

SKRIPSI

**PERBANDINGAN METODE DETEKSI OBJEK DALAM
CITRA PENCAHAYAAN RENDAH MENGGUNAKAN
METODE CNN BASED DAN TRANSFORMER BASED**



**RIDHO PANDHU AFRIANTO
NIM. 162112133062**

**PROGRAM SARJANA
TEKNOLOGI SAINS DATA
DEPARTEMEN TEKNIK
FAKULTAS TEKNOLOGI MAJU DAN MULTIDIPLIN
UNIVERSITAS AIRLANGGA
2025**

LEMBAR PENGESAHAN

PERBANDINGAN METODE DETEKSI OBJEK DALAM CITRA PENCAHAYAAN RENDAH MENGGUNAKAN CNN BASED DAN TRANSFORMER BASED

Nama : Ridho Pandhu Afrianto
NIM : 162112133062
Tanggal Sidang Skripsi :

Surabaya, 09 Mei 2025

Pembimbing I

Pembimbing II

Dr. Aziz Fajar, S.Kom., M.Kom.
NIP. 199410052024023101 **Ratih Ardiati Ningrum, S.Si., M.Stat.**
NIP. 199501262020013201

Mengetahui,
Koordinator Program Studi
S1 Teknologi Sains Data

Dr. Dwi Rantini, S.Si.
NIP. 199406152022033201

PERNYATAAN ORISINALITAS SKRIPSI

Saya, Ridho Pandhu Afrianto, 162112133062, penulis Skripsi yang berjudul **Perbandingan Metode Deteksi Objek dalam Citra Pencahayaan Rendah menggunakan CNN Based dan Transformer Based** menyatakan bahwa:

1. Skripsi ini adalah asli dan benar-benar hasil karya saya sendiri, bukan hasil karya pihak lain dengan mengatasnamakan saya, bukan merupakan hasil tiruan atau jiplakan (*plagiarism*) dari karya pihak lain, dan/atau bukan tulisan yang dibuat dengan kecerdasan buatan.
2. Skripsi ini belum pernah diajukan untuk mendapatkan gelar akademik, baik di Universitas Airlangga, maupun di perguruan tinggi lainnya.
3. Dalam Skripsi ini tidak terdapat karya atau pendapat yang telah ditulis atau dipublikasikan orang lain, kecuali secara tertulis dengan jelas dicantumkan sebagai acuan dengan disebutkan nama pengarang dan dicantumkan dalam daftar kepustakaan.

Pernyataan ini saya buat dengan sebenar-benarnya, dan apabila dikemudian hari terdapat penyimpangan dan ketidakbenaran dalam pernyataan ini, maka saya bersedia menerima sanksi akademik berupa pencabutan gelar yang telah diperoleh karna karya tulis Skripsi ini, serta sanksi-sanksi lainnya sesuai dengan norna dan peraturan yang berlaku di Universitas Airlangga.

Surabaya, 23 April 2025

Ridho Pandhu Afrianto
NIM 162112133062

KETENTUAN PENGGUNAAN SKRIPSI

Ketentuan hak cipta bagi skripsi yang tidak dipublikasikan, terdaftar, tersedia, serta terbuka untuk umum di Perpustakaan Universitas Airlangga, dimiliki penulis dengan mengikuti aturan HKI yang berlaku di Universitas Airlangga. Referensi kepustakaan diperkenankan dicatat, tetapi pengutipan atau peringkasan hanya dapat dilakukan dengan seizin penulis dan harus disertasi dengan kaidah ilmiah. Memperbanyak atau menerbitkan sebagian atau seluruh skripsi haruslah seizin Penulis.

Situs Skripsi ini dapat ditulis sebagai berikut:

Afrianto, R.P. (2025). Perbandingan Metode Deteksi Objek dalam Citra Pencahayaan Rendah menggunakan CNN Based dan Transformer Based. Skripsi. Surabaya: Universitas Airlangga

Afrianto, R.P. (2025). *A Comparative Study of Object Detection Methods in Low-Light Images Using CNN-Based and Transformer-Based Approaches*. Undergraduate Thesis. Surabaya: Universitas Airlangga.

KATA PENGANTAR

Puji syukur saya panjatkan ke hadirat Tuhan Yang Maha Esa atas rahmat dan karunia-Nya sehingga saya dapat menyelesaikan skripsi ini yang berjudul "Perbandingan Metode Deteksi Objek dalam Citra Pencahayaan Rendah Menggunakan CNN-Based dan Transformer-Based". Skripsi ini disusun sebagai syarat untuk mendapat gelar sarjana di Program Studi Teknologi Sains Data, Fakultas Sains dan Teknologi, Universitas Airlangga.

Saya menyadari bahwa dalam skripsi ini, banyak pihak yang telah memberikan bantuan dan dukungan, baik berupa bimbingan, saran, maupun dorongan moral. Oleh karena itu, saya mengucapkan terima kasih yang sebesar-besarnya kepada kedua orang tua penulis, Freddy Afrianto dan Sri Moempoeni yang mendukung penuh kegiatan skripsi penulis. Tanpa dukungan keluarga dan orang-orang terdekat yang tidak bisa disebutkan satu persatu, penulis tidak mungkin dapat menyelesaikan keseluruhan skripsi dengan lancar.

Tentu saja, penulis tidak lupa mengucapkan terimakasih kepada dosen pembimbing penulis, Dr. Aziz Fajar, S.Kom., M.Kom. dan Ratih Ardiati Ningrum, S.Si., M.Stat. atas bimbingannya yang mengajarkan segala hal mulai dari teknis hingga kepenulisan, serta memberi masukan dan pengarahan ketika penulis mengalami kesulitan pada proses penulisan skripsi ini.

Terimakasih tambahan juga penulis utarakan pada *Konka Coffee* dan *Kedai Masjarakat* yang sudah menyediakan kopi murah berkualitas, teman menulis naskah skripsi. Terakhir, terimakasih sebesar-besarnya pada *Hindia*, terutama untuk album *Lagipula Hidup Akan Berakhir* yang menjadi favorit penulis karena telah membantu menenangkan jiwa dan raga penulis saat menyusun skripsi yang penuh tantangan ini.

Surabaya, 23 April 2025

Penulis

ABSTRAK

PERBANDINGAN METODE DETEksi OBJEK DALAM CITRA PENCAHAYAAN RENDAH MENGGUNAKAN METODE CNN BASED DAN TRANSFORMER BASED

Oleh

Ridho Pandhu Afrianto

NIM: 162112133062

Program Sarjana Teknologi Sains Data

Penelitian ini melakukan perbandingan kinerja metode deteksi objek pada citra berpencahayaan rendah menggunakan tiga pendekata, berbasis CNN (YOLOv9), berbasis Transformer (RT-DETR), dan *hybrid CNN-Transformer* (YOLOv10). Dataset ExDark yang terdiri dari 7.363 citra dengan 12 kelas objek digunakan sebagai bahan pelatihan dan validasi. Selain itu, untuk pengujian *real-world*, digunakan 12 rekaman video CCTV berdurasi 5-30 detik yang mencakup objek-objek serupa, diambil untuk menilai performa model pada frame kondisi nyata. Proses penelitian meliputi *preprocessing data*, termasuk *Lanczos Downsampling*, konversi format *bounding box*, penajaman kontras dengan CLAHE, dan augmentasi data, yang akan dibagi menjadi *train set* (80%) dan *test set* (20%). Model dilatih dengan *grid search* pada *hyperparameter learning rate* (0,001; 0,01) dan *batch size* (8; 16), kemudian dievaluasi berdasarkan *mean Average Precision* pada IoU 0,5 (mAP50) dan waktu inferensi. Analisis Pareto Frontier menunjukkan bahwa YOLOv9m mencapai *trade-off* terbaik dengan mAP50 0,5739 pada citra bercahaya cukup dan 0,5707 untuk citra bercahaya rendah, serta inferensi 606,62 ms – 588,45 ms. Model CNN umumnya lebih cepat daripada Transformer, sedangkan YOLOv10s tercatat tercepat (221,08 ms) dengan akurasi mAP50 0,3960.

Kata Kunci: Deteksi objek, Pencahayaan rendah, CNN, Transformer, YOLOv9, RT-DETR, YOLOv10, Pareto Frontier, ExDark, mAP50, CCTV

ABSTRACT

A COMPARATIVE STUDY OF OBJECT DETECTION METHODS IN LOW-LIGHT IMAGES USING CNN-BASED AND TRANSFORMER- BASED APPROACHES

By

Ridho Pandhu Afrianto

Student ID Number: 162112133062

Undergraduate Program in Data Science Technology

This study presents a comparative analysis of object detection methods in low-light images using three distinct approaches, CNN-based (YOLOv9), Transformer-based (RT-DETR), and hybrid CNN-Transformer (YOLOv10). The ExDark dataset, consisting of 7,363 images across 12 object classes, was utilized for training and validation. In addition, to assess real-world applicability, 12 CCTV video recordings, each lasting 5 to 30 seconds, were employed as a test set, featuring similar object categories to evaluate model performance on real environmental frames. The research workflow involved a comprehensive data preprocessing pipeline, including Lanczos downsampling, bounding box format conversion, contrast enhancement via CLAHE, and data augmentation. The dataset was split into training (80%) and validation (20%) subsets. Model training was conducted using a grid search strategy over two hyperparameters, learning rate (0.001 and 0.01) and batch size (8 and 16). Performance evaluation was based on mean Average Precision at an IoU threshold of 0.5 (mAP50) and inference time. Pareto Frontier analysis identified YOLOv9m as the most balanced model, achieving an mAP50 of 0.5739 under well-lit conditions and 0.5707 in low-light scenarios, with inference times of 606.62 ms and 588.45 ms, respectively. In general, CNN-based models outperformed Transformer-based models in terms of inference speed. YOLOv10s demonstrated the fastest inference time (221.08 ms) among all models but with a trade-off in accuracy, reaching an mAP50 of only 0.3960.

Keywords: Object detection, Low-light images, CNN, Transformer, YOLOv9, RT-DETR, YOLOv10, Pareto Frontier, ExDark, mAP50, CCTV

DAFTAR ISI

HALAMAN JUDUL	i
LEMBAR PENGESAHAN	ii
KATA PENGANTAR.....	v
DAFTAR ISI.....	iv
DAFTAR TABEL	viii
DAFTAR GAMBAR.....	viii
BAB I PENDAHULUAN.....	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	5
1.3 Tujuan Penelitian	5
1.4 Manfaat Penelitian	6
1.5 Batasan Masalah.....	6
BAB II TINJAUAN PUSTAKA.....	7
2.1 <i>Computer Vision</i>	7
2.2 Analisis <i>Object Detection</i>	8
2.3 Data <i>Preprocessing</i>	8
2.3.1 Contrast Enhancement	9
2.3.2 Downsampling	10
2.3.3 Augmentasi Data.....	11
2.4 <i>Convolutional Neural Network</i>	12
2.5 <i>You Only Look Once</i>	14
2.6 Transformer.....	16
2.7 CNN-Transformer (YOLOv10)	19
2.8 <i>Hyperparameter Tuning</i>	20
2.9 <i>Loss Function</i>	22
2.10 Metrik Evaluasi Model.....	25
2.11 Pareto Frontier.....	27
2.12 <i>Optimizer</i>	28

BAB III METODE PENELITIAN	28
3.1 Lokasi dan Waktu	28
3.2 Bahan dan Alat.....	28
3.3 Cara Kerja	29
3.3.1 Data Preprocessing.....	31
3.3.3 Metode Perbandingan Model	40
BAB IV HASIL DAN PEMBAHASAN	42
4.1 Pengambilan data	42
4.2 <i>Data Preprocessing</i>	43
4.2.1 <i>Downsampling</i>	43
4.2.2 Pengubahan format label <i>bounding box</i>	45
4.2.3 <i>Data Augmentation</i>	46
4.3 Implementasi YOLOv9	47
4.3.1 Pelatihan YOLOv9	47
4.3.2 Kombinasi YOLOv9 terbaik.....	49
4.3.2.1 YOLOv9s	49
4.3.2.2 YOLOv9m	33
4.3.2.3 YOLOv9c	30
4.4 Implementasi RT-DETR	34
4.4.1 Pelatihan RT-DETR.....	34
4.4.2 Kombinasi RTDETR terbaik	35
4.4.2.1 RTDETR-L	36
4.4.2.2 REDETR-X.....	40
4.5 Implementasi YOLOv10.....	47
4.5.1 Pelatihan YOLOv10.....	47
4.5.2 Kombinasi YOLOv10 terbaik.....	48
4.5.2.1 YOLOv10s	49
4.5.2.2 YOLOv10m	53
4.5.2.3 YOLOv10b.....	57
4.6 Testing Model.....	61
4.6.1 YOLOv9s.....	62

4.6.1.1 <i>Well-lit Detection</i>	62
4.6.1.2 <i>Low-light Detection</i>	65
4.6.2 YOLOv9m	68
4.6.2.1 <i>Well-lit Detection</i>	68
4.6.2.2 <i>Low-light Detection</i>	71
4.6.3 YOLOv9c.....	74
4.6.3.1 <i>Well-lit Detection</i>	74
4.6.3.2 <i>Low-light Detection</i>	77
4.6.4 RTDETR-L	80
4.6.4.1 <i>Well-lit Detection</i>	80
4.6.4.2 <i>Low-light Detection</i>	83
4.6.5 RTDETR-X.....	86
4.6.5.1 <i>Well-lit Detection</i>	86
4.6.5.2 <i>Low-light Detection</i>	89
4.6.6 YOLOv10s.....	92
4.6.6.1 <i>Well-lit Detection</i>	92
4.6.6.2 <i>Low-light Detection</i>	95
4.6.7 YOLOv10m	97
4.6.7.1 <i>Well-lit Detection</i>	97
4.6.7.2 <i>Low-light Detection</i>	100
4.6.8 YOLOv10b	103
4.6.8.1 <i>Well-lit Detection</i>	103
4.6.8.2 <i>Low-light Detection</i>	105
4.7 Pareto Frontier Analysis	107
4.8 Diskusi.....	109
4.8.1 Studi yang Konsisten dengan Temuan Penelitian.....	110
4.8.2 Studi yang Bertentangan dengan Temuan Penelitian	111
4.8.3 Analisis Penyebab Perbedaan	112
BAB V KESIMPULAN DAN SARAN	116
5.1 Kesimpulan.....	116
5.2 Saran	117

DAFTAR PUSTAKA	118
-----------------------------	-----

DAFTAR TABEL

Tabel 3.1 Deskripsi Parameter Augmentasi.....	33
Tabel 3.2 Tipe Model YOLOv9.....	35
Tabel 3.3 Tipe Model RT-DETR	37
Tabel 3.4 Tipe Model YOLOv10.....	39
Tabel 3.5 Deskripsi Hyperparameter yang Diuji	40
Tabel 4.1 Hasil grid search untuk YOLOv9.....	48
Tabel 4.2 Kombinasi terbaik untuk YOLOv9.....	49
Tabel 4.3 Label vs Prediksi untuk YOLOv9s	32
Tabel 4.4 Label vs Prediksi untuk YOLOv9m.....	29
Tabel 4.5 Label vs Prediksi untuk YOLOv9c	33
Tabel 4.6 Hasil grid search untuk RT-DETR.....	34
Tabel 4.7 Kombinasi terbaik untuk RT-DETR.....	35
Tabel 4.8 Label vs Prediksi untuk RTDETR-L	39
Tabel 4.9 Label vs Prediksi untuk RTDETR-X	46
Tabel 4.10 Hasil grid search untuk YOLOv10	47
Tabel 4.11 Kombinasi terbaik untuk YOLOv10.....	48
Tabel 4.12 Label vs Prediksi untuk YOLOv10s	52
Tabel 4.13 Label vs Prediksi untuk YOLOv10m	56
Tabel 4.14 Label vs Prediksi untuk YOLOv10b.....	60
Tabel 4.15 Label vs Prediksi untuk YOLOv9s pada <i>well-lit test</i>	62
Tabel 4.16 Hasil evaluasi YOLOv9s pada <i>well-lit test</i>	64
Tabel 4.17 Label vs Prediksi untuk YOLOv9s pada <i>low-light test</i>	65
Tabel 4.18 Hasil evaluasi YOLOv9s pada <i>low-light test</i>	67
Tabel 4.19 Label vs Prediksi untuk YOLOv9m pada <i>well-lit test</i>	68
Tabel 4.20 Hasil evaluasi YOLOv9m pada <i>well-lit test</i>	70
Tabel 4.21 Label vs Prediksi untuk YOLOv9m pada <i>low-light test</i>	71
Tabel 4.22 Hasil evaluasi YOLOv9m pada <i>low-light test</i>	72
Tabel 4.23 Label vs Prediksi untuk YOLOv9c pada <i>well-lit test</i>	74
Tabel 4.24 Hasil evaluasi YOLOv9c pada <i>well-lit test</i>	76
Tabel 4.25 Label vs Prediksi untuk YOLOv9c pada <i>low-light test</i>	77
Tabel 4.26 Hasil evaluasi YOLOv9c pada <i>low-light test</i>	79
Tabel 4.27 Label vs Prediksi untuk RTDETR-L pada <i>well-lit test</i>	80
Tabel 4.28 Hasil evaluasi RTDETR-L pada <i>well-lit test</i>	82
Tabel 4.29 Label vs Prediksi untuk RTDETR-L pada <i>low-light test</i>	83
Tabel 4.30 Hasil evaluasi RTDETR-L pada <i>low-light test</i>	84
Tabel 4.31 Label vs Prediksi untuk RTDETR-X pada <i>well-lit test</i>	86
Tabel 4.32 Hasil evaluasi RTDETR-X pada <i>well-lit test</i>	87
Tabel 4.33 Label vs Prediksi untuk RTDETR-X pada <i>low-light test</i>	89

Tabel 4.34 Hasil evaluasi RTDETR-X pada <i>low-light test</i>	91
Tabel 4.35 Label vs Prediksi untuk YOLOv10s pada <i>well-lit test</i>	92
Tabel 4.36 Hasil evaluasi YOLOv10s pada <i>well-lit test</i>	93
Tabel 4.37 Label vs Prediksi untuk YOLOv10s pada <i>low-light test</i>	95
Tabel 4.38 Hasil evaluasi YOLOv10s pada <i>low-light test</i>	96
Tabel 4.39 Label vs Prediksi untuk YOLOv10m pada <i>well-lit test</i>	97
Tabel 4.40 Hasil evaluasi YOLOv10m pada <i>well-lit test</i>	99
Tabel 4.41 Label vs Prediksi untuk YOLOv10m pada <i>low-light test</i>	100
Tabel 4.42 Hasil evaluasi YOLOv10m pada <i>low-light test</i>	102
Tabel 4.43 Label vs Prediksi untuk YOLOv10b pada <i>well-lit test</i>	103
Tabel 4.44 Hasil evaluasi YOLOv10b pada <i>well-lit test</i>	104
Tabel 4.45 Label vs Prediksi untuk YOLOv10b pada <i>low-light test</i>	105
Tabel 4.46 Hasil evaluasi YOLOv10b pada <i>low-light test</i>	107

DAFTAR GAMBAR

Gambar 2. 1 Object detection dengan satu atau lebih objek	8
Gambar 2. 2 Proses Konvolusi 2D.....	13
Gambar 2.3 Sistem deteksi objek dengan YOLO	14
Gambar 2.4 Arsitektur YOLO	14
Gambar 2.5 Arsitektur YOLOv9	15
Gambar 2.6 Arsitektur Transformer.....	17
Gambar 2.7 Arsitektur RT-DETR	18
Gambar 2.8 Arsitektur YOLOv10	20
Gambar 2.9 Perbedaan ℓ_2 loss dan IoU loss	23
Gambar 3.1 Sampel Exclusively Dark dataset.....	29
Gambar 3.2 Flowchart Keseluruhan Proses	31
Gambar 3.3 Proses Inferensi Model berbasis CNN	34
Gambar 3.4 Proses Inferensi Model Berbasis Transformer (RE-DETR)	36
Gambar 3.5 Proses Inferensi Model berbasis CNN-Transformer (YOLOv10)	38
Gambar 4.1 Sampel Exclusively Dark Dataset.....	42
Gambar 4.2 Distribusi citra untuk setiap objek.....	43
Gambar 4.3 Downsampling sebanyak 20%	44
Gambar 4.4 Downsampling sebanyak 40%	44
Gambar 4.5 Sampel hasil CLAHE	46
Gambar 4.6 Sampel hasil augmentasi data	47
Gambar 4.7 Sampel hasil mosaic augmentati	47
Gambar 4.8 Kurva hasil pelatihan model YOLOv9s	49
Gambar 4.9 Confusion Matrix Normalized untuk YOLOv9s.....	30
Gambar 4.10 Kurva F1-Confidence untuk YOLOv9s	31
Gambar 4.11 Kurva hasil pelatihan model YOLOv9m	33
Gambar 4. 12 Confusion Matrix Normalized untuk YOLOv9m	29
Gambar 4.13 Kurva F1-Confidence untuk YOLOv9m	30
Gambar 4.14 Kurva hasil pelatihan model.....	30
Gambar 4.15 Confusion Matrix Normalized untuk YOLOv9c	31
Gambar 4.16 Kurva F1-Confidence untuk YOLOv9c.....	32
Gambar 4.17 Kurva hasil pelatihan model RTDETR-L	36
Gambar 4.18 Confusion Matrix Normalized untuk REDETR-L	37
Gambar 4.19 Kurva F-Confidence untuk REDETR-L	38
Gambar 4.20 Kurva hasil pelatihan model REDETR-X	40
Gambar 4.21 Confusion matrix Normalized untuk REDETR-X	43
Gambar 4.22 Kurva F1-Confidence untuk RTDETR-X	44
Gambar 4.23 Kurva hasil pelatihan model YOLOv10s	49
Gambar 4.24 Confusion Matrix Normalized untuk YOLOv10s.....	50
Gambar 4.25 Kurva F1-Confidence untuk YOLOv10s	51

Gambar 4.26 Kurva hasil pelatihan model YOLOv10m	53
Gambar 4.27 Confusion Matrix Normalized untuk YOLOv10s.....	54
Gambar 4. 28 Kurva F1-Confidence untuk YOLOv10m	55
Gambar 4.29 Kurva hasil pelatihan model YOLOv10b.....	57
Gambar 4.30 Confusion Matrix Normalized untuk YOLOv10b	58
Gambar 4.31 Kurva F1-Confidence untuk YOLOv10b.....	59
Gambar 4.32 Distribusi instances dan frames untuk setiap kelas	61
Gambar 4.33 YOLOv9s <i>well-lit test confusion matrix</i>	63
Gambar 4.34 YOLOv9s test <i>confusion matrix</i>	66
Gambar 4.35 YOLOv9m <i>well-lit test confusion matrix</i>	69
Gambar 4.36 YOLOv9m <i>low-light test confusion matrix</i>	72
Gambar 4.37 YOLOv9c <i>well-lit test confusion matrix</i>	75
Gambar 4.38 YOLOv9c <i>low-light test confusion matrix</i>	78
Gambar 4.39 RTDETR-L <i>well-lit test confusion matrix</i>	81
Gambar 4.40 RTDETR-L <i>low-light test confusion matrix</i>	84
Gambar 4.41 RTDETR-X <i>well-lit test confusion matrix</i>	87
Gambar 4.42 RTDETR-X <i>low-light test confusion matrix</i>	90
Gambar 4.43 YOLOv10s <i>well-lit test confusion matrix</i>	93
Gambar 4.44 YOLOv10s <i>low-light test confusion matrix</i>	96
Gambar 4.45 YOLOv10m <i>well-lit test confusion matrix</i>	98
Gambar 4.46 YOLOv10m <i>low-light test confusion matrix</i>	101
Gambar 4.47 YOLOv10b <i>well-lit test confusion matrix</i>	104
Gambar 4.49 Pareto Frontier.....	108

BAB I

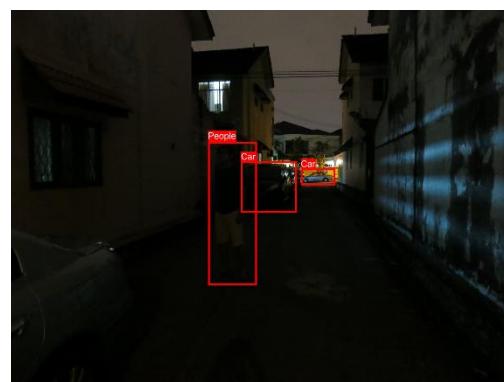
PENDAHULUAN

1.1 Latar Belakang

Dalam dunia keamanan modern, penggunaan CCTV telah menjadi pilihan utama dalam aktivitas di berbagai ruang publik dan privat, mulai dari pemantauan lalu lintas hingga perlindungan objek vital seperti bandara dan gedung pemerintahan. Namun, meskipun teknologi CCTV telah berkembang, masalah mendasar terkait deteksi objek dalam kondisi pencahayaan rendah tetap menjadi kendala signifikan. Pada malam hari atau di area yang minim penerangan, CCTV konvensional sering kali gagal menangkap gerakan mencurigakan dengan jelas, sehingga memberikan celah bagi tindakan kejahatan, termasuk terorisme. Kasus-kasus seperti pemboman dan serangan di ruang publik yang terekam CCTV menunjukkan bahwa saat ini identifikasi objek yang ada di dalamnya masih mengalami situasi yang sulit dan lambat (Putri, 2021). Keterbatasan ini menyoroti perlunya pengembangan lebih lanjut dalam teknologi pengawasan yang mampu beradaptasi dengan berbagai kondisi cahaya untuk memastikan keamanan maksimal.



(a)



(b)

Gambar 1.1 (a) Citra berpencahayaan rendah, (b) Contoh hasil dari model deteksi objek

Sumber: Loh & Chan (2018)

Salah satu permasalahan dari perangkat yang memerlukan waktu kerja 24 jam non-stop seperti CCTV, adalah kemampuannya untuk mendeteksi sebuah objek jika objek tersebut berada di pencahayaan yang rendah atau malam hari. Berdasarkan Gambar 1.1 (a) sudah dapat disimpulkan bahwa belum tentu manusia dapat mengetahui objek apa saja yang berada di gambar tersebut apalagi mengawasinya selama 24 jam. Oleh karena itu, perlu bantuan model deteksi objek seperti yang dicontohkan dalam Gambar 1.1 (b) untuk melakukan tugas demikian. Menurut Chen & Shah (2021), faktor-faktor seperti *noise*, *glare*, *lousy illumination*, *low contrast*, *shadows*, dan *reflectance* membuat deteksi objek dalam kondisi tersebut menjadi sulit. Berbagai penelitian dengan pendekatan yang beragam telah dilakukan untuk mengatasi masalah tersebut. Sebagian besar penelitian dalam peningkatan citra berfokus pada mempercantik subjek dan meningkatkan kualitas estetika gambar, dibandingkan dengan pemulihan fitur yang diperlukan untuk analisis lebih lanjut. Beberapa penelitian juga mencoba mengatasi masalah tersebut dengan menggunakan *hardware* yang lebih canggih, seperti kamera yang dilengkapi dengan sensor inframerah dan gambar termal. Namun, metode ini memiliki biaya yang tinggi dan sering kali menghasilkan citra yang kurang realistik (Chen & Shah, 2021). Salah satu pendekatan untuk meningkatkan kecerahan dalam kondisi pencahayaan rendah adalah dengan menaikkan ISO atau memperpanjang *exposure time*. Namun, strategi ini masing-masing cenderung meningkatkan *noise* dan menyebabkan *motion blur* (Chen *et al.*, 2018). Pendekatan lain yang dapat diterapkan adalah memanfaatkan *software* seperti Photoshop atau Lightroom untuk melakukan penyesuaian pencahayaan secara digital. Meskipun demikian, *software* ini memerlukan keterampilan artistik dan tidak efisien untuk dataset skala besar dengan kondisi pencahayaan yang beragam (Zheng & Gupta, 2022). Selebihnya, pendekatan-pendekatan tersebut menjadi kurang layak untuk aplikasi seperti CCTV yang memerlukan operasi non-stop selama 24 jam dan menuntut kecepatan maupun ketepatan deteksi.

Menyikapi masalah tersebut, Loh & Chan (2018) membuat sebuah dataset bernama ExDark atau *Exclusively Dark* untuk memfasilitasi para peneliti dalam

memahami fenomena citra di pencahayaan rendah. ExDark memiliki 7363 citra yang terdiri dari citra berpencahayaan rendah hingga citra di berpencahayaan fajar dengan 12 kelas objek di dalamnya sehingga cocok untuk riset berbasis aplikasi seperti *object detection* (Loh & Chan, 2018). Dataset ExDark ini kemudian diadopsi oleh model PE-YOLO atau *Pyramid Enhancement - You Only Look Once* yang secara khusus dirancang untuk mendeteksi objek dalam pencahayaan rendah dengan menggunakan pendekatan *Convolutional Neural Networks* (CNN) (Yin *et al.*, 2023). Kelemahan model ini adalah pada model *backbone* yang terbilang lama (YOLOv3), serta mekanisme di dalamnya yang masih memiliki keterbatasan dalam penanganan *noise* terutama pada frekuensi tinggi. Pada penelitian lain, Cui dkk. (2022) mengembangkan model *Multitask Auto-Encoding Transformation* (MAET). MAET menawarkan pendekatan baru dalam mendeteksi objek di kondisi pencahayaan rendah dengan memanfaatkan regularisasi ortogonal untuk memisahkan fitur degradasi pencahayaan dan deteksi objek. Model ini menggunakan pipeline *image signal processor* (ISP) yang lebih realistik untuk mensintesis gambar berpencahayaan rendah, yang memungkinkan model untuk mengoptimalkan deteksi objek di lingkungan gelap tanpa menghasilkan artefak yang sering terjadi pada metode pemulihan pencahayaan tradisional. Disamping skor mAP yang bernilai 0,74 pada dataset ExDark, terdapat beberapa kelemahan yang membatasi model ini (Cui *et al.*, 2022). Penggunaan regularisasi ortogonal menambah kompleksitas komputasi yang dapat menghambat implementasi model ini pada perangkat dengan sumber daya terbatas. Selain itu, MAET lagi-lagi juga masih mengandalkan YOLOv3 sebagai *backbone* yang merupakan arsitektur lama. Menggunakan pendekatan yang berbeda, Cui dkk. (2022) melakukan penelitian lain dengan mekanisme Transformer untuk mengembangkan *Illumination Adaptive Transformer* (IAT) untuk memulihkan gambar sRGB dengan pencahayaan normal dari kondisi *low-light* atau *under/exposure*. IAT mampu mencapai skor mAP sebesar 77,2 pada dataset ExDark dengan *time inference* 0,04 detik berkat komponen *simplified image signal processor* (ISP) yang dimilikinya (Cui *et al.*, 2022). Komponen *Simplified ISP* inilah yang membantu IAT mampu merealisasikan inferensi yang sangat cepat dengan parameter yang relatif kecil

yaitu sebesar 90.000 parameter. Namun, justru dari *simplified* ISP inilah IAT mengalami keterbatasan generalisasi ketika diterapkan pada dataset dengan kondisi yang beragam.

Kedua pendekatan yang populer pada *object detection*, yaitu CNN dan Transformer telah terbukti bahwa selalu ada *trade-off* masing-masing. Oleh karena itu, akan sangat menarik untuk mengeksplorasi apakah penggunaan kedua pendekatan yang berbeda pada model-model *state of the art* terbaru berpotensi menghasilkan peningkatan performa yang signifikan dan *trade-off* yang lebih optimal dibandingkan dengan penelitian-penelitian terdahulu. Definisi optimal dalam konteks ini adalah model yang mampu memberikan skor *mean Average Precision* (mAP) yang tinggi sembari meminimalkan waktu inferensi. Model-model *state-of-the-art* saat ini yang menggunakan pendekatan CNN, Transformer, dan Hybrid (CNN-Transformer) adalah YOLOv9 yang dirancang oleh Wang dkk. (2024), RT-DETR oleh Zhao dkk. (2023), dan YOLOv10 oleh peneliti lain Wang dkk. (2024). Pemilihan model-model ini didasarkan tidak hanya pada statusnya sebagai model *state-of-the-art* saat ini, tetapi juga karena kemampuan mereka dalam melakukan deteksi multi-objek secara *real-time* dengan kecepatan yang tinggi, menjadikannya ideal untuk aplikasi yang membutuhkan deteksi cepat dan akurat.

Dalam membandingkan dua jenis metrik yang berbeda, pendekatan dengan perbandingan secara langsung sangat tidak memungkinkan untuk dilakukan karena mAP diukur dalam skala skor sementara waktu inferensi diukur dalam satuan waktu (*ms*). Oleh karena itu, pendekatan yang tepat untuk mengevaluasi kedua metrik ini adalah dengan menggunakan *Pareto Frontier*. *Pareto Frontier* memungkinkan evaluasi *trade-off* antara dua atau lebih metrik, di mana sebuah solusi dianggap optimal jika tidak ada solusi lain yang lebih baik dalam kedua metrik secara bersamaan. Perbandingan antara mAP dan waktu inferensi menjadi sangat penting, karena perangkat seperti CCTV memerlukan model yang tidak hanya cepat tetapi juga akurat.

Penelitian ini ditujukan untuk memberikan kontribusi yang signifikan dalam pengembangan metode deteksi objek pada citra berpencahayaan rendah dengan

mengeksplorasi dan membandingkan model berbasis CNN dan pendekatan Transformer. Dengan mengatasi tantangan yang dihadapi oleh model deteksi objek dalam kondisi pencahayaan rendah, penelitian ini dapat membantu mengarahkan untuk meningkatkan akurasi serta kecepatan deteksi dalam berbagai aplikasi dunia nyata, seperti pengawasan CCTV dan industri lainnya yang bergantung pada deteksi objek secara *real-time*. Tujuan dari penelitian ini adalah dapat membantu mengidentifikasi solusi yang lebih efisien dan efektif baik secara biaya maupun sumber daya, terutama dalam kondisi pencahayaan yang beragam. Selainnya, optimalisasi pada penelitian ini juga diharapkan dapat menjadi solusi alternatif pengganti *thermal-cam* dan *infrared* untuk CCTV konvensional yang ada di Indonesia.

1.2 Rumusan Masalah

Rumusan masalah pada penelitian ini adalah sebagai berikut:

1. Bagaimana performa metode YOLOv9 terhadap objek dengan pencahayaan rendah dan apa faktor yang menyebabkannya?
2. Bagaimana performa metode RT-DETR terhadap objek dengan pencahayaan rendah dan apa faktor yang menyebabkannya?
3. Bagaimana performa metode YOLOv10 terhadap objek dengan pencahayaan rendah dan apa faktor yang menyebabkannya?
4. Apa metode yang terbaik dari segi *mean Average Precision* (mAP) dan inferensi waktu untuk mendekripsi objek berpencahayaan rendah?

