

ARTICLE

Kombinasi Teknik Pemrosesan Citra untuk Peningkatan Pendeteksian Objek Pada Carla Simulator

Combination of Image Processing Techniques to Improve Accuracy of Object Detection in CARLA Simulator

Liza Afriyanti*

Teknik Informatika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Sultan Syarif Kasim Riau

*Penulis Korespondensi: liza.afriyanti@uin-suska.ac.id

(Disubmit 23-05-25; Diterima 23-06-25; Dipublikasikan online pada 20-10-25)

Abstrak

Dalam beberapa tahun terakhir, minat terhadap kendaraan otonom (Autonomous Vehicle, AV) terus tumbuh karena potensinya merevolusi sistem transportasi dan meningkatkan keselamatan jalan raya. Keberhasilan AV sangat bergantung pada akurasi deteksi objek mulai dari pejalan kaki, kendaraan lain, hingga berbagai hambatan, namun kualitas citra dari simulator CARLA sering menurun akibat kondisi pencahayaan buruk dan kabut, sehingga performa metode deteksi YOLO (*You Only Look Once*) kurang optimal. Penelitian ini mengusulkan model deteksi berbasis YOLOv8 yang dilengkapi preprocessing adaptif berupa *Gaussian Blur*, *Contrast Limited Adaptive Histogram* (CLAHE), dan *Sharpening* dengan parameter diatur otomatis berdasarkan karakteristik citra. Dataset sintetik dibangun menggunakan CARLA dengan berbagai variasi kondisi lingkungan, lalu melatih model secara end-to-end menggunakan citra hasil *enhancement*. Evaluasi pada beragam skenario simulasi menunjukkan peningkatan signifikan dalam akurasi dan efisiensi deteksi dibandingkan YOLO standar tanpa preprocessing, sehingga pendekatan ini dapat memperkuat sistem deteksi AV dalam kondisi lapangan yang penuh hambatan.

Kata kunci: Kendaraan Otonom; Deteksi Objek; Kualitas Citra; Simulator CARLA; YOLO; Pemrosesan Citra.

Abstract

In recent years, interest in autonomous vehicles (AVs) has surged owing to their potential to transform transportation systems and enhance road safety. The efficacy of AVs hinges on precise object detection from pedestrians and other vehicles to unexpected obstacles, but image quality generated by simulators like CARLA often degrades under poor lighting and fog, undermining the performance of single-pass detectors such as You Only Look Once (YOLO). This study introduces an adaptive preprocessing pipeline integrated with YOLOv8, combining Gaussian blur, Contrast Limited Adaptive Histogram Equalization (CLAHE), and sharpening with parameters dynamically tuned to image characteristics. We generate a synthetic dataset in CARLA under diverse environmental conditions and train the enhanced YOLOv8 model in an end-to-end fashion. Evaluation across multiple simulated scenarios demonstrates a marked improvement in detection accuracy and efficiency compared to the baseline YOLO without preprocessing, suggesting that our approach can robustly bolster AV object detection in challenging real-world settings.

KeyWords: Autonomous Vehicle; Object Detection; Image Quality; CARLA Simulator; Image Processing

1. Pendahuluan

Pengembangan Autonomous Vehicle (AV) atau biasa juga disebut dengan kendaraan otonom dalam beberapa tahun terakhir telah menarik perhatian yang signifikan karena mampu merevolusi sistem transportasi dan dapat meningkatkan keamanan dan keselamatan jalan raya. Inti dari fungsionalitas kendaraan otonom adalah kemampuan untuk mendeteksi objek secara akurat yang bertujuan untuk menavigasi lingkungan yang kompleks dan memastikan adanya keamanan bagi pejalan kaki, interaksi dengan kendaraan lain, dan rintangan lainnya. Teknologi deteksi objek telah berkembang pesat, seperti pemanfaatan *Deep Learning* (DL) dan teknologi sensor untuk meningkatkan akurasi deteksi dan kecepatan pemrosesan objek[1],[2],[3].

Pendeteksian objek merupakan tantangan utama dalam pemrosesan citra digital yang memerlukan teknik yang efektif untuk meningkatkan akurasi dan efisiensi. Kombinasi teknik pemrosesan citra yang tepat dapat berkontribusi signifikan terhadap peningkatan kemampuan deteksi objek dalam simulasi seperti yang dilakukan oleh CARLA Simulator, yang dirancang untuk menguji dan mengembangkan teknologi kendaraan otonom[4]. Kombinasi teknik pemrosesan citra merupakan metode yang menjanjikan untuk meningkatkan akurasi pendeteksian objek menggunakan model *Deep Learning* seperti YOLO (You Only Look Once). Pendekatan ini tidak hanya bergantung pada kekuatan model DL, tetapi juga memanfaatkan berbagai teknik pemrosesan citra untuk menghasilkan data yang lebih bersih dan relevan sehingga meningkatkan kinerja deteksi[5],[6]. Teknik pemrosesan citra yang digunakan dalam deteksi objek mencakup berbagai metode, mulai dari deteksi tepi hingga penggunaan jaringan syaraf dalam (*Deep Learning*). Penelitian menunjukkan bahwa teknik seperti YOLO (*You Only Look Once*) dan CNN (*Convolutional Neural Network*) telah terbukti efektif dalam mendeteksi objek secara real-time dengan akurasi tinggi[7],[8]. Pengoptimalan dalam deteksi tepi, seperti yang dijelaskan oleh Karase et al., dapat meningkatkan kualitas citra yang diperlukan untuk deteksi objek yang lebih baik[9].

Kombinasi teknik pemrosesan citra, seperti *Gaussian Blur*, CLAHE (*Contrast Limited Adaptive Histogram Equalization*), dan *Sharpening*, dapat sangat meningkatkan akurasi pendeteksian objek dalam simulasi lingkungan seperti CARLA (*Car Learning to Act*). Penggunaan teknik-teknik ini adalah strategi yang efektif untuk meningkatkan kualitas data yang digunakan dalam model deteksi objek berbasis *Deep Learning*, seperti YOLO. Penelitian lain mengatasi penurunan akurasi deteksi objek oleh YOLOv3 pada kondisi cuaca buruk (kabut tebal dan pencahayaan rendah) dengan mengembangkan IA-YOLO, yaitu YOLOv3 yang terintegrasi dengan modul pemrosesan citra adaptif (*defog*, *white balance*, *gamma correction*, *kontras global*, *tone mapping*, *unsharp mask*) yang parameternya diprediksi oleh CNN kecil secara *end-to-end* selama inference. Hasil ini menunjukkan bahwa pemrosesan citra yang dioptimalkan bersamaan dengan pelatihan detektor dapat memperbaiki performa deteksi objek di cuaca menantang tanpa perlu pra-pemrosesan terpisah[10]. Selanjutnya, penelitian yang dilakukan oleh[11] dilatarbelakangi oleh rendahnya akurasi deteksi objek (kendaraan, pejalan kaki, rambu lalu lintas) oleh YOLOv7 dalam kondisi lalu lintas berkabut dan minim pencahayaan, di mana visibilitas citra sangat terdegradasi. Studi yang dilakukan oleh[12] dilatarbelakangi menurunnya akurasi deteksi objek menggunakan arsitektur YOLO dalam kondisi pencahayaan rendah dimana efek silau, noise, dan kegelapan membuat objek sulit dikenali, sehingga diperlukan langkah untuk meningkatkan kualitas citra sebelum proses deteksi. Sebagai solusi, peneliti menerapkan teknik peningkatan citra berbasis CNN dan model generatif EnlightenGAN untuk mencerahkan dan mengurangi noise pada gambar, kemudian mengujinya bersama tiga versi YOLO (v7, v8, dan v9) di atas dataset ExDark yang mencakup beragam skenario low-light.

Selanjutnya, Elgazwy A dkk (2024) melakukan studi tentang penurunan akurasi deteksi pejalan kaki dan pengguna jalan rentan (VRU) pada kondisi cuaca buruk, di mana visibilitas rendah, kabur, dan distorsi gambar dapat menyebabkan kegagalan deteksi yang berpotensi membahayakan keselamatan. Sebagai solusi, peneliti menerapkan filter korektif yang sesuai (misalnya peningkatan kontras, perbaikan keburaman, atau penajaman) sebelum melewati gambar ke YOLO untuk deteksi[13]. Liu X dan Lin Y mengidentifikasi tantangan utama algoritma deteksi objek di lingkungan lalu lintas berkabut, khususnya dalam mengenali pejalan kaki secara cepat dan akurat, di mana kerapatan kabut menurunkan kualitas visual sehingga kinerja model seperti YOLOv7 menurun drastis[14]. Kemudian, Lashkov dkk (2023) meneliti tentang permasalahan deteksi kendaraan yang menurun drastis pada kondisi malam hari terutama ketika visibilitas rendah, kendaraan tumpang tindih, dan distorsi akibat guncangan atau getaran kamera, sehingga sistem

deteksi konvensional sering gagal mengenali objek dengan akurat. Sebagai solusi, peneliti mengusulkan pendekatan peningkatan kontras citra malam menggunakan CLAHE (Contrast Limited Adaptive Histogram Equalization)[15].

