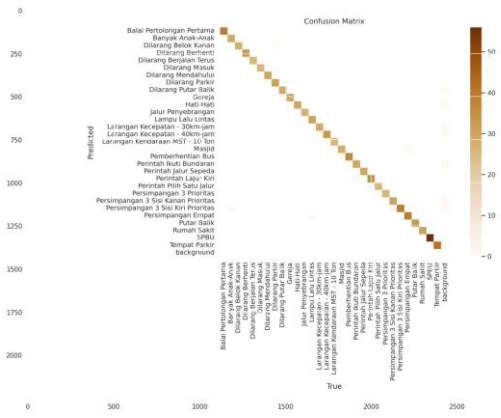
1. Presisi Kotak Pembatas (Box(P)): Nilai ini berkisar antara 0.978 hingga 1, yang menunjukkan bahwa model sangat akurat dalam memprediksi lokasi objek dalam gambar.
2. *Recall* (R): Nilai ini adalah 1 untuk semua kelas, yang berarti model berhasil menemukan semua instansi objek dalam gambar.
3. *Mean Average Precision* (mAP50 : Nilai ini adalah 0.995 untuk semua kelas, yang menunjukkan bahwa model memiliki presisi rata-rata yang sangat tinggi saat IoU diatur ke 0.50.
4. *Mean Average Precision* (mAP50-95): Nilai ini adalah rata-rata presisi rata-rata untuk IoU dari 0.50 hingga 0.95. Nilai ini memberikan gambaran tentang seberapa baik model bekerja pada berbagai tingkat IoU.

Secara keseluruhan, hasil ini menunjukkan bahwa model telah dilatih dengan baik dan mampu mendeteksi dan melokalisasi objek dengan akurasi yang sangat tinggi. Ini menunjukkan kualitas yang baik dari dataset yang digunakan untuk pelatihan dan efektivitas arsitektur model (dalam hal ini, YOLOv8).

6. *Testing*

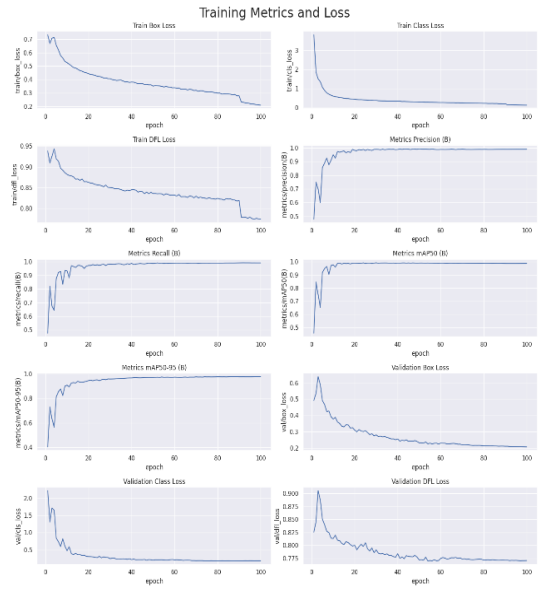
Tahap pengujian dalam pengembangan model deteksi objek seperti YOLOv8 sangat penting. Pertama, model diuji untuk melihat akurasi dalam mendeteksi objek dalam gambar yang tidak dikenal. Ini melibatkan penggunaan dataset pengujian yang berbeda dari dataset pelatihan. Metrik seperti *Presisi*, *Recall*, dan *Mean Average Precision* (mAP) biasanya digunakan untuk mengukur kinerja mode1(Oklilas, Sukemi, and Apriliyanto, 2023). Setelah model menunjukkan akurasi yang memadai, label prediksi diuji. Ini berarti memeriksa apakah model tidak hanya dengan benar mendeteksi keberadaan objek dalam

gambar, tetapi juga dengan benar mengidentifikasi kelas objek tersebut.



Gambar 10. Gambar Matriks *confusion*

Berdasarkan matriks *confusion* menunjukkan bahwa model mampu mengklasifikasikan berbagai jenis rambu lalu lintas dengan benar. Matriks ini dikodekan warna, dengan warna yang lebih gelap menunjukkan nilai yang lebih tinggi; tampaknya mewakili frekuensi prediksi. Bilah warna di sebelah kanan menunjukkan nilai dari 0 hingga 50. Elemen diagonal dari kiri atas ke kanan bawah berwarna gelap, menunjukkan prediksi yang benar oleh model. Oleh karena itu, hasilnya menunjukkan bahwa model telah berhasil dalam tugas klasifikasi rambu lalu lintas.



Gambar 11. Grafik *loss*, matriks *training* dan validasi model

Selama proses pelatihan dan validasi, grafik *loss* (termasuk *box loss*, *object loss*, dan *class loss*) dan grafik metrik (seperti *precision*, *recall*, mAP@0.5 atau mAP50, dan mAP@0.5:0.95 atau mAP50-95) ditampilkan. Seperti yang ditunjukkan pada Gambar 13, grafik tersebut menunjukkan penurunan yang konsisten dalam tingkat loss dan peningkatan yang stabil dalam nilai metrik selama 100 epoch. Dalam hal ini, metrik keseluruhan untuk 30 label rambu lalu lintas, model mencapai precision sebesar 0,993, recall sebesar 0,999, mAP50 sebesar 0,995, dan mAP50-95 sebesar 0,984. Ini menunjukkan kinerja yang baik dari model dalam mengklasifikasikan berbagai jenis rambu lalu lintas.