1.3 Tujuan Penelitian

Tujuan yang ingin dicapai dalam penelitian ini adalah sebagai berikut:

1. Untuk mengetahui performa metode YOLOv9 terhadap objek dengan pencahayaan rendah dan mencari faktor yang menyebabkannya.
2. Untuk mengetahui performa metode RT- DETR terhadap objek dengan pencahayaan rendah dan mencari faktor yang menyebabkannya.
3. Untuk mengetahui performa metode YOLOv10 terhadap objek dengan pencahayaan rendah dan mencari faktor yang menyebabkannya.

4. Untuk mencari metode yang terbaik dalam *mean Average Precision* (mAP) dan inferensi waktu dalam mendekripsi objek dengan pencahayaan rendah.

1.4 Manfaat Penelitian

Manfaat yang ingin dicapai dalam penelitian ini adalah sebagai berikut:

1. Menjadi referensi bagi penelitian selanjutnya dalam pengembangan model berbasis CNN ataupun Transformer untuk mendekripsi objek.
2. Model yang lebih optimal dalam mendekripsi objek dalam pencahayaan rendah akan bermanfaat untuk meningkatkan kapabilitas *hardware* dengan sumber daya terbatas dalam mendekripsi objek.

1.5 Batasan Masalah

Batasan masalah pada penelitian ini jumlah kelas yang dapat didekripsi hanya 12 kelas karena sesuai dengan data yang didapat dari ExDark.

BAB II

TINJAUAN PUSTAKA

2.1 Computer Vision

Computer vision adalah kemampuan komputer untuk memahami dan menganalisis data visual, terutama dari gambar atau video, dengan tujuan mengidentifikasi objek, pola, dan fitur lainnya. Ini adalah cabang ilmu komputer yang berkaitan dengan pengembangan sistem yang memungkinkan komputer untuk “melihat” dan memahami dunia sekitarnya melalui data visual yang diberikan kepadanya (Wibowo, 2016).

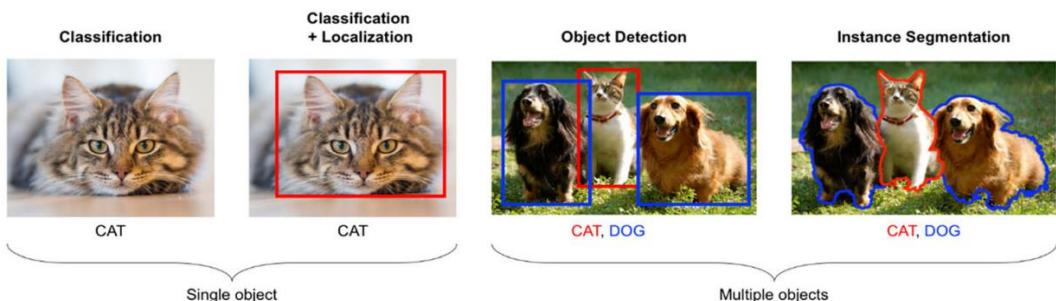
Computer Vision telah menjadi topik yang luas, meliputi mulai dari merekam data mentah hingga ekstraksi pola gambar dan interpretasi informasi. Ini menggabungkan konsep, teknik, dan ide dari pemrosesan gambar digital, pengenalan pola, kecerdasan buatan, dan grafika komputer. Sebagian besar tugas dalam *computer vision* berkaitan dengan proses mendapatkan informasi tentang peristiwa atau deskripsi dari gambar digital dan ekstraksi fitur. Tujuan utama *computer vision* adalah menciptakan model dan mengekstrak data serta informasi dari gambar, sementara pemrosesan gambar berkaitan dengan menerapkan transformasi komputasi pada gambar. *Computer vision* juga memiliki keterkaitan dengan *Human-Computer Interaction* (HCI), yang fokus pada desain, *interface*, dan interaksi antara manusia dan komputer (Wiley & Lucas, 2018).

Meskipun fungsinya serupa, sistem *computer vision* memiliki keterbatasan dibandingkan dengan mata manusia, terutama dalam hal sensitivitas parameter, kekuatan algoritma, dan akurasi hasil. Beberapa tantangan signifikan dalam teknik mereka termasuk pengukuran kinerja dan evaluasi algoritma untuk mencapai akurasi, kekuatan, atau skalabilitas. Ada upaya yang komprehensif untuk mengembangkan dan mengategorikan *computer vision* ke dalam berbagai aplikasi spesifik, seperti otomatisasi garis perakitan, sensor jarak jauh, robotika, komunikasi komputer dan manusia, serta alat bagi kaum difabel visual (Wiley & Lucas, 2018).

2.2 Analisis *Object Detection*

Deteksi objek merupakan proses penting dalam *computer vision* yang bertujuan untuk mengidentifikasi instansi dari objek-objek semantik tertentu dalam gambar digital dan video. Salah satu pendekatan umum dalam deteksi objek adalah dengan menciptakan sejumlah besar *candidate windows* yang kemudian dianalisis menggunakan fitur-fitur *Convolutional Neural Networks* (Voulovodimos, 2018).

Pendeteksian objek digunakan untuk mengklasifikasikan dan menemukan lokasi setiap objek dalam kotak pembatas atau *bounding box*. Penelitian oleh Lee (2017) menghasilkan sebuah metode bernama *sliding window* yang mampu mengurangi kompleksitas waktu yang sudah menjadi teknik dasar untuk mendeteksi objek. Dalam metode ini, sebuah *window* atau jendela dengan ukuran misalnya M x N, dipilih untuk mencari di atas gambar objektif. Pada awalnya, sebuah pengklasifikasi disiapkan pada serangkaian gambar *train*, menyebar di atas objek yang dituju untuk dideteksi sebagai satu kelas dan objek tidak teratur sebagai kelas yang berbeda. Tes yang termasuk dalam objek yang dituju untuk dideteksi disebut gambar positif, sementara sampel tidak teratur disebut gambar negatif (Lee *et al.*, 2017).



Gambar 2. 1 *Object detection* dengan satu atau lebih objek

Sumber: Diwan, *et al.* (2022)

2.3 Data *Preprocessing*

Pada domain *computer vision*, *data preprocessing* adalah langkah penting yang bertujuan untuk meningkatkan akurasi dan efisiensi model. Dalam banyak studi terbaru, seperti yang dijelaskan oleh Rouhbakhshmeghrazi & Alizadeh

(2023), teknik ini mencakup serangkaian metode seperti augmentasi gambar, peningkatan kontras, dan pengurangan *noise*, yang diterapkan untuk mempersiapkan gambar sebelum proses pelatihan model. Sesuai dasar tersebut, penelitian ini akan melakukan *data preprocessing* berupa normalisasi data dan augmentasi data.

2.3.1 Contrast Enhancement

Contrast Enhancement merupakan teknik dalam pemrosesan citra yang bertujuan untuk meningkatkan kualitas visual dengan memperbesar perbedaan intensitas antara objek dan latar belakangnya. Salah satu metode yang paling sering digunakan untuk meningkatkan kontras suatu citra adalah CLAHE.

Contrast Limited Adaptive Histogram Equalization (CLAHE) adalah teknik peningkatan gambar yang bertujuan untuk memperbaiki kontras pada area gambar dengan menggunakan histogram *equalization*, tetapi dengan batasan untuk menghindari amplifikasi *noise* secara berlebihan. Berbeda dengan *histogram equalization* biasa, CLAHE membagi gambar menjadi blok-blok kecil dan melakukan proses *equalization* secara lokal, yang membantu dalam mengatasi masalah *over-amplification* pada area terang atau gelap. Seperti yang dijelaskan oleh Yusuf dkk. (2024), CLAHE telah digunakan dalam berbagai aplikasi untuk meningkatkan visibilitas objek pada gambar yang memiliki variasi pencahayaan, seperti pada deteksi target menggunakan citra UAV. Teknik ini membantu meningkatkan detail gambar sambil mengurangi *noise* yang tidak diinginkan.

Selain itu, Manongga dkk. (2024) juga menunjukkan bahwa CLAHE digunakan dalam deteksi tanda jalan di kondisi pencahayaan rendah, dimana kombinasi CLAHE dan YOLOv7 membantu meningkatkan deteksi secara signifikan. CLAHE memastikan bahwa objek pada gambar yang memiliki kontras rendah tetap terdeteksi dengan jelas tanpa menghasilkan distorsi berlebih. Pendekatan ini sangat penting dalam aplikasi *real-time* yang memerlukan pengolahan gambar yang cepat dan akurat.

2.3.2 Downsampling

Dalam *preprocessing image data*, *downsampling* atau yang juga dikenal sebagai desimasi, merupakan proses fundamental dalam pengolahan citra digital yang bertujuan untuk mengurangi resolusi spasial suatu citra dengan mengurangi jumlah pikselnya (Paul, 2021). Proses ini menghasilkan representasi citra yang lebih kecil dan membutuhkan lebih sedikit sumber daya komputasi untuk diproses dan disimpan. Metode *downsampling* yang umum meliputi teknik seperti *nearest neighbor*, *bilinear*, dan *bicubic interpolation*. Masing-masing metode memiliki keunggulan dan keterbatasannya sendiri dalam hal kecepatan, kompleksitas, dan kualitas gambar yang dihasilkan.

Salah satu metode *downsampling* yang banyak digunakan saat ini adalah *Lanczos Resampling*. Metode ini merupakan salah satu metode interpolasi yang telah banyak diadopsi dalam proses downsampling karena kemampuannya menghasilkan citra dengan kualitas tinggi. Metode ini menggunakan fungsi sinc yang dilipat dengan sebuah *window Lanczos*, sehingga diperoleh *kernel support* terbatas yang mampu mengaproksimasi fungsi sinc (Sheibaniard & Yu, 2023). Secara matematis, kernel Lanczos didefinisikan sebagai:

$$L(x) = \begin{cases} \text{sinc}(\pi x)\text{sinc}\left(\frac{\pi x}{a}\right), & \text{jika } |x| < a \\ 0, & \text{sebaliknya} \end{cases} \quad (2.1)$$

Dimana x adalah jarak antara titik interpolasi dan sampel terdekat, sedangkan a merupakan parameter yang menentukan lebar *support kernel*. Fungi ini hanya bernilai ketika $|x| < a$ dan nol di luar interval tersebut, sehingga memiliki sifat lokal. Penggunaan fungsi sinc yang dilipat atau *windowed sinc* memungkinkan kernel Lanczos mendekati karakteristik ideal dari interpolasi sinyal secara teoritis, namun tetap efisien secara komputasi.

Lanczos resampling telah digunakan dalam berbagai aplikasi pengolahan citra, termasuk penajaman citra, analisis citra medis, dan penginderaan jauh. Dalam konteks penelitian terkini, Mottola dkk. (2021) menemukan bahwa interpolasi Lanczos paling efektif dalam mempertahankan informasi asli selama *resampling*.

citra CT untuk analisis radiomik pada pasien kanker ginjal. Selain itu, Li dkk. (2023) dalam studi berjudul *Learning Steerable Function for Efficient Image Resampling* mengusulkan modifikasi kernel Lanczos dengan menambahkan adaptabilitas orientasi, sehingga mampu mengoptimalkan proses *resampling* pada citra medis dan citra penginderaan jauh dengan hasil yang lebih tajam dan detail. Sementara itu, penelitian oleh SheibaniFard dan Yu (2023) yang mengembangkan representasi neural implisit untuk data volumetrik juga mengintegrasikan modul *downsampling* berbasis Lanczos, yang secara signifikan mengurangi beban pelatihan dan penggunaan memori GPU tanpa mengorbankan kualitas visual. Hasil-hasil penelitian tersebut menegaskan metode Lanczos menjadi pilihan utama dalam *downsampling* karena kemampuannya dalam mempertahankan informasi frekuensi tinggi dan menghasilkan citra berkualitas tinggi.

2.3.3 Augmentasi Data

Augmentasi data adalah teknik yang digunakan dalam bidang *computer vision* untuk meningkatkan variasi data latih tanpa harus mengumpulkan data baru. Teknik ini menjadi sangat penting dalam pelatihan model *machine learning*, khususnya *deep learning*, yang membutuhkan jumlah data yang besar untuk mencapai kinerja optimal. Augmentasi data membantu mengatasi masalah *overfitting* dengan menyediakan variasi yang lebih besar dalam data latih, sehingga model dapat lebih general dan mampu menangani data baru dengan lebih baik.

Beberapa teknik umum yang digunakan dalam augmentasi data meliputi rotasi, skala, translasi, *flipping*, pemotongan, perubahan kecerahan dan kontras, serta penambahan *noise*. Teknik ini memungkinkan model untuk belajar dari berbagai variasi gambar yang mungkin ditemui di dunia nyata. Studi terbaru oleh Shorten dan Khoshgoftaar (2019) menunjukkan bahwa augmentasi data dapat secara signifikan meningkatkan kinerja model *deep learning* dalam berbagai tugas *computer vision*. Selain itu, penelitian ini mengungkapkan bahwa kombinasi berbagai teknik augmentasi seringkali memberikan hasil terbaik.

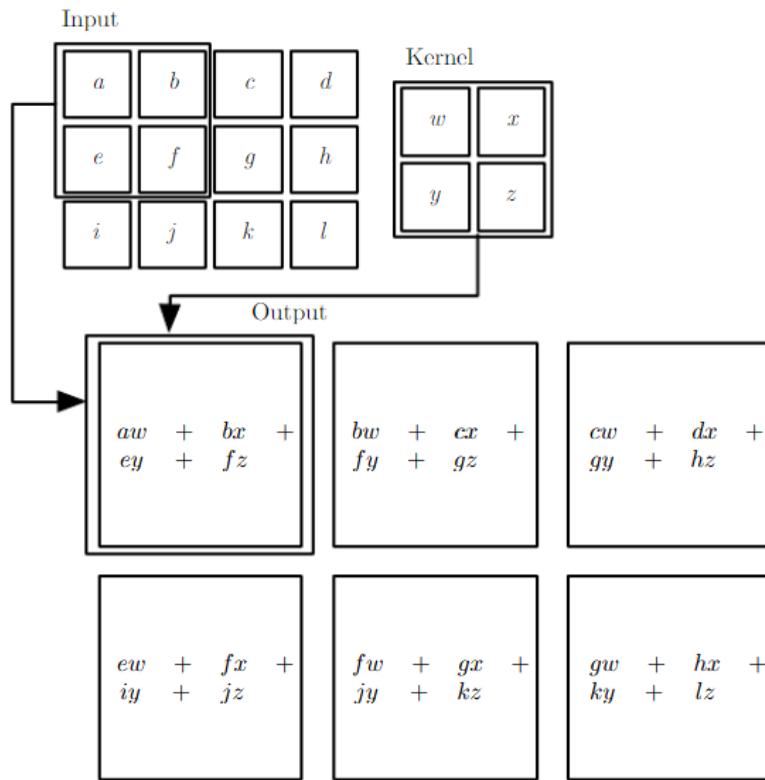
Selain itu, terdapat jenis lain dari augmentasi data, yaitu *mosaic data augmentation*. Teknik ini merupakan teknik augmentasi yang menggabungkan

potongan-potngna dari beberapa citra asli menjadi satu citra komposit. Teknik ini dirancang untuk menambah keragaman data *train* tanpa harus meningkatkan jumlah dataset secara signifikan. Sebagai contoh, dalam studi YOLOv4 yang dilakukan oleh Sandy dkk. (2024) , *mosaic augmentation* diterapkan dengan car amengintegrasikan empat citra secara simultan, sehingga menghasilkan variasi skala, latar belakang, dan konteks yang lebih kaya. Hasil penelitian tersebut menunjukkan bahwa penggunaan *mosaic data augmentation* mampu meningkatkan performa deteksi terutama dalam hal akurasi pendekripsi objek kecil dan membuat model lebih tahan terhadap variasi dalam *train set*.

2.4 *Convolutional Neural Network*

Jaringan saraf konvolusional atau *Convolutional Neural Network (CNN)* merupakan sebuah konsep yang diperkenalkan oleh LeCun dkk. pada tahun 1989. Penelitian tersebut mengacu pada penggunaan metode *backpropagation* yang diaplikasikan pada *Handwritten Zip Code* untuk *U.S. Postal Service*. Data yang digunakan pada penelitian tersebut berjumlah 9298 gambar tulisan tangan untuk sebuah kode pos yang ditulis oleh banyak orang yang berbeda, menggunakan berbagai macam ukuran, gaya penulisan, dan instrumen, dengan tingkat penulisan yang bervariasi. Pada penelitian tersebut, LeCun berhasil membuat sebuah jaringan tunggal yang dapat mempelajari seluruh operasi pengenalan, mulai dari gambar karakter yang dinormalisasi hingga klasifikasi akhir dan diberi nama *convolution networks* atau *convolutional neural network*.

Sesuai namanya, jaringan saraf konvolusional menunjukkan bahwa jaringan tersebut menggunakan operasi matematika yang disebut konvolusi. Konvolusi adalah jenis operasi linier khusus. Jaringan *convolutional* hanyalah jaringan saraf yang menggunakan konvolusi sebagai pengganti perkalian matriks umum di setidaknya satu lapisannya (Goodfellow, 2016).



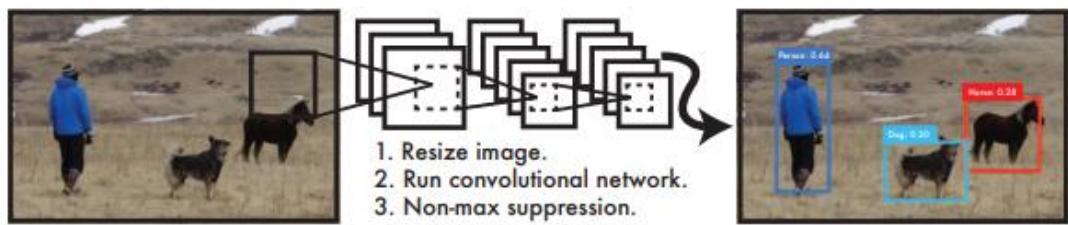
Gambar 2. 2 Proses Konvolusi 2D

Sumber: Goodfellow (2016)

Saat ini, CNN telah menjadi konsep yang sangat penting dalam bidang *computer vision* terutama dalam analisis citra. Arsitektur CNN meniru cara manusia memproses visual dengan mengadopsi konsep seperti *filter receptive field* dan penggabungan fitur hierarkis. Struktur CNN terdiri dari beberapa lapisan yang bekerja secara bersama-sama untuk mempelajari representasi fitur dari gambar input. Contohnya adalah lapisan konvolusi yang menggunakan *filter* atau *kernel* untuk memindai gambar dan mengekstrak fitur-fitur penting seperti tepi, tekstur, atau pola yang lebih kompleks. Setelah itu, lapisan aktivasi diterapkan untuk memperkenalkan non-linearitas ke dalam jaringan, yang membantu dalam pembelajaran representasi fitur yang lebih kompleks (Li *et al.*, 2021).

2.5 You Only Look Once

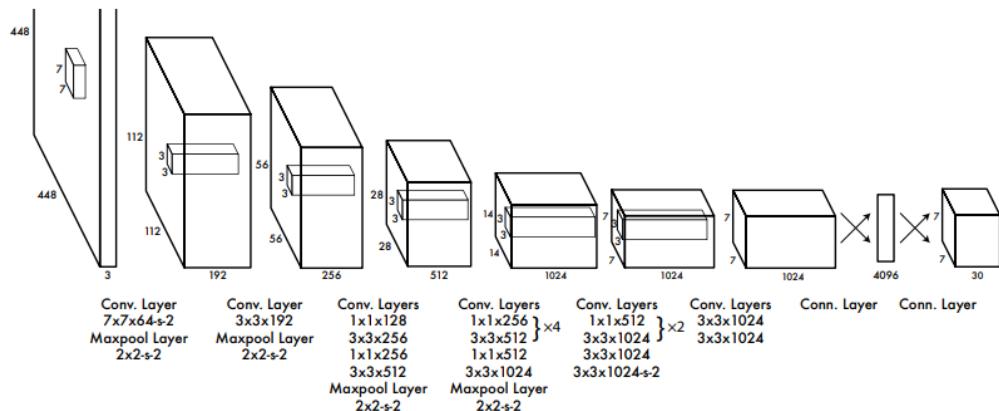
You Only Look Once (YOLO) adalah sebuah model deteksi objek revolusioner yang pertama kali diperkenalkan oleh Joseph Redmon dan Santosh Divvala pada tahun 2015. YOLO merombak pendekatan tradisional dalam deteksi objek dengan mengusulkan pendekatan yang lebih efisien dan cepat.



Gambar 2.3 Sistem deteksi objek dengan YOLO

Sumber: Redmon, et al. (2015)

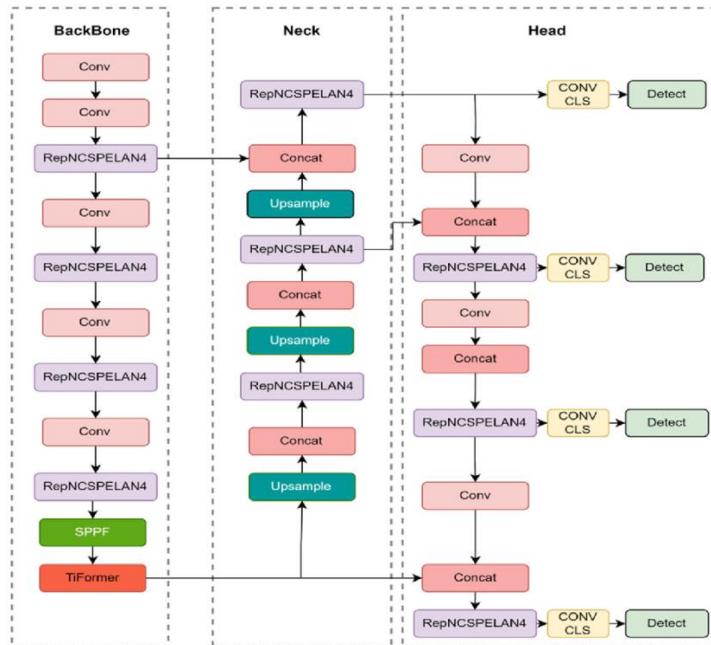
Arsitektur YOLO terdiri dari beberapa lapisan konvolusi yang menghasilkan *grid* dari prediksi *bounding box* dan probabilitas kelas. Lapisan-lapisan konvolusi ini digabungkan dengan lapisan-lapisan pooling untuk menghasilkan representasi fitur dari gambar input. Kemudian, menggunakan *fully connected layer*, prediksi untuk *bounding box* dan probabilitas kelas dilakukan secara bersamaan (Redmon et al., 2019).



Gambar 2.4 Arsitektur YOLO

Sumber: Redmon, et al. (2015)

Salah satu keunggulan utama dari YOLO adalah kemampuannya untuk mendeteksi objek dengan cepat dalam waktu *realtime*, karena memproses seluruh gambar dalam satu langkah. Hal ini membuatnya sangat cocok untuk aplikasi seperti deteksi objek dalam video atau pada perangkat dengan komputasi yang terbatas (Wang & Liao, 2024).



Gambar 2.5 Arsitektur YOLOv9

Sumber: Li, et al. (2024)

Seiring berkembangnya ilmu pengetahuan dan teknologi, muncul berbagai varian dari YOLO mulai dari YOLO9000 (YOLOv2), IA-YOLO, dan juga YOLOv9. YOLOv9 merupakan pengembangan dari sistem deteksi objek berbasis *Convolutional Neural Network* (CNN) yang mengombinasikan dua komponen utama, yaitu *Programmable Gradient Information* (PGI) dan *Generalized Efficient Layer Aggregation Network* (GELAN). Arsitektur YOLOv9 dimulai dengan pemrosesan input melalui serangkaian lapisan konvolusi pada GELAN, yang dirancang untuk memaksimalkan efisiensi jalur gradien dan parameter jaringan. PGI berperan dalam mengatasi kehilangan informasi yang sering terjadi selama proses *feedforward* dalam jaringan dalam (*information bottleneck*) dengan memperkenalkan mekanisme *reversible branch*. Mekanisme ini memastikan bahwa gradien yang diperoleh untuk memperbarui bobot jaringan optimal, sehingga

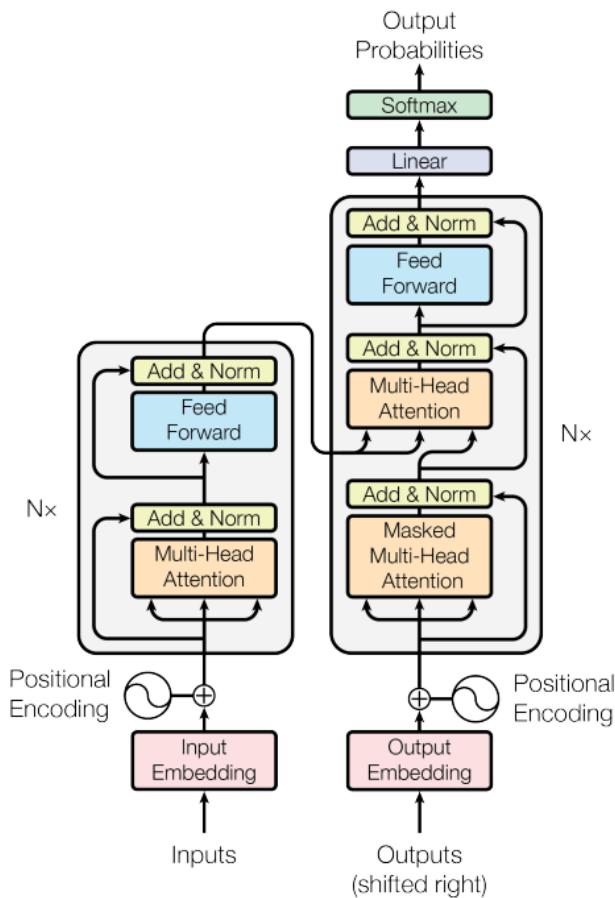
jaringan dapat mempertahankan informasi penting dari fitur yang lebih dalam. Dalam YOLOv9, arsitektur GELAN menggunakan blok konvolusi yang dirancang untuk mencapai keseimbangan antara akurasi, *computational complexity*, dan *inference time*. Di tahap akhir, prediksi dilakukan menggunakan *head* deteksi yang telah dioptimalkan untuk efisiensi deteksi objek *multiscale*. Kombinasi PGI dan GELAN memungkinkan YOLOv9 untuk mencapai kinerja deteksi objek yang unggul dibandingkan dengan model sebelumnya, dengan parameter yang lebih sedikit namun akurasi yang lebih tinggi, khususnya pada dataset MS COCO (Wang *et al.*, 2024).

Walau demikian, varian terbaru YOLO saat ini adalah YOLOv10 yang sudah menggunakan mekanisme Transformer di dalamnya. Setiap adanya kemunculan baru dari varian YOLO ini membuat YOLO menjadi lebih optimal yang secara spesifik dibutuhkan oleh pengembangnya. Jadi, bukan berarti karena YOLOv10 adalah versi terbaru, varian tersebut akan secara pasti mengungguli varian-varian lainnya dalam seluruh aspek maupun tugas yang dikerjakan, melainkan varian-varian dari YOLO ini akan menyesuaikan antara *trade-off* yang dibutuhkan. (Wang & Liao, 2024).

2.6 Transformer

Transformer merupakan sebuah arsitektur *neural network* yang menjadi *State of The Art* pada bidang *natural language processing* (NLP) sejak diperkenalkan oleh Vaswani dkk. (2017). Arsitektur Transformer menyajikan pendekatan yang berbeda dengan model *recurrent* atau CNN tradisional yang sebelumnya dominan dalam NLP.

Pusat dari Transformer adalah mekanisme *attention*, yang memungkinkan model untuk memberikan perhatian yang berbeda pada setiap bagian dari input *sequence*, terlepas dari jaraknya. Ini memungkinkan Transformer untuk memahami konteks semantik secara global dari sebuah *sequence*, yang merupakan keuntungan besar dalam NLP.

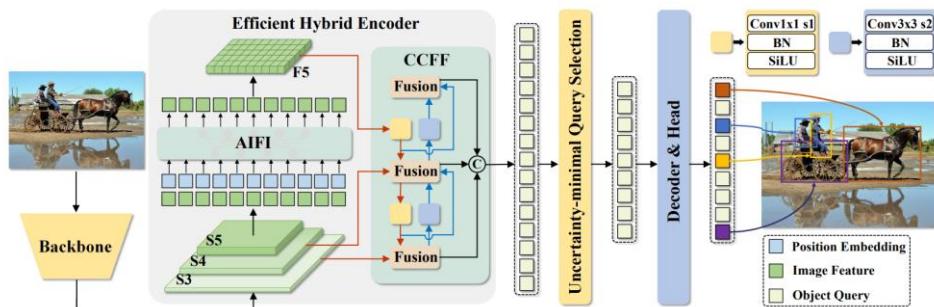


Gambar 2.6 Arsitektur Transformer

Sumber: Vaswani, et al. (2017)

Transformer terdiri dari dua komponen utama yaitu *encoder* dan *decoder*. *Encoder* bertugas untuk mengubah *input sequence* menjadi representasi fitur yang semantik, sedangkan *decoder* bertanggung jawab untuk menghasilkan *output sequence* dari representasi yang diberikan oleh *encoder*. Masing-masing *encoder* dan *decoder* terdiri dari beberapa *layer* yang terdiri dari *sub-layer*. *Sub-layer* dalam *encoder* terdiri dari *multi-head self-attention layer*, diikuti oleh *feed-forward neural network layer*. Di sisi lain, *sub-layer* dalam *decoder* memiliki *multi-head self-attention layer*, diikuti oleh *masked multi-head self-attention layer* dan *feed-forward neural network layer*. *Sub-layer* ini bekerja bersama-sama untuk memperoleh representasi fitur yang semantik dan menghasilkan *output* yang akurat (Vaswani et al. , 2017).

Selain diterapkan dalam NLP, Transformer juga telah digunakan secara luas dalam bidang *computer vision*, khususnya dalam tugas-tugas seperti segmentasi gambar, pemrosesan gambar, dan pengenalan objek. Penggunaan Transformer dalam domain non-NLP menunjukkan fleksibilitas dan kemampuan adaptasi yang luar biasa dari arsitektur tersebut (Zhang *et al.*, 2021). Perluasan Transformer ke dalam domain *computer vision* ini salah satunya dipelopori oleh Carion dkk. di tahun 2020. Carion menciptakan suatu rancangan transformer yang dapat memahami konteks secara spasial dan diberi nama *DEtection TRansformer* atau DETR. DETR ini dapat menyaingi model *Faster R-CNN* secara signifikan dalam objek-objek besar berkat mekanisme *self-attention*-nya (Carion *et al.*, 2020). DETR ini kemudian dikembangkan oleh Zhao di tahun 2024 yang memungkinkan penggunaannya menjadi *realtime* dengan beberapa modifikasi seperti penghapusan mekanisme NMS, penerapan metode *Efficient Hybrid Encoder*, *Uncertainty-Minimal Query Selection*, dan menciptakan *Flexible Speed Tuning* yang dapat mengatur jumlah *decoder layers* sesuai kebutuhan tanpa perlu *retraining*. Varian *realtime* dari DETR yang dikembangkan oleh Zhao ini diberi nama *RealTime-DETR* atau RT-DETR (Zhao *et al.*, 2024)



Gambar 2.7 Arsitektur RT-DETR

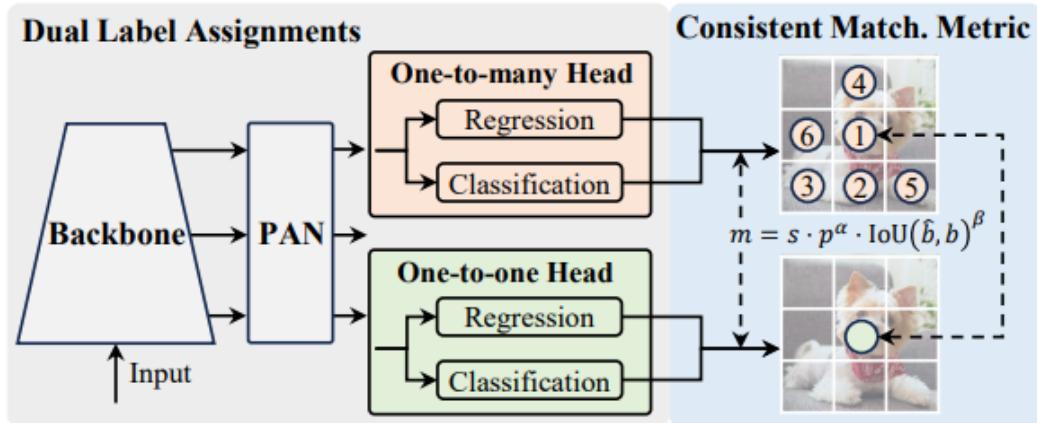
Sumber: Zhao, *et al.* (2024)

RT-DETR merupakan salah satu pengembangan dari arsitektur Transformer untuk deteksi objek secara *realtime*, dirancang untuk mengatasi keterbatasan model YOLO yang memerlukan *Non-Maximum Suppression* (NMS) dalam tahap *post-processing*. Arsitektur RT-DETR terdiri dari tiga komponen utama, yaitu *backbone*, *efficient hybrid encoder*, dan *decoder Transformer*. Fitur *multiscale* dari *backbone* diproses oleh *hybrid encoder* yang terdiri dari dua modul, yaitu *Attention-based*

Intra-scale Feature Interaction (AIFI) untuk interaksi fitur *intrascale*, dan *CNN-based Cross-scale Feature Fusion* (CCFF) untuk *cross-scale fusion*. RT-DETR menggunakan mekanisme *uncertainty-minimal query selection* untuk memilih fitur *encoder* dengan ketepatan yang tinggi sebagai *query* awal untuk *decoder*, yang kemudian secara iteratif mengoptimalkan prediksi objek. Keunggulan utama dari RT-DETR adalah kemampuannya untuk menyelaraskan kecepatan dan akurasi, menghilangkan kebutuhan akan NMS, serta mendukung *tuning* kecepatan dengan menyesuaikan jumlah lapisan *decoder* tanpa perlu pelatihan ulang, menjadikannya solusi yang fleksibel untuk berbagai skenario deteksi *realtime* (Zhao *et al.*, 2024).