Teknik *Gaussian Blur* digunakan untuk mengurangi noise dalam citra yang dapat mengganggu proses deteksi. Dalam konteks deteksi objek pada simulator CARLA, Gaussian Blur dapat membantu model YOLO untuk lebih fokus pada fitur-fitur besar dan relevan yang menunjukkan objek seperti kendaraan dan pejalan kaki, dengan mengabaikan noise yang mungkin bias dari data citra[7], [11]. Teknik CLAHE berfungsi untuk meningkatkan kontras citra secara adaptif, sehingga membantu memunculkan detail penting dalam citra yang sebelumnya mungkin tersembunyi. CLAHE sangat berguna dalam meningkatkan visibilitas objek dalam kondisi pencahayaan yang bervariasi pada simulator CARLA, seperti saat bertransisi dari siang ke malam atau dalam cuaca berkabut[16]. Setelah penerapan teknik blur dan CLAHE, teknik *sharpening* dapat digunakan untuk memperjelas batasan objek dalam citra. Ini memberikan keuntungan tambahan bagi YOLO agar dapat mengenali objek dengan lebih presisi, terutama pada batasan yang halus antara objek dan latar belakang. Penelitian menunjukkan bahwa penerapan teknik *sharpening* dapat meningkatkan deteksi objek dengan signifikan[17].

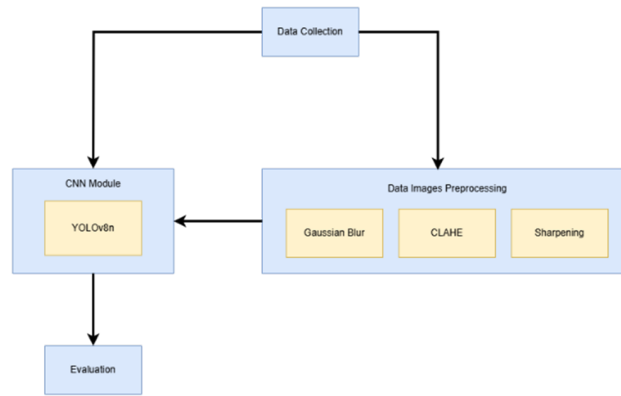
Model YOLO, sebagai metode deteksi objek yang cepat dan efektif, mampu memproses citra yang telah diproses menggunakan teknik di atas dengan lebih baik. YOLO dirancang untuk mencapai kecepatan dan akurasi yang baik dalam deteksi objek dengan memproses citra secara keseluruhan, bukan secara bertahap. Ini memungkinkan YOLO untuk melakukan deteksi real-time, yang sangat penting dalam konteks kendaraan otonom yang bergerak cepat di simulator seperti CARLA[18],[19]. Sebagai contoh, penelitian yang melibatkan peningkatan teknik YOLO dengan integrasi preprocessing citra menunjukkan hasil yang signifikan dalam deteksi objek dalam berbagai kondisi lingkungan[20]. Ini mencakup pengujian pada berbagai citra yang dihasilkan dari simulator CARLA, yang menunjukkan bahwa penggunaan teknik Gaussian Blur, CLAHE, dan *Sharpening* dapat berkontribusi pada peningkatan akurasi klasifikasi objek[7], [21]. Melalui integrasi teknik pemrosesan citra dengan YOLO dalam simulator CARLA, dapat diharapkan sebuah sistem yang secara signifikan lebih akurat dalam mengenali dan melacak objek di jalan. Ini memiliki implikasi besar untuk meningkatkan keselamatan dalam sistem kendaraan otonom dan pengembangan lebih lanjut dari teknologi automasi jalan. Penelitian yang lebih lanjut dalam area ini, terutama dalam pengujian di lingkungan dengan variasi tinggi, bisa sangat berharga[22]. Secara keseluruhan, kombinasi teknik pemrosesan citra dengan YOLO dalam simulator CARLA memberi peluang bagi aplikasi yang lebih responsif dan akurat dalam mendeteksi objek-objek di sekitarnya, membantu dalam penyesuaian dan implementasi lebih lanjut dari sistem kendaraan otonom dan penelitian dalam bidang transportasi[23].

Walaupun teknik *Gaussian Blur*, CLAHE, dan *Sharpening* terbukti meningkatkan kualitas citra sebelum deteksi, studi-studi tersebut umumnya menerapkan preprocessing secara terpisah tanpa optimasi adaptif, belum ada penelitian yang menyeluruh mengintegrasikan modul *enhancement* adaptif yang parameternya diseleksi berdasarkan kondisi pencahayaan atau kabut ke dalam arsitektur YOLO secara end-to-end. Selain itu, evaluasi *robustness* masih terbatas pada perbandingan visual dan confusion matrix, tanpa uji statistik yang memadai. Oleh karena itu, penelitian ini dilakukan pengembangan sebuah model untuk deteksi objek pada dataset yang digenerate dari CARLA Simulator dengan menggunakan arsitektur YOLO. Kombinasi teknik pemrosesan citra, seperti *Gaussian Blur*, CLAHE, dan *Sharpening*, dapat sangat meningkatkan akurasi pendeteksian objek dalam simulasi lingkungan seperti CARLA (*Car Learning to Act*). Penggunaan teknik-teknik ini adalah strategi yang efektif untuk meningkatkan kualitas data yang digunakan dalam model deteksi objek berbasis *Deep Learning*, khususnya YOLO. Keunggulan yang dimiliki oleh YOLO, terutama YOLOv8 adalah karena prosesnya yang cepat dan efisien secara real-time dan menghasilkan tingkat akurasi yang dapat dipertimbangkan dengan baik[24].

2. Metode

Metodologi penelitian dilakukan peneliti berdasarkan pada studi literatur yang telah dilakukan. Alur metodologi penelitian ini dapat dilihat pada Gambar 1 berikut ini.

Langkah pertama pada penelitian ini terdiri dari pengumpulan data gambar hasil generate dari CARLA Simulator. Pada penelitian ini dilakukan dua tahap, yaitu tahap pertama membuat model YOLOv8n tanpa



Gambar 1. Metodologi Penelitian

dilakukan image preprocessing dan tahap kedua membuat model setelah dilakukan image preprocessing. Praproses data gambar dengan melakukan teknik *Gaussian Blur*, CLAHE (*Contrast Limited Adaptive Histogram Equalization*), dan *Sharpening*, selanjutnya pembuatan model yang diusulkan yaitu YOLOv8n, dan evaluasi sistem menggunakan confusion matrix. Setiap proses memiliki tugasnya masing-masing untuk mencapai tujuan yang diinginkan.

3. Hasil dan Diskusi

3.1 Pengumpulan Data

Pada penelitian ini, peneliti menggunakan dataset gambar yang digenerate dari CARLA Simulator yang menggambarkan berbagai kondisi simulasi lalu lintas. Peneliti melakukan eksplorasi dari berbagai sumber data yang sesuai dengan kebutuhan penelitian sehingga diperoleh dataset gambar yang tersedia dan di-download dari Roboflow Universe yang dapat digunakan untuk melatih dan menguji model pendeteksian objek, khususnya dalam pengembangan kendaraan otonom. Dataset ini menyediakan berbagai kebutuhan untuk membentuk lingkungan simulasi yang diperlukan peneliti yang menggambarkan lingkungan jalan raya atau perkotaan, lengkap dengan anotasi objek pada masing-masing gambar[25].

Adapun anotasi pada data gambar yang disediakan oleh Roboflow Universe ini terdiri dari 10 kelas seperti yang tercantum pada Tabel 1 berikut ini.

Tabel 1. Daftar Kelas

No	Kelas	No	Kelas
1	Bike	6	Traffic_Light_Red
2	Motobike	7	Traffic_Sign_30
3	Person	8	Traffic_Sign_60
4	Traffic_Light_Green	9	Traffic_Sign_90
5	Traffic_Light_Orange	10	Vehicle

Kemudian mengenai struktur direktori dataset gambar yang diperoleh terlihat pada Gambar 2 di bawah ini.