KESIMPULAN

Dalam penelitian ini, pengenalan jenis rambu lalu lintas yang mencakup 30 label berbeda berhasil dilakukan dengan menggunakan model YOLOv8 . Hasil pengujian menunjukkan bahwa model ini memiliki kinerja yang cukup baik, dengan nilai *Precision* sebesar 0,993, *Recall* sebesar 0,999, mAP50 sebesar 0,995, dan mAP50-95 sebesar 0,984. Nilai-nilai ini menunjukkan bahwa model mampu mengidentifikasi dan mengklasifikasikan rambu lalu lintas dengan akurasi yang tinggi. Meskipun demikian, masih ada ruang untuk peningkatan. Salah satu cara untuk meningkatkan kinerja model adalah dengan menambahkan lebih banyak data *training* dan data *testing* untuk setiap label. Dengan data yang lebih banyak, model akan memiliki lebih banyak kesempatan untuk belajar dan memahami berbagai jenis rambu lalu lintas, yang pada akhirnya dapat meningkatkan akurasi dan kinerja model secara keseluruhan. Oleh karena itu, penelitian ini menunjukkan potensi yang signifikan dalam penggunaan model YOLOv8 untuk pengenalan rambu lalu lintas, dan menunjukkan arah yang menjanjikan untuk penelitian dan pengembangan lebih lanjut dalam bidang ini.

REFERENSI

Arief R. (2021). *Tesis Sistem Deteksi Dan Pengenalan Rambu Lalu Lintas Di Indonesia Menggunakan Algoritma*

Yolov4. Universutas Hasanudin, Makassar.

Fahim, S. H., Abdal, A., Rasel, S., Sarker, A. R., and Chowdhury, T. (2024). Bangladeshi Vehicle Identification via YOLO v8-Based License Plate Detection. *Cambridge Open Engage*. https://doi.org/10.33774/coe-2024-5k5w6

Gong, C., Li, A., Song, Y., Xu, N., and He, W. (2022). Traffic Sign Recognition Based on the YOLOv3 Algorithm. *Sensors*, *22*(23). https://doi.org/10.3390/s22239345

Gregorius, N., Goenawan, A. N., and Tjondrowiguno, L. (2022). Pengenalan Rambu Lalu Lintas di Indonesia Secara Real-time Menggunakan YOLOv4-tiny. *Jurnal Infra*, *10*.

Janapriya, A. (2023). Pengenalan Jenis Rambu Lalu Lintas menggunakan Metode YOLO V5. *Jurnal Elektronik Ilmu Komputer Udayana* , *11*(4), 2654–5101. https://doi.org/10.24843/JLK.2023.v11.i04.p32

Liu, Y., Shi, G., Li, Y., and Zhao, Z. (2022). M-YOLO: Traffic Sign Detection Algorithm Applicable to Complex Scenarios. *Symmetry*, *14*(5). https://doi.org/10.3390/sym14050952

Munawar, M. (2023, January 10). Train YOLOv8 on Custom Data? Retrieved January 7, 2024, from https://medium.com/

Nugroho, A., and Cahyono, M. R. A. (2022). Implementasi Object Recognition Pada Rambu-Rambu Dan Lampu Lalu Lintas Dengan Raspberry Pi Dengan Algoritma Yolov5. *Sebatik*, *26*(2), 549–556. https://doi.org/10.46984/sebatik.v26i2.2047

Oklilas, A. F., Sukemi, S., and Apriliyanto, R. (2023). Model Yolo Versi 4 Pada Pengenalan Kendaraan Di Jalan Raya Kota Palembang. *Transmisi: Jurnal Ilmiah Teknik Elektro*, *25*(3), 136–139. https://doi.org/10.14710/transmisi.25.3.136-139

Raza M. (2023, October 16). Yolo V8: A Deep Dive Into Its Advanced Functions and New Features. Retrieved January 7, 2024, from https://medium.com/

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). *You Only Look Once: Unified, Real-Time Object Detection*. Retrieved from http://pjreddie.com/yolo/

Shahab, M. (2023, October 8). YOLO v8. Retrieved January 7, 2024, from https://medium.com/

Shen, L., Lang, B., and Song, Z. (2023). DS-YOLOv8-Based Object Detection Method for Remote Sensing Images. *IEEE Access*, *11*, 125122–125137.

Takarob R. (2022). *Deteksi Dan Identifikasi Rambu-Rambu Lalu Lintas Berbasis Algoritma You Only Look Once (Yolo)*. Jakarta.

Xuan-Ha Nguyen, T.-T. N. D.-A. N. (2022). Development of Real-Time Traffic-Object and Traffic-Sign Detection Models Applied for Autonomous Intelligent Vehicles. *JST: Smart Systems and Devices*, *32*(1), 17–24. https://doi.org/10.51316/jst.155.ssad.2022.32.1.3

Zhang, R., Zheng, K., Shi, P., Mei, Y., Li, H., and Qiu, T. (2023). Traffic Sign Detection Based on the Improved YOLOv5. *Applied Sciences (Switzerland)*, *13*(17). https://doi.org/10.3390/app13179748