2.7 CNN-Transformer (YOLOv10)

Sesuai namanya, metode berbasis CNN-Transformer menggabungkan *Convolutional Neural Network* (CNN) dengan mekanisme Transformer. CNN dikenal unggul dalam mengekstraksi fitur spasial dari citra melalui jaringan konvolusi bertumpuk, sementara Transformer, yang awalnya digunakan dalam bidang *Natural Language Processing* (NLP), mampu menangkap hubungan jangka panjang dengan memanfaatkan mekanisme *self-attention*. Kombinasi ini bertujuan untuk mengatasi keterbatasan masing-masing metode, yaitu CNN yang terbatas dalam menangkap dependensi jarak jauh dan Transformer yang kurang optimal untuk pengolahan citra tanpa bantuan lapisan konvolusi. Metode *hybrid* ini memungkinkan sistem untuk melakukan deteksi objek secara lebih akurat, terutama dalam lingkungan yang kompleks seperti pencahayaan rendah atau perspektif yang variatif.



Gambar 2.8 Arsitektur YOLOv10

Sumber: Wang, *et al.* (2024)

YOLOv10 memperkenalkan serangkaian peningkatan signifikan dalam arsitektur deteksi objek berbasis CNN yang bertujuan untuk meningkatkan efisiensi dan akurasi. Arsitektur YOLOv10 mengimplementasikan strategi *dual label assignments* untuk pelatihan tanpa *Non-Maximum Suppression* (NMS), yang menghilangkan kebutuhan *post-processing* dalam inferensi. Ini memungkinkan YOLOv10 mencapai deteksi *real-time* yang lebih efisien dengan menggabungkan kekuatan dari penugasan *one-to-many* dan *one-to-one*. Selain itu, desain arsitektur yang mengedepankan efisiensi-akurasi mencakup penggunaan *lightweight classification head*, *downsampling* yang terpisah antara *spatial* dan *channel*, serta *rank-guided block* untuk mengurangi redundansi komputasi tanpa mengorbankan kinerja. YOLOv10 juga memanfaatkan large-kernel convolution dan modul *partial self-attention* untuk memperluas *receptive field* dan meningkatkan kemampuan representasi global. Hasilnya, YOLOv10 berhasil melampaui model YOLO sebelumnya dan model *state-of-the-art* lainnya dalam hal latensi dan jumlah parameter, dengan tetap mempertahankan atau meningkatkan akurasi pada dataset MS COCO (Wang *et al.*, 2024).

2.8 Hyperparameter Tuning

Hyperparameter adalah parameter yang digunakan untuk mengontrol proses pelatihan model dan tidak dipelajari langsung dari data (Goodfellow *et al.*,

2016). Berbeda dengan parameter model yang *di-update* selama pelatihan, seperti bobot dalam jaringan saraf, *hyperparameter* ditentukan sebelum proses pelatihan dimulai dan berperan penting dalam menentukan performa model. Beberapa contoh *hyperparameter* yang umum meliputi *Learning Rate*, *Batch Size*, dan *Epochs*.

Hyperparameter Tuning adalah proses mengoptimalkan nilai *hyperparameter* agar model dapat mencapai performa terbaik pada data tertentu (Bergstra & Bengio, 2012). Proses *tuning* ini biasanya melibatkan eksperimen dengan berbagai kombinasi nilai *hyperparameter*, menggunakan teknik seperti *grid search* atau *random search*, untuk menemukan konfigurasi yang paling sesuai dengan data dan tugas yang dihadapi. *Hyperparameter tuning* sangat penting karena *hyperparameter* yang tidak tepat dapat menyebabkan model *underfitting* ataupun *overfitting*, dimana model tidak mampu menangkap pola yang kompleks dari data atau justru terlalu menyesuaikan diri dengan data latih sehingga kinerjanya buruk pada data uji. Salah satu cara untuk mengoptimalkan performa model dengan *hyperparameter tuning* adalah mengkonfigurasi parameter *learning rate*.

Learning Rate menentukan seberapa besar perubahan bobot dalam jaringan saraf terjadi setiap kali ada *update* berdasarkan *error* yang dihitung (Smith, 2017). *Learning Rate* yang terlalu tinggi dapat menyebabkan model menjadi tidak stabil karena langkah-langkah perbaikan yang terlalu besar sehingga tidak pernah mencapai titik minimum *error*. Sebaliknya, *Learning Rate* yang terlalu rendah dapat menyebabkan pelatihan model menjadi sangat lambat, memerlukan lebih banyak waktu dan sumber daya untuk mencapai konvergensi, dan bahkan bisa terjebak pada *local minimum* (Goodfellow *et al.*, 2016).

Batch Size adalah *hyperparameter* lain yang memiliki pengaruh signifikan terhadap proses pelatihan model. *Batch Size* menentukan jumlah contoh data yang diproses sebelum bobot model diperbarui (Masters & Luschi, 2018). Pada umumnya, *Batch Size* yang lebih besar memungkinkan pelatihan yang lebih cepat dan stabil karena bisa lebih efisien dalam pemrosesan komputasi, terutama saat menggunakan GPU. Namun, *Batch Size* yang besar juga bisa mengurangi kemampuan model untuk keluar dari *local minimum*, yang dapat menyebabkan kinerja yang buruk pada generalisasi (Keskar *et al.*, 2017). Sebaliknya, *Batch Size*

yang kecil seringkali menghasilkan variabilitas yang lebih tinggi dalam pembaruan bobot, yang bisa membantu model menghindari *local minimum*, meskipun dengan risiko pelatihan yang lebih lama.

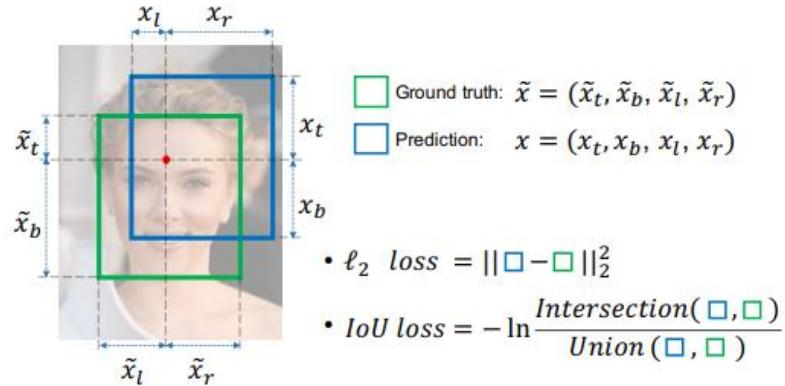
Salah satu parameter penting dalam tugas *object detection* adalah *confidence*. Parameter ini mengukur seberapa yakin model terhadap keberadaan objek di dalam setiap *bounding box* yang dihasilkan, semakin tinggi nilai *confidence*, semakin besar keyakinan model akan deteksi tersebut. Beberapa penelitian terdahulu secara konsisten menetapkan nilai *default confidence* sebesar 0,25 untuk menyeimbangkan *trade-off* antara *precision* dan *recall*. Gao dkk. (2024) dalam studi pengembangan YOLOv8 menggunakan *threshold confidence* 0,25 sebagai batas deteksi awal. Baik dkk. (2025) dalam evaluasi YOLOv7 pada dataset *traffic controller* juga menerapkan *default confidence* 0,25, serta Guemas dkk. (2024) pada implementasi RT-DETR menetapkan *cutoff confidence* 0,25 untuk menjaga konsistensi metrik performa. Dengan demikian, penerapan nilai *confidence* 0,25 telah teruji baik pada arsitektur *CNN-based* maupun *Transformer-based*.

2.9 Loss Function

Loss function merupakan komponen penting dalam pelatihan model terutama untuk *deep learning* dalam mendeteksi objek. *Loss function* digunakan untuk mengukur seberapa baik model memprediksi hasil yang diinginkan selama proses pelatihan. Secara umum, *object detection* memerlukan dua *loss function*, yaitu *loss function* untuk klasifikasi objek itu sendiri, dan yang *loss function* untuk *bounding box regression*. Salah satu *loss function* yang paling umum digunakan untuk *bounding box regression* adalah *Intersection over Union* (IoU), yang mengukur tumpang tindih antara prediksi *box* dengan *ground truth*. IoU loss ini pertama kali diperkenalkan oleh Yu (2016) untuk mengatasi permasalahan yang ada pada *loss function* sebelumnya yaitu ℓ_2 loss (Yu *et al.*, 2016).

Kelemahan utama dari ℓ_2 loss terletak pada koordinat sebuah *bounding box* yang dioptimalkan sebagai empat variabel yang independen. Asumsi ini mengabaikan fakta bahwa batas-batas suatu objek sangat berkorelasi. Hal ini dapat

mengakibatkan beberapa kasus kegagalan di mana satu atau dua batas dari *bounding box* yang diprediksi sangat dekat dengan *ground truth*, tetapi keseluruhan *bounding box* tidak dapat diterima. IoU menyelesaikan masalah ini dengan menganggap satu *bounding box* adalah sebagai satu unit. Dengan begitu IoU loss dapat memberikan prediksi yang lebih akurat daripada ℓ_2 loss (Yu *et al.* 2016). Secara umum, perbedaan ℓ_2 loss dan IoU loss dapat dilihat pada Gambar 2.9.



Gambar 2.9 Perbedaan ℓ_2 loss dan IoU loss

Sumber: Yu, *et al.* (2016)

Namun, IoU loss ini hanya digunakan untuk memprediksi *bounding box* pada *ground truth* dan belum dapat mengklasifikasikan kelas objek. Agar dapat mengklasifikasikan kelas objek, terdapat sebuah *loss function* bernama *Focal Loss* (FL) yang menerapkan konsep modulasi pada *cross entropy loss* untuk memfokuskan pembelajaran pada contoh-contoh yang sulit dan mengurangi bobot dari *easy negatives* (Lin *et al.*, 2017). Persamaan *Focal Loss* ini dijelaskan sebagai berikut.

$$\mathbf{FL}(p) = -(1 - p_t)^\gamma \log(p_t), p_t = \begin{cases} p, & \text{when } y = 1 \\ 1 - p, & \text{when } y = 0 \end{cases} \quad (2.2)$$

Dimana $y \in \{1, 0\}$ menentukan kelas *ground-truth* dan $p \in [0, 1]$ menunjukkan estimasi probabilitas untuk kelas dengan label $y = 1$, γ adalah parameter fokus yang dapat dilakukan *tuning*. Secara spesifik, FL terdiri dari bagian *cross entropy – log(pt)* dan bagian *dynamically scaling factor* $(1 - pt)^\gamma$, dimana

scaling factor ($1 - pt$) γ secara otomatis menurunkan bobot kontribusi dari *easy examples* selama pelatihan dan dengan cepat memfokuskan model pada *hard examples*.

Focal Loss ini kembali dikembangkan menjadi *Generalized Focal Loss* (GFL) yang menggeneralisasi *Focal Loss* dari diskrit {1,0} ke versi kontinu. GFL dapat dikhususkan menjadi *Quality Focal loss* (QFL) dan *Distribution Focal Loss* (DFL), di mana QFL mendorong untuk mempelajari representasi gabungan yang lebih baik dari kualitas klasifikasi dan lokalisasi, dan DFL memberikan estimasi *bounding box* yang lebih informatif dan tepat dengan memodelkan lokasinya sebagai *General Distributions* (Li *et al.*, 2020).

IoU juga dikembangkan menjadi *Generalized IoU* (GIoU) *loss*. GIoU adalah perbaikan dari IoU *loss* yang digunakan untuk menghitung seberapa baik *bounding box* prediksi sesuai dengan *ground truth*. IoU biasa hanya mengukur tumpang tindih antara dua kotak, GIoU memperbaiki kelemahan IoU ketika tidak ada tumpang tindih antara prediksi dan *ground truth*. GIoU memperkenalkan area pembungkus minimal yang mencakup kedua kotak tersebut dan menghitung rasio antara area pembungkus dan selisih area antara kotak prediksi dengan *ground truth*. Dengan pendekatan ini, GIoU memberikan nilai yang lebih representatif bahkan ketika *bounding box* prediksi dan *ground truth* tidak bertumpang tindih, yang sering terjadi pada awal pelatihan model deteksi objek (Rezatofighi *et al.*, 2019). Persamaan GIoU ini dijelaskan sebagai berikut.

$$GIoU = IoU - \frac{|C \setminus (A \cup B)|}{|C|} \quad (2.3)$$

Dimana A dan B merupakan dua bentuk *convex* sembarang dan C adalah bentuk *convex* terkecil yang dapat menyelimuti kedua bentuk A dan B secara keseluruhan.

Seiring dengan perkembangan *loss function* untuk prediksi *bounding box*, terdapat pula kemajuan dalam *loss function* untuk *object classification* yang dikenal sebagai *Binary Cross Entropy with Logits Loss* (*BCEWithLogitsLoss*). *BCEWithLogitsLoss* menggabungkan dua operasi penting yaitu *sigmoid activation*

dan *binary cross-entropy loss* dalam satu langkah yang lebih efisien. Pada dasarnya, *loss* ini digunakan untuk mengukur kesalahan antara prediksi model (*logits*) dan target yang diinginkan (label biner). Pada fungsi BCE biasa, tahap yang dilakukan pertama kali adalah aktivasi sigmoid pada *output* model untuk mengubah nilai *logit* menjadi probabilitas antara 0 dan 1. Kemudian, probabilitas ini dibandingkan dengan target menggunakan *binary cross-entropy loss*, yang menghitung seberapa jauh perbedaan antara distribusi prediksi dan target. Namun, dalam *BCEWithLogitsLoss*, proses aktivasi sigmoid dilakukan secara internal, sehingga mengurangi risiko masalah numerik seperti *gradient vanishing* yang sering menggunakan sigmoid secara terpisah (Ruby & Yendapalli, 2020). Persamaan *BCEWithLogitsLoss* ini dijelaskan sebagai berikut.

$$BCEWithLogitsLoss = -(y \log(\sigma(x)) + (1 - y) \log(1 - \sigma(x))) \quad (4)$$

Dimana y adalah label sebenarnya (0 atau 1), x adalah *logit* atau *output* dari model, dan $\sigma(x)$ adalah fungsi sigmoid dari *logit* tersebut.

2.10 Metrik Evaluasi Model

Metrik evaluasi model digunakan untuk mengukur kinerja model. Dalam penelitian yang berkaitan dengan deteksi objek pada kondisi pencahayaan rendah, pemilihan metrik evaluasi model menjadi elemen krusial dalam menentukan performa model secara akurat. *F1-Score*, *Precision*, *Recall*, *Accuracy*, dan *Mean Average Precision* (mAP) adalah beberapa metrik yang sering digunakan dalam berbagai penelitian untuk menilai efektivitas model deteksi objek. *F1-Score* menggabungkan *Precision* dan *Recall* dalam satu ukuran, memberikan bobot yang seimbang antara *false positives* dan *false negatives*, yang sangat penting dalam kasus data yang tidak seimbang (Chicco & Jurman, 2020). *Precision* menggambarkan proporsi dari prediksi positif yang benar-benar akurat, sementara *Recall* mengukur sejauh mana model dapat mendeteksi semua contoh positif yang ada, keduanya adalah metrik yang esensial dalam konteks pengenalan objek yang

sering kali mengalami masalah *misalignment* atau *noise* pada citra (Powers, 2020).

Persamaan untuk metrik-metrik tersebut dijelaskan sebagai berikut,

$$precision = \frac{TP}{TP + FP}, \quad (2.5)$$

$$recall = \frac{TP}{TP + FN}, \quad (2.6)$$

$$F1 = \frac{2 \times precision \times recall}{precision + recall}, \quad (2.7)$$

$$accuracy = \frac{TP + TN}{TP + FN + TN + FP} \quad (2.8)$$

Dimana *TP* adalah *True Positive*, yaitu jumlah data di kelas positif yang berhasil diprediksi dengan benar sebagai positif, dan *TN* adalah *True Negative*, yaitu jumlah data di kelas negatif yang juga diprediksi dengan benar sebagai negatif. Sebaliknya, *FN* (*False Negative*) mengacu pada kasus di mana data yang seharusnya berada di kelas positif justru diprediksi sebagai negatif, dan *FP* (*False Positive*) terjadi ketika data dari kelas negatif salah diklasifikasikan sebagai positif.

Metrik-metrik dasar ini sangat penting dalam mengevaluasi performa model karena memberikan informasi rinci tentang jenis kesalahan yang dibuat oleh model. *Precision*, misalnya, sangat sensitif terhadap *False Positives* dan berguna dalam skenario di mana kesalahan deteksi positif (misalnya, mendekksi objek yang sebenarnya tidak ada) harus diminimalkan. *Recall*, di sisi lain, lebih relevan dalam situasi di mana tidak terlewatkan satu pun deteksi benar sangat penting, seperti pada sistem keamanan atau aplikasi medis, di mana *False Negatives* (objek yang ada tetapi tidak terdeteksi) lebih berbahaya.

Accuracy sering kali digunakan sebagai metrik dasar dalam evaluasi model, tetapi metrik ini bisa menyesatkan terutama pada dataset yang tidak seimbang, di mana jumlah contoh dari satu kelas jauh lebih besar dibandingkan kelas lainnya (Saito *et al.*, 2015). Oleh karena itu, untuk deteksi objek dalam citra, *mean Average Precision* (mAP) dianggap sebagai metrik yang lebih komprehensif karena memperhitungkan akurasi deteksi untuk setiap kelas objek, dan memberikan gambaran keseluruhan performa model dalam mendekksi berbagai jenis objek

(Zhao *et al.*, 2019). Penggunaan mAP dalam tugas deteksi objek telah menjadi standar industri, terutama dalam kompetisi seperti COCO dan PASCAL VOC, yang mengharuskan model tidak hanya mendekripsi objek dengan benar, tetapi juga menentukan lokasi dengan presisi tinggi (Everingham *et al.*, 2015). Persamaan mAP ini dijelaskan sebagai berikut,

$$mAP = \frac{1}{n} \sum_{k=1}^n AP_k \quad (2.9)$$

Dimana n adalah jumlah kelas yang ada dan AP_k adalah rerata nilai *precision* dari kelas k .

2.11 Pareto Frontier

Pareto Frontier, atau juga dikenal dengan Pareto Optimum, merupakan suatu *framework* yang digunakan dalam *multi-objective optimization* dan berasal dari teori ekonomi. Dalam domain *machine learning*, Pareto Frontier juga digunakan untuk mengidentifikasi solusi optimal ketika terdapat lebih dari satu tujuan yang saling bertentangan, seperti akurasi model dan waktu inferensi.

Solusi pada Pareto Frontier adalah solusi yang dinamakan Pareto Optimal, yang berarti tidak ada solusi lain yang dapat meningkatkan satu tujuan tanpa merugikan tujuan lainnya. Dengan demikian, Pareto Frontier memungkinkan identifikasi model yang mencapai keseimbangan optimal antara berbagai metrik (Benmeziane *et al.*, 2022).

Dalam *multi-objective optimization*, tujuan yang sering dicapai adalah untuk meminimalkan atau memaksimalkan beberapa fungsi objektif secara bersamaan. Dalam optimasi dua *objective*, $f_1(x)$ dan $f_2(x)$, solusi x^* dianggap Pareto Optimal jika tidak ada solusi x yang memenuhi:

$$f_1(x) \leq f_1(x^*) \text{ dan } f_2(x) \leq f_2(x^*) \text{ dengan } f_1(x) < f_1(x^*) \text{ atau } f_2(x) < f_2(x^*) \quad (2.10)$$

Dengan kata lain, solusi x^* adalah optimal jika setiap perbaikan pada salah satu tujuan akan menyebabkan kerugian pada tujuan lainnya. Sekumpulan solusi

Pareto optimal ini membentuk Pareto Frontier, yang menggambarkan *trade-off* antara tujuan-tujuan yang saling bertentangan tersebut (Deb, 2001). Konsep Pareto Frontier telah terbukti sangat berguna dalam berbagai aplikasi termasuk *machine learning*. Salah satu contoh yang menonjol adalah dalam *Neural Architecture Search* (NAS), dimana Pareto Frontier digunakan untuk mengidentifikasi arsitektur yang optimal dalam hal latensi dan akurasi inferensi (Ye *et al.*, 2024). Selain itu, penelitian juga menunjukkan potensi Pareto Frontier dalam mengoptimalkan beberapa tujuan secara simultan, seperti *fairness*, *privacy*, dan *utility* dalam *modeling* (Yaghini *et al.*, 2023).

2.12 Optimizer

Optimizer adalah sebuah konsep dalam *deeplearning* yang berfungsi untuk meminimalkan *loss function* dengan cara memperbarui bobot atau *weights* jaringan secara iteratif. Menurut Cacciola dkk. (2023) proses pembaruan parameter melalui *optimizer* sangat menentukan laju konvergensi dan kestabilan pelatihan, terutama ketika menangani dataset berukuran besar. Li dan Liang (2018) menunjukkan bahwa *optimizer*, terutama metode *Stochastic Gradient Descent* (SGD), memiliki efek regularisasi implisit yang mendorong model menuju solusi yang lebih general dan mencegah *overfitting*. Wang dan Wu (2024) menjelaskan *Optimizer SGD* ini dengan persamaan berikut:

$$\theta_{t+1} = \theta_t - \eta \nabla \theta L(\theta_t; xi, yi) \quad (2.11)$$

Persamaan SGD di atas memperlihatkan mekanisme dasar dari optimasi model melalui pembaruan parameter secara stokastik. Variabel θ_t merepresentasikan sekumpulan bobot atau parameter jaringan pada iterasi ke- t , yang bertugas mengatur perilaku model. *Learning rate* η menentukan seberapa besar perubahan atau *step* yang diambil dalam setiap iterasi, nilai yang terlalu besar dapat menyebabkan konvergensi tidak stabil, sedangkan nilai yang terlalu kecil dapat membuat proses pelatihan berjalan terlalu lambat. Gradien $\nabla \theta L(\theta_t; xi, yi)$ adalah vektor gradien yang menggambarkan arah peningkatan tercepat dari *loss function* ketika parameter berada pada kondisi θ_t . Oleh karena itu, dengan mengurangkan gradien ini, SGD secara bertahap menyesuaikan parameter agar fungsi kerugian berkurang. Perhitungan gradien dilakukan berdasarkan satu atau

sekumpulan kecil data acak (*minibatch*), sehingga metode ini efisien dalam menghadapi dataset berukuran besar dan memiliki efek regularisasi implisit.

BAB III

METODE PENELITIAN

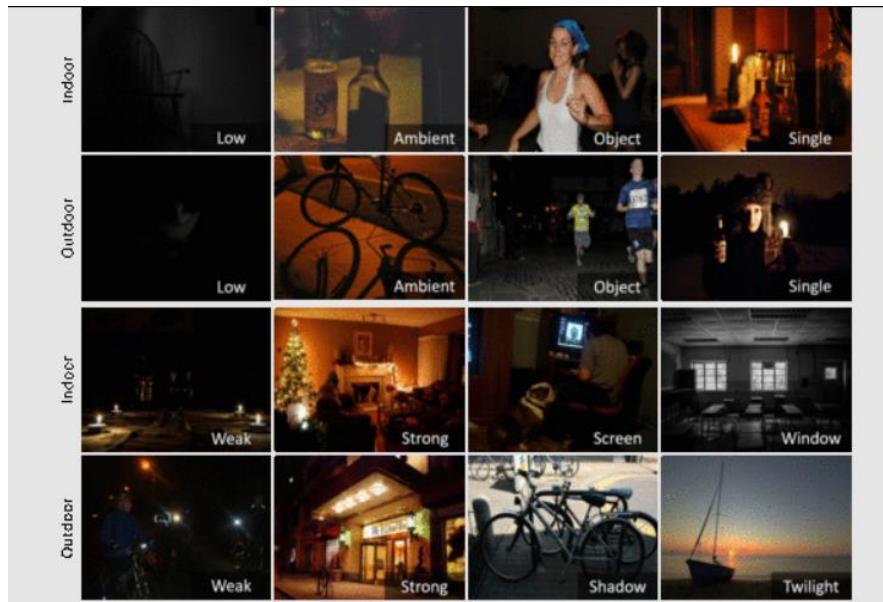
3.1 Lokasi dan Waktu

Penelitian dilakukan di Laboratorium MPL, Gedung Kuliah Bersama, Kampus C Universitas Airlangga yang beralamat di Jalan Dr. Ir. H. Soekarno, Mulyorejo, Surabaya, Jawa Timur. Penelitian ini berlangsung dari bulan Juli 2024 hingga bulan Desember 2024, sesuai dengan jadwal yang telah ditentukan. Dalam periode tersebut, peneliti melakukan perencanaan, pengumpulan data, dan penyusunan skripsi.

Penelitian ini dilaksanakan dengan menggunakan *hardware* yang terdiri dari 12.7 GB RAM, GPU NVIDIA Tesla T4, dan CPU Intel(R) Xeon(R). Penggunaan platform perangkat lunak melibatkan torch-2.1.0+cu121 dan Python-3.10.12.

3.2 Bahan dan Alat

Data yang digunakan pada penelitian ini merupakan data sekunder *The Exclusively Dark* atau *ExDark* yang dibuat oleh Loh, Y. P., & Chan, C. S. (2019). Data ini merupakan kumpulan dari 7363 citra dengan pencahayaan sangat rendah hingga pencahayaan fajar dengan 12 kelas objek. Objek-objek tersebut adalah *table*, *people*, *motorbike*, *dog*, *cup*, *chair*, *bicycle*, *boat*, *bottle*, *bus*, *car*, dan *cat*. Sebagian besar data citra ini diperoleh dari *website* dan *search engine* seperti *Flickr.com*, *Photo-bucket.com*, *Imgur.com*, *Deviantart.com*, *Gettyimages.com* dan *Google Search* dengan kata kunci *dark*, *low-light*, *nighttime*, dan lain-lain. *ExDark* juga mengambil sampel dari dataset-dataset besar seperti PASCAL VOC, ImageNet, dan Microsoft COCO serta menambah variasi data dengan mengekstraksi setiap *frame* berpencahayaan rendah dalam film-film dan mengambil gambar secara manual menggunakan beragam model *smartphone* dan kamera.



Gambar 3.1 Sampel *Exclusively Dark* dataset

Sumber: Loh, Y. P., & Chan, C. S. (2019)

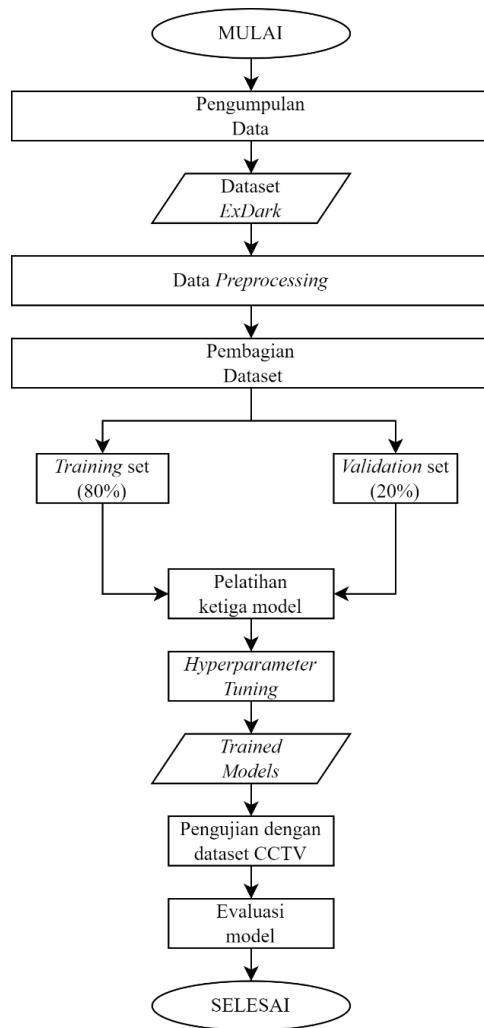
3.3 Cara Kerja

Penelitian ini tidak secara langsung menguji model pada perangkat CCTV atau sistem dengan sumber daya terbatas. Namun, salah satu kekuatan utama dari model deteksi objek modern adalah kemampuannya untuk melakukan generalisasi dari berbagai jenis dataset, termasuk dari dataset ExDark yang mengandung gambar dalam kondisi pencahayaan rendah yang sangat mirip dengan gambar yang dihasilkan oleh kamera CCTV di malam hari atau area yang gelap dengan berbagai macam sudut kamera.

Metode penelitian ini terdiri dari beberapa tahapan. Tahapan pertama dalam penelitian ini adalah dengan membagi dataset ke dalam dua set, yaitu *train* (80%), *validation* (20%). Pada tahap ini, pemilihan *train* set dan *validation* set akan menggunakan teknik *Stratified Random Sampling*. Teknik ini dipilih karena setiap gambar dalam dataset terbagi ke dalam 12 kelas yang berbeda, dan tujuan dari stratifikasi adalah memastikan bahwa distribusi kelas dalam set *validation* tetap seimbang, sehingga mewakili proporsi yang sama dengan dataset asli. Teknik ini penting untuk menghindari bias dalam proses validasi model dan memastikan bahwa semua kelas terwakili secara proporsional selama evaluasi.

Setelah pembagian data, dilakukan beberapa tahapan *preprocessing* seperti pengurangan resolusi menggunakan *lanczos resampling* lalu pengubahan format label *bounding box* yang disesuaikan untuk kebutuhan model YOLO dan RT-DETR. Data kemudian ditingkatkan kualitasnya dengan teknik *contrast enhancement* menggunakan CLAHE. Tahap terakhir dalam *preprocessing* ini adalah dengan augmentasi data menggunakan parameter-parameter yang ditentukan. Tahap augmentasi data ini juga mencakup algoritma *mosaic data*.

Setelah tahap *preprocessing* dan pembagian data, *train set* dan *validation set* digunakan untuk melatih tiga model (YOLOv9, RT-DETR, dan YOLOv10) dengan proses *hyperparameter tuning*. Setelah pelatihan selesai, model terlatih (*trained models*) akan dievaluasi berdasarkan performanya, termasuk *mean Average Precision* (mAP) dan waktu inferensi, kemudian dibandingkan menggunakan analisis *Pareto Frontier*. Model terlatih ini kemudian diuji menggunakan *test set* yang berupa dataset rekaman CCTV asli. Rekaman CCTV diperoleh dari sumber daring dengan kata kunci "*CCTV footage of a [subject]*", dimana *subject* mengacu pada jenis objek yang terdapat dalam dataset ExDark. Durasi rekaman bervariasi antara 5 hingga 30 detik, dengan 1 *frame* per detik.



Gambar 3.2 *Flowchart* Keseluruhan Proses

3.3.1 Data Preprocessing

Pada tahap ini dilakukan *data preprocessing* pada *train set* yang meliputi:

3.3.1.1 *Downsampling*

Tahap pertama dalam proses preprocessing adalah penerapan *downsampling* menggunakan algoritma *Lanczos Resampling*. Proses ini dilakukan di awal untuk mengurangi kompleksitas komputasi pada tahapan *preprocessing* berikutnya. Penting untuk dicatat bahwa model-model yang akan dilatih tidak menggunakan citra dengan resolusi hasil *downsampling*, karena masing-masing model telah mengkonfigurasi parameter input secara *default* sebesar 640×640 piksel. Dengan demikian, berapapun resolusi citra

awal, semua akan diseragamkan menjadi 640×640 piksel. Langkah *downsampling* ini berfungsi untuk mengurangi detail-detail yang dianggap kurang signifikan, sehingga memungkinkan model mencapai generalisasi yang lebih baik ketika dilakukan resampling atau interpolasi otomatis melalui arsitektur model yang telah disesuaikan.

3.3.1.2 Pengubahan format label *bounding box*

Setiap citra yang sudah melalui tahap *downsampling* akan membutuhkan anotasi *bounding box* baru. Setiap anotasi di dataset ExDark mencakup beberapa informasi, antara lain 16 karakter pertama yang berisi data dari alat anotasi, kolom pertama yang berisi nama kelas objek, serta kolom kedua hingga kelima yang menyatakan koordinat *bounding box* dalam bentuk $[l \ t \ w \ h]$, di mana l adalah jarak dari tepi kiri gambar, t adalah jarak dari tepi atas gambar, w adalah lebar *bounding box*, dan h adalah tinggi *bounding box*. Selain itu, anotasi ini juga memiliki informasi tambahan mengenai oklusi dan orientasi yang tidak akan digunakan.

Untuk mengubah format tersebut menjadi format yang dapat diterima oleh seluruh model, dilakukan proses konversi yang melibatkan beberapa langkah penting. Pertama, nama kelas objek dikonversi menjadi ID kelas yang sesuai. Selanjutnya, koordinat *bounding box* yang awalnya dinyatakan dalam bentuk $[l \ t \ w \ h]$ dikonversi menjadi $[center_x, center_y, width, height]$ di mana $center_x$ dan $center_y$ adalah koordinat pusat *bounding box*, sedangkan $width$ dan $height$ tetap merupakan dimensi dari *bounding box*. Semua nilai ini kemudian dinormalisasi dengan membagi nilai x , y , $width$, dan $height$ dengan lebar dan tinggi gambar. Hasil akhir dari proses ini dituliskan ke dalam *file* dengan format *.txt* yang sesuai untuk digunakan dalam pelatihan model.

3.3.1.3 *Contrast Enhancement*

Setelah citra pada *train set* dilakukan *downsampling*, tahap selanjutnya dalam metodologi ini adalah penerapan teknik normalisasi data

menggunakan *Contrast Limited Adaptive Histogram Equalization* (CLAHE). Teknik ini dipilih karena kemampuannya dalam meningkatkan kontras gambar, khususnya pada kondisi pencahayaan rendah, seperti yang terdapat pada dataset ExDark. CLAHE bekerja dengan membagi gambar menjadi blok-blok kecil, kemudian melakukan *equalization* pada histogram setiap blok secara adaptif. Langkah ini memastikan bahwa kontras lokal pada setiap bagian gambar dapat ditingkatkan secara proporsional, tanpa menyebabkan amplifikasi *noise* yang berlebihan.

Dalam penelitian ini, CLAHE diterapkan untuk meningkatkan visibilitas objek pada gambar, dengan tujuan memaksimalkan akurasi deteksi objek pada kondisi pencahayaan yang beragam. Proses ini penting karena peningkatan kontras yang dihasilkan dari CLAHE membantu model deteksi objek dalam mengidentifikasi objek secara lebih efektif, terutama dalam skenario dengan pencahayaan yang tidak merata.

3.3.1.4 Data Augmentation

Setelah membagi dataset ke dalam *train set* dan juga *validation set*, tahap terakhir adalah melakukan augmentasi data pada *train set* yang sudah dibuat. Proses augmentasi data ini dilakukan dengan menggunakan bantuan dari modul *albumentations*. Parameter-parameter augmentasi untuk digunakan dalam modul *albumentations* dijelaskan dalam Tabel 3.1.