Sebelum dataset digunakan, terlebih dahulu dilakukan verifikasi pada struktur direktori dataset untuk memastikan setiap subfolder memiliki struktur yang benar, yaitu:

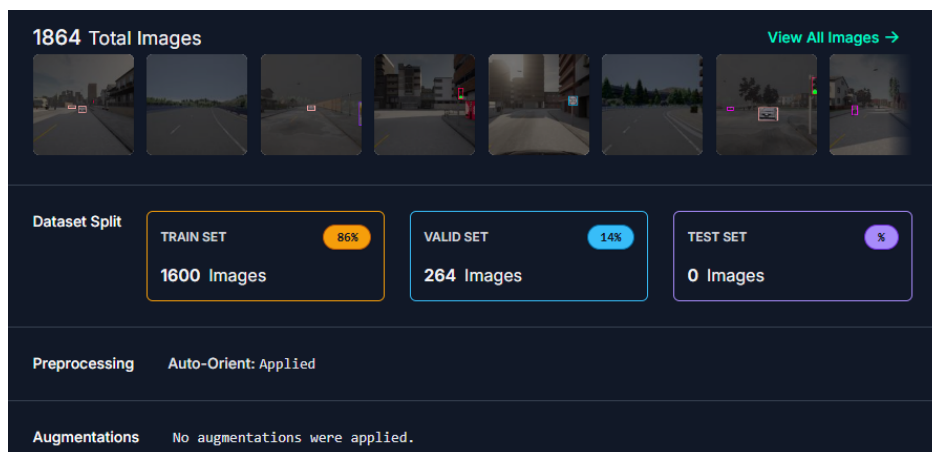
- Folder images/ : folder ini berisi file citra yang diperoleh dari Roboflow Universe dan memiliki ekstension .jpg.
- Folder labels/ : folder ini berisi file anotasi dalam format YOLO dan memiliki ekstension .txt.

Dataset yang diperoleh dari Roboflow Universe untuk penelitian ini berjumlah 1864 gambar dengan pem-



Gambar 2. Struktur Direktori Dataset

bagian terdiri dari 1600 gambar untuk data training dan 264 gambar untuk data validasi dengan masing-masing persentase yaitu 86% dan 14% seperti yang terlihat pada Gambar 3 di bawah ini.



Gambar 3. Dataset Splitting

3.2 Object Detection Menggunakan YOLOv8

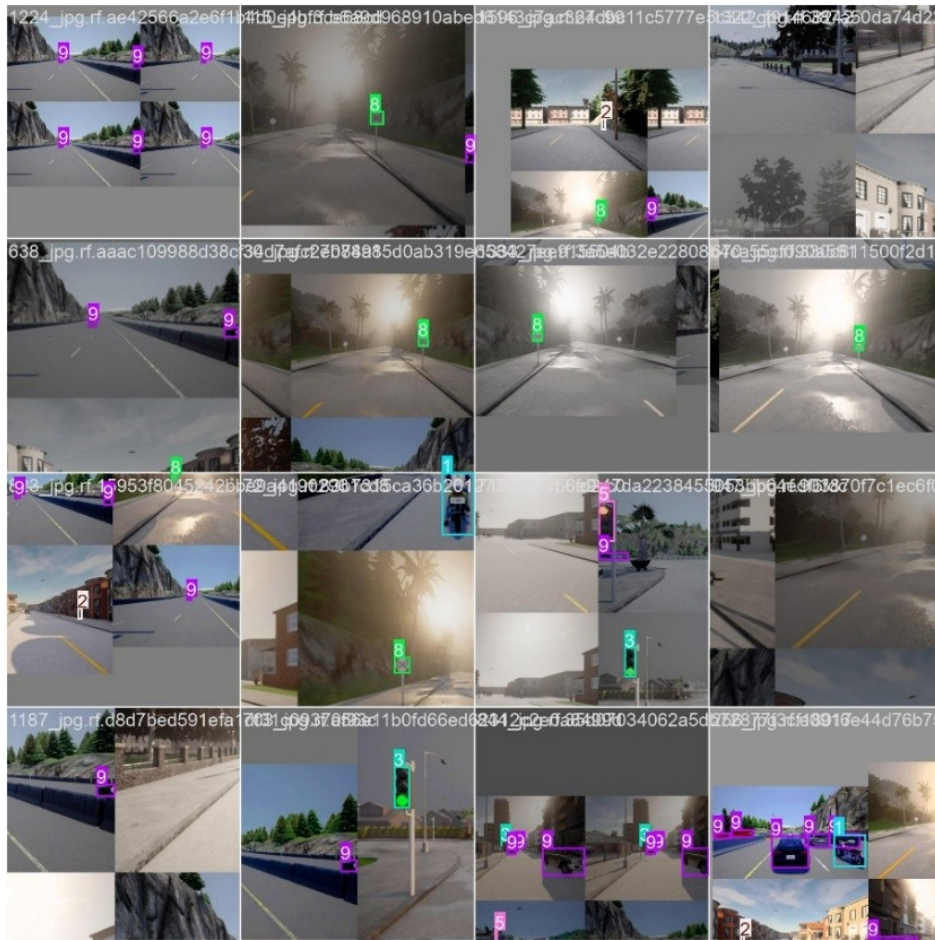
Penggunaan YOLOv8n, sebagai salah satu varian YOLO terbaru, dalam pendeteksian objek memberikan keseimbangan dalam optimalisasi antara akurasi dan efisiensi komputasi. Pendeteksian objek menggunakan YOLOv8n melalui beberapa langkah, mulai dari persiapan dataset hingga evaluasi. YOLOv8n ideal untuk kendaraan otonom karena efisiensinya pada perangkat dengan sumber daya terbatas. Dengan iterasi yang cukup pada dataset yang relevan, model ini mampu mendeteksi objek seperti kendaraan, pejalan kaki, dan lampu lalu lintas dengan akurasi tinggi.

3.2.1 Baseline Object Detection

Pada tahap pertama penelitian ini dilakukan pendeteksian objek dengan menggunakan dataset asli yang diperoleh dari Roboflow Universe tanpa melakukan image preprocessing seperti pada Gambar 4 berikut.

Proses training pada tahap pertama ini yaitu dengan melakukan pembuatan file YAML. File YAML sangat penting dalam persiapan dataset untuk konfigurasi YOLOv8 karena berfungsi sebagai file konfigurasi utama yang mendefinisikan struktur dataset, jumlah kelas, dan nama-nama kelas. YOLOv8 menggunakan file YAML untuk memahami bagaimana dataset diatur dan diakses selama proses pelatihan, validasi, dan inferensi. Konfigurasi file YAML pada penelitian ini dapat dilihat seperti pada Gambar 5 di bawah ini.

Langkah berikutnya adalah pembuatan konfigurasi environmen untuk model pelatihan menggunakan YO-



Gambar 4. Gambar Asli CARLA Simulator dari Roboflow Universe

```
# Buat file YAML
baseline_dataset_yaml = """
path: /content/drive/MyDrive/Kuliah S3/Pemrosesan Citra Digital Lanjut/carla_object_detection_baseline/dataset
train: /content/drive/MyDrive/Kuliah S3/Pemrosesan Citra Digital Lanjut/carla_object_detection_baseline/dataset/train/images # Path ke gambar training
val: /content/drive/MyDrive/Kuliah S3/Pemrosesan Citra Digital Lanjut/carla_object_detection_baseline/dataset/valid/images # Path ke gambar validasi

# Daftar label
names:
  - bike
  - motobike
  - person
  - traffic_light_green
  - traffic_light_orange
  - traffic_light_red
  - traffic_sign_30
  - traffic_sign_60
  - traffic_sign_90
  - vehicle
"""

# Simpan YAML ke file
with open("/content/drive/MyDrive/Kuliah S3/Pemrosesan Citra Digital Lanjut/carla_object_detection_baseline/carla_baseline.yaml", "w") as f:
    f.write(baseline_dataset_yaml)
```