Tabel 3.1 Deskripsi Parameter Augmentasi

<i>Parameter</i>	<i>Value Range</i>
<i>Rotation augmentation range</i>	0,0 ; 45,0
<i>Translation augmentation range</i>	0,0 ; 0,9
<i>Scaling augmentation range</i>	0,0 ; 0,9
<i>Shear augmentation range</i>	0,0 ; 10,0
<i>Perspective augmentation range</i>	0,0 ; 0,001
<i>Horizontal flip augmentation probability</i>	0,0 ; 1,0

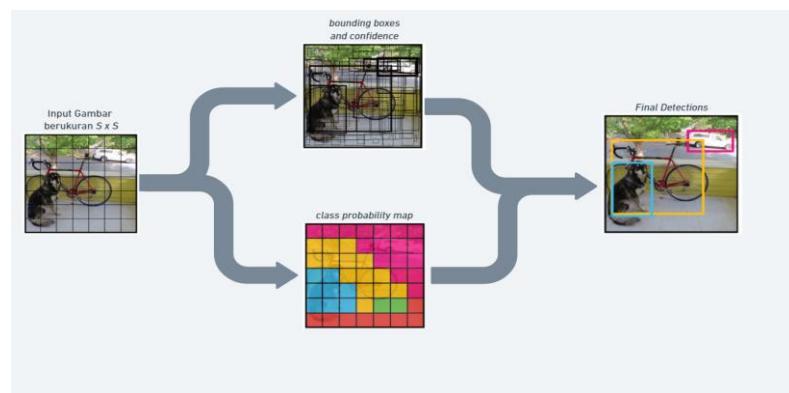
Setelah seluruh citra mengalami proses augmentasi sesuai dengan parameter yang telah ditetapkan, langkah berikutnya adalah penerapan konsep mosaic data untuk meningkatkan keberagaman data yang dapat dipelajari model. Proses mosaic ini dilakukan pada citra asli sebelum augmentasi dan mengelompokkan citra ke dalam satu set yang terdiri dari enam citra, tanpa dilakukan pengambilan data secara berulang.

3.3.2 Implementasi Model

Pada tahap ini model berbasis CNN, Transformer, dan CNN-Transformer akan dilatih menggunakan *training set* serta akan divalidasi dengan *validation set*. Setelah itu akan dilakukan proses *hyperparameter tuning* untuk mendapatkan kombinasi parameter optimal.

3.3.2.1 Model berbasis CNN

Proses inferensi model berbasis CNN yaitu YOLOv9 yang digunakan pada penelitian ini ditunjukkan pada Gambar 3.3. Proses dimulai dengan sebuah gambar *input* berukuran $S \times S$. dimana setiap sel grid memiliki tugas untuk mendekripsi objek yang pusatnya jatuh dalam sel tersebut.



Gambar 3.3 Proses Inferensi Model berbasis CNN

YOLO bekerja dengan dua cabang utama. Cabang pertama bertugas memprediksi *bounding boxes* dan *confidence scores*. Setiap sel pada grid memprediksi beberapa *bounding boxes*, yang masing-masing berisi informasi tentang koordinat (x, y), lebar (*width*), tinggi (*height*), dan nilai

kepercayaan yang mencerminkan seberapa yakin model bahwa kotak tersebut mengandung objek dan seberapa tepat prediksi tersebut dibandingkan dengan objek sebenarnya.

Cabang kedua dari arsitektur ini menghasilkan *class probability map*. Setiap sel *grid* juga memprediksi probabilitas kelas untuk berbagai jenis objek yang mungkin berada dalam sel tersebut. Probabilitas ini menunjukkan seberapa besar kemungkinan objek dalam sel tersebut termasuk dalam setiap kelas yang ada.

Hasil dari kedua cabang ini digabungkan untuk menghasilkan deteksi *final*. *Bounding boxes* yang memiliki nilai kepercayaan tinggi dan probabilitas kelas yang signifikan akan dipilih sebagai hasil deteksi akhir, ditampilkan sebagai kotak yang mengelilingi objek yang terdeteksi dalam gambar.

YOLOv9 memiliki lima tipe model yang dibedakan berdasarkan jumlah parameternya. Namun, karena keterbatasan sumber daya, penelitian ini hanya akan menggunakan tiga tipe model yang berada di tengah-tengah spektrum jumlah parameter. Pemilihan ketiga tipe ini dilakukan dengan tujuan untuk memprediksi tren kinerja, yang diharapkan dapat mewakili performa dua tipe model lainnya yang tidak digunakan dalam penelitian ini. Tipe-tipe model YOLOv9 yang akan diuji pada penelitian ini dijelaskan dalam Tabel 3.2.

Tabel 3.2 Tipe Model YOLOv9

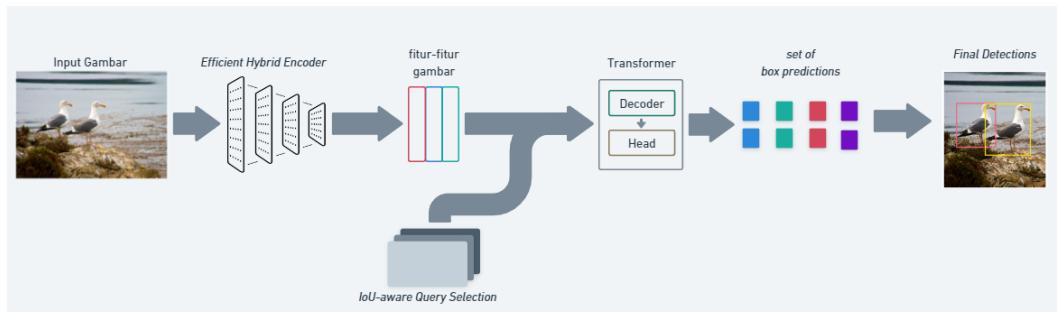
Tipe	Jumlah Parameter (dalam juta)
YOLOv9s	7,2
YOLOv9m	20,1
YOLOv9c	25,5

Proses inferensi model berbasis CNN, yaitu YOLOv9, yang digunakan dalam penelitian ini akan bersumber dari library *Ultralytics*. Library ini menyediakan implementasi YOLO yang telah dioptimalkan

untuk berbagai aplikasi deteksi objek secara *real-time*. Dalam implementasinya, *Ultralytics* menggunakan beberapa *loss function* yang penting untuk meningkatkan kinerja deteksi objek pada YOLOv9, yaitu *box-loss*, *cls-loss*, dan *dfl-loss*. Berdasarkan dokumentasi *Ultralytics*, algoritma yang digunakan oleh *box-loss* adalah *Intersection over Union loss*, untuk *cls-loss* adalah *BCEWithLogitsLoss*, dan *dfl-loss* adalah *Distribution Focal Loss*. Optimizer SGD akan dipilih sebagai *optimizer* pada ketiga tipe model YOLOv9.

3.3.2.2 Model berbasis Transformer

Proses inferensi model berbasis Transformer yang menggunakan model *Real-Time DEtection TTransformer* (RT-DETR) yang digunakan pada penelitian ini ditunjukkan pada Gambar 3.4. Proses dimulai dengan memasukkan gambar *input* ke dalam *convolutional neural network* (CNN). CNN berfungsi untuk mengekstraksi fitur-fitur penting dari gambar, menghasilkan representasi fitur yang lebih terperinci dan kompak.



Gambar 3.4 Proses Inferensi Model Berbasis Transformer (RE-DETR)

Fitur-fitur gambar yang dihasilkan dari CNN kemudian diberi pengkodean posisi (*positional encoding*) untuk mempertahankan informasi spasial dari gambar asli. *Positional encoding* ini penting karena Transformer memerlukan informasi posisi untuk memproses urutan data secara efektif.

Selanjutnya, fitur-fitur gambar yang telah diberi pengkodean posisi ini dimasukkan ke dalam Transformer. Transformer terdiri dari dua

komponen utama, *Encoder* dan *Decoder*. *Encoder* bertugas untuk memproses dan memahami representasi fitur gambar secara menyeluruh, sementara *Decoder* menghasilkan prediksi *bounding boxes* dan kelas objek dari fitur-fitur yang telah diproses.

Set *box prediction* yang dihasilkan oleh *Decoder Transformer* mencakup informasi koordinat dan kategori objek yang terdeteksi. Pada tahap akhir, prediksi-prediksi ini digabungkan untuk menghasilkan deteksi akhir, di mana objek-objek yang terdeteksi ditandai dengan *bounding boxes* pada gambar asli.

Tipe-tipe model RT-DETR yang akan diuji pada penelitian ini dijelaskan dalam Tabel 3.3.

Tabel 3.3 Tipe Model RT-DETR

Tipe	Jumlah Parameter (dalam juta)
RT-DETR-L	32,9
RT-DETR-X	67,4

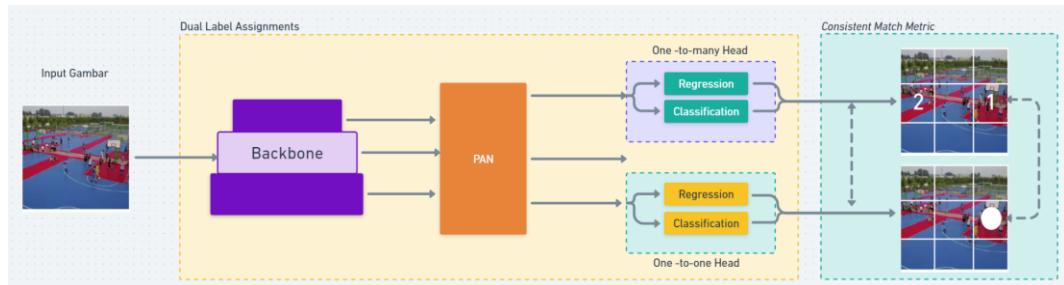
Proses inferensi model berbasis Transformer, yaitu RT-DETR, yang digunakan dalam penelitian ini akan bersumber dari library *Ultralytics*. Dalam implementasinya, *Ultralytics* menggunakan beberapa *loss function* yang penting untuk meningkatkan kinerja deteksi objek pada RT-DETR, yaitu *GIoU-loss* dan *cls-loss*. Berdasarkan dokumentasi *Ultralytics*, algoritma *loss function* yang digunakan oleh adalah *Generalized Intersection over Union loss* dan untuk *cls-loss* adalah *BCEWithLogitsLoss*. *Optimizer SGD* akan dipilih sebagai *optimizer* pada kedua tipe model RT-DETR.

3.3.2.3 Model berbasis CNN-Transformer

Proses inferensi model berbasis CNN-Transformer yaitu model YOLOv10 yang digunakan pada penelitian ini ditunjukkan pada Gambar 3.5. YOLOv10 mengimplementasikan *Dual Label Assignments* untuk

meningkatkan kinerja deteksi objek, yaitu melalui pendekatan *one-to-one* dan *one-to-many*. Pada pendekatan *one-to-one*, setiap prediksi dicocokkan dengan satu *ground truth*. Strategi ini menghindari penggunaan proses *post-processing* seperti *Non-Maximum Suppression* (NMS) yang biasa digunakan untuk menangani prediksi *overlapping predictions*. Namun, kelemahan dari *one-to-one* adalah kurangnya pengawasan yang optimal, yang dapat mempengaruhi akurasi dan kecepatan konvergensi selama pelatihan.

Sebaliknya, pendekatan *one-to-many* memungkinkan satu prediksi dicocokkan dengan beberapa *ground truth*. Meskipun pendekatan ini memberikan *supervised signal* yang lebih baik, metode ini memerlukan NMS untuk mengurangi *overlapping predictions*. YOLOv10 menggabungkan kedua metode ini melalui *dual label assignments*. Dalam tahap pelatihan, dua "head" model yang mewakili kedua pendekatan ini dioptimalkan secara bersamaan. Namun, pada tahap inferensi, hanya *one-to-one head* yang digunakan untuk membuat prediksi, sehingga memungkinkan penerapan YOLOv10 secara *end-to-end* tanpa *cost* tambahan pada waktu inferensi.



Gambar 3.5 Proses Inferensi Model berbasis CNN-Transformer (YOLOv10)

Selama proses *label assignment*, baik pendekatan *one-to-one* maupun *one-to-many* menggunakan *Consistent Matching Metric* untuk mengevaluasi kesesuaian antara prediksi dengan *instance* objek. Metrik yang digunakan oleh YOLOv10 adalah fungsi yang mempertimbangkan skor klasifikasi, IoU, dan prior spasial, yaitu apakah titik prediksi berada di dalam *instance* objek.

YOLOv10 memiliki enam tipe model yang dibedakan berdasarkan jumlah parameternya. Namun, karena keterbatasan sumber daya, penelitian ini hanya akan menggunakan tiga tipe model yang berada di tengah-tengah spektrum jumlah parameter. Pemilihan ketiga tipe ini dilakukan dengan tujuan untuk memprediksi tren kinerja, yang diharapkan dapat mewakili performa tiga tipe model lainnya yang tidak digunakan dalam penelitian ini. Tipe-tipe model YOLOv10 yang akan diuji pada penelitian ini dijelaskan dalam Tabel 3.4.

Tabel 3.4 Tipe Model YOLOv10

Tipe	Jumlah Parameter (dalam juta)
YOLOv10-S	7,2
YOLOv10-M	15,4
YOLOv10-B	20,5

Proses inferensi model berbasis CNN-Transformer, yaitu YOLOv10, yang digunakan dalam penelitian ini akan bersumber dari library *Ultralytics*. Sama halnya seperti YOLOv9, *Ultralytics* menggunakan beberapa *loss function* pada YOLOv10, yaitu *box-loss*, *cls-loss*, dan *dfl-loss*. Algoritma yang digunakan oleh *box-loss* adalah *Intersection over Union loss*, untuk *cls-loss* adalah *BCEWithLogitsLoss*, dan *dfl-loss* adalah *Distribution Focal Loss*. Optimizer SGD akan dipilih sebagai *optimizer* pada ketiga tipe model YOLOv10.

3.3.2.4 Hyperparameter Tuning

Pada ketiga model dilakukan *hyperparameter tuning* dengan parameter yang ditunjukkan pada Tabel 3.5.

Tabel 3.5 Deskripsi *Hyperparameter* yang Diuji

Parameter	Value
<i>Learning Rate</i>	0,001 ; 0,01
<i>Batch Size</i>	8 ; 16

Dalam penelitian ini, konfigurasi *Learning rate* & *Batch Size* menjadi esensial karena model deteksi objek harus dapat menangani kompleksitas citra dengan pencahayaan rendah. Sementara itu, Epochs tidak dijadikan sebagai parameter yang *di-tuning* karena penggunaan *callbacks* seperti *early stopping* dan *patience* sudah cukup efektif dalam menghindari *overfitting*, dengan cara menghentikan pelatihan ketika tidak ada perbaikan pada performa model dalam beberapa iterasi (Prechelt, 1998). Epochs yang akan digunakan adalah 1000 dengan *patience* 30. Pendekatan ini memastikan bahwa model tidak dilatih terlalu lama sehingga tetap efisien dan optimal dalam performa.

3.3.3 Metode Perbandingan Model

Dalam penelitian ini, perbandingan kinerja antara ketiga model, yaitu YOLOv9, RT-DETR, dan YOLOv10, akan dilakukan dengan mempertimbangkan akurasi dan *time complexity* menggunakan Pareto Frontier. Prosedur perbandingan model ini adalah sebagai berikut:

1. Pengumpulan Data Waktu Inferensi dan Nilai *mean Average Precision* (mAP)

Setiap model akan diuji untuk mengumpulkan data mAP dan waktu inferensi pada dataset yang sama. Selain itu, *cls_loss* (*BCEWithLogitsLoss*) akan digunakan sebagai alat ukur (*loss function*) untuk ketiga model dikarenakan *loss function* ini dapat digunakan untuk ketiga model yang bersangkutan, yakni YOLOv9, RT-DETR, dan YOLOv10.

2. Konstruksi *Pareto Frontier*

Berdasarkan hasil pengujian, nilai-nilai akurasi dan waktu inferensi dari tiap model akan dibentuk ke dalam plot untuk menghasilkan *Pareto Frontier*. *Pareto Frontier* mengidentifikasi model yang optimal secara relatif, yaitu model yang menawarkan kombinasi terbaik dari mAP tinggi dan waktu inferensi rendah.

3. Analisis Perbandingan

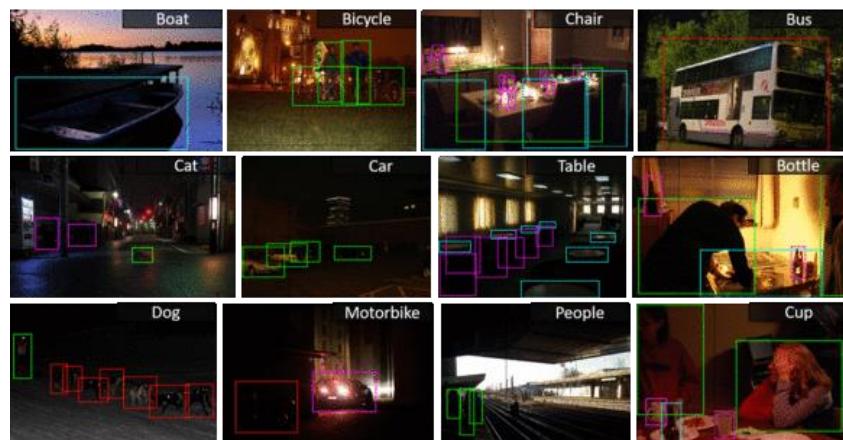
Model-model yang terletak pada *Pareto Frontier* dikategorikan sebagai solusi Pareto optimal. Jika dua model memiliki tingkat mAP yang setara, maka model dengan waktu inferensi yang lebih singkat akan lebih diprioritaskan. Sebaliknya, jika waktu inferensi kedua model serupa, model dengan akurasi yang lebih tinggi akan dianggap lebih unggul. Model-model yang berada di luar *Pareto Frontier* dinilai kurang efisien karena terdapat model lain yang dapat memberikan kinerja yang lebih bagus, baik dalam mAP maupun waktu inferensi.

BAB IV

HASIL DAN PEMBAHASAN

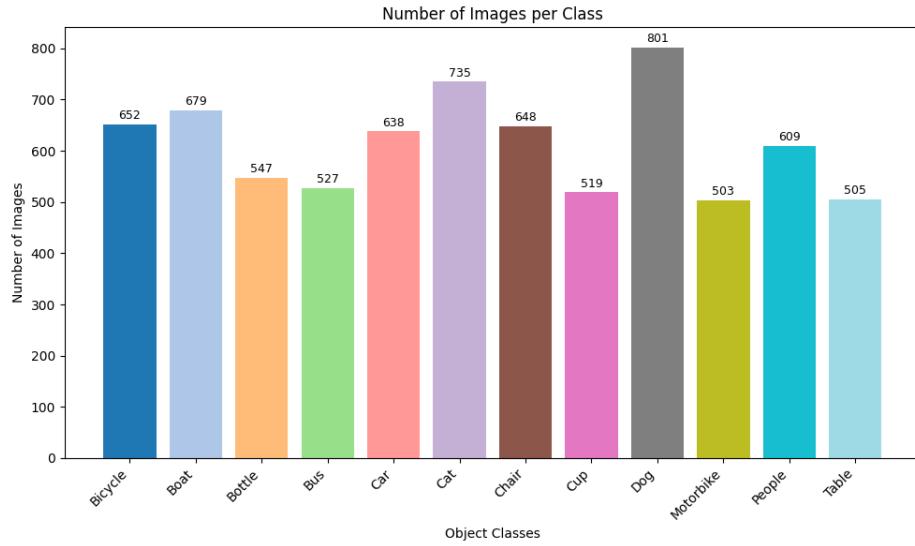
4.1 Pengambilan data

Dataset yang digunakan dalam penelitian ini diperoleh dari sebuah publikasi ilmiah yang menyediakan tautan menuju repositori GitHub. Repositori tersebut telah berisi seluruh citra yang dibutuhkan, yang telah dikelompokkan berdasarkan kelas masing-masing, serta dilengkapi dengan anotasi bounding box yang sesuai. Gambar di bawah ini menunjukkan contoh data yang digunakan dalam penelitian ini:



Gambar 4.1 Sampel *Exclusively Dark Dataset*

Jumlah citra yang mewakili setiap objek kelas yang ada pada *ExDark* dijelaskan oleh diagram batang di bawah:



Gambar 4.2 Distribusi citra untuk setiap objek

Visualisasi pada gambar di atas hanya mewakili jumlah citra untuk setiap objek, dan bukan representasi dari setiap *instance* yang ada untuk setiap objeknya.

Setelah seluruh dataset beserta label anotasinya berhasil dikumpulkan, langkah selanjutnya adalah membagi dataset tersebut menjadi dua bagian, yaitu *training set* sebesar 80% dan *validation set* sebesar 20%. Tujuan dari pemisahan ini adalah untuk menjaga kemurnian validation set dari proses preprocessing, sehingga data tersebut dapat merepresentasikan kondisi nyata di dunia sebenarnya. Proses pembagian dilakukan menggunakan teknik stratified random sampling guna memastikan distribusi kelas objek tetap seimbang pada masing-masing subset.

4.2 Data Preprocessing

4.2.1 Downsampling

Tahap awal dalam proses data preprocessing adalah penyesuaian resolusi citra melalui metode downsampling menggunakan algoritma *Lanczos resampling*. Proses ini diterapkan pada training set dengan tujuan

untuk mengurangi kompleksitas data sekaligus mempercepat proses pelatihan model.

Langkah pertama dalam penyesuaian resolusi ini dilakukan dengan menguji sejauh mana pengurangan resolusi sebesar 20% dari ukuran asli tetap menghasilkan citra yang dapat digunakan secara efektif. Gambar berikut menunjukkan contoh hasil *downsampling* sebesar 20% dari resolusi awal:



Gambar 4.3 *Downsampling* sebanyak 20%

Berdasarkan penilaian subjektif dari peneliti, kualitas visual citra hasil *downsampling* sebesar 20% masih dianggap memadai untuk digunakan dalam pelatihan model. Oleh karena itu, dilakukan pengujian lanjutan dengan mengurangi resolusi lebih lanjut, yakni sebesar 40% dari resolusi asli. Contoh hasil *downsampling* sebesar 40% ditunjukkan pada gambar berikut:



Gambar 4.4 *Downsampling* sebanyak 40%

Hasil visual yang ditunjukkan di atas menunjukkan bahwa *downsampling* hingga 40% tetap menghasilkan citra yang layak untuk digunakan. Oleh karena itu, pengurangan resolusi sebesar ini dianggap sesuai untuk digunakan dalam tahap *preprocessing*, karena selain mempercepat *preprocessing* data, juga dapat membantu mengurangi beban yang akan ditanggung oleh *memory hardware* pada saat *training*.

4.2.2 Pengubahan format label *bounding box*

Setelah memperoleh citra beserta anotasi *bounding box*-nya, langkah selanjutnya adalah melakukan konversi format *bounding box* ke dalam bentuk yang sesuai dengan standar yang dapat diterima oleh model-model yang digunakan dalam penelitian ini, yaitu YOLOv9, RT-DETR, dan YOLOv10. Format awal anotasi *bounding box* yang diperoleh adalah sebagai berikut:

```
% bbGt version=3
Bicycle 204 28 271 193 0 0 0 0 0 0 0 0
```

Format tersebut kemudian dikonversi ke dalam format baru yang telah disesuaikan dengan kebutuhan masing-masing arsitektur model. Berikut adalah representasi hasil konversi *bounding box* ke dalam format akhir yang digunakan dalam proses pelatihan model:

```
0 0.679000 0.332000 0.542000 0.514667
```

Digit pertama pada format baru ini menunjukkan kode objek yang bersangkutan yang telah diurutkan sesuai abjad secara *ascending*. Nilai 0 pada digit pertama ini mengacu pada kelas *Bicycle* atau sepeda. Digit kedua dan seterusnya adalah pendefinisian koordinat yang sudah dinormalisasi untuk setiap objek dalam citra. Dimulai dari digit kedua, *x_center*, *y_center*, *width*, dan terakhir digit kelima *height*.

4.2.2 Contrast Enhancement

Teknik *contrast enhancement* yang diterapkan dalam penelitian ini adalah *Contrast Limited Adaptive Histogram Equalization* (CLAHE). Metode CLAHE bekerja dengan menerapkan peningkatan kontras secara lokal pada gambar, sehingga menghasilkan citra dalam *grayscale* dengan tingkat kecerahan yang disesuaikan secara adaptif, terutama pada area-area yang sebelumnya kurang pencahayaan. Contoh hasil pemrosesan citra menggunakan metode CLAHE ditunjukkan pada gambar berikut:



Gambar 4.5 Sampel hasil CLAHE

4.2.3 Data Augmentation

Tahap terakhir dalam proses *preprocessing* adalah *data augmentation*, yang dilakukan menggunakan modul *Albumentations*. Augmentasi ini diterapkan pada data *training* sebanyak 5886 citra, sehingga jumlah total citra setelah augmentasi menjadi dua kali lipat, yaitu sebanyak 11.772. Dengan menggunakan parameter-parameter augmentasi yang telah ditentukan, contoh hasil augmentasi dapat dilihat pada gambar berikut:



Gambar 4.6 Sampel hasil augmentasi data

Langkah lanjutan dalam proses augmentasi adalah penerapan algoritma *mosaic*, yang menggabungkan empat gambar berbeda dalam satu citra gabungan. Dengan konfigurasi seperti ini, jumlah total citra pada *training set* bertambah menjadi 17,115 citra. Contoh hasil dari teknik *mosaic augmentation* ditunjukkan pada gambar berikut:



Gambar 4.7 Sampel hasil *mosaic augmentati*

4.3 Implementasi YOLOv9

4.3.1 Pelatihan YOLOv9

Model YOLOv9 yang digunakan dalam penelitian ini terdiri dari tiga varian, yaitu YOLOv9s, YOLOv9m, dan YOLOv9c. Setiap varian model dilatih menggunakan konfigurasi *hyperparameter* yang telah ditentukan sebelumnya melalui pendekatan *grid search*, yang menghasilkan

empat kombinasi parameter berbeda. Dengan demikian, total model yang dihasilkan dari proses pelatihan YOLOv9 adalah sebanyak 12 model (3 varian \times 4 kombinasi). Seluruh hasil pelatihan dari model-model tersebut dirangkum dalam Tabel 4.1.

Tabel 4.1 Hasil *grid search* untuk YOLOv9

BATCH 8	Lr 0.01			Lr 0.001		
	s	m	c	s	m	c
total_epoch	153	123	182	50	36	38
total_time	24528.6	17141.7	27158.1	7983.14	4994.8	5658.07
train/box_loss	116.087	100.368	0.85547	116.476	104.929	0.97495
train/cls_loss	0.93253	0.74765	0.61091	0.91759	0.75938	0.66978
train/dfl_loss	13.208	119.547	109.861	131.245	121.221	115.929
metrics/precision(B)	0.79881	0.80415	0.78834	0.77952	0.81313	0.82603
metrics/recall(B)	0.70113	0.70362	0.71021	0.69448	0.72082	0.72515
metrics/mAP50(B)	0.7792	0.77866	0.77686	0.76621	0.80037	0.80051
metrics/mAP50-95(B)	0.50553	0.5069	0.50874	0.4986	0.52844	0.53259
val/box_loss	133.479	135.412	137.849	131.088	129.887	129.319
val/cls_loss	0.90956	0.92488	0.95876	0.93619	0.88004	0.87685
val/dfl_loss	154.166	155.132	160.583	14.645	144.284	144.121
BATCH 16	Lr 0.01			Lr 0.001		
	s	m	c	s	m	c
total_epoch	139	97	31	46	36	43
total_time	13430.4	11619.9	4144.79	4433.32	4323.72	5735.19
train/box_loss	113.872	100.394	119.269	114.262	101.155	0.91356
train/cls_loss	0.88159	0.73445	0.95252	0.8772	0.69955	0.60739
train/dfl_loss	129.174	118.485	132.722	127.248	117.283	110.651
metrics/precision(B)	0.77846	0.78923	0.77959	0.78191	0.81178	0.82808
metrics/recall(B)	0.70355	0.71027	0.66315	0.68447	0.70649	0.71274
metrics/mAP50(B)	0.77258	0.7757	0.75295	0.7596	0.789	0.79429
metrics/mAP50-95(B)	0.50055	0.50545	0.47686	0.49252	0.51436	0.52796
val/box_loss	134.283	134.797	136.475	132.618	133.798	131.601
val/cls_loss	0.931	0.93049	100.895	0.96442	0.90462	0.90149
val/dfl_loss	154.146	154.237	152.299	14.601	146.205	14.593

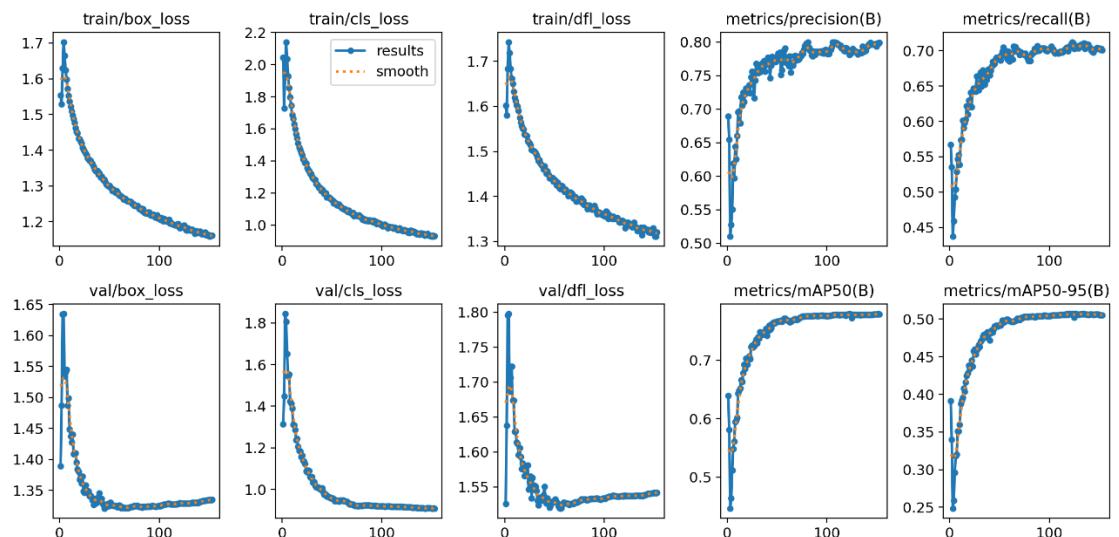
4.3.2 Kombinasi YOLOv9 terbaik

Berdasarkan evaluasi menggunakan metrik yang tercantum pada Tabel 4.2, diperoleh kombinasi *hyperparameter* terbaik untuk masing-masing varian YOLOv9 sebagaimana dijelaskan pada bagian berikut.

Tabel 4.2 Kombinasi terbaik untuk YOLOv9

Tipe Model	Batch Size	Learning Rate	mAP50	map50-95	val/cls_loss
YOLOv9s	8	0.01	0.7792	0.50553	0.90956
YOLOv9m	8	0.001	0.80037	0.52844	0.88004
YOLOv9c	8	0.001	0.80051	0.53259	0.87685

4.3.2.1 YOLOv9s

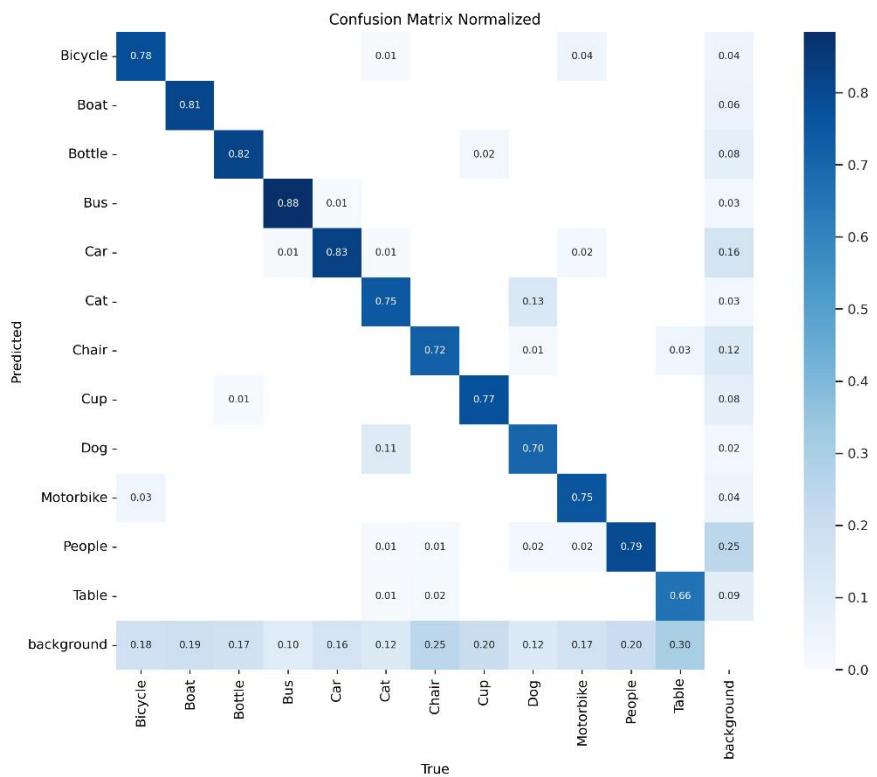


Gambar 4.8 Kurva hasil pelatihan model YOLOv9s

Gambar 4.8 di atas menunjukkan grafik hasil pelatihan model YOLOv9s dengan konfigurasi *epoch* 1000, *optimizer SGD*, dan *patience* 30. Grafik *train/box_loss*, *train/cls_loss*, dan *train/dfl_loss* secara konsisten menurun dari nilai yang relatif tinggi di awal menuju nilai yang lebih rendah dan stabil di akhir pelatihan. Pada grafik *metrics/precision(B)* dan

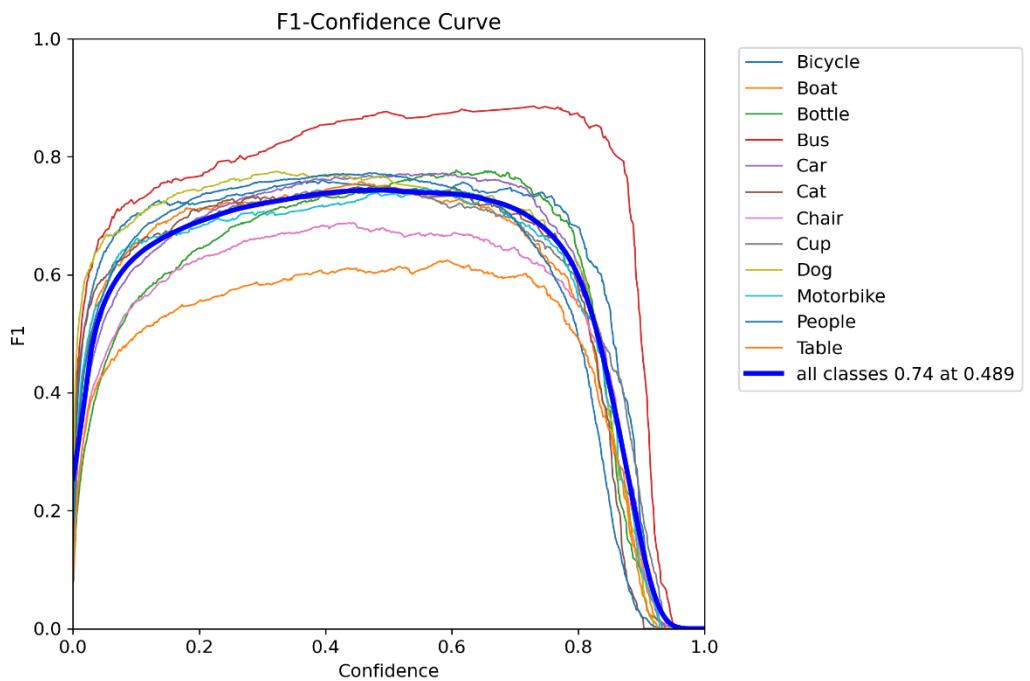
metrics/recall(B), nilai kedua metrik cenderung meningkat secara bertahap, menandakan model semakin mampu mengurangi *false positives* dan *false negatives* seiring bertambahnya epoch.

Selanjutnya, tren *loss* pada *validation set*, yaitu *val/box_loss*, *val/cls_loss*, dan *val/dfl_loss*, menunjukkan pola fluktuatif cukup luas yang mungkin saja terjadi karena kapasitas modelnya yang kecil sehingga lebih sensitif terhadap perubahan kondisi data dan *hyperparameter*. *Overfitting* juga dapat terjadi jika model terlalu terpaku pada *train set*, tetapi penerapan mekanisme *early stopping* berupa *patience 30* dapat menahan masalah ini agar tidak berlanjut secara ekstrim. Peningkatan metrik *mAP50(B)* dan *map50-95(B)* di *validation set* memperlihatkan bahwa secara umum performa deteksi YOLOv9s kian membaik setelah beberapa epoch tertentu, menandakan kemampuan model untuk secara konsisten menempatkan *bounding box* dengan tepat pada berbagai rentang IoU. Meski demikian, karena YOLOv9s memiliki arsitektur yang sangat ringan, lanju peningkatan mAP mungkin sedikit lambat untuk mencapai nilai puncak yang setara dengan model yang secara parameter lebih besar dan memiliki jaringan yang lebih dalam.



Gambar 4.9 *Confusion Matrix Normalized* untuk YOLOv9s

Gambar 4.9 merupakan gambar *confusion matrix normalized* untuk YOLOv9s yang memperlihatkan hasil cukup akurat, dapat telihat dari nilai diagonal yang relatif tinggi. Namun, beberapa kelas dengan kemiripan visual seperti *Bicycle* dan *Motorbike* atau *Chair* dan *Table*, masih sering tertukar, terutama karena kesulitan mengekstrasi fitur halus terlebih lagi pada kondisi pencahayaan rendah. Selain itu, beberapa objek yang cenderung kecil atau kurang kontras seperti *Bottle* terkadang diklasifikasikan sebagai latar belakang.



Gambar 4.10 Kurva F1-Confidence untuk YOLOv9s

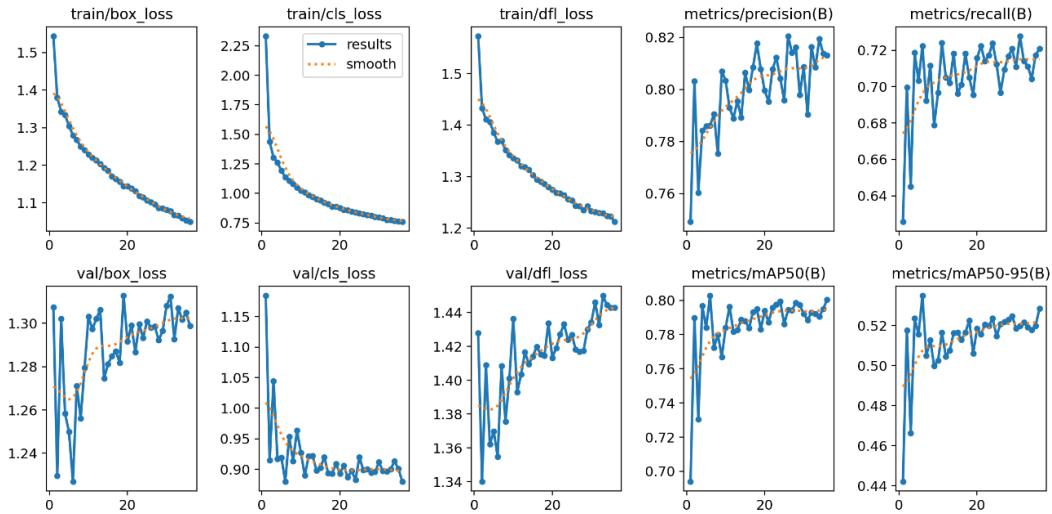
Gambar 4.10 di atas adalah kurva *F1-Confidence* untuk YOLOv9s yang menunjukkan bagaimana *precision* dan *recall* saling berinteraksi pada variasi nilai *threshold confidence* pada model. Beberapa kelas yang memiliki kontur visual lebih jelas, seperti *Bus* dan *Car*, terlihat mencapai puncak F1 pada level *confidence* menengah hingga tinggi, menandakan bahwa model memiliki keyakinan yang cukup kuat untuk mendeteksi objek tersebut tanpa banyak mengorbankan *recall*. Namun, kurva untuk kelas-kelas yang cenderung mirip atau sering tumpang tindih seperti *Chair* dan *Table*, relatif lebih rentan terpengaruh oleh perubahan *threshold confidence*.

Tabel 4.3 Label vs Prediksi untuk YOLOv9s

No.	<i>Ground Truth (a)</i>	<i>Predicted (b)</i>
1.		
2.		
3.		

Tabel 4.3 di atas menunjukkan sampel dari hasil prediksi YOLOv9s yang umumnya sudah mendekati *ground truth*, dimana beberapa objek berhasil terdeteksi dengan *confidence* yang relatif baik meski berada pada kondisi cahaya minim. YOLOv9s ini juga mampu mendeteksi objek yang tidak diberi label dalam *ground truth*, contohnya seperti objek *Table* pada gambar 1b. Namun, tampak pula adanya perbedaan letak dan skala *bounding box* pada beberapa objek seperti, khususnya jika objek tersebut tumpang tindih atau berukuran kecil seperti pada gambar 2a dan 2b.

4.3.2.2 YOLOv9m



Gambar 4.11 Kurva hasil pelatihan model YOLOv9m

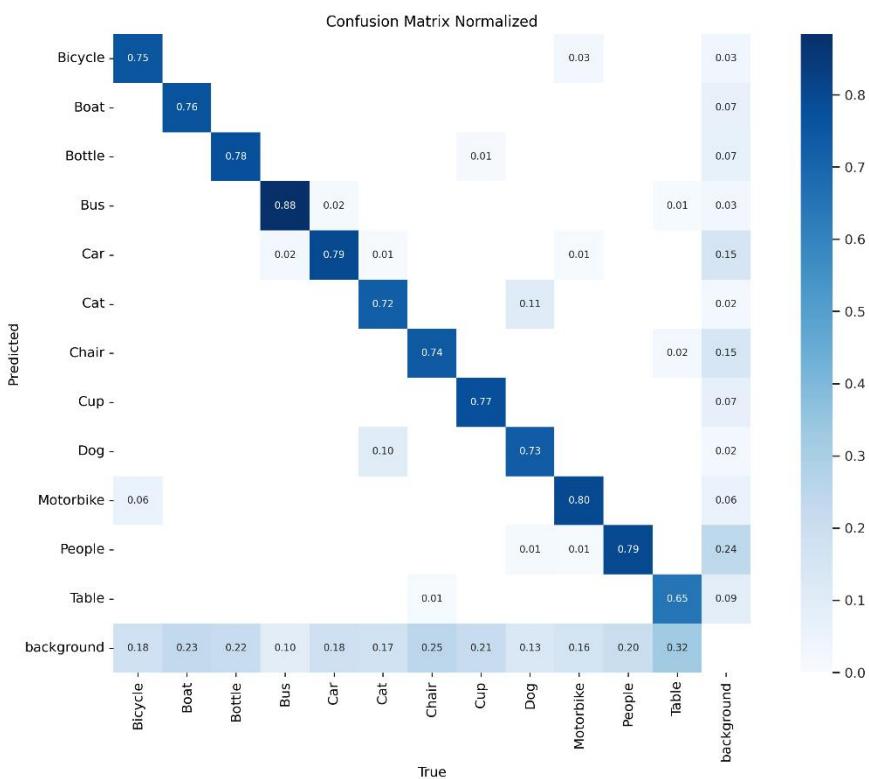
Gambar 4.11 menunjukkan beberapa metrik utama yang digunakan untuk mengevaluasi kinerja kombinasi terbaik model YOLOv9m. Pada *train/box_loss*, *train/cls_loss*, dan *train/dfl_loss*, secara umum terlihat tren menurun yang menandakan bahwa proses pelatihan berjalan cukup baik. Pada awal pelatihan, nilai ketiga *loss* tersebut cukup tinggi, kemudian secara konsisten menurun dan semakin stabil mendekati titik konvergen di akhir pelatihan. Data ini menunjukkan bahwa model semakin terlatih dalam mengekstraksi fitur dan menyesuaikan *weights* sesuai target.

Selanjutnya pada kurva metrik evaluasi di bagian training untuk *precision* dan *recall*, tampak terjadi peningkatan bertahap sepanjang epoch. Peningkatan *precision* mengindikasikan bahwa model semakin jarang memberikan *false positives*. Sementara itu peningkatan *recall* menandakan model semakin jarang melakukan *false negatives*.

Jika dibandingkan dengan grafik pada *validation set*, *val/box_loss*, *val/cls_loss*, dan *val/dfl_loss*, terlihat bahwa ketiga loss di set validasi cenderung fluktuatif. Pada beberapa titik bahkan mengalami kenaikan, yang dapat menandakan potensi *overfitting*. Meski demikian, adanya mekanisme *early stopping* dengan *patience = 30* dapat membantu menghentikan pelatihan ketika

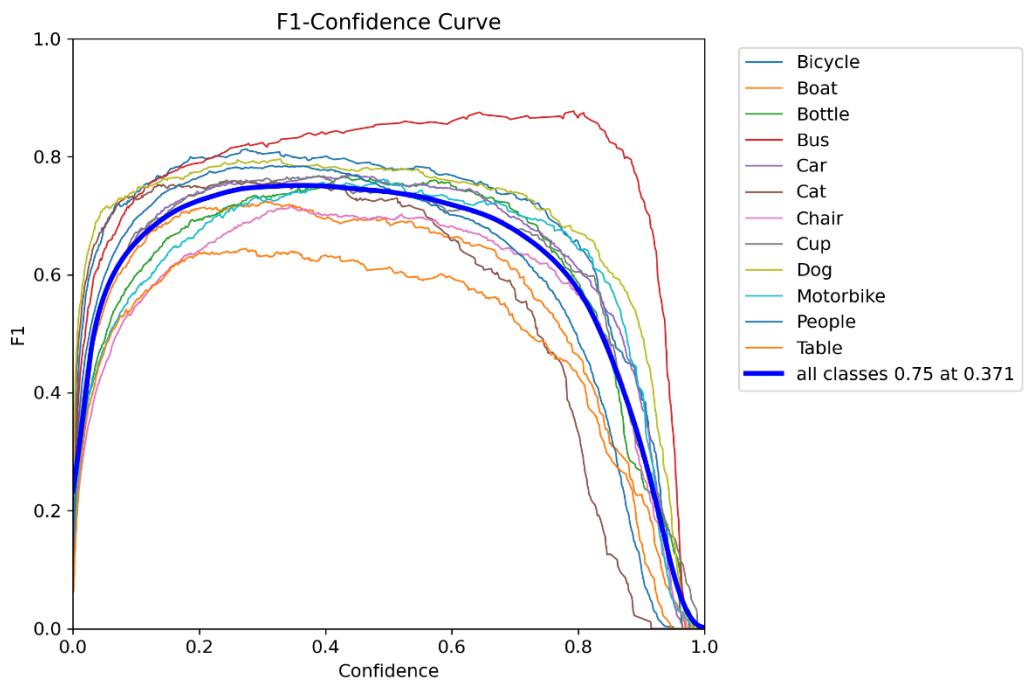
performa validasi tidak lagi meningkat dalam 30 epoch, sehingga diharapkan dapat mengurangi efek *overfitting* ini.

Di sisi lain, *map50* dan *map50-95* pada set validasi menunjukkan peningkatan secara umum, meskipun terlihat sedikit ketidakstabilan di beberapa bagian. Data ini menunjukkan bahwa model semakin mampu melakukan deteksi dengan akurasi lebih baik pada berbagai nilai IoU.



Gambar 4. 12 *Confusion Matrix Normalized* untuk YOLOv9m

Nilai diagonal yang tinggi pada Gambar 4.12 di atas menunjukkan bahwa meskipun model secara umum mampu melakukan deteksi dengan benar, terdapat kesalahan klasifikasi antar kelas dengan kemiripan fitur, seperti *Bicycle* dengan *Motorbike* dan *Chair* dengan *Table*, yang mengindikasikan ambiguitas dalam ekstraksi fitur halus pada kondisi pencahayaan rendah.



Gambar 4.13 Kurva F1-Confidence untuk YOLOv9m

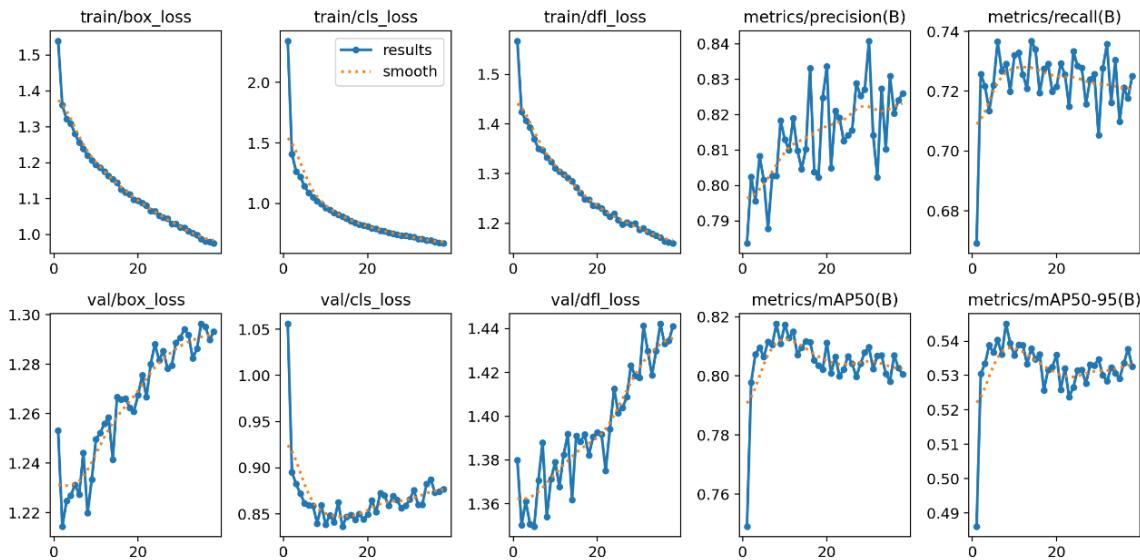
Gambar 4.13 di atas menunjukkan sebagian besar kelas mencapai puncak F1 pada kisaran *confidence* yang relatif mirip, mencerminkan bahwa terdapat *sweet spot* dimana model cukup yakin saat mendekripsi objek tanpa mengorbankan terlalu banyak *recall*. Dalam kondisi pencahayaan rendah, fluktuasi bentuk kurva di area puncak mengindikasikan bahwa untuk beberapa kelas, perubahan sedikit saja pada *threshold* dapat memengaruhi *precision* dan *recall* secara signifikan.

Tabel 4.4 Label vs Prediksi untuk YOLOv9m

No.	<i>Ground Truth (a)</i>	<i>Predicted (b)</i>
1.	<p>2015_04405.png</p> <p>Chair, People, Cup</p>	<p>2015_04405.png</p> <p>Chair 0.41, People 0.49, Cup 0.49</p>
2.	<p>2015_07323.png</p> <p>Chair, Table, Chair</p>	<p>2015_07323.png</p> <p>People 0.6, Chair 0.3, Chair 0.3</p>
3.	<p>2015_03014.png</p> <p>Car, Car, Car, Car, Car</p>	<p>2015_03014.png</p> <p>Car 0.9, Car 0.9, Car 0.6</p>

Tabel 4.4 di atas adalah sampel hasil dari pendekripsi objek oleh model YOLOv9m. YOLOv9m yang memiliki jumlah parameter lebih besar daripada YOLOv9s tampak lebih mampu mendekripsi objek dengan lebih akurat dan stabil. Dari tabel terlihat bahwa *bounding box* yang dihasilkan lebih mendekati *ground truth*, khususnya pada objek yang berukuran kecil atau sedikit tumpang tindih seperti pada gambar 2b dan 3b. Walau demikian, masih terdapat pula beberapa prediksi yang belum sempurna seperti terdeteksinya dua *People* dalam gambar 1b. Secara keseluruhan, peningkatan jumlah parameter pada YOLOv9m membantu model mengenali objek dengan *confidence* yang lebih besar dibandingkan YOLOv9s.

4.3.2.3 YOLOv9c



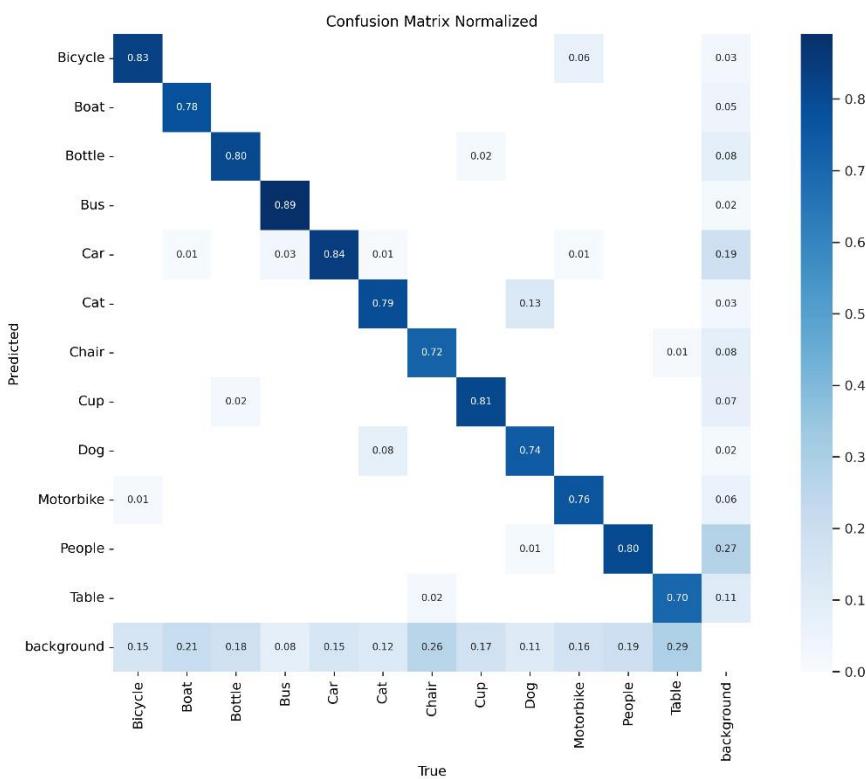
Gambar 4.14 Kurva hasil pelatihan model

Gambar 4.14 di atas menunjukkan grafik dari hasil pelatihan YOLOv9c dengan *optimizer SGD*, selama 1000 epoch. Terlihat bahwa *train loss* secara konsisten menurun dari nilai awal yang tinggi menuju nilai yang lebih stabil di akhir proses. Hal ini menunjukkan bahwa model mampu mempelajari fitur-fitur penting dalam dataset. Selain itu, metrik *precision* dan *recall* di sisi *training* mengalami peningkatan seiring waktu, menandakan bahwa model semakin tepat dalam melakukan prediksi *bounding box* serta semakin jarang melewatkannya objek.

Pada bagian validasi, *val loss* cenderung kebih fluktuatif meski terdapat kecenderungan penurunan secara umum. Fluktuasi ini bisa saja menandakan bahwa model mengalami kesulitan dalam menyelaraskan pembelajaran pada data yang belum pernah dilihat atau adanya kecenderungan *overfitting*, khususnya saat jumlah epoch terus bertambah. Namun, mekanisme *early stopping* dengan *patience 30* dapat membantu mengontrol hal tersebut agar pelatihan berhenti ketika peningkatan kinerja tidak lagi signifikan.

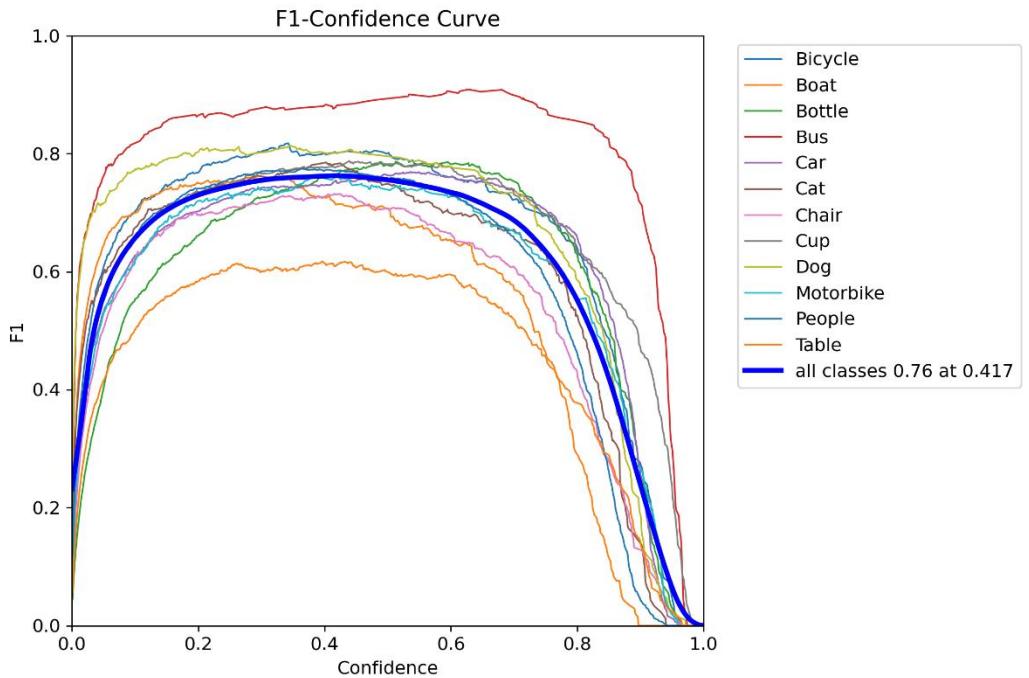
Dari segi metrik mAP, baik mAP50(B) maupun mAP50-95(B) mengalami tren kenaikan yang menggambarkan kemampuan model untuk mendeteksi objek dengan berbagai tingkat IoU. Peningkatan ini mengindikasikan bahwa YOLOv9c

berhasil menyerap pola-pola penting pada kondisi pencahayaan rendah dan mempertahankan kinerja generalisasi yang relatif baik.



Gambar 4.15 *Confusion Matrix Normalized* untuk YOLOv9c

Gambar 4.15 di atas merupakan *confusion matrix normalized* untuk YOLOv9c yang menunjukkan peningkatan akurasi cukup signifikan dengan nilai diagonal rata-rata lebih tinggi pada kelas *Bus* atau *Bicycle* yang mencapai lebih dari 0,8. Namun terlihat pula beberapa *misclassifications* yang masih terjadi, seperti *People* yang kadan terkласifikasi menjadi *background* atau *Motorbike*, membuktikan bahwa objek dengan kontur yang tidak terlalu tajam atau posisi yang tumpang tindih masih menjadi hal yang cukup sulit bahkan bagi model terbesar untuk YOLOv9 pada penelitian ini.



Gambar 4. 16 Kurva F1-Confidence untuk YOLOv9c

Gambar 4.16 di atas merupakan kurva *F1-Confidence* untuk YOLOv9c. Kurva masing-masing kelas cenderung naik hingga mencapai puncak F1 di kisaran *confidence* tertentu, kemudian turun drastis ketika *threshold* semakin ketat. Hal ini mengindikasikan bahwa model YOLOv9c, dengan kapasitas jaringan yang lebih besar, mampu menyeimbangkan *precision* dan *recall* yang lebih optimal.

Dari sisi implementasi, penentuan *threshold* yang tepat menjadi kunci untuk mendapatkan hasil deteksi terbaik. Pada titik F1 tertinggi (misalnya pada *confidence* 0,4-0,5), model dapat menghasilkan prediksi yang relatif baik dengan minim *false positives* maupun *false negatives*. Namun di kelas tertentu yang bentuknya lebih samar atau sering tumpang tindih, kurvanya dapat berbeda sedikit lebih rendah dan rentan berubah ketika *threshold* dinaikkan atau diturunkan.

Tabel 4. 5 Label vs Prediksi untuk YOLOv9c

No.	<i>Ground Truth (a)</i>	<i>Predicted (b)</i>
1.	<p>2015_04405.png</p> <p>Chair, People, Cup</p>	<p>2015_04405.png</p> <p>Chair 0.8, People 0.9, Cup 0.3, Bottle 0.5</p>
2.	<p>2015_07323.png</p> <p>People, Chair, Table, Chair</p>	<p>2015_07323.png</p> <p>People 0.9, Table 0.5, Chair 0.8, Chair 0.5</p>
3.	<p>2015_03014.png</p> <p>Car, Car, Car, Car, Car</p>	<p>2015_03014.png</p> <p>Car 0.3, Car 0.8, Car 0.8, Car 0.8, Car 0.8</p>

Pada tabel 4.5, terlihat sampel hasil prediksi YOLOv9c yang menunjukkan deteksi objek yang lebih mendekati *ground truth* dari segi *bounding box* dengan *confidence* yang lebih stabil. Meski masih terdapat sedikit kekeliruan pada beberapa objek yang penampakannya serupa atau berdekatan seperti pada gambar 1a, YOLOv9c dengan parameter terbesar di keluarga YOLOv9 membantu mengenali ciri-ciri visual lebih baik, sehingga secara umum hasil deteksi berada lebih dekat dengan *ground truth* dibanding model-model sebelumnya.

4.4 Implementasi RT-DETR

4.4.1 Pelatihan RT-DETR

Model RT-DETR yang digunakan terdiri dari dua varian, yaitu RTDETR-L dan RTDETR-X. Sama seperti pada YOLOv9, masing-masing varian dilatih menggunakan empat kombinasi *hyperparameter* hasil dari teknik *grid search*, sehingga menghasilkan total 8 model (2 varian \times 4 kombinasi). Hasil dari seluruh pelatihan model RT-DETR ini dijelaskan dalam Tabel 4.6.

Tabel 4. 6 Hasil *grid search* untuk RT-DETR

BATCH 8	Lr 0.01		Lr 0.001	
	L	X	L	X
total_epoch	113	138	54	59
total_time	33011.3	50155.7	15668.5	21347.9
train/giou_loss	0.44008	0.50669	0.41779	0.38656
train/cls_loss	0.6008	0.58061	0.59373	0.56301
train/l1_loss	0.19798	0.20241	0.20204	0.18394
metrics/precision(B)	0.80956	0.81877	0.83277	0.84243
metrics/recall(B)	0.71213	0.69621	0.68728	0.72232
metrics/mAP50(B)	0.76971	0.76535	0.75949	0.79138
metrics/mAP50-95(B)	0.4838	0.47706	0.48304	0.50915
val/giou_loss	0.46754	0.47455	0.46317	0.45104
val/cls_loss	0.67114	0.66686	0.68041	0.64739
val/l1_loss	0.38581	0.38788	0.37794	0.36973

BATCH 16	Lr 0.01		Lr 0.001	
	L	X	L	X
total_epoch	104	61	48	48
total_time	23291	17971.5	10759.5	14122.8
train/giou_loss	0.47121	0.45605	0.41929	0.39596
train/cls_loss	0.5811	0.58457	0.58013	0.55788
train/l1_loss	0.19763	0.19389	0.19391	0.18118
metrics/precision(B)	0.80266	0.81163	0.81956	0.85686
metrics/recall(B)	0.68876	0.70686	0.68933	0.73583
metrics/mAP50(B)	0.74552	0.76933	0.75277	0.80794
metrics/mAP50-95(B)	0.47541	0.48726	0.47302	0.51841
val/giou_loss	0.46774	0.47212	0.48153	0.44628

val/cls_loss	0.69491	0.67639	0.69523	0.64611
val/l1_loss	0.37382	0.38233	0.38661	0.35431

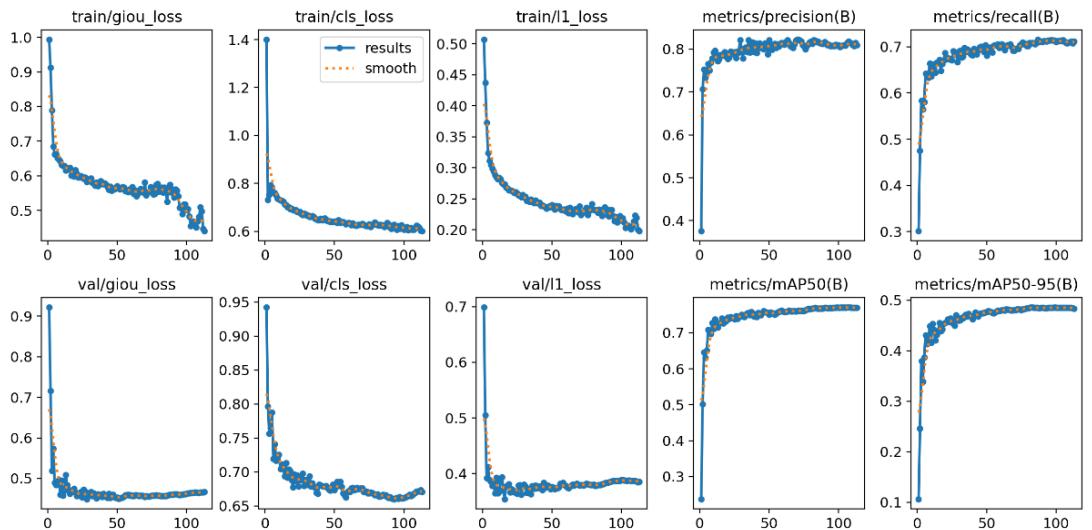
4.4.2 Kombinasi RTDETR terbaik

Berdasarkan metrik yang ditampilkan pada Tabel 4.7, diperoleh konfigurasi *hyperparameter* terbaik untuk masing-masing tipe RT-DETR, yang dijelaskan lebih lanjut pada bagian berikut.

Tabel 4. 7 Kombinasi terbaik untuk RT-DETR

Tipe Model	Batch Size	Learning Rate	mAP50	map50-95	val/cls_loss
RTDETR-L	8	0.01	0.76971	0.4838	0.67114
RTDETR-X	16	0.001	0.80794	0.51841	0.64611

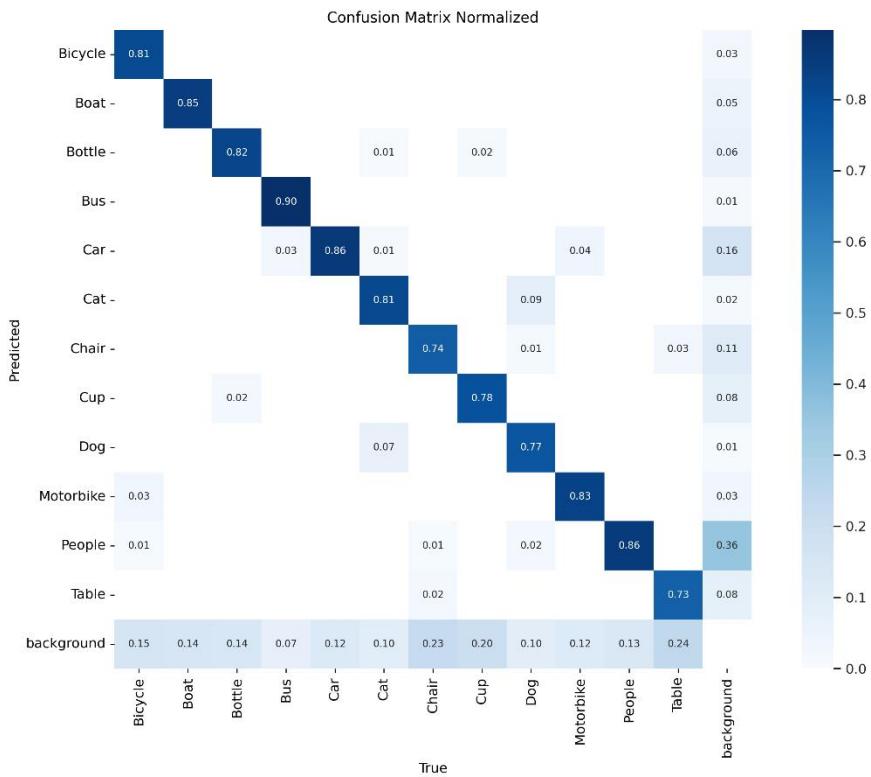
4.4.2.1 RTDETR-L



Gambar 4. 17 Kurva hasil pelatihan model RTDETR-L

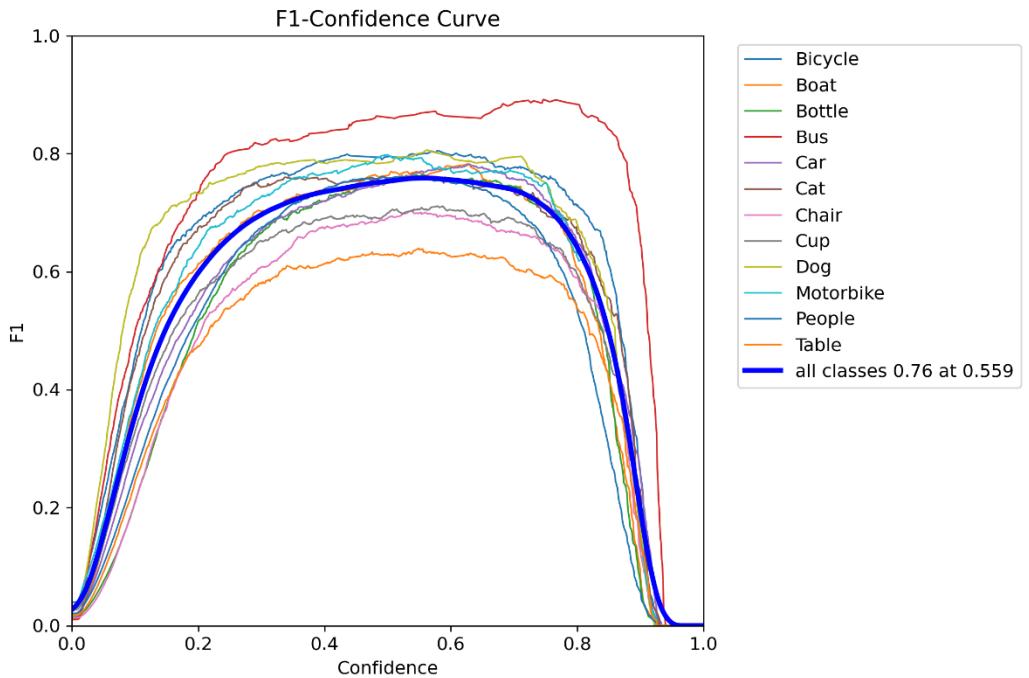
Gambar 4.17 merupakan kurva hasil pelatihan RTDETR-L. Terlihat bahwa performa model meningkat secara signifikan seiring bertambahnya epoch. Hal ini ditunjukkan dari tren penurunan konsisten pada metrik *train loss* utama yaitu *train/giou_loss*, *train/cls_loss*, dan *train/l1_loss*, yang mengindikasikan bahwa model berhasil mempelajari representasi fitur spasial, kelas objek, dan posisi dengan baik. Kinerja ini juga tercermin pada metrik evaluasi seperti *precision* dan *recall* di data pelatihan yang mengalami peningkatan cukup tajam di awal dan kemudian stabil pada nilai tinggi.