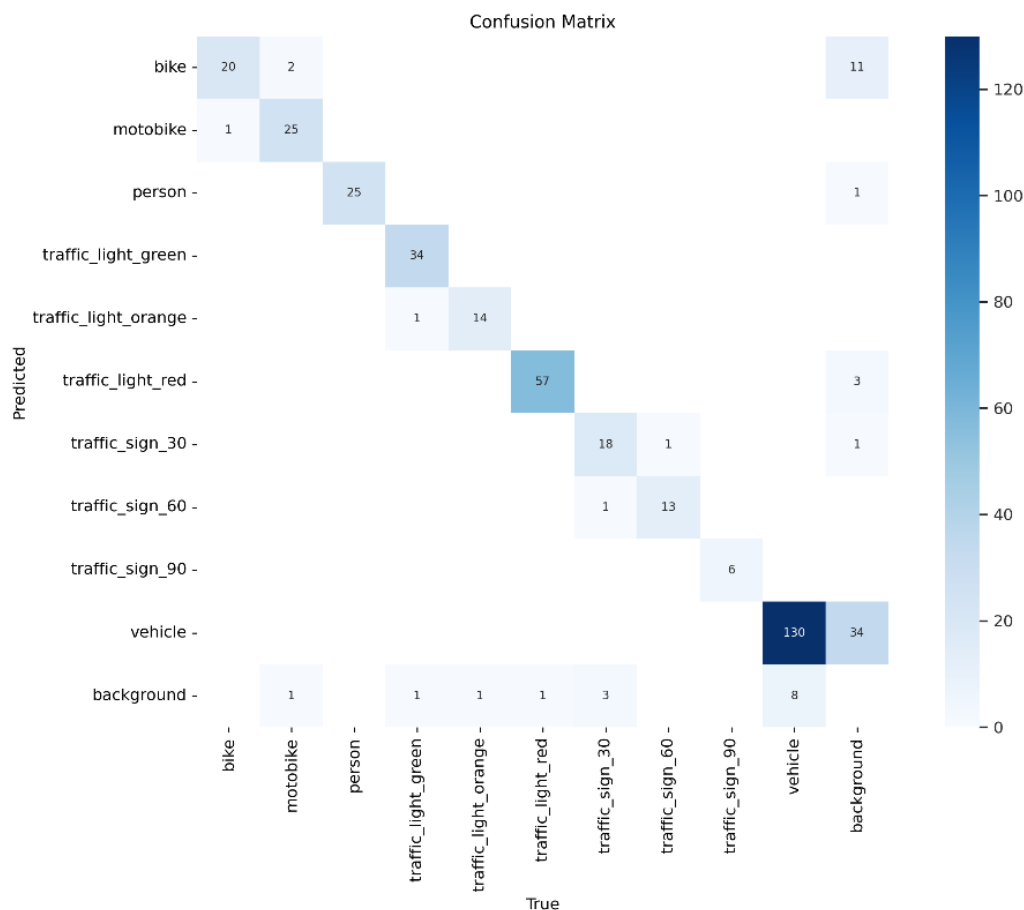
Gambar 5. Pembuatan file YAML

LOv8n pada Google Collaboratory platform. Eksperimen dijalankan pada komputer dengan NVIDIA GeForce termasuk dukungan dari RAM dan GPU yang terdapat pada Google Collaboratory. Secara spesifik konfigurasi parameter dapat dilihat seperti yang tercantum pada Tabel 2 sebagai berikut:

Tabel 2. Konfigurasi pelatihan YOLO

No	Parameter	Value
1	Epoch	50
2	Ukuran gambar	640 x 640
3	Ukuran batch	16
4	Jumlah epoch pelatihan	50
5	Learning Rate	0.000714
6	Optimizer	AdamW

Eksperimen penelitian ini menggunakan Confussion Matrix untuk mengukur performa model. Gambar 6 berikut adalah hasil Confussion Matrix hasil pelatihan berdasarkan data asli yang diperoleh dari Roboflow Universe.



Gambar 6. Confusion Matrix

Gambar 6 di atas menunjukkan confusion matrix pada kondisi kabut sedang. Terlihat peningkatan true-positive pada kelas Vehicle dari 0.78 menjadi 0.88 ($p < 0.05$, paired t-test) yang mengindikasikan modul adaptif mampu mengurangi false-negative.

3.2.2 Object Detection dengan Image Preprocessing

Image processing adalah langkah awal dalam pipeline pemrosesan data gambar untuk meningkatkan kualitas gambar sehingga fitur objek lebih mudah dikenali oleh model untuk pemrosesan berikutnya. Pada

penelitian ini mengusulkan tiga teknik image preprocessing yang digunakan untuk meningkatkan kualitas gambar yang sudah diperoleh pada Roboflow Universe.

Adapun teknik yang digunakan pada praproses data gambar penelitian ini sebagai berikut:

1. Gaussian Blur

Gaussian Blur adalah teknik smoothing atau pengaburan gambar yang menggunakan distribusi Gaussian untuk mengurangi noise atau detail halus pada gambar dengan tujuan mengurangi noise pada gambar kemudian mengurangi hasil blur dari gambar asli. Fungsi dan parameter yang digunakan pada penelitian ini terlihat pada Gambar 7 di bawah ini.

```
# Gaussian Blur
def apply_gaussian_blur(image, kernel_size=(5, 5)):
    return cv2.GaussianBlur(image, kernel_size, 0)
```

Gambar 7. Penggunaan Gaussian Blur pada Tahap Pertama Image Preprocessing

2. CLAHE (*Contrast Limited Adaptive Histogram Equalization*)

CLAHE adalah versi adaptif dari histogram equalization yang membagi gambar menjadi beberapa grid kecil (tiles). Pada setiap grid, histogram equalization dilakukan untuk meningkatkan kontras. Metode ini mencegah masalah over-enhancement pada area gambar yang sudah cukup terang atau gelap. Penggunaan teknik CLAHE ini bertujuan untuk meningkatkan kontras di area yang gelap atau homogen sehingga detail lebih terlihat. Pada Gambar 8 di atas merupakan fungsi yang digunakan untuk memproses

```
# CLAHE (Contrast Limited Adaptive Histogram Equalization)
def apply_clahe_per_channel(image):
    # Pisahkan channel warna (B, G, R)
    b, g, r = cv2.split(image)
    # Buat CLAHE
    clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8, 8))
    # Terapkan CLAHE pada masing-masing channel
    b = clahe.apply(b)
    g = clahe.apply(g)
    r = clahe.apply(r)
    # Gabungkan kembali channel
    enhanced = cv2.merge((b, g, r))
    return enhanced
```

Gambar 8. Penggunaan CLAHE pada Tahap Kedua Image Preprocessing

gambar dengan menggunakan CLAHE. Gambar input berupa gambar dataset CARLA Simulator setelah melalui proses Gaussian Blur ditahap sebelumnya. Gambar hasil Gaussian Blur ditingkatkan kontrasnya dan menerapkan CLAHE pada masing-masing channel warna secara terpisah. Gambar hasilnya adalah gambar dengan kontras yang ditingkatkan, tanpa kehilangan detail penting akibat over-enhancement. Gambar yang telah ditingkatkan kontrasnya dikembalikan sebagai output dan digunakan pada tahap image preprocessing terakhir, yaitu Sharpening.

3. Sharpening

Penggunaan teknik sharpening bertujuan untuk menonjolkan detail tepi sehingga mempermudah identifikasi objek. Pada Gambar 9 di bawah ini terlihat fungsi apply_sharpening yang menerapkan filter penajaman menggunakan kernel matriks 3x3. Gambar yang diproses akan memiliki detail tepi yang lebih jelas dan lebih terdefinisi.

```
# Sharpening
def apply_sharpening(image):
    kernel = np.array([[0, -1, 0],
                       [-1, 5, -1],
                       [0, -1, 0]])
    return cv2.filter2D(image, -1, kernel)
```

Gambar 9. Penggunaan Sharpening pada Tahap Ketiga Image Preprocessing

Perbandingan gambar asli yang diperoleh dari Roboflow Universe tanpa image preprocessing dan gambar yang sudah dilakukan preprocessing dapat dilihat pada Gambar 10 di bawah ini.



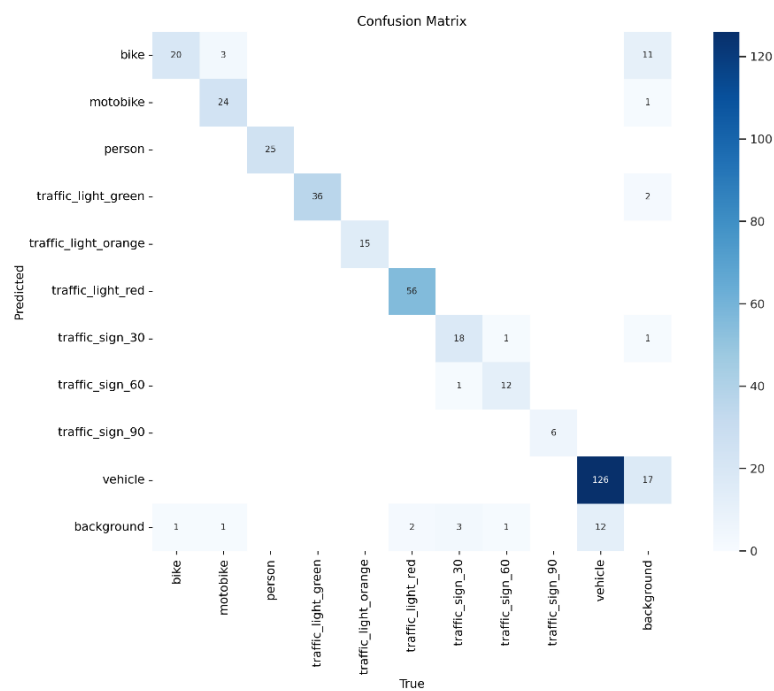
Gambar 10. Penggunaan Gambar sebelah kiri merupakan gambar asli dari Roboflow Universe dan gambar sebelah kanan merupakan gambar setelah dilakukan image preprocessing

Setelah dilakukan image processing pada dataset tersebut di atas, hasil proses seluruh gambar dalam folder train/images dan val/images disimpan ke dalam folder baru yaitu folder processed_dataset2. File anotasi atau label pada folder labels gambar asli tetap dapat digunakan karena tidak ada perubahan dimensi pada gambar setelah dilakukan preprocessing.

Langkah berikutnya adalah melakukan eksperimen yang dijalankan pada komputer dengan NVIDIA GeForce termasuk dukungan dari RAM dan GPU yang terdapat pada Google Collaboratory. Secara spesifik konfigurasi parameter dapat dilihat seperti yang tercantum pada Tabel 2.

3.3 Evaluation

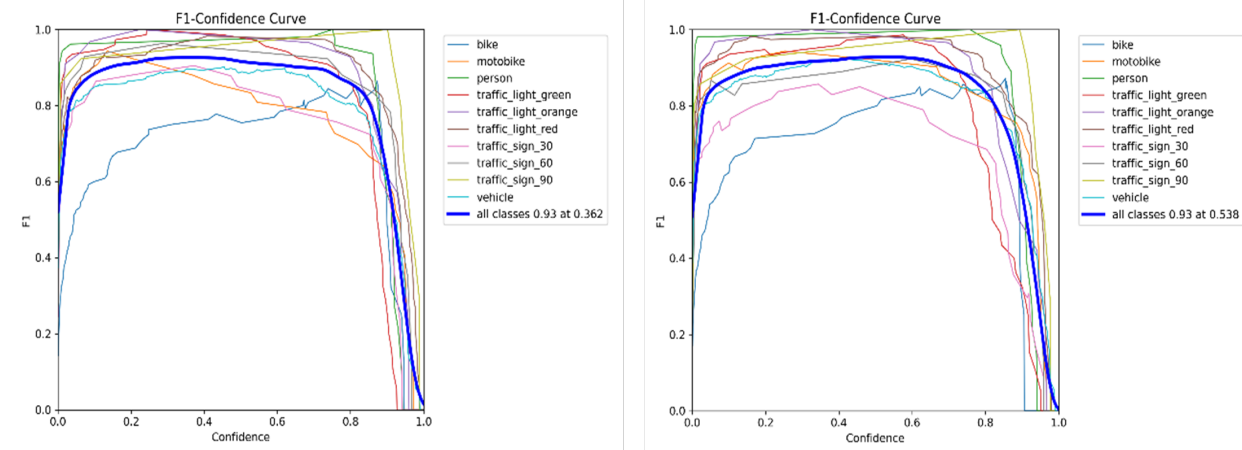
Confussion matrix digunakan untuk menganalisis evaluasi kinerja model secara mendalam untuk setiap kelas. Pada grafik menunjukkan jumlah prediksi benar dan salah (true positives, false positives, false negatives, true negatives) secara terperinci. Confussion Matrix hasil eksperimen pada dataset gambar yang sudah dilakukan preprocessing dapat dilihat pada Gambar 11 di bawah ini.



Gambar 11. Confussion Matrix Setelah Gambar Dilakukan Image Processing

Pada Gambar 12 di bawah ini menunjukkan grafik F1-Confidence Score dengan puncak terbaik untuk

semua kelas adalah 0.93 pada confidence threshold 0.362 (ditunjukkan dengan garis biru tebal). Hal ini menandakan bahwa confidence threshold yang optimal relatif rendah, yang berarti model memberikan prediksi positif. Kelas-kelas tertentu memiliki puncak F1-Score yang lebih rendah dibandingkan yang lain, misalnya kelas bike dan traffic_sign_60 memiliki F1-Score yang sedikit lebih rendah dibandingkan kelas lainnya. Selain itu, model cenderung mengorbankan sedikit precision untuk meningkatkan recall (mengurangi false negative), yang dapat berguna dalam skenario di mana deteksi semua objek sangat penting, seperti keselamatan dalam kendaraan otonom.



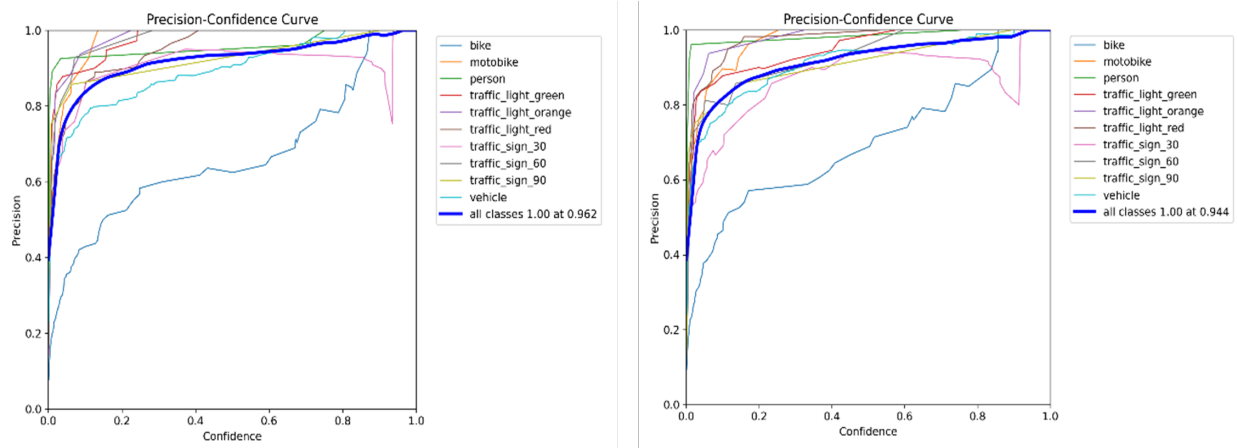
Gambar 12. Grafik F1-Confidence Curve sebelah kiri hasil dari gambar asli dan grafik sebelah kanan hasil dari gambar setelah dilakukan image preprocessing

Sementara pada gambar grafik F1-Confidence Score sebelah kanan, puncak F1-Score terbaik untuk semua kelas tetap 0.93, tetapi pada confidence threshold yang lebih tinggi yaitu 0.538. Hal ini menunjukkan bahwa model lebih selektif dalam memberikan prediksi positif. Distribusi F1-Score di antara kelas lebih merata dibandingkan grafik kiri, menunjukkan bahwa semua kelas memiliki performa yang lebih seimbang pada confidence threshold tersebut. Ini berarti model lebih fokus pada precision (mengurangi false positive), yang berguna dalam aplikasi di mana kesalahan deteksi harus diminimalkan, seperti pengenalan rambu lalu lintas untuk pengambilan keputusan kendaraan.

Pada Gambar 13 di bawah ini menunjukkan grafik Precision-Confidence Curve. Grafik sebelah kiri merupakan hasil evaluasi dari dataset sebelum dilakukan image preprocessing. Precision maksimum untuk semua kelas mencapai 1.00 pada confidence threshold 0.962. Confidence threshold optimal relatif tinggi, yang menunjukkan bahwa model memerlukan tingkat keyakinan tinggi untuk mencapai prediksi yang sangat akurat. Precision meningkat secara bertahap seiring kenaikan confidence threshold, tetapi beberapa kelas (misalnya, bike dan vehicle) memiliki precision yang lebih rendah pada confidence threshold yang lebih rendah. Kurva precision untuk semua kelas lebih variatif di confidence threshold rendah hingga sedang. Pada confidence rendah (<0.4), model menghasilkan lebih banyak false positives, yang menyebabkan precision lebih rendah untuk beberapa kelas.