Evaluasi pada data validasi pun memberikan hasil yang sejalan. Grafik *val/giou_loss*, *val/cls_loss*, dan *val/l1_loss* menunjukkan pola penurunan yang stabil hingga mencapai titik konvergen, tanpa lonjakan signifikan yang menandakan *overfitting*. Hal ini memperkuat bukti bahwa model mampu melakukan generalisasi terhadap data baru yang tidak dilihat selama pelatihan. Selain itu, metrik mAP50(B) dan mAP50-95(B) yang terus meningkat dan stabil di akhir pelatihan menjadi indikator utama bahwa RTDETR-L menghasilkan prediksi *bounding box* yang presisi pada berbagai tingkatan *threshold IoU*.



Gambar 4. 18 Confusion Matrix Normalized untuk REDETR-L

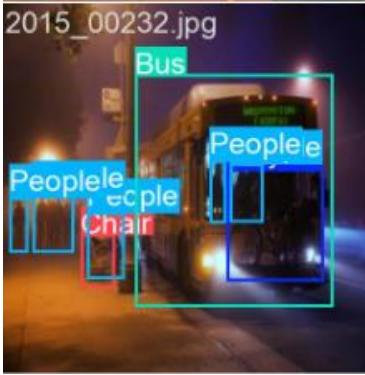
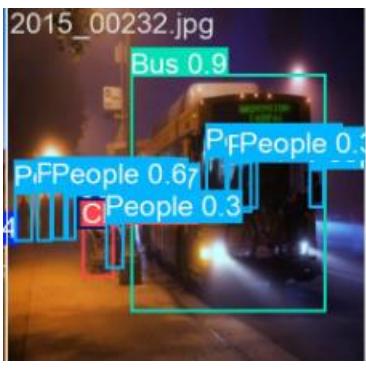
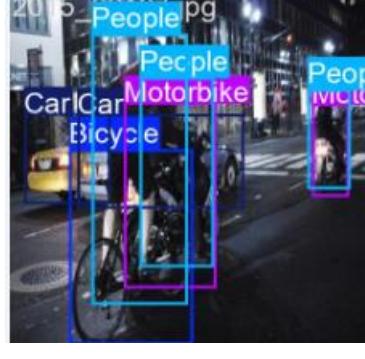
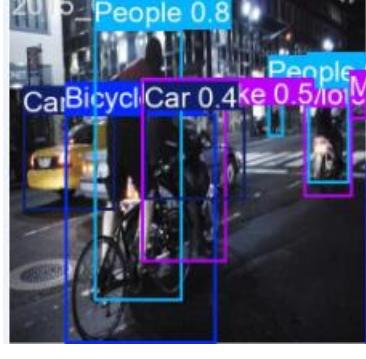
Gambar 4.8 di atas adalah gambar *confusion matrix normalized* untuk RTDETR-L yang memperlihatkan bahwa model dengan parameter besar ini mampu membedakan mayoritas kelas dengan akurasi tinggi, dapat dilihat dari nilai diagonal yang dominan di atas 0,8 untuk beberapa kelas seperti *Bus*, *Car*, dan *Dog*. Hal ini menunjukkan kapasitas model yang lebih besar membantu mengenali detail visual walaupun pencahayaan minim. Meski demikian, terlihat pula kekeliruan pada beberapa kelas yang memiliki kesamaan bentuk, misalnya *Motorbike* yang terkadang terklasifikasi sebagai *Bicycle*, menandakan bahwa fitur dan kontur halus lagi-lagi masih sulit untuk dibedakan sepenuhnya oleh model.



Gambar 4. 19 Kurva F-Confidence untuk REDETR-L

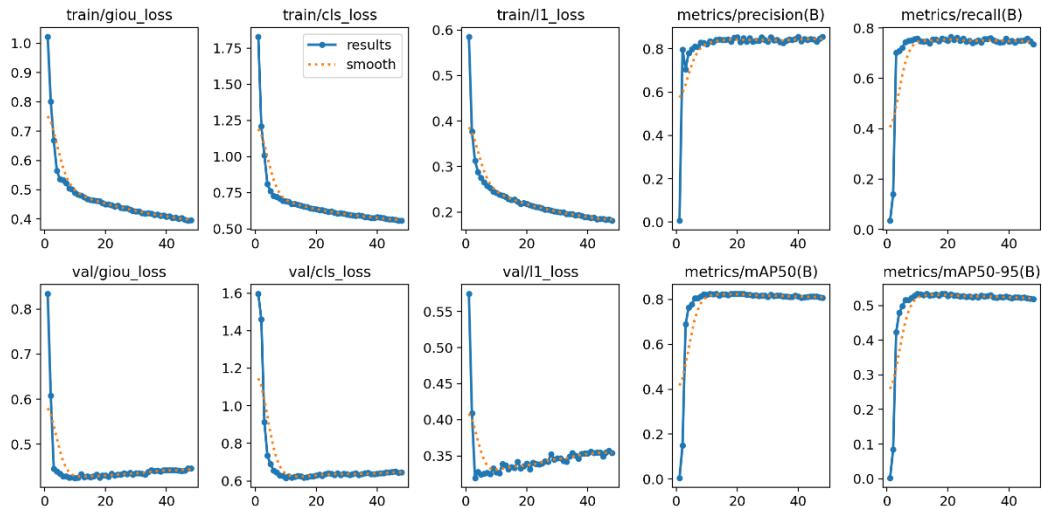
Gambar 4.19 di atas merupakan kurva *F1-Confidence* untuk RTDETR-L yang memperlihatkan bahwa model ini mampu mencapai keseimbangan *precision* dan *recall* di rentang *confidence* tertentu yang relatif lebih luas dibandingkan model YOLOv9 secara keseluruhan. Secara umum, mayoritas kelas menunjukkan kenaikan F1 yang cukup tajam di awal saat *confidence threshold* masih rendah, kemudian mencapai titik puncak dan mulai menurun ketika *threshold* semakin tinggi. Meskipun demikian, terdapat pula kelas tertentu yang kurvanya relatif lebih cepat turun saat *confidence threshold* meningkat, mengindikasikan bahwa perbedaan visual objek pada cahaya rendah masih saja menimbulkan tantangan pada pemisahan fitur yang halus.

Tabel 4. 8 Label vs Prediksi untuk RTDETR-L

No.	<i>Ground Truth (a)</i>	<i>Predicted (b)</i>
1.	 <p>2015_00232.jpg</p> <p>Bus People Chair People</p>	 <p>2015_00232.jpg</p> <p>Bus 0.9 PfPeople 0.3 People 0.67 C People 0.3</p>
2.	 <p>2015_00217.jpg</p> <p>Bicycle Dog</p>	 <p>2015_00217.jpg</p> <p>Table 0.6 0.9 Dog 0.8</p>
3.	 <p>2015_People.jpg</p> <p>Car People Motorbike Bicycle</p>	 <p>2015_People.jpg</p> <p>Car Bicycl People Car 0.4 ke 0.5 / o M</p>

Tabel 4.8 di atas merupakan sampel hasil dari model RTDETR-L. Dengan parameter yang lebih besar dibandingkan dengan YOLOv9, RTDETR-L tampak menandingi bahkan melebihi akurasi deteksi, terutama pada objek yang lebih kecil atau saling berdekatan seperti pada gambar 3b. Model ini secara keseluruhan memiliki prediksi *bounding box* yang lebih tepat dan *confidence score* yang relatif stabil.

4.4.2.2 REDETR-X



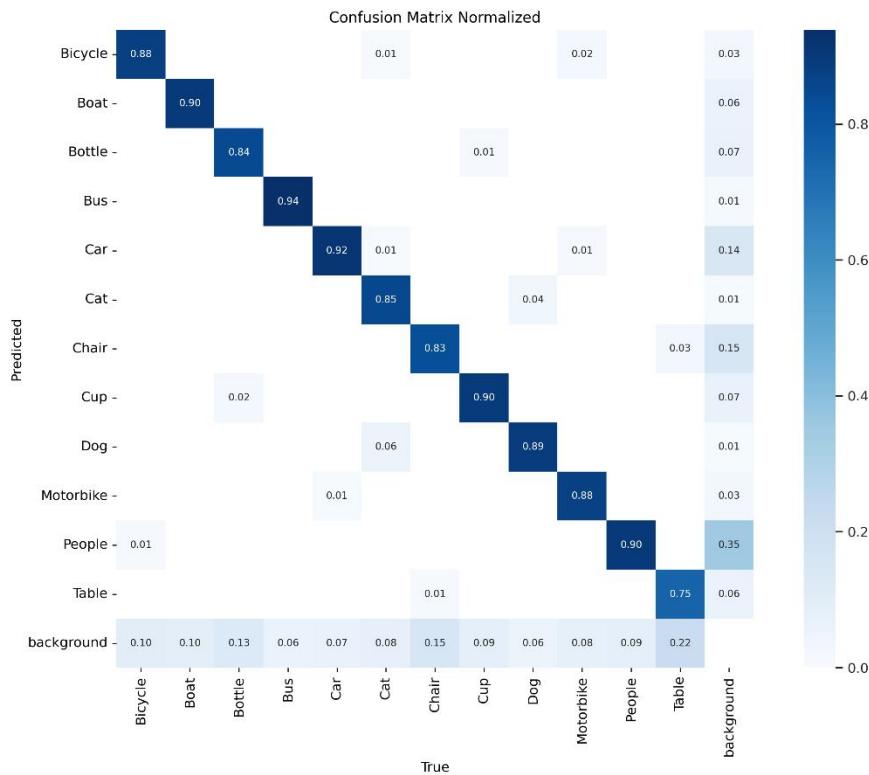
Gambar 4. 20 Kurva hasil pelatihan model REDETR-X

Gambar 4.20 di atas adalah gambar performa model RTDETR-X selama pelatihan. Tampak bahwa *train loss* turun secara konsisten, menandakan bahwa model dengan arsitektur serta parameter yang paling besar ini mampu mempelajari fitur-fitur penting pada citra dengan efektif. Tren kurva *precision* dan *recall* pada *train set* juga menunjukkan peningkatan yang stabil hingga mencapai nilai mendekati konvergen di akhir pelatihan, menandakan model semakin baik dalam mendekripsi objek dengan akurasi tinggi dan minim kesalahan deteksi.

Dari sisi validasi, kurva-kurvanya cenderung menurun secara umum meski sempat mengalami fluktuasi pada beberapa titik. Namun, mekanisme *early stopping* dengan *patience* 30 membantu menjaga model agar tidak *overfitting*. Peningkatan pada metrik mAP50(B) dan mAP50-95(B) di *validation set* juga mendukung pemahaman bahwa RTDETR-X dapat menangkap berbagai variasi objek dalam kondisi cahaya minim dengan rentang IoU yang luas.

Dibandingkan dengan model-model sebelumnya, RTDETR-X memiliki parameter dan kedalaman arsitektur yang jauh lebih besar, sehingga mampu menyerap lebih banyak informasi visual. Meskipun

memerlukan sumber daya komputasi yang lebih besar, hasil pelatihan yang konsisten dan peningkatan mAP mencerminkan daya generalisasi dan ketepatan deteksi yang lebih tinggi.

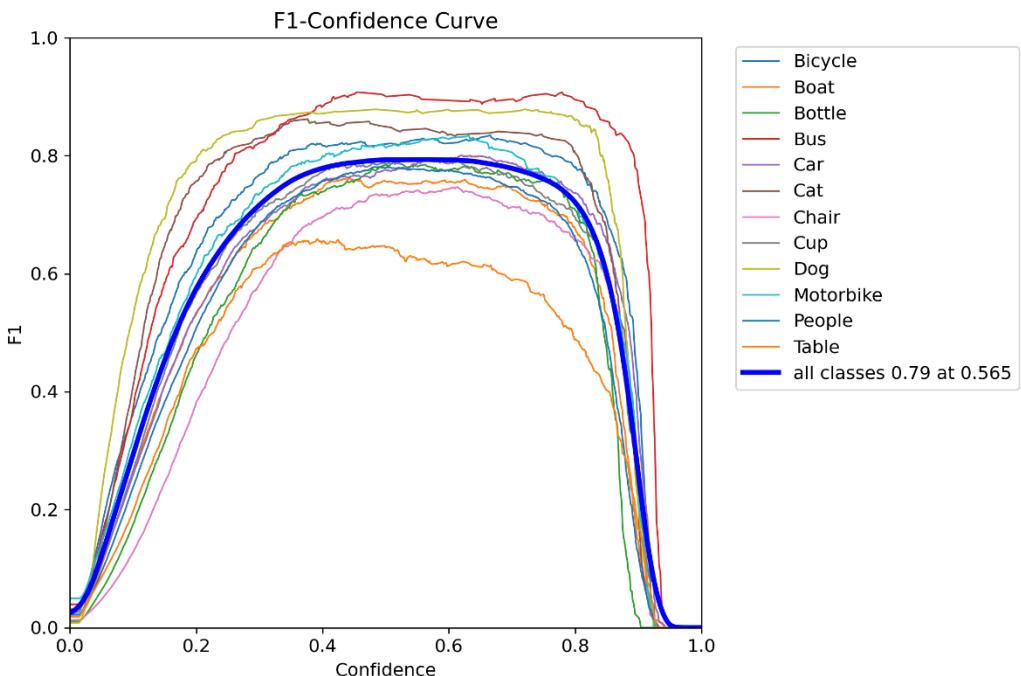


Gambar 4. 21 Confusion matrix Normalized untuk REDETR-X

Confusion Matrix Normalized pada gambar 4.21 menunjukkan bahwa model dengan parameter jauh lebih besar mampu mengklasifikasikan mayoritas kelas dengan akurasi tinggi, terlihat dari nilai diagonal yang dominan di atas 0,8. Pada beberapa kelas seperti *Bus* dan *Car*, kemampuan deteksi di kondisi pencahayaan rendah tampak semakin baik. Meski demikian, lagi-lagi masih terdapat kebingungan kecil di antara kelas yang mirip, seperti *Bicycle* dan *Motorbike*, menandakan bahwa bentuk, sudut, atau kontur yang serupa dapat menimbulkan *misclassification*.

Dari sisi performa secara keseluruhan, RTDETR-X menunjukkan peningkatan yang substansial dibandingkan model-model sebelumnya,

terutama dalam mengenali objek berukuran kecil relatif kecil atau objek yang cenderung tersamar di *background*.

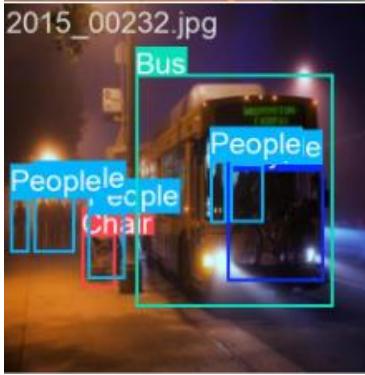
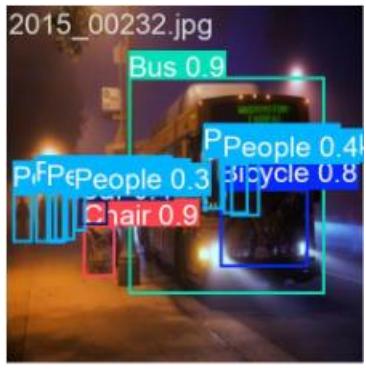
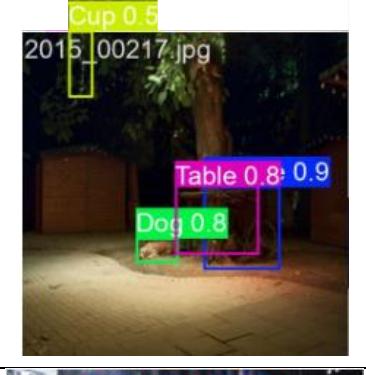
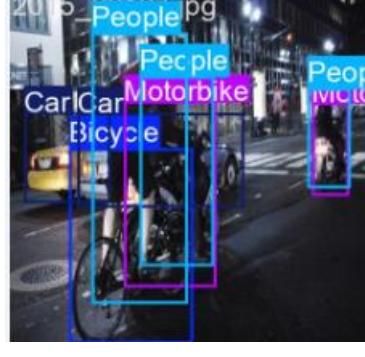
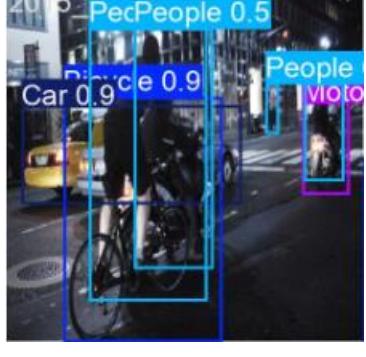


Gambar 4. 22 Kurva F1-Confidence untuk RTDETR-X

Kurva *F1-Confidence* pada Gambar 4.22 memperlihatkan bahwa model RTDETR-X mampu mencapai keseimbangan *precision* dan *recall* yang cukup baik di rentang *confidence* menengah, terlihat kenaikan F1 yang stabil sebelum akhirnya menurun saat ambang *confidence* makin tinggi. Pada beberapa kelas, seperti *Car* dan *Bus*, kurva cenderung lebih tinggi dan lebih stabil dibanding kelas yang memiliki kemiripan ciri fisik seperti *Bicycle* dan *Motorbike* sehingga rentan terjadi kesalahan klasifikasi. Hal ini mengindikasikan bahwa kapasitas model yang lebih besar membantu pengenalan ciri objek yang jelas, tetapi tetap memiliki tantangan yang sama seperti model-model sebelumnya saat fitur objek berbeda tipis dalam pencahayaan gelap.

Dibandingkan dengan model berparameter lebih kecil, RTDETR-X menampilkan puncak F1 yang lebih tinggi dan area puncak yang cenderung lebih lebar, menegaskan kemampuan model untuk mempertahankan keseimbangan antara menghindari *false positives* dan tidak melewatkkan objek. Kondisi *low-light* pada ExDark masih menjadi kendala di kelas-kelas tertentu, namun peningkatan F1 di berbagai rentang *confidence* menekankan bahwa RTDETR-X lebih tangguh dalam mengenali perbedaan tekstur dan kontras yang minimal.

Tabel 4. 9 Label vs Prediksi untuk RTDETR-X

No.	<i>Ground Truth (a)</i>	<i>Predicted (b)</i>
1.	 <p>2015_00232.jpg</p> <p>Bus People People Chair</p>	 <p>2015_00232.jpg</p> <p>Bus 0.9 People 0.4 People 0.3 Bicycle 0.8 Chair 0.9</p>
2.	 <p>2015_00217.jpg</p> <p>Bicycle Dog</p>	 <p>2015_00217.jpg</p> <p>Cup 0.5 Table 0.8 Dog 0.8</p>
3.	 <p>2015_00127.jpg</p> <p>Car People Motorbike Bicycle</p>	 <p>2015_00127.jpg</p> <p>Car 0.9 People 0.5 People 0.9 Bicycle 0.8</p>

Tabel 4.9 di atas merupakan sampel hasil dari deteksi objek menggunakan model RTDETR-X. Terlihat bahwa model yang memiliki parameter terbesar ini memiliki *bounding box* yang lebih mendekati *ground truth*, terutama untuk objek yang kecil atau tumpang tindih dengan *confidence* yang lebih tinggi seperti pada gambar 3b. Tetapi, justru menciptakan kebingungan baru seperti pada gambar 2b yang cukup jauh berbeda dengan gambar 2a. Selebihnya, model ini juga dapat

memilah atau mengambil suatu objek secara lebih detail seperti ditunjukkan pada objek-objek *People* di gambar 1b yang terlihat sangat terpisah dengan rapi.

4.5 Implementasi YOLOv10

4.5.1 Pelatihan YOLOv10

Model YOLOv10 yang digunakan terdiri dari tiga varian, yaitu YOLOv10s, YOLOv10m, dan YOLOv10b. Seperti halnya dua model sebelumnya, masing-masing varian YOLOv10 diuji dengan empat kombinasi *hyperparameter* melalui teknik *grid search*, menghasilkan total 12 model ($3 \text{ varian} \times 4 \text{ kombinasi}$). Seluruh hasil pelatihan dari model YOLOv10 ini disajikan dalam Tabel 4.10.

Tabel 4.10 Hasil *grid search* untuk YOLOv10

BATCH 8	Lr 0.01			Lr 0.001		
	s	m	b	s	m	b
total_epoch	109	174	150	117	188	54
total_time	12322	23914.4	23594.6	13003.2	16472.6	8470.51
train/box_loss	243.436	201.078	192.587	199.731	198.211	190.896
train/cls_loss	198.149	148.982	141.913	144.895	147.546	134.995
train/dfl_loss	240.514	231.333	228.432	21.366	229.442	223.828
metrics/precision(B)	0.79927	0.78046	0.82146	0.80476	0.80271	0.82051
metrics/recall(B)	0.67451	0.71035	0.67349	0.68341	0.69555	0.70915
metrics/mAP50(B)	0.75557	0.77337	0.77444	0.75887	0.77108	0.79314
metrics/mAP50-95(B)	0.48525	0.50709	0.51212	0.49627	0.50846	0.52548
val/box_loss	280.277	278.851	279.928	280.489	278.668	270.202
val/cls_loss	209.757	209.591	206.692	223.116	208.539	203.039
val/dfl_loss	282.986	317.003	323.184	279.423	316.092	301.239

BATCH 16	Lr 0.01			Lr 0.001		
	s	m	b	s	m	b
total_epoch	145	139	107	128	54	38
total_time	10807.6	15082.1	14816.3	9511.46	5874.28	5244.12
train/box_loss	226.507	200.573	198.476	189.915	196.896	199.209
train/cls_loss	172.874	146.065	142.712	132.568	138.849	141.819
train/dfl_loss	227.508	229.004	229.662	207.403	224.978	227.501
metrics/precision(B)	0.77576	0.81938	0.81252	0.80703	0.80623	0.80625
metrics/recall(B)	0.67603	0.67797	0.6814	0.67497	0.70078	0.69801
metrics/mAP50(B)	0.74654	0.7749	0.76618	0.75515	0.77668	0.78232
metrics/mAP50-95(B)	0.48057	0.50907	0.49944	0.49512	0.51339	0.52283
val/box_loss	286.094	278.814	279.254	281.667	272.104	268.403
val/cls_loss	218.469	209.035	213.922	228.229	21.165	202.321
val/dfl_loss	285.825	313.941	31.927	280.646	300.242	295.285

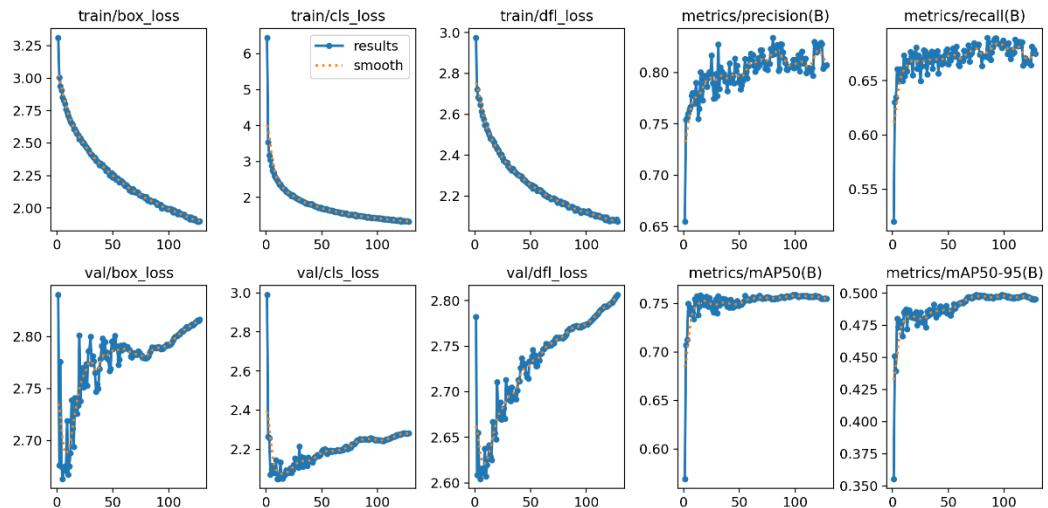
4.5.2 Kombinasi YOLOv10 terbaik

Berdasarkan metrik yang tercantum pada Tabel 4.11, diperoleh kombinasi *hyperparameter* terbaik untuk masing-masing varian YOLOv10 yang akan diuraikan pada bagian berikut.

Tabel 4. 11 Kombinasi terbaik untuk YOLOv10

Tipe Model	Batch Size	Learning Rate	mAP50	map50-95	val/cls_loss
YOLOv10s	16	0.001	0.75515	0.49512	228.229
YOLOv10m	16	0.001	0.77668	0.51339	0.21165
YOLOv10b	8	0.01	0.77444	0.51212	206.692

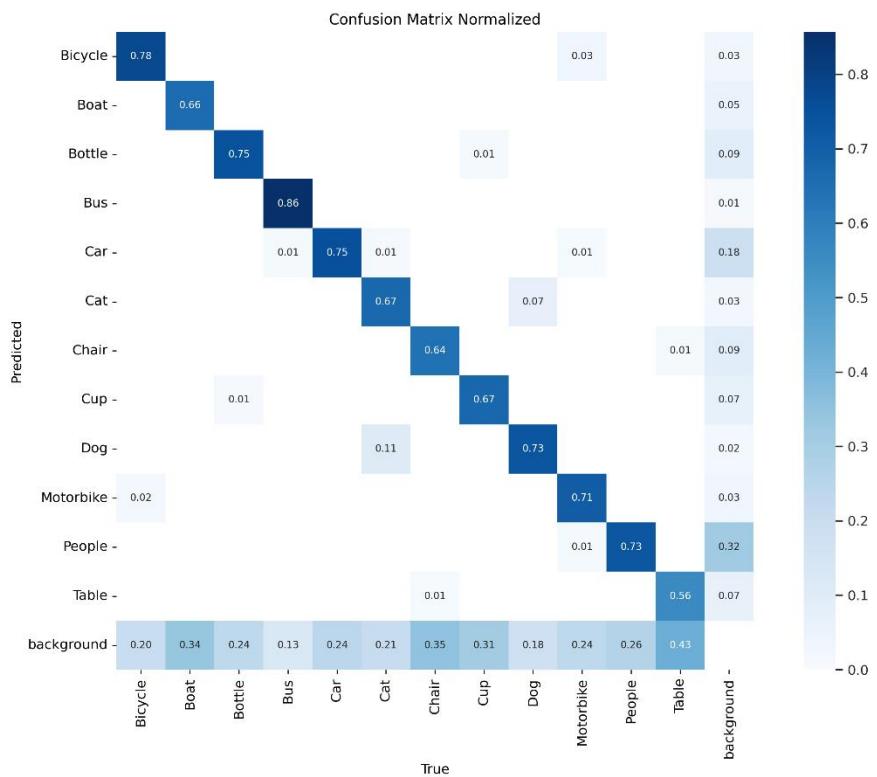
4.5.2.1 YOLOv10s



Gambar 4.23 Kurva hasil pelatihan model YOLOv10s

Pada gambar 4.23, grafik hasil pelatihan YOLOv10s, ketiga loss utama di bagian *training* tampak menurun secara konsisten dari nilai tinggi di awal menuju tingkat yang semakin stabil. Peningkatan metrik *precision(B)* dan *recall(B)* mengindikasikan bahwa model semakin baik dalam mengurangi *false positives* serta jarang melewatkkan objek yang seharusnya terdeteksi.

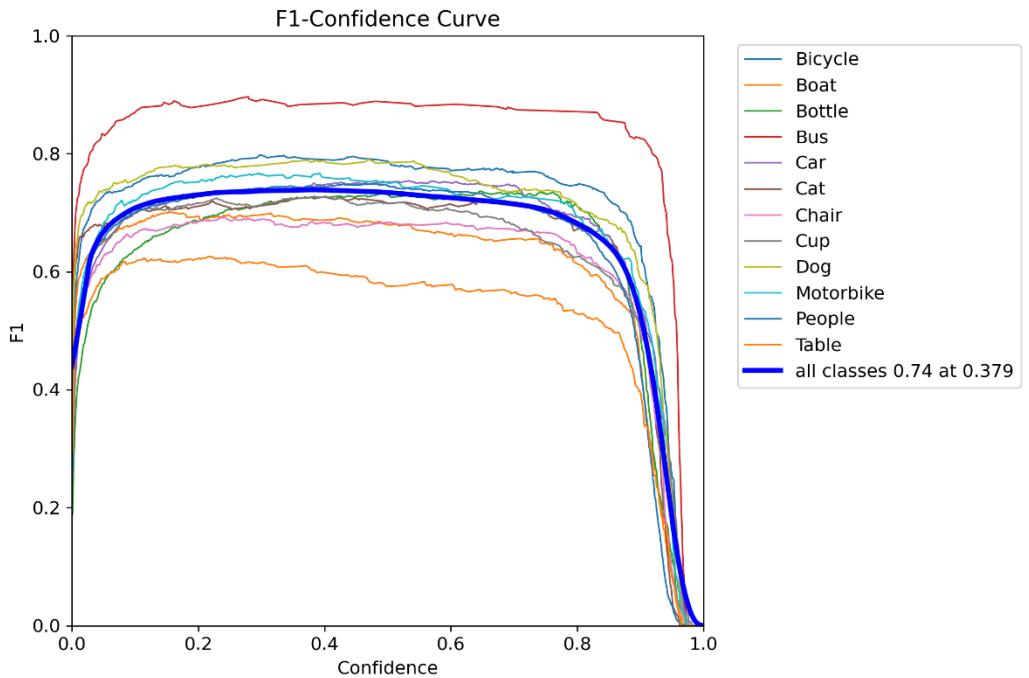
Pada *validation set* cenderung mengalami fluktuasi, namun masih membentuk tren penurunan hingga titik tertentu. Meskipun sempat naik di beberapa titik, mekanisme *early stopping* dengan *patience 30* membantu menjaga model dari *overfitting*. Grafik mAP50(B) dan mAP50-95(B) memperlihatkan kenaikan kinerja yang signifikan di fase awal, kemudian tetap stabil dan sedikit meningkat di akhir pelatihan.



Gambar 4.24 Confusion Matrix Normalized untuk YOLOv10s

Confusion Matrix Normalized untuk YOLOv10s pada Gambar 4.24 menunjukkan bahwa model ini secara umum berhasil mengklasifikasikan objek dengan akurasi yang cukup baik, terbukti dari nilai diagonal yang cukup tinggi. Meski demikian, terlihat pula adanya kekeliruan pada kelas tertentu yang berukuran kecil seperti *Cup* yang kadang terdeteksi sebagai *Bottle*, dan lagi-lagi *Bicycle* dan *Motorbike* yang sesekali tertukar.

Secara keseluruhan, meskipun beberapa objek masih menimbulkan kebingungan dengan *background*, hasil ini menandakan bahwa tipe terkecil dari YOLOv10 yaitu YOLOv10s sudah cukup baik untuk mendeteksi objek untuk citra berpencahayaan rendah.



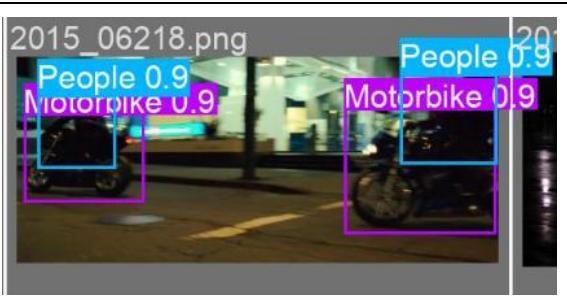
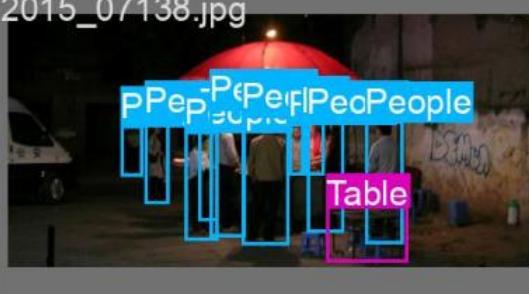
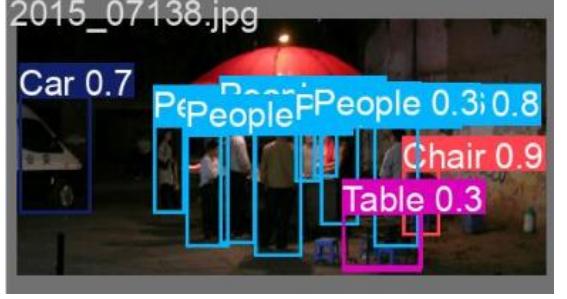
Gambar 4.25 Kurva F1-Confidence untuk YOLOv10s

Gambar 4.25 menunjukkan *F1-Confidence Curve* untuk YOLOv10s yang membuktikan bahwa mayoritas kelas memiliki puncak F1 yang cukup tinggi di rentang *threshold* menengah, kemudian menurun drastis ketika *threshold* diperketat. Hal ini menandakan bahwa meski model semakin yakin terhadap prediksinya, YOLOv10s juga lebih rentan kehilangan beberapa deteksi. Pada beberapa kelas dengan ciri fisik yang jelas seperti *Bus* atau *Car*, puncak F1 terlihat lebih tinggi dan stabil dibanding kelas yang memiliki kemiripan ciri seperti *Bicycle* dan *Motorbike*.