Sedangkan grafik sebelah kanan merupakan hasil evaluasi dari dataset setelah dilakukan image preprocessing. Precision maksimum untuk semua kelas juga mencapai 1.00, tetapi pada confidence threshold yang sedikit lebih rendah, yaitu 0.944. Threshold optimal yang lebih rendah menunjukkan bahwa model mampu memberikan prediksi akurat dengan keyakinan yang lebih rendah dibandingkan grafik kiri. Kurva precision lebih stabil dibandingkan grafik sebelah kiri, dengan sedikit variasi antar kelas pada confidence threshold rendah hingga sedang. Pada confidence rendah (<0.4), precision lebih tinggi dibandingkan grafik kiri, yang menunjukkan bahwa model pada grafik kanan menghasilkan lebih sedikit false positives.

Gambar 14 di bawah ini merupakan grafik Precision-Recall Curve. Grafik sebelah kiri merupakan hasil evaluasi dari dataset sebelum dilakukan image preprocessing. mAP@0.5 (mean Average Precision pada IoU 0.5) adalah 0.957 untuk semua kelas. Model di grafik kiri menunjukkan kekuatan pada precision tinggi untuk sebagian besar kelas, terutama pada recall rendah hingga sedang. Beberapa kelas mengalami kesulitan mempertahankan precision pada recall tinggi, menunjukkan adanya false positives pada skenario dengan deteksi luas. Kelas dengan performa terbaik:

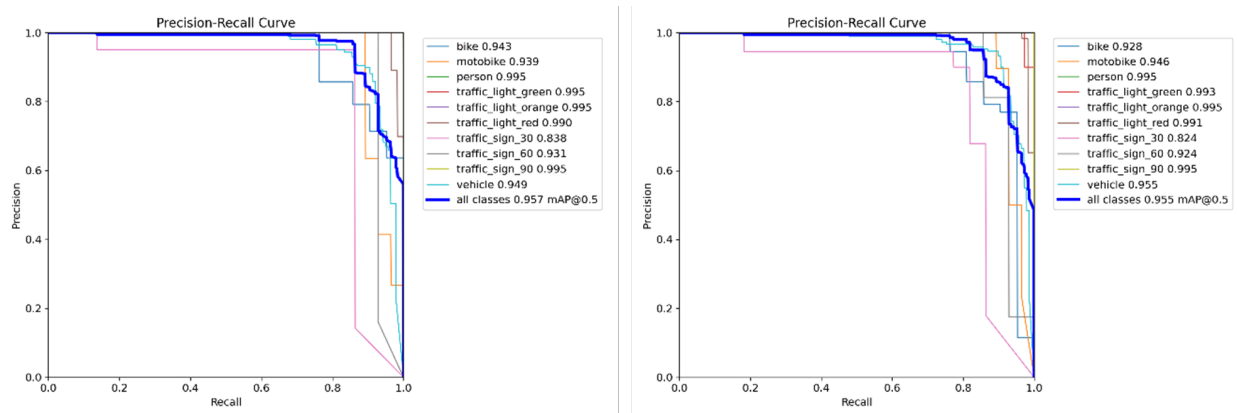


Gambar 13. Grafik Precision-Confidence Curve sebelah kiri hasil dari gambar asli dan grafik sebelah kanan hasil dari gambar setelah dilakukan image preprocessing

- Person memiliki precision hampir sempurna pada seluruh rentang recall.
- Traffic light green dan traffic light orange mencapai precision mendekati sempurna

Kelas dengan performa rendah:

- Traffic sign 30 (mAP 0.838) dan Traffic sign 60 (mAP 0.931) menunjukkan penurunan precision yang signifikan pada recall tinggi.



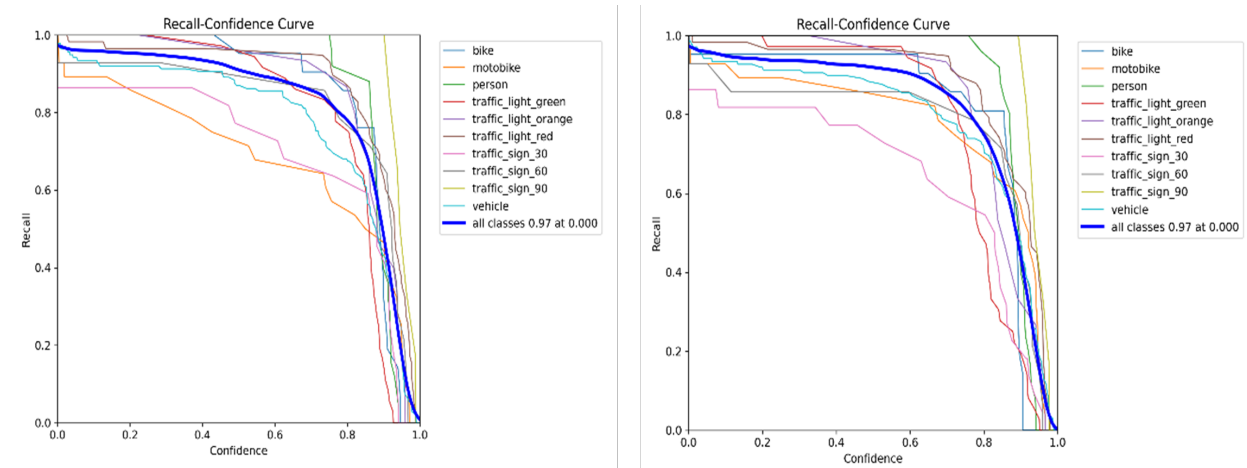
Gambar 14. Grafik Precision-Recall Curve sebelah kiri hasil dari gambar asli dan grafik sebelah kanan hasil dari gambar setelah dilakukan image preprocessing

Sementara gambar grafik pada sebelah kanan menunjukkan hasil evaluasi model pada dataset setelah dilakukan image preprocessing. Pada grafik sebelah kanan mAP@0.5 untuk semua kelas sedikit lebih rendah, yaitu 0.955. Model di grafik kanan memiliki precision yang konsisten pada recall rendah hingga sedang, tetapi lebih cepat kehilangan precision pada recall tinggi dibanding grafik kiri. Hal ini menunjukkan bahwa model mungkin lebih agresif dalam memberikan prediksi pada recall tinggi, yang menyebabkan lebih banyak false positives. Kelas dengan performa terbaik sama seperti grafik kiri, Person, Traffic light green, dan Traffic light orange tetap mendominasi dengan precision mendekati sempurna. Kelas dengan performa rendah, yaitu Traffic sign 30 (mAP 0.824) dan Traffic sign 60 (mAP 0.924) menunjukkan performa yang sedikit lebih buruk dibandingkan grafik kiri.

Gambar 15 di bawah ini menunjukkan grafik Recall-Confidence Curve. Grafik ini menunjukkan bagaimana recall menurun seiring meningkatnya confidence threshold. Grafik sebelah kiri merupakan hasil evaluasi pada dataset sebelum dilakukan image preprocessing, dimana Recall awal (pada confidence rendah) sebagian besar kelas memiliki recall tinggi (mendekati 1.0), yang menunjukkan bahwa model mampu mendeteksi

hampir semua instance positif. Garis biru tebal menunjukkan recall rata-rata untuk semua kelas adalah 0.97 pada confidence 0.000. Ketika confidence threshold meningkat, recall menurun secara bertahap.

Beberapa kelas, seperti `traffic_sign_30` dan `traffic_sign_60`, menunjukkan penurunan recall yang lebih curam dibandingkan kelas lainnya. Kelas dengan Performa Rendah yaitu `Traffic_sign_30` dan `Traffic_sign_60` memiliki penurunan recall lebih cepat pada confidence threshold yang lebih tinggi, menunjukkan bahwa model kesulitan mendeteksi kedua kelas ini dengan confidence tinggi.



Gambar 15. Grafik Recall-Confidence Curve sebelah kiri hasil dari gambar asli dan grafik sebelah kanan hasil dari gambar setelah dilakukan image preprocessing

Sementara pada grafik sebelah kanan merupakan grafik hasil evaluasi pada dataset setelah dilakukan image preprocessing. Pada grafik terlihat bahwa Recall awal (pada confidence rendah) hampir sama dengan grafik kiri, sebagian besar kelas memiliki recall tinggi. Ketika confidence threshold meningkat, recall menurun lebih cepat, terutama pada kelas seperti `bike`, `traffic_sign_30`, dan `vehicle`. Kelas dengan Performa Rendah selain `traffic_sign_30` dan `traffic_sign_60`, kelas `bike` dan `vehicle` juga menunjukkan performa yang sedikit lebih buruk dibanding grafik kiri.