Secara keseluruhan, capaian F1 di angka 0,84 (pada *confidence* 0,379), memberikan indikasi bahwa model telah menemukan *sweet spot* untuk menyeimbangkan *false positives* dan *false negatives*. Namun, adanya variasi kurva antarkelas menunjukkan bahwa pencahayaan minim dan fitur objek yang serupa masih dapat menimbulkan kesalahan klasifikasi. Meskipun demikian, bentuk kurva yang relatif mulus dan puncak yang cukup stabil pada beberapa kelas menegaskan kemampuan YOLOv10s

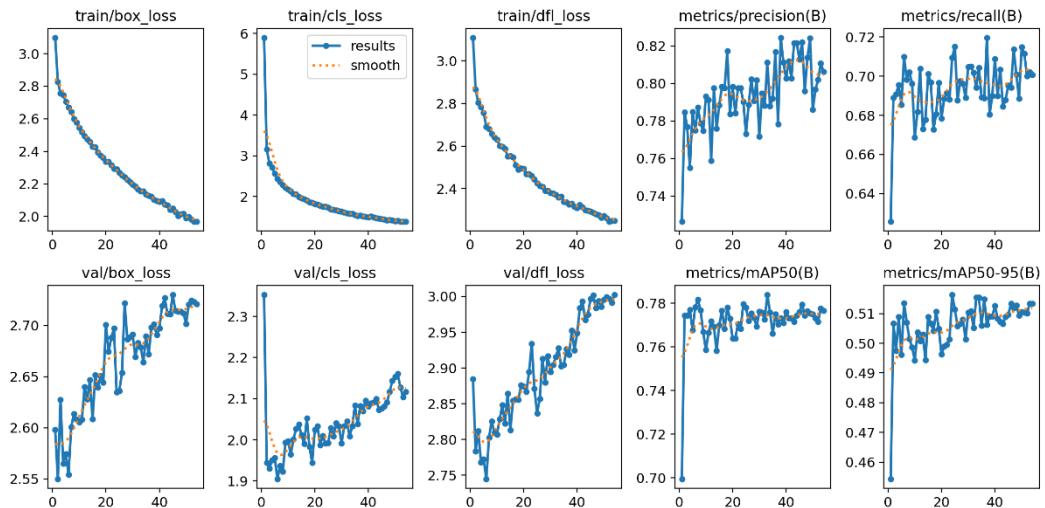
sebagai parameter terkecil dari YOLOv10 lainnya, mampu untuk mempertahankan kinerja yang baik.

Tabel 4.12 Label vs Prediksi untuk YOLOv10s

No.	<i>Ground Truth (a)</i>	<i>Predicted (b)</i>
1.	 <p>2015_06218.png People Motorbike</p>	 <p>2015_06218.png People 0.9 Motorbike 0.9</p>
2.	 <p>2015_07138.jpg People Car Chair Table</p>	 <p>2015_07138.jpg Car 0.7 People 0.3 People 0.8 Chair 0.9 Table 0.3</p>
3.	 <p>2015_01461.jpg Bottle Bottle Bottle Bottle Bottle</p>	 <p>2015_01461.jpg Bottle 0.9 Table 0.8</p>

Tabel 4.12 di atas adalah sampel hasil dari deteksi oleh model YOLOv10s yang memiliki parameter terkecil diantara YOLOv10 lainnya. Tabel di atas menunjukkan kemampuan deteksi yang cukup baik secara *bounding box* walau masih tedapat sedikit ketidakakuratan pada kelas-kelas yang saling tumpang tindih atau objek kecil. Namun uniknya, model ini mampu mendeteksi objek yang tidak terdaftar dalam *ground truth* tetapi terlihat secara kasat mata secara cukup meyakinkan seperti pada objek *Car* di gambar 2b dan *Table* di gambar 3b.

4.5.2.2 YOLOv10m

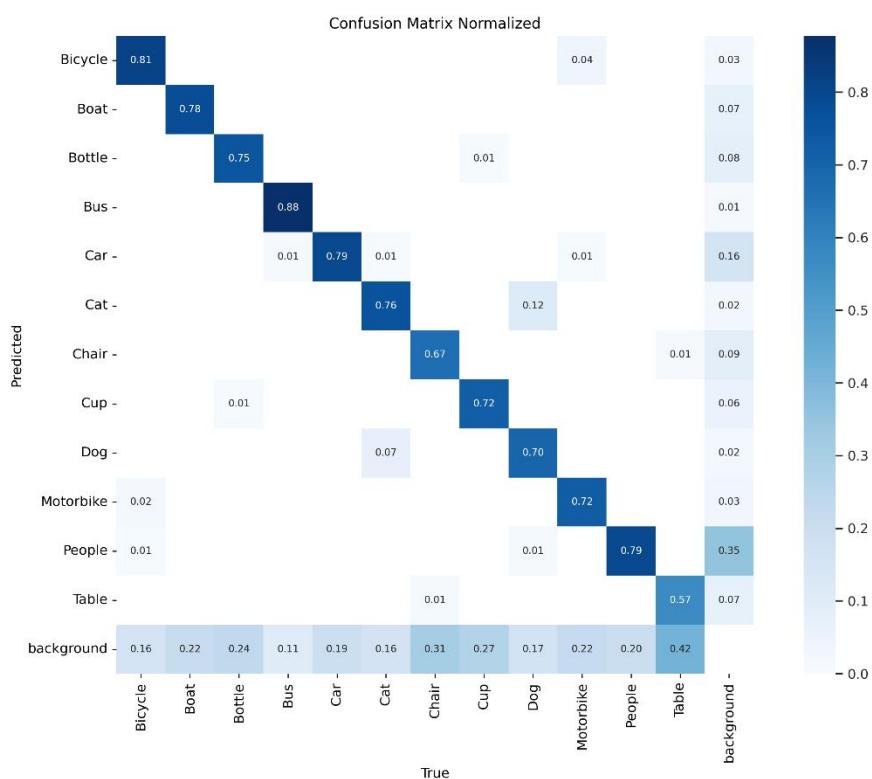


Gambar 4.26 Kurva hasil pelatihan model YOLOv10m

Kurva-kurva untuk model YOLOv10m pada gambar 4.26 di atas menunjukkan bahwa selama pelatihan, ketiga *loss* utama di fase *training* menurun secara konsisten hingga mencapai titik yang makin stabil di akhir. Di saat yang sama, metrik *precision(B)* dan *recall(B)* menunjukkan tren peningkatan, menandakan kemampuan model yang kian baik dalam menempatkan *bounding box* dengan akurasi tinggi dan semakin jarang melewatkannya objek.

Pada bagian validasi, kurva *val/box_loss*, *val/cls_loss*, dan *val/dfl_loss* memperlihatkan pola yang cenderung naik-turun, namun secara umum masih membentuk tren penurunan di beberapa segmen, sebelum akhirnya stagnan atau fluktuatif. Meski menandakan kecenderungan *overfitting* di beberapa titik yang ditunjukkan oleh *val loss* yang kadang naik, penerapan mekanisme *early stopping* membantu mencegah peuranan yang lebih parah. Hal

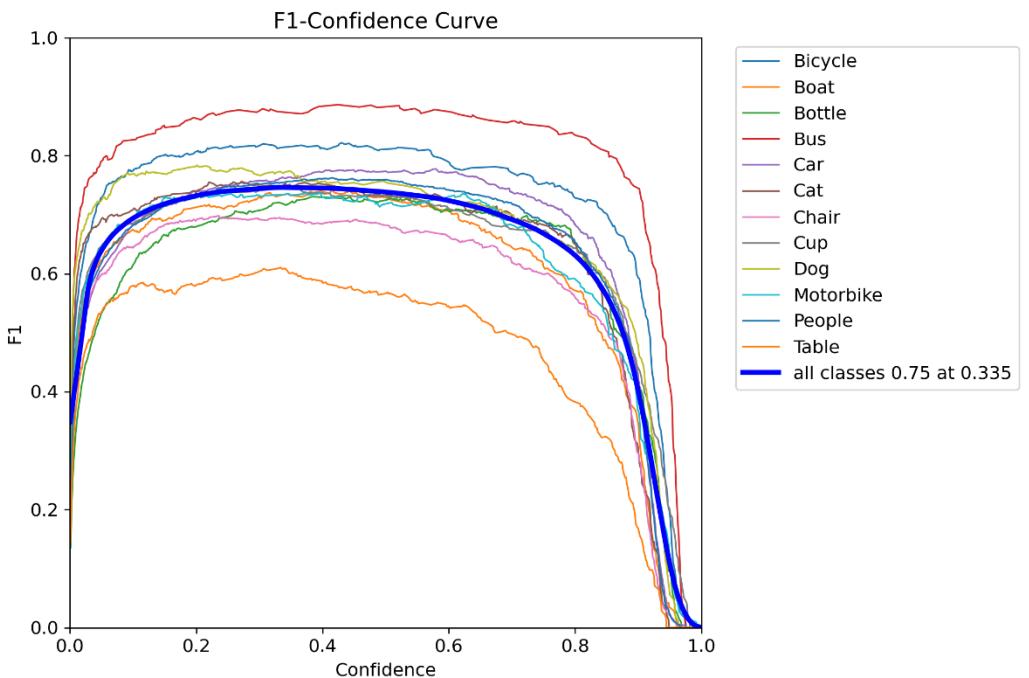
ini didukung oleh kenaikan metrik mAP50(B) dan mAP50-95(B) yang menggambarkan kemampuan deteksi model pada berbagai rentang IoU.



Gambar 4. 27 *Confusion Matrix Normalized* untuk YOLOv10s

Gambar 4.27 di atas adalah *confusion matrix* yang sudah dinormalisasi untuk YOLOv10m, yang menunjukkan bahwa model secara umum memiliki tingkat akurasi klasifikasi yang cukup tinggi, terlihat dari nilai diagonal yang dominan pada beberapa kelas seperti *Bus* dan *Car*. Namun, lagi-lagi masih saja terdapat kekeliruan pada kelas dengan bentuk yang mirip seperti *Bicycle* dan *Motorbike*, serta objek-objek berukuran kecil yang cenderung sulit diidentifikasi pada kondisi cahaya rendah. Beberapa *misclassifications* juga muncul antara *People* dan *background*, menandakan bahwa model belum sepenuhnya mampu membedakan objek berkontras rendah dari latar belakang.

Secara keseluruhan, hasil ini menekankan bahwa meski YOLOv10m dapat mengenali mayoritas kelas dengan baik, kondisi pencahayaan minim masih menjadi tantangan yang memicu kebingungan di antara kelas tertentu.

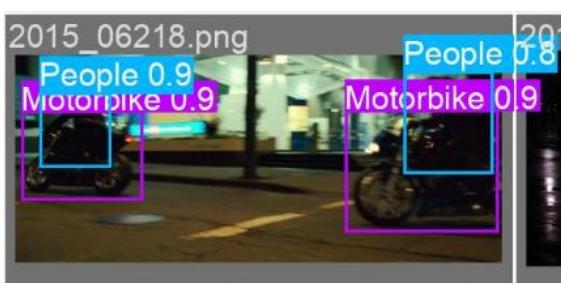
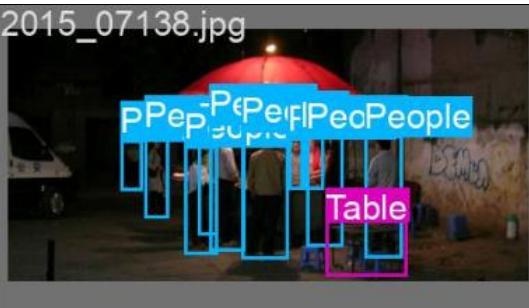


Gambar 4. 28 Kurva F1-Confidence untuk YOLOv10m

Kurva *F1-Confidence* pada gambar 4.28 di atas memperlihatkan bahwa YOLOv10m mencapai puncak keseimbangan antara *precision* dan *recall* di rentang *confidence* menengah. Nilai F1 tertinggi di kisaran 0,75 pada *confidence* sekitar 0,335 menggambarkan bahwa di titik tersebut, model dapat meminimalkan kesalahan deteksi, baik *false positives* maupun *false negatives*, dengan mempertahankan tingkat keyakinan yang memadai terhadap prediksinya.

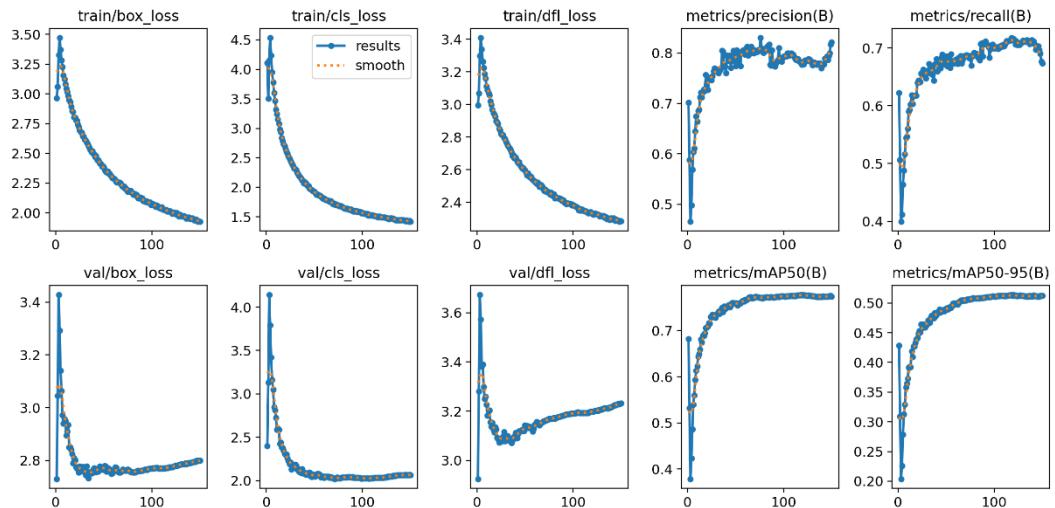
Di beberapa kelas, terutama yang cirinya lebih kontras seperti *Car* atau *Bus*, kurva terlihat lebih stabil dan mencapai puncak yang lebih tinggi, menunjukkan model lebih mudah mengenali objek yang jelas meski dalam kondisi cahaya rendah. Namun pada kelas dengan detail serupa atau berukuran kecil, seperti *Bottle* dan *Cup*, lagi-lagi kurva cenderung kurang stabil dan puncaknya lebih rendah, menandakan tantangan dalam membedakan ciri-ciri visual yang samar.

Tabel 4. 13 Label vs Prediksi untuk YOLOv10m

No.	<i>Ground Truth (a)</i>	<i>Predicted (b)</i>
1.	2015_06218.png 	2015_06218.png 
2.	2015_07138.jpg 	2015_07138.jpg 
3.	2015_01461.jpg 	2015_01461.jpg 

Sampel perbandingan *ground truth* dan *predicted* pada tabel 4.13 di atas menunjukkan bahwa model kemampuan deteksi YOLOv10m semakin mendekati *ground truth*. Perbedaan paling terlihat adalah pada gambar 3b yang memiliki tingkat kemiripan dan *confidence* yang sangat tinggi. Terlebih lagi, objek *Table* yang sebelumnya muncul di gambar 3b, tidak lagi muncul pada YOLOv10m ini. Tapi tetap saja, pada gambar 2b masih terdeteksi *Car* yang sebenarnya tidak terdaftar pada *grund truth label*. Secara umum model dengan parameter yang lebih besar daripada YOLOv10s ini menunjukkan perkembangan yang lebih baik dari segi *bounding box* pada objek-objek yang saling tumpang tindih.

4.5.2.3 YOLOv10b



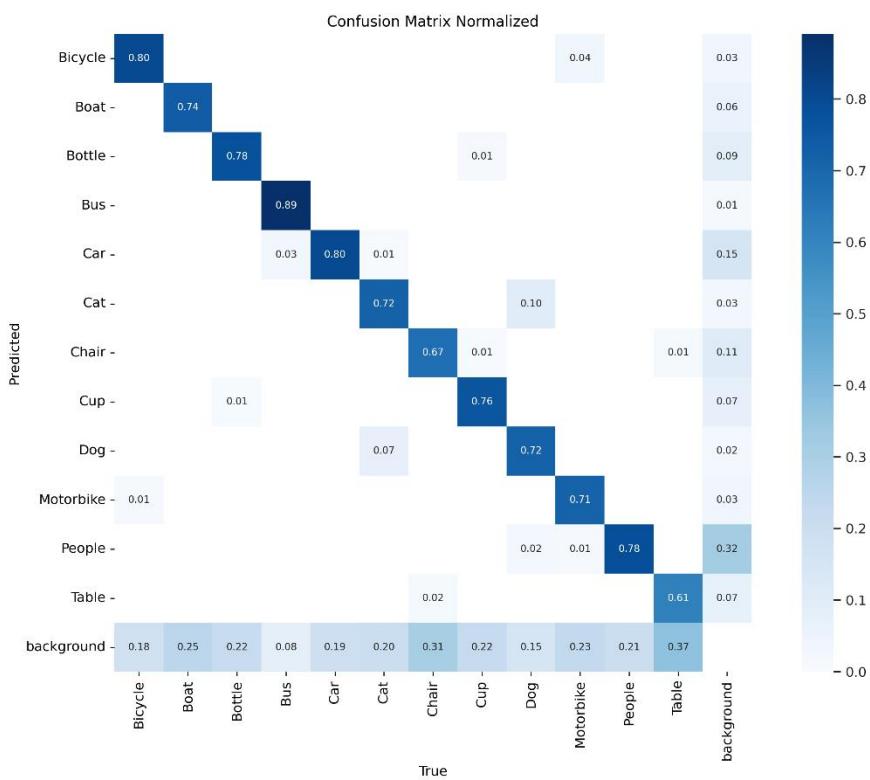
Gambar 4.29 Kurva hasil pelatihan model YOLOv10b

Kurva-kurva dari Gambar 4.29 di atas menunjukkan pola konvergensi yang baik. Pada grafik *loss training* (*train/box_loss*, *train/cls_loss*, *train/dfl_loss*), terlihat penurunan nilai loss yang signifikan pada awal training dan kemudian stabil pada nilai yang lebih rendah, menandakan proses pembelajaran yang efektif. Perbandingan antara kurva hasil aktual dan kurva *smoothing* menunjukkan stabilitas proses training tanpa fluktuasi yang berlebihan.

Pada metrik validasi, *val/box_loss* dan *val/cls_loss* menunjukkan pola penurunan yang serupa dengan metrik *training*, mengindikasikan bahwa model tidak mengalami *overfitting* yang signifikan. Namun, terdapat sedikit peningkatan pada *val/dfl_loss* setelah epoch ke-100, yang menunjukkan potensi keterbatasan model dalam memprediksi distribusi lokasi objek pada dataset validasi. Metrik performa model menunjukkan hasil yang menjanjikan, dengan nilai *precision* mencapai sekitar 0,8 dan *recall* sekitar 0,7 yang mengindikasikan keseimbangan

yang baik antara *true positive* dan *false positive* dalam deteksi objek pada kondisi pencahayaan rendah.

Metrik mAP50(B) yang mencapai nilai sekitar 0,75 menunjukkan performa deteksi yang baik pada threshold IoU 0,5 , sedangkan mAP50-95(B) yang mencapai sekitar 0,5 menggambarkan kemampuan model dalam mendekripsi objek dengan presisi lokasi yang bervariasi.

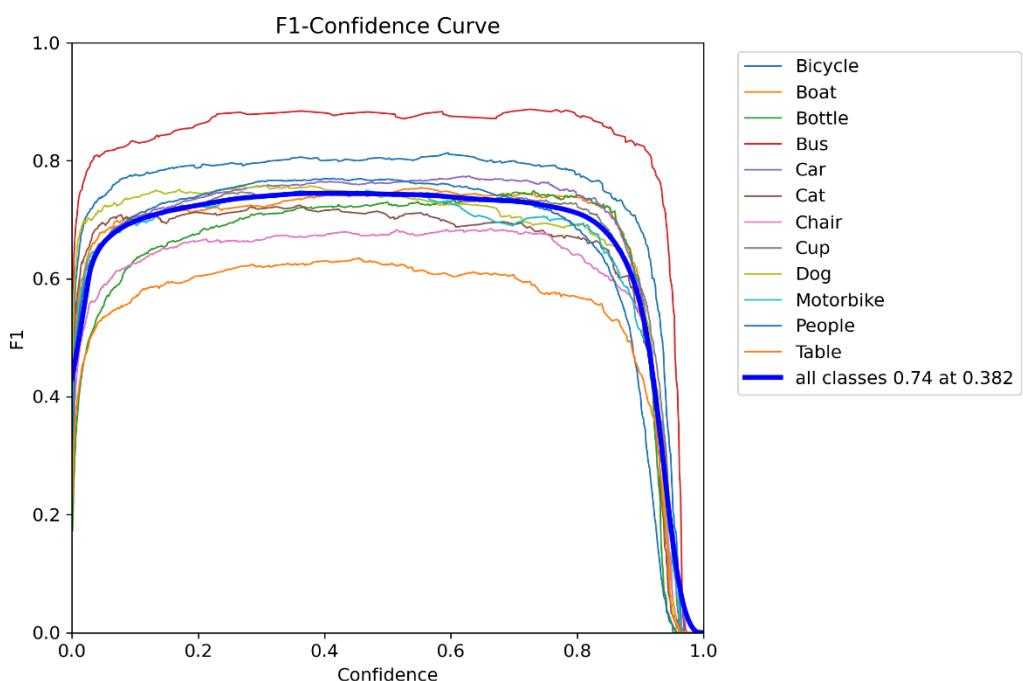


Gambar 4.30 Confusion Matrix Normalized untuk YOLOv10b

Gambar 4.30 menunjukkan *confusion matrix* yang sudah dinormalisasi untuk YOLOv10b, yang membuktikan dominasi nilai diagonal tinggi pada hampir semua kelas, misalnya *Bus* dan *Car* yang berada di atas 0,8. Hal ini mencerminkan keunggulan arsitektur YOLOv10b yang memiliki parameter terbesar di kalangan YOLOv10, dalam mengekstraksi fitur lebih detail. Meski demikian, masih terlihat kekeliruan pada beberapa

kelas yang bentuknya mirip, seperti *Bicycle* dan *Motorbike*, menandakan tantangannya masih saja sama seperti model-model sebelumnya.

Secara keseluruhan, hasil ini mengindikasikan bahwa YOLOv10b dapat memberikan performa deteksi yang baik meskipun berada di lingkungan pencahayaan rendah. Hal ini dapat dilihat dari kemampuan model dalam membedakan objek yang memiliki kontras lebih tegas serta menekan kesalahan prediksi antara objek dan latar belakang.



Gambar 4.31 Kurva F1-Confidence untuk YOLOv10b

Gambar 4.31 di atas menunjukkan kurva *F1-Confidence* untuk YOLOv10b yang menampilkan peningkatan keseimbangan *precision* dan *recall* yang lebih konsisten di rentang *confidence* menengah dibandingkan model serupa dengan parameter lebih kecil. Mayoritas kurva kelas mencapai puncak F1 pada *confidence* sekitar 0,3-0,4 , mencerminkan bahwa model telah mempelajari ciri-ciri objek secara mendalam sehingga mampu mengurangi *false positives* sekaligus mempertahankan *recall*. Kelas-kelas dengan kontur jelas seperti *Car* atau *Bus*

cenderung memperlihatkan puncak F1 yang lebih stabil dan lebih tinggi dibanding kelas yang cirinya saling menyerupai seperti *Bicycle* vs *Motorbike*.

Meskipun demikian, beberapa kelas menurun cukup drastis ketika *confidence* lebih tinggi dari 0,7 , menandakan bahwa model memang semakin selektif dalam menyatakan objek tetapi dengan risiko melewatkantarget yang lebih samar (menurunkan *recall*). Secara keseluruhan, hasil tersebut menggarisbawahi bahwa kapasitas besar YOLOv10b mampu mempelajari beragam pola visual pada kondisi pencahayaan rendah, namun pemilihan *confidence threshold* tetap menjadi faktor penting guna memperoleh kinerja deteksi terbaik untuk tiap kelas.

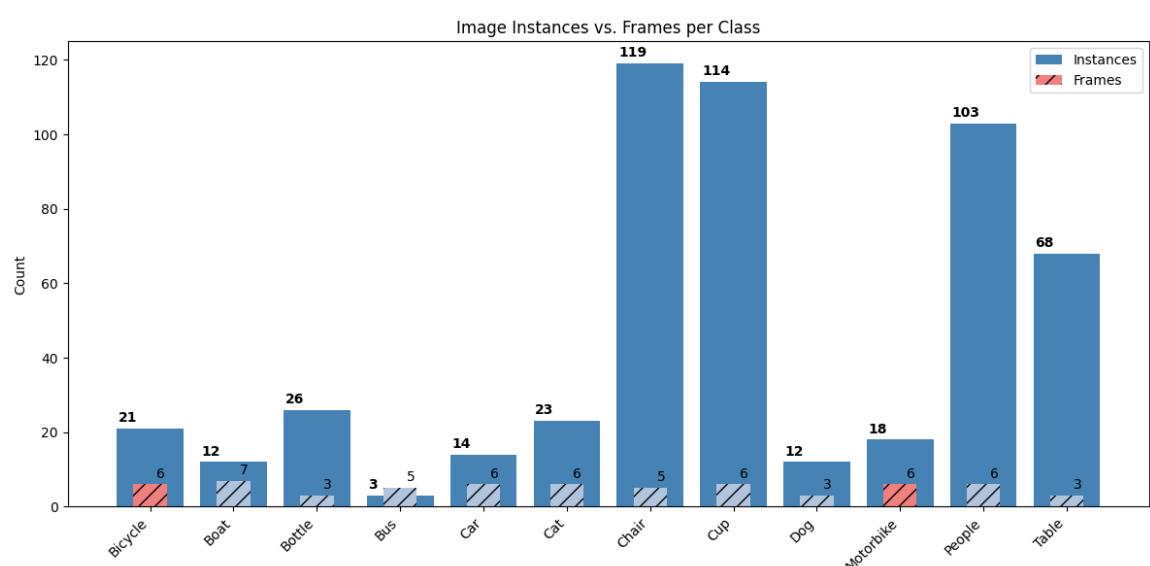
Tabel 4.14 Label vs Prediksi untuk YOLOv10b

No.	Ground Truth (a)	Predicted (b)
1.	<p>2015_06218.png</p>	<p>2015_06218.png</p>
2.	<p>2015_07138.jpg</p>	<p>2015_07138.jpg</p>
3.	<p>2015_01461.jpg</p>	<p>2015_01461.jpg</p>

Tabel 4.14 di atas menunjukkan perbandingan dari sampel *ground truth* dan *predicted* untuk model YOLOv10b yang tampak tidak terlalu berubah secara signifikan dibandingkan model YOLOv10m sebelumnya. Dari sampel di atas, terlihat bahwa yang dapat langsung dikenali adalah nilai *confidence* yang lebih tinggi dan skala serta posisi *bounding box* yang semakin akurat. Walau demikian model masih saja kesulitan untuk mengenali objek tertentu yang terlihat mirip ataupun dekat seperti objek *Chair* pada gambar 2b.

4.6 Testing Model

Dataset uji yang digunakan pada penelitian ini adalah hasil rekaman CCTV asli yang didapatkan dari internet. Total rekaman video yang diambil adalah sejumlah 12 video, dengan satu di antaranya merupakan hasil penggabungan dua kelas objek, yaitu *Bicycle* dan *Motorbike*, dalam satu video yang sama. Dari keseluruhan video tersebut, sebanyak 56 citra telah diambil dan dianotasi secara manual oleh peneliti untuk keperluan evaluasi model. Distribusi jumlah instance dan frame untuk masing-masing kelas objek dapat dilihat pada Gambar 4.32.



Gambar 4.32 Distribusi instances dan frames untuk setiap kelas

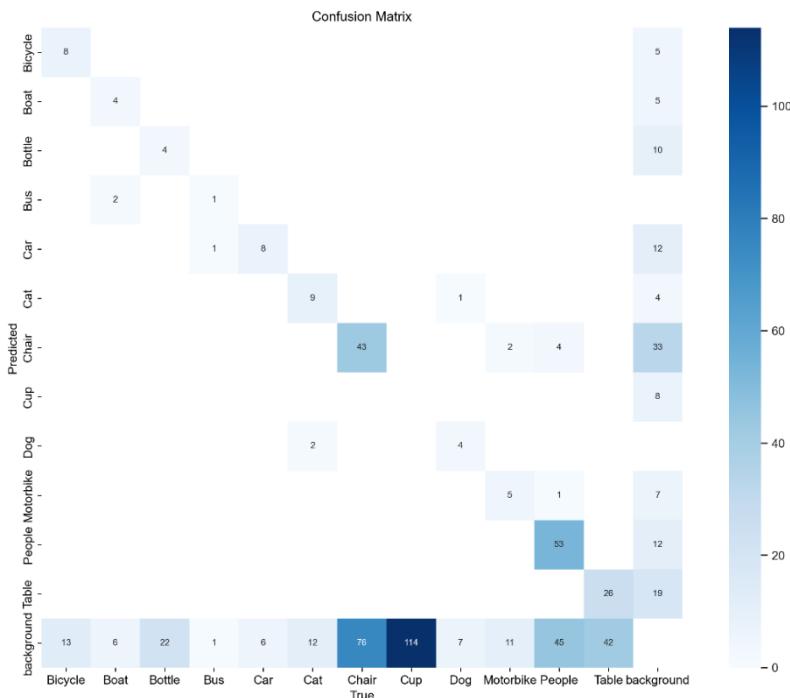
4.6.1 YOLOv9s

4.6.1.1 Well-lit Detection

Tabel 4.15 Label vs Prediksi untuk YOLOv9s pada *well-lit test*

<i>Ground Truth</i>	<i>Predicted</i>
	
	
	
	
	
	

Tabel 4.15 memperlihatkan contoh hasil deteksi objek oleh YOLOv9s pada enam *frame* rekaman CCTV dengan pencahayaan cukup. Kolom kiri menampilkan label asli (*ground truth*), sedangkan kolom kanan menunjukkan prediksi model. Secara umum, YOLOv9s berhasil mengenali objek-objek berskala cukup besar, seperti manusia (*People*), dan bahkan mendeteksi objek yang lebih kecil, misalnya sepeda (*Bicycle*). Meski demikian, masih ditemukan beberapa kesalahan, seperti munculnya prediksi kursi (*Chair*) di area yang sebenarnya kosong, serta salah klasifikasi sepeda menjadi sepeda motor (*Motorbike*). Selain itu, model kesulitan mendeteksi objek yang sangat kecil atau berada pada jarak jauh, kasus ini terutama terjadi pada deteksi manusia pada *frame* tertentu.



Gambar 4.33 YOLOv9s well-lit test confusion matrix

Gambar 4.33 menunjukkan *confusion matrix* yang tidak dinormalisasi untuk hasil deteksi YOLOv9s pada kondisi pencahayaan cukup. Terlihat bahwa cukup banyak objek terkласifikasi sebagai latar belakang (*background*), terutama untuk kelas berukuran kecil seperti cangkir (*Cup*), yang sama sekali tidak terdeteksi. Nilai diagonal pada *confusion matrix* relatif rendah, meski model menunjukkan performa terbaik pada kelas *People*, *Chair*, dan *Table*. Kesalahan

klasifikasi lain yang masih muncul antara lain prediksi Boat yang keliru sebagai Bus, serta beberapa objek *People* dikira sebagai *Chair*.

Tabel 4.16 Hasil evaluasi YOLOv9s pada *well-lit test*

Model	mAP50	mAP50-95	Inference Time (ms)	Total Time (s)
YOLOv9s	0,3972	0,1752	256,12	20,05

Tabel 4.16 di atas, adalah tabel metrik hasil dari *testing* model YOLOv9s pada citra berpencahayaan cukup. Rendahnya akurasi yang tergambar pada *confusion matrix* tercermin pada nilai mAP50 sebesar 0,3972. Di sisi lain, model mampu melakukan inferensi sangat cepat dengan rata-rata 256,12 ms per gambar, sehingga proses deteksi pada 56 citra selesai dalam waktu total 20,05 detik.

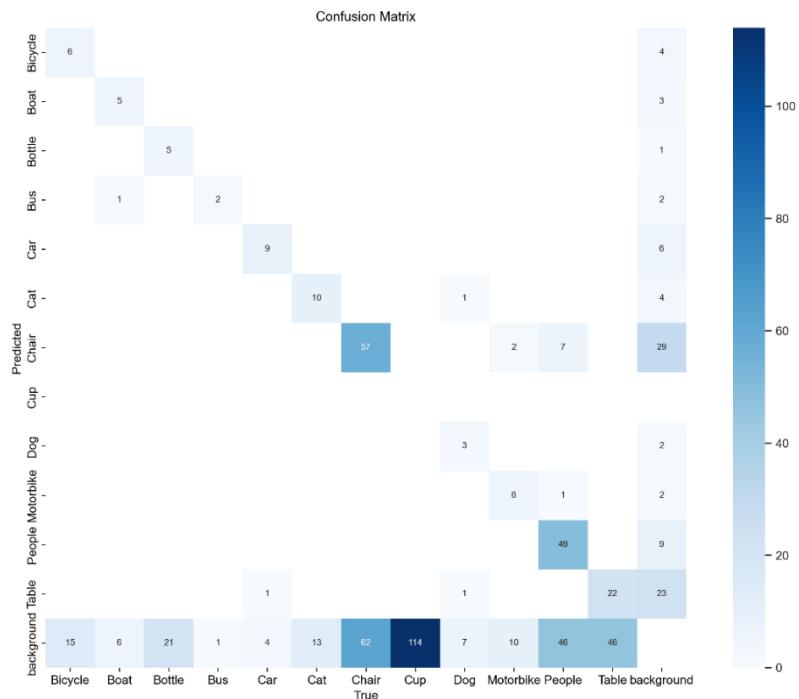
4.6.1.2 Low-light Detection

Tabel 4.17 Label vs Prediksi untuk YOLOv9s pada *low-light test*

<i>Ground Truth</i>	<i>Predicted</i>
30.jpg 	30.jpg
36.jpg 	36.jpg
40.jpg 	40.jpg
43.jpg 	43.jpg
44.jpg 	44.jpg
45.jpg 	45.jpg

Tabel 4.17 memperlihatkan contoh hasil deteksi objek oleh YOLOv9s pada enam *frame* rekaman CCTV dengan pencahayaan rendah. Kolom kiri menampilkan label asli (*ground truth*), sedangkan kolom kanan menunjukkan prediksi model. Pada citra berpencahayaan rendah, secara umum YOLOv9s berhasil mengenali objek-objek berskala cukup besar, seperti manusia (*People*), dan bahkan

mendeteksi objek yang lebih kecil, misalnya sepeda (*Bicycle*). Tetapi masih saja ditemukan beberapa kesalahan, seperti munculnya prediksi kursi (*Chair*) di area yang sebenarnya kosong. Uniknya, model mampu mengenali objek kecil atau jauh (seperti manusia) dengan lebih baik daripada citra berpencahayaan cukup.