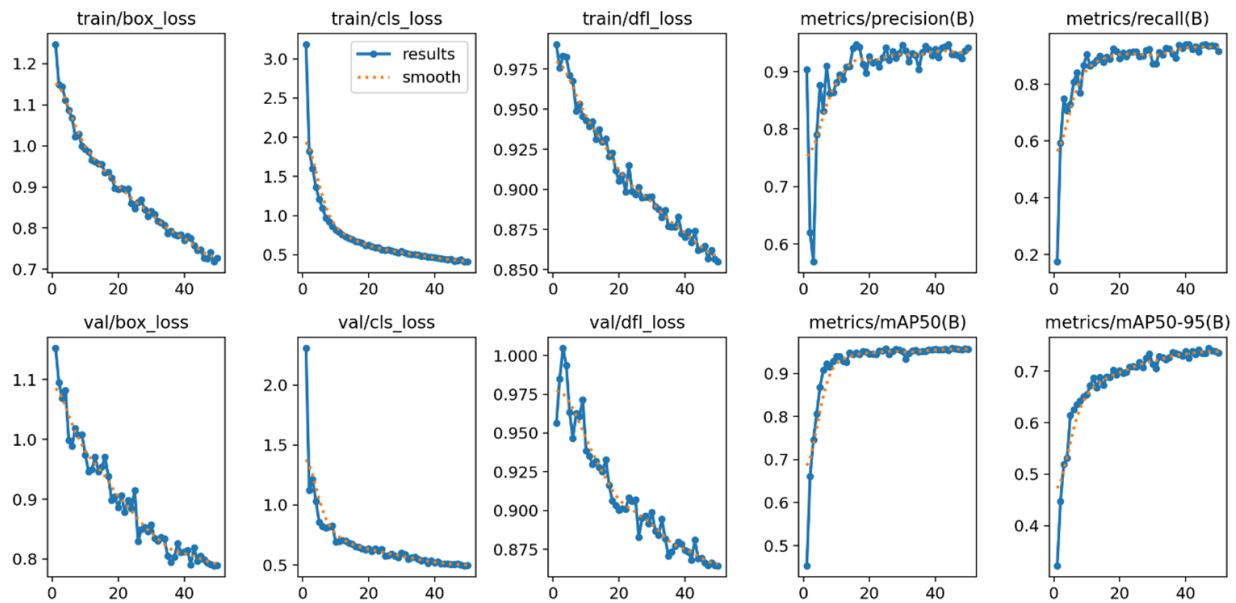
Pada Gambar 16 di bawah ini menunjukkan hasil pelatihan dan evaluasi model YOLOv8n yang sudah dilakukan pada dataset gambar sebelum dilakukan image preprocessing. Grafik ini menampilkan berbagai metrik dan loss selama pelatihan dan validasi, memberikan gambaran tentang kinerja model. Pada grafik `train/box_loss` dan `val/box_loss` menunjukkan bahwa grafik menunjukkan penurunan yang stabil, yang berarti model semakin baik dalam memprediksi posisi bounding box seiring bertambahnya epoch.

Pada grafik `train/cls_loss` dan `val/cls_loss` menunjukkan pengukuran kesalahan pada klasifikasi objek selama pelatihan dan validasi. Hasil Loss klasifikasi juga menurun secara konsisten, menunjukkan bahwa model semakin baik dalam mengidentifikasi kelas objek.

Pada `train/dfl_loss` dan `val/dfl_loss`, menunjukkan pengukuran kesalahan pada distribusi fokus lokal (DFL), yaitu cara model memprediksi posisi piksel pada bounding box dengan lebih presisi. Hasil penurunan loss menunjukkan model semakin akurat dalam menentukan posisi bounding box pada level piksel.

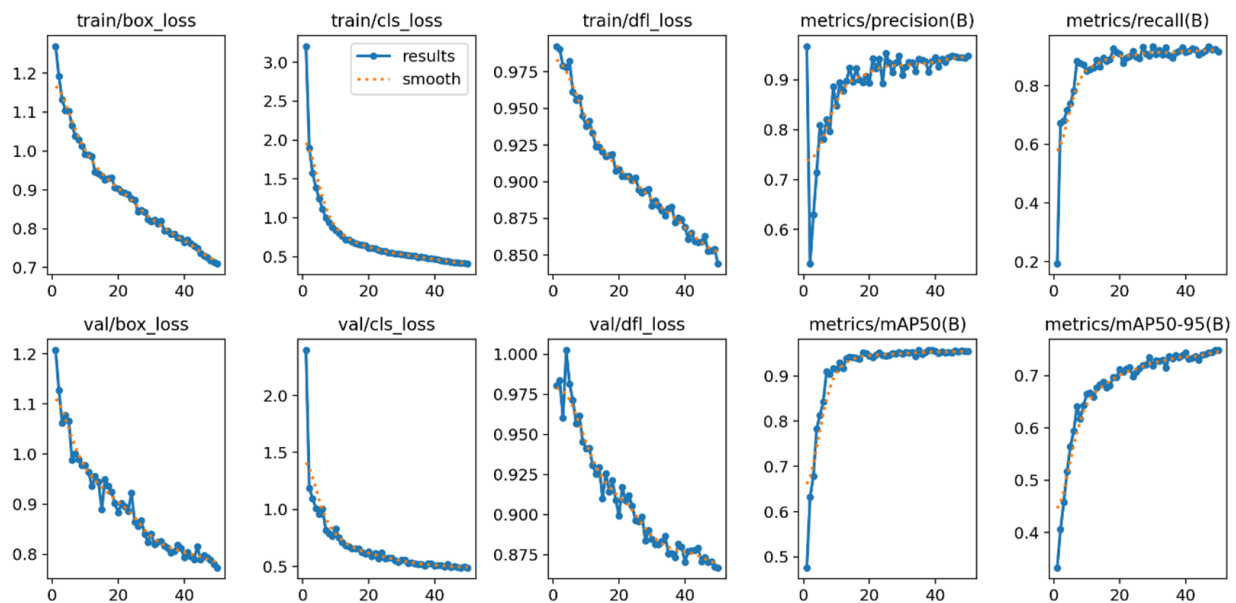
Selanjutnya grafik Precision dan Recall. Metrik Precision mengukur proporsi prediksi positif yang benar. Dapat dilihat hasil Precision meningkat tajam pada awal pelatihan dan mencapai nilai mendekati 0.9 di akhir pelatihan, menunjukkan bahwa model semakin akurat dalam menghindari false positives. Sementara matriks recall digunakan untuk mengukur proporsi instance positif yang berhasil dideteksi. Hasil Recall juga meningkat stabil hingga mendekati 0.85, menunjukkan bahwa model mampu mendeteksi sebagian besar objek yang relevan.

Terakhir grafik Map (*Mean Average Precision*), pertama `matrix/Map50` memperlihatkan pengukuran rata-rata presisi pada Intersection over Unit (IoU) threshold 50%. Grafik menunjukkan peningkatan stabil hingga mencapai nilai mendekati 0.95, menunjukkan kinerja yang sangat baik pada prediksi bounding box dan klasifikasi pada IoU rendah. Sementara grafik `matrix/mAP50-95` untuk mengukur rata-rata presisi pada



Gambar 16. Grafik Hasil Training dan Loss Hasil Pelatihan Model Sebelum dilakukan Image Preprocessing

berbagai IoU threshold (50% hingga 95%). Grafik juga meningkat stabil, meskipun nilainya sedikit lebih rendah (sekitar 0.85), menunjukkan bahwa model tetap akurat bahkan pada IoU yang lebih ketat.



Gambar 17. Grafik Hasil Training dan Loss Hasil Pelatihan Model setelah dilakukan Image Preprocessing

Pada Gambar 17 di atas menunjukkan hasil training dan loss hasil pelatihan model setelah dilakukan image preprocessing. Hasilnya tidak jauh berbeda dengan hasil evaluasi pelatihan dataset sebelum dilakukan image preprocessing. Perbedaan hasil evaluasi antara dataset sebelum dilakukan image preprocessing dan setelah dilakukan image preprocessing dapat dilihat pada Tabel 3 berikut ini.

4. Simpulan

Berdasarkan hasil evaluasi yang telah dilakukan, dapat disimpulkan bahwa model YOLOv8n dengan melakukan *image preprocessing* pada dataset gambar CARLA Simulator ini menunjukkan kinerja yang sangat baik, dengan akurasi tinggi dan pelatihan yang stabil. Model ini cocok untuk diterapkan pada tugas deteksi objek di lingkungan nyata. Pada matrix evaluasi penurunan loss pada performa pelatihan dan validasi

Tabel 3. Perbedaan Hasil Evaluasi Antara Dataset Sebelum Dilakukan Image Preprocessing Dan Setelah Dilakukan Image Preprocessing

Aspek	Hasil Evaluasi Sebelum Dilakukan Image Preprocessing	Hasil Evaluasi Setelah Dilakukan Image Preprocessing
Train/Validation	Penurunan lebih cepat dan nilai akhir lebih rendah.	Penurunan lebih lambat, tetapi tetap konsisten.
Precision	Precision mencapai ~0.9 lebih awal dan lebih stabil.	Precision meningkat bertahap, tetapi hasil akhir setara.
Recall	Recall lebih stabil selama pelatihan.	Recall sedikit lebih rendah di awal pelatihan.
mAP50	Konvergensi lebih cepat ke nilai ~0.95.	Konvergensi lebih lambat tetapi mencapai hasil akhir yang sama.
Map50-95	Lebih stabil tanpa fluktuasi signifikan.	Meningkat bertahap dengan sedikit variasi di awal pelatihan.

menunjukkan bahwa penurunan loss menunjukkan bahwa model berhasil belajar dari data tanpa indikasi *overfitting*. Konsistensi antara *loss* pelatihan dan validasi menunjukkan bahwa model memiliki generalisasi yang baik. Untuk tingkat akurasi, precision dan recall menunjukkan kinerja yang baik, dengan precision mencapai 0.9 dan recall mencapai 0.85. Nilai mAP yang tinggi (mendekati 0.95 untuk IoU 50% dan 0.85 untuk IoU 50-95%) mengindikasikan kemampuan model yang sangat baik dalam deteksi objek. Dalam hal stabilitas model, kurva loss dan metrik lainnya menunjukkan konsistensi selama pelatihan, tanpa lonjakan besar yang mengindikasikan masalah dalam optimisasi.

Keterbatasan pada penelitian ini yaitu masih mengandalkan data simulasi CARLA dan sintetik, walaupun modul adaptif terbukti efektif, validasi pada data dunia nyata dengan *noise sensor* dan kondisi cuaca sesungguhnya masih diperlukan. Selain itu, *joint training* menambah beban komputasi, sehingga perlu optimasi lebih lanjut untuk *deployment* pada *embedded hardware*.

Pustaka

- [1] L. Du, "Object detectors in autonomous vehicles: Analysis of deep learning techniques," *International Journal of Advanced Computer Science and Applications*, vol. 14, no. 10, pp. 217–224, 2023.
- [2] Y. Peng, Y. Qin, X. Tang, Z. Zhang, and L. Deng, "Survey on image and point-cloud fusion-based object detection in autonomous vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 12, pp. 22 772–22 789, 2022.
- [3] N. A. Razak, N. A. A. Sabri, J. Johari, F. A. Ruslan, M. M. Kamal, and M. A. Aziz, "Investigation of object detection and identification at different lighting conditions for autonomous vehicle application," *International Journal of Automotive and Mechanical Engineering*, vol. 20, no. 3, pp. 10 649–10 658, 2023.
- [4] L. Orgován, T. Bécsi, and S. Aradi, "Autonomous drifting using reinforcement learning," *Periodica Polytechnica Transportation Engineering*, vol. 49, no. 3, pp. 292–300, 2021.
- [5] H. Majidifard, P. Jin, Y. Adu-Gyamfi, and W. G. Buttlar, "Pavement image datasets: A new benchmark dataset to classify and density pavement distresses," *Transportation Research Record*, vol. 2674, no. 2, pp. 328–339, 2020.
- [6] C. Lin, D. Tian, X. Duan, J. Zhou, D. Zhao, and D. Cao, "Da-rdd: Toward domain adaptive road damage detection across different countries," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 3, pp. 3091–3103, 2023.
- [7] C. Chavan, "Vehicle detection using yolov5," *International Journal of Scientific Research in Engineering and Management*, vol. 7, no. 5, pp. 1–9, 2023.
- [8] K. Sharada, W. Alghamdi, K. Karthika, A. H. Alawadi, G. Nozima, and V. Vijayan, "Deep learning techniques for image recognition and object detection," *E3S Web of Conferences*, vol. 399, pp. 1–11, 2023.
- [9] P. Kasare, J. Kulkarni, and R. Bichkar, "Optimization in edge detection using ant colony optimization," *International Journal of Recent Technology and Engineering*, vol. 8, no. 3, pp. 8167–8170, 2019.
- [10] W. Liu, G. Ren, R. Yu, S. Guo, J. Zhu, and L. Zhang, "Image-adaptive yolo for object detection in adverse weather conditions," in *Proceedings of the 36th AAAI Conference on Artificial Intelligence (AAAI 2022)*, vol. 36, 2022, pp. 1792–1800.

- [11] Y. Qiu, Y. Lu, Y. Wang, and H. Jiang, "Idod-yolov7: Image-dehazing yolov7 for object detection in low-light foggy traffic environments," *Sensors*, vol. 23, no. 3, 2023.
- [12] S. Muhammad, A. Ali, and M. Farhan, "Enhancing object detection in low light environments using image enhancement techniques and yolo architectures," in *2024 26th International Multi-Topic Conference (INMIC)*, 2024, pp. 1–6.
- [13] A. Elgazwy *et al.*, "Image enhancement for better vru detection in challenging weather conditions," in *2024 6th International Conference on Communications, Signal Processing, and their Applications (ICCSPA 2024)*, 2024, pp. 1–8.
- [14] X. Liu and Y. Lin, "Yolo-gw: Quickly and accurately detecting pedestrians in a foggy traffic environment," *Sensors*, vol. 23, no. 12, 2023.
- [15] I. Lashkov, R. Yuan, and G. Zhang, "Edge-computing-facilitated nighttime vehicle detection investigations with clahe-enhanced images," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 11, pp. 13 370–13 383, 2023.
- [16] L. Liu *et al.*, "Deep learning for generic object detection: A survey," *International Journal of Computer Vision*, vol. 128, no. 2, pp. 261–318, 2020.
- [17] Q. C. Mao, H. M. Sun, Y. B. Liu, and R. S. Jia, "Mini-yolov3: Real-time object detector for embedded applications," *IEEE Access*, vol. 7, pp. 133 529–133 538, 2019.
- [18] M. S. Beg and M. Y. Ismail, "Investigation of collision estimation with vehicle and pedestrian using carla simulation software," *Journal of Mechanical Engineering and Sciences*, vol. 18, no. 1, pp. 9949–9958, 2024.
- [19] H. Wang, Y. Yu, Y. Cai, X. Chen, L. Chen, and Q. Liu, "A comparative study of state-of-the-art deep learning algorithms for vehicle detection," *IEEE Intelligent Transportation Systems Magazine*, vol. 11, no. 2, pp. 82–95, 2019.
- [20] J. Lee, K. Lim, and J. Cho, "Improved monitoring of wildlife invasion through data augmentation by extract–append of a segmented entity," *Sensors*, vol. 22, no. 19, 2022.
- [21] S. Zhao, X. Yang, Y. Chen, and X. Liu, "Research on pedestrian detection algorithms combined with lightweight networks," in *2022 2nd International Conference on Algorithms, High Performance Computing and Artificial Intelligence (AHPCAI 2022)*, 2022, pp. 62–67.
- [22] A. E. Sallab, M. Abdou, E. Perot, and S. Yogamani, "Deep reinforcement learning framework for autonomous driving," in *IS and T International Symposium on Electronic Imaging Science and Technology*, 2017, pp. 70–76.
- [23] M. Abdou and H. A. Kamal, "Sdc-net: End-to-end multitask self-driving car camera cocoon iot-based system," *Sensors*, vol. 22, no. 23, pp. 1–19, 2022.
- [24] A. A. Rasjid, B. Rahmat, and A. N. Sihananto, "Implementasi yolov8 pada robot deteksi objek," *Journal of Technology and System Information*, vol. 1, no. 3, p. 9, 2024.
- [25] A. Mendhe, H. B. Chaudhari, A. Diwan, S. M. Rathod, and A. Sharma, "Object detection and tracking for autonomous vehicle using ai in carla," in *2022 International Conference on Industry 4.0 Technology (I4Tech 2022)*, 2022, pp. 1–5.