Gambar 4.34 YOLOv9s *test confusion matrix*

Gambar 4.34 menunjukkan *confusion matrix* yang tidak dinormalisasi untuk hasil deteksi YOLOv9s pada kondisi pencahayaan rendah. Terlihat bahwa masih banyak objek terklasifikasi sebagai latar belakang (*background*), terutama untuk kelas berukuran kecil seperti cangkir (*Cup*), yang sama sekali tidak terdeteksi. Nilai diagonal pada *confusion matrix* relatif masih rendah, meski model juga menunjukkan performa lebih baik pada kelas *People*, *Chair*, dan *Table*. Kesalahan klasifikasi lain yang masih muncul antara lain prediksi *Boat* yang keliru sebagai *Bus*, berbeda dari kasus sebelumnya yang hanya 4 dari 6 *Boat* benar dideteksi, kali ini 5 dari 6 kasus adalah deteksi yang benar. Secara keseluruhan citra yang kurang pencahayaan justru membantu model untuk mengidentifikasi objek dengan lebih baik.

Tabel 4.18 Hasil evaluasi YOLOv9s pada *low-light test*

Model	mAP50	mAP50-95	Inference Time (ms)	Total Time (s)
YOLOv9s	0,4439	0,1827	236,60	18,66

Tabel 4.18 di atas, adalah tabel metrik hasil dari *testing* model YOLOv9s pada citra berpencahayaan rendah. Akurasi yang tergambar pada *confusion matrix* juga tercermin pada nilai mAP50 sebesar 0,4439, yang bernilai lebih besar daripada citra berpencahayaan cukup . Di sisi lain, model mampu melakukan inferensi sangat cepat dengan rata-rata 236,60 ms per gambar yang sedikit lebih cepat daripada kasus sebelumnya. Hal ini menghasilkan proses deteksi pada 56 citra selesai dalam waktu total 18,66 detik, yaitu 1,39 detik lebih cepat daripada kasus sebelumnya.

4.6.2 YOLOv9m

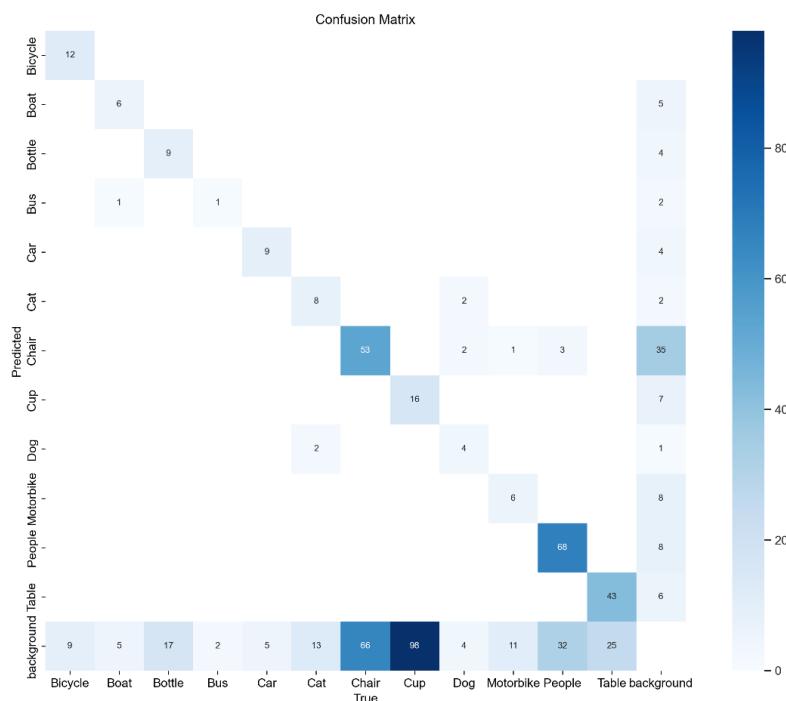
4.6.2.1 Well-lit Detection

Tabel 4.19 Label vs Prediksi untuk YOLOv9m pada *well-lit test*

<i>Ground Truth</i>	<i>Predicted</i>
30.jpg 	30.jpg
36.jpg 	36.jpg
40.jpg 	40.jpg
43.jpg 	43.jpg
44.jpg 	44.jpg
45.jpg 	45.jpg

Tabel 4.19 memperlihatkan contoh hasil deteksi objek oleh YOLOv9m yaitu model dengan parameter lebih besar daripada YOLOv9s, pada enam *frame* rekaman CCTV dengan pencahayaan cukup. Kolom kiri menampilkan label asli (*ground truth*), sedangkan kolom kanan menunjukkan prediksi model. Secara

umum, YOLOv9m berhasil mengenali objek-objek berskala cukup besar, seperti manusia (*People*), dan bahkan mendeteksi objek yang lebih kecil, misalnya sepeda (*Bicycle*) lebih baik daripada YOLOv9s. Terlebih lagi, model ini tidak mengalami kesalahan deteksi untuk kursi (*Chair*) dan motor (*Motorbike*) seperti pada kasus sebelumnya. Berbeda dengan YOLOv9s, model YOLOv9m juga mulai dapat mendeteksi objek yang sangat kecil atau berada pada jarak jauh seperti pada objek manusia.



Gambar 4.35 YOLOv9m *well-lit test confusion matrix*

Gambar 4.35 menunjukkan *confusion matrix* yang tidak dinormalisasi untuk hasil deteksi YOLOv9m pada kondisi pencahayaan cukup. Terlihat bahwa masih banyak objek terklasifikasi sebagai latar belakang (*background*), terutama untuk kelas berukuran kecil seperti cangkir (*Cup*) yang walaupun jauh lebih baik ketimbang kasus pada YOLOv9s. Nilai diagonal pada *confusion matrix* relatif masih rendah walau sudah sedikit meningkat daripada YOLOv9s. Seperti kasus sebelumnya, model juga menunjukkan performa lebih baik pada kelas *People*,

Chair, dan *Table*. Secara keseluruhan, model ini mampu mengungguli YOLOv9s walaupun masih dalam citra yang cukup pencahayaannya.

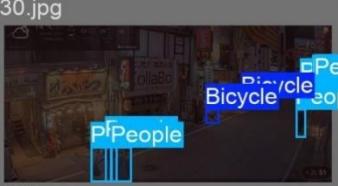
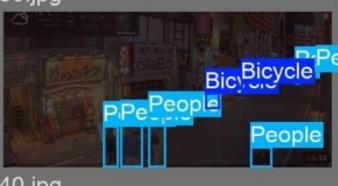
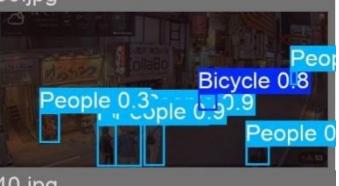
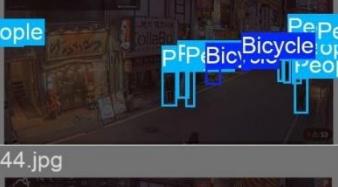
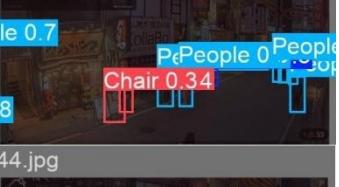
Tabel 4.20 Hasil evaluasi YOLOv9m pada *well-lit test*

Model	mAP50	mAP50-95	Inference Time (ms)	Total Time (s)
YOLOv9m	0,5739	0,3059	606,62	40,06

Tabel 4.20 di atas, adalah tabel metrik hasil dari *testing* model YOLOv9m pada citra berpencahayaan cukup. Akurasi yang tergambar pada *confusion matrix* juga tercermin pada nilai mAP50 sebesar 0,5739, yang bernilai jauh lebih besar daripada model YOLOv9s. Di sisi lain, model mampu melakukan inferensi yang dapat dibilang cukup cepat dengan rata-rata 606,62 ms, atau 350,5 ms lebih lama dari YOLOv9s. Hal ini menghasilkan proses deteksi pada 56 citra selesai dalam waktu total 40,06 detik, yaitu 20,01 detik lebih lama daripada YOLOv9s.

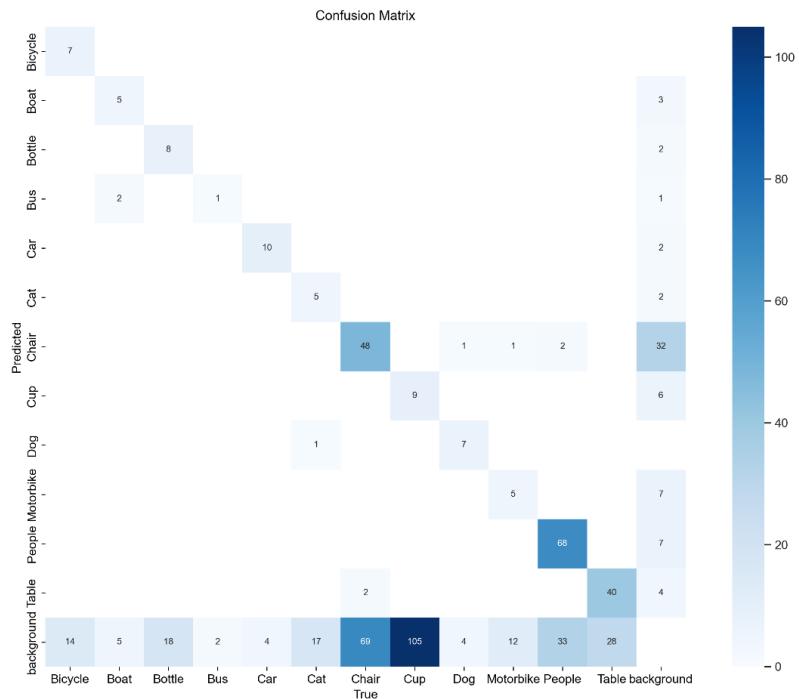
4.6.2.2 Low-light Detection

Tabel 4.21 Label vs Prediksi untuk YOLOv9m pada *low-light test*

<i>Ground Truth</i>	<i>Predicted</i>
30.jpg 	30.jpg 
36.jpg 	36.jpg 
40.jpg 	40.jpg 
43.jpg 	43.jpg 
44.jpg 	44.jpg 
45.jpg 	45.jpg 

Tabel 4.21 memperlihatkan contoh hasil deteksi objek oleh YOLOv9m, pada enam *frame* rekaman CCTV dengan pencahayaan rendah. Kolom kiri menampilkan label asli (*ground truth*), sedangkan kolom kanan menunjukkan prediksi model. Secara umum, YOLOv9m berhasil mengenali objek-objek berskala cukup besar dan kecil seperti pada manusia (*People*) yang jauh maupun dekat.

Namun, menariknya adalah model YOLOv9m yang mendeteksi citra berpencahayaan rendah ini justru seringkali salah mendeteksi *Chair* yang harusnya tidak ada, berbeda dengan YOLOv9s sebelumnya.



Gambar 4.36 YOLOv9m *low-light test confusion matrix*

Gambar 4.36 menunjukkan *confusion matrix* yang tidak dinormalisasi untuk hasil deteksi YOLOv9m pada kondisi pencahayaan rendah. Terlihat bahwa masih banyak objek terklasifikasi sebagai latar belakang (*background*), terutama untuk kelas berukuran kecil seperti cangkir (*Cup*) yang kali ini lebih rendah daripada YOLOv9m untuk citra yang berpencahayaan cukup. Nilai diagonal pada *confusion matrix* relatif masih rendah walau sudah sedikit meningkat daripada YOLOv9s. Secara keseluruhan, model ini mampu mengungguli YOLOv9s walau masih dalam citra yang cukup pencahayaannya, tetapi masih belum mampu mengungguli YOLOv9m pada citra bercahaya cukup.

Tabel 4. 22 Hasil evaluasi YOLOv9m pada *low-light test*

Model	mAP50	mAP50-95	Inference Time (ms)	Total Time (s)
YOLOv9m	0,5707	0,3132	588,45	38,35

Tabel 4.22 di atas, adalah tabel metrik hasil dari *testing* model YOLOv9m pada citra berpencahayaan rendah. Sesuai dengan *confusion matrix* sebelumnya, nilai metrik mAP50 untuk YOLOv9m di cahaya rendah 0,0032 poin lebih rendah daripada untuk citra berpencahayaan cukup. Ini berbanding terbalik dengan mAP50-95 yang nilainya sedikit lebih tinggi daripada untuk citra berpencahayaan cukup. Dari segi waktu, citra berpencahayaan rendah mempercepat model untuk melakukan inferensi, yaitu 588,45 ms atau 18,17 ms lebih cepat daripada kasus sebelumnya. Menghasilkan total waktu untuk 56 citra *test* selama 38,35 detik.

4.6.3 YOLOv9c

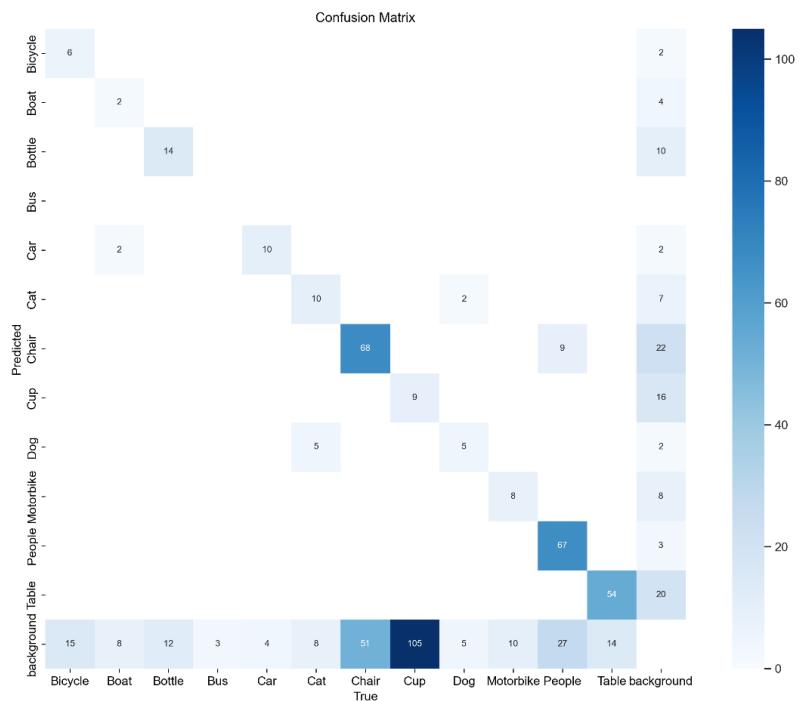
4.6.3.1 Well-lit Detection

Tabel 4. 23 Label vs Prediksi untuk YOLOv9c pada *well-lit test*

<i>Ground Truth</i>	<i>Predicted</i>
30.jpg 	30.jpg
36.jpg 	36.jpg
40.jpg 	40.jpg
43.jpg 	43.jpg
44.jpg 	44.jpg
45.jpg 	45.jpg

Tabel 4.23 memperlihatkan contoh hasil deteksi objek oleh YOLOv9c yaitu model yang memiliki parameter terbesar untuk keluarga YOLOv9 di penelitian ini, pada enam *frame* rekaman CCTV dengan pencahayaan yang cukup. Secara umum, YOLOv9c berhasil mengenali objek-objek berskala cukup besar dan kecil seperti

pada manusia (*People*) yang jauh maupun dekat secara lebih akurat dibanding YOLOv9s. Tetapi jika hanya dilihat dari hasil enam frame di atas, kemampuannya masih belum dapat menyaingi YOLOv9m dikarenakan terdapat kesalahan deteksi pada salah satu frame yaitu objek botol (*Bottle*) yang seharusnya tidak ada.



Gambar 4.37 YOLOv9c *well-lit test confusion matrix*

Gambar 4.37 menunjukkan *confusion matrix* yang tidak dinormalisasi untuk hasil deteksi YOLOv9c pada kondisi pencahayaan cukup. Terlihat bahwa masih saja banyak objek terkласifikasi sebagai latar belakang (*background*). Kesalahan-deteksi terhadap objek yang sangat kecil seperti cangkir (*Cup*) dan objek yang sangat besar seperti (*Boat*) juga masih lebih banyak dibandingkan dengan YOLOv9m. Nilai diagonal pada *confusion matrix* relatif masih rendah walau sudah sedikit meningkat daripada YOLOv9s tetapi belum dapat menandingi YOLOv9m.

Tabel 4.24 Hasil evaluasi YOLOv9c pada *well-lit test*

Model	mAP50	mAP50-95	Inference Time (ms)	Total Time (s)
YOLOv9c	0,5324	0,2672	860,54	55,14

Tabel 4.24 di atas, adalah tabel metrik hasil dari *testing* model YOLOv9c pada citra berpencahayaan cukup. Sesuai dengan *confusion matrix* sebelumnya, nilai metrik mAP50 untuk YOLOv9c di cahaya cukup 0,0383 poin lebih rendah daripada model YOLOv9m, dan begitu juga untuk metrik mAP50-95 yang turun 0,046. Dari segi waktu, model ini memerlukan waktu yang cukup lama dibanding dengan YOLOv9m yaitu 860,54 ms menghasilkan total waktu selama 55,14 detik atau 16,79 detik lebih lama daripada YOLOv9m.

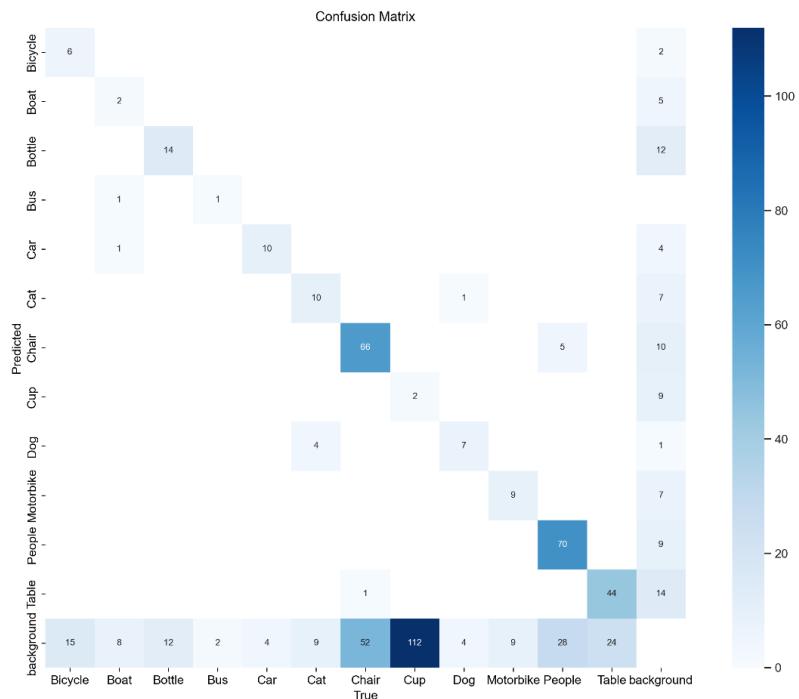
4.6.3.2 Low-light Detection

Tabel 4.25 Label vs Prediksi untuk YOLOv9c pada *low-light test*

<i>Ground Truth</i>	<i>Predicted</i>
30.jpg 	30.jpg
36.jpg 	36.jpg
40.jpg 	40.jpg
43.jpg 	43.jpg
44.jpg 	44.jpg
45.jpg 	45.jpg

Tabel 4.25 memperlihatkan contoh hasil deteksi objek oleh YOLOv9c yaitu model yang memiliki parameter terbesar untuk keluarga YOLOv9 di penelitian ini, pada enam *frame* rekaman CCTV dengan pencahayaan yang rendah. Secara umum, YOLOv9c berhasil mengenali objek-objek berskala cukup besar dan kecil seperti

pada manusia (*People*) yang jauh maupun dekat secara lebih akurat dibanding YOLOv9s dan mungkin juga YOLOv9m. Kesalahan deteksi pada YOLOv9c untuk citra berpencahayaan cukup yaitu deteksi botol (*Bottle*) juga tidak ditunjukkan pada kasus citra pencahayaan rendah ini.



Gambar 4.38 YOLOv9c low-light test confusion matrix

Gambar 4.38 menunjukkan *confusion matrix* yang tidak dinormalisasi untuk hasil deteksi YOLOv9c pada kondisi pencahayaan rendah. Terlihat bahwa masih saja banyak objek terkласifikasi sebagai latar belakang (*background*). Kesalahan-deteksi terhadap objek yang sangat kecil seperti cangkir (*Cup*) dan objek yang sangat besar seperti (*Boat*) juga masih lebih banyak dibandingkan dengan YOLOv9m. Nilai diagonal pada *confusion matrix* relatif masih rendah walau sudah sedikit meningkat daripada YOLOv9c pada kasus pencahayaan cukup.

Tabel 4.26 Hasil evaluasi YOLOv9c pada *low-light test*

Model	mAP50	mAP50-95	Inference Time (ms)	Total Time (s)
YOLOv9c	0,5704	0,2755	829,73	53,53

Tabel 4.26 di atas, adalah tabel metrik hasil dari *testing* model YOLOv9c pada citra berpencahayaan rendah. Metrik mAP50 untuk YOLOv9c di cahaya rendah bernilai 0,0003 poin lebih buruk daripada YOLOv9m di kasus yang sama. Begitu juga untuk metrik mAP50-95 yang turun 0,0377 dibanding YOLOv9m. Dari segi waktu, model berparameter besar ini memerlukan waktu yang cukup lama dibanding dengan YOLOv9m tetapi masih lebih cepat daripada kasus citra berpencahayaan cukup, yaitu di angka 829,73 ms yang menghasilkan total waktu seluruh deteksi 56 citra sebesar 53,53 detik.

4.6.4 RTDETR-L

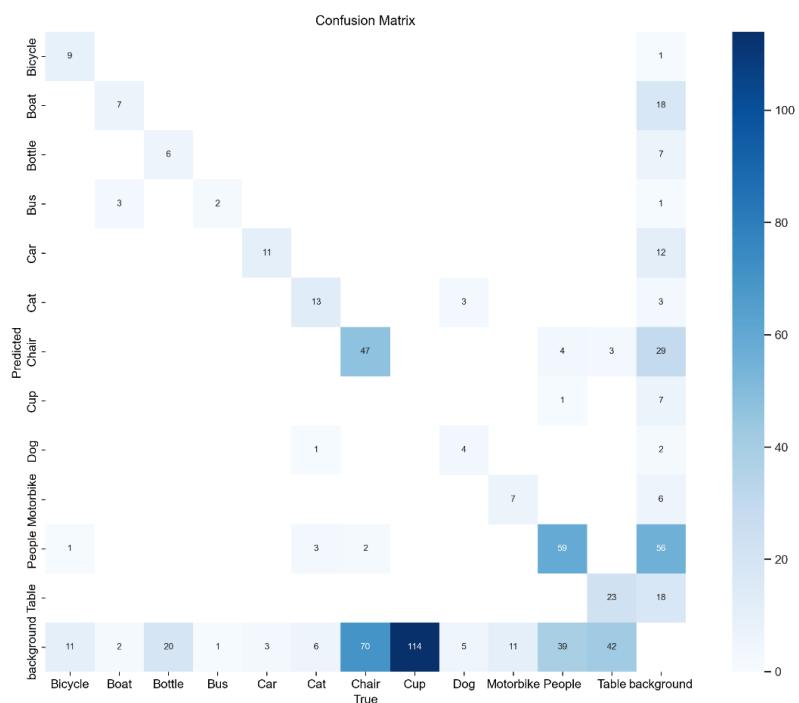
4.6.4.1 Well-lit Detection

Tabel 4.27 Label vs Prediksi untuk RTDETR-L pada *well-lit test*

<i>Ground Truth</i>	<i>Predicted</i>
30.jpg 	
36.jpg 	
40.jpg 	
43.jpg 	
44.jpg 	
45.jpg 	

Tabel 4.27 memperlihatkan contoh hasil deteksi objek oleh RTDETR-L yaitu model yang memiliki pendekatan Transformer dan berparameter lebih besar

daripada YOLOv9, pada enam *frame* rekaman CCTV dengan pencahayaan yang cukup. Secara umum, RTDETR-L berhasil mengenali objek-objek berskala cukup besar dan kecil seperti pada manusia (*People*) yang jauh maupun dekat secara lebih banyak namun belum begitu akurat jika dibanding YOLOv9. Terlihat dengan sekilas bahwa RTDETR-L masih seringkali salah menghitung jumlah manusia yang ada pada gambar, serta salah mengklasifikasi kursi (*Chair*) yang seharusnya tidak ada.



Gambar 4.39 RTDETR-L well-lit test confusion matrix

Gambar 4.39 menunjukkan *confusion matrix* yang tidak dinormalisasi untuk hasil deteksi RTDETR-L pada kondisi pencahayaan cukup. Terlihat bahwa masih saja banyak objek yang salah deteksi dan terkласifikasi sebagai latar belakang (*background*). Walau dengan parameter yang lebih besar daripada YOLOv9, RTDETR-L tidak menunjukkan performa yang baik jika dilihat dari nilai diagonalnya. Terlebih lagi model ini tidak dapat sekalipun mendeteksi objek kecil seperti cangkir (*Cup*).

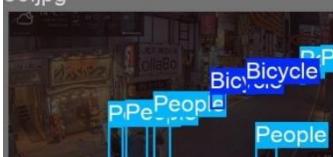
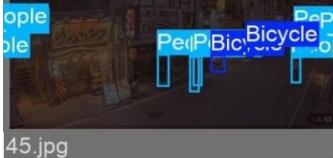
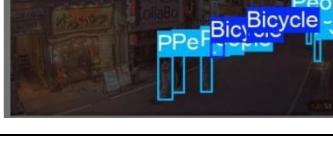
Tabel 4.28 Hasil evaluasi RTDETR-L pada *well-lit test*

Model	mAP50	mAP50-95	Inference Time (ms)	Total Time (s)
RTDETR-L	0,3632	0,1486	1380,11	93,36

Tabel 4.28 di atas, adalah tabel metrik hasil dari *testing* model RTDETR-L pada citra berpencahayaan cukup. Nilai-nilai metrik ini tercermin pula dari *confusion matrix* sebelumnya. Terlebih lagi RTDETR-L memiliki mAP50 dan mAP50-95 sebesar 0,03 poin lebih buruk daripada YOLOv9s di kasus yang sama yaitu citra berpencahayaan cukup. Dari segi waktu, karena model ini memiliki parameter yang dapat dibilang besar, waktu inferensi yang dibutuhkan 5 kali lipat lebih lama dibanding YOLOv9s, yaitu 1380,11 detik (1123,99 ms lebih lama) yang menghasilkan total waktu deteksi 56 citra selama 93,36 detik.

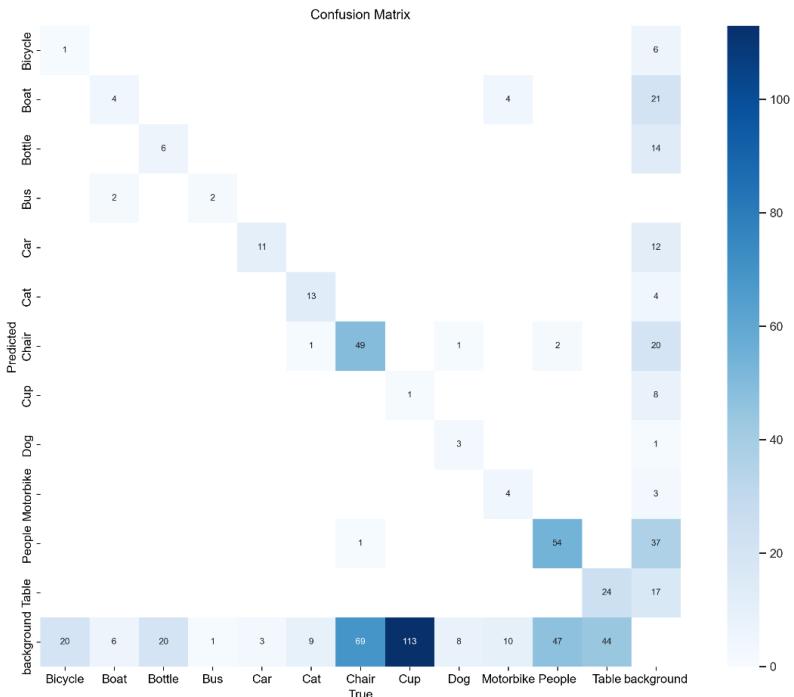
4.6.4.2 Low-light Detection

Tabel 4.29 Label vs Prediksi untuk RTDETR-L pada *low-light test*

<i>Ground Truth</i>	<i>Predicted</i>
30.jpg	
36.jpg	
40.jpg	
43.jpg	
44.jpg	
45.jpg	

Tabel 4.29 memperlihatkan contoh hasil deteksi objek oleh RTDETR-L yaitu model yang memiliki pendekatan Transformer dan berparameter lebih besar daripada YOLOv9, pada enam *frame* rekaman CCTV dengan pencahayaan yang rendah. Secara umum, RTDETR-L di kasus citra berpencahayaan rendah ini memiliki performa lebih buruk dibandingkan kasus sebelumnya. Model ini

mengalami lebih banyak salah deteksi manusia (*People*) dan lagi-lagi kursi (*Chair*) yang sebenarnya tidak ada.



Gambar 4.40 RTDETR-L *low-light test confusion matrix*

Gambar 4.40 menunjukkan *confusion matrix* yang tidak dinormalisasi untuk hasil deteksi RTDETR-L pada kondisi pencahayaan rendah. Terlihat bahwa masih saja banyak objek yang salah deteksi dan terkласifikasi sebagai latar belakang (*background*). Walau kasus cahaya rendah membantu RTDETR-L untuk membantu mendeteksi cangkir (*Cup*) yang sebelumnya sama sekali tidak terdeteksi, nilai diagonal pada model ini dapat terbilang lebih buruk daripada RTDETR-L dalam kasus cahaya cukup.

Tabel 4. 30 Hasil evaluasi RTDETR-L pada *low-light test*

Model	mAP50	mAP50-95	Inference Time (ms)	Total Time (s)
RTDETR-L	0,3513	0,1443	1289,16	88,29

Tabel 4.30 di atas, adalah tabel metrik hasil dari *testing* model RTDETR-L pada citra berpencahayaan cukup. Nilai-nilai metrik ini tercermin pula dari *confusion matrix* yang buruk sebelumnya. Metrik mAP50 dan mAP50-95 untuk kasus ini memiliki nilai

lebih buruk daripada kasus citra pencahayaan cukup sebelumnya. Hal terbaik yang dapat diambil dari tabel hanyalah inferensi waktunya yang lebih cepat dibanding kasus cahaya cukup, yaitu 1289,16 ms atau 90,95 ms lebih cepat. Walau lagi-lagi masih sangat jauh jika dibandinkan dengan seluruh model pada YOLOv9.

4.6.5 RTDETR-X

4.6.5.1 Well-lit Detection

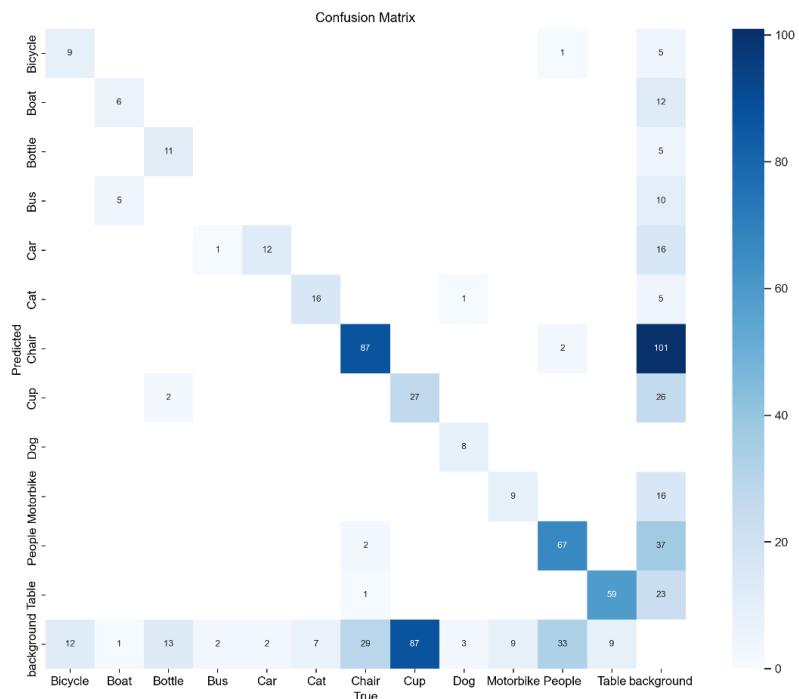
Tabel 4.31 Label vs Prediksi untuk RTDETR-X pada well-lit test

Tabel 4.31 Label vs Prediksi untuk RTDETR-X pada well-lit test

<i>Ground Truth</i>	<i>Predicted</i>
30.jpg 	
36.jpg 	
40.jpg 	
43.jpg 	
44.jpg 	
45.jpg 	

Tabel 4.31 memperlihatkan contoh hasil deteksi objek oleh RTDETR-X yaitu model yang memiliki pendekatan Transformer dan berparameter paling besar di penelitian ini, pada enam *frame* rekaman CCTV dengan pencahayaan yang

cukup. Secara umum, RTDETR-X berhasil mengenali objek-objek berskala cukup besar dan kecil seperti pada manusia (*People*) dan sepeda (*Bicycle*) yang jauh maupun dekat secara lebih akurat secara *bounding box* dibanding RTDETR-L. Namun, terlihat bahwa model ini masih saja mengalami kebingungan terhadap objek yang seharusnya tidak ada seperti mobil (*Car*) dan bis (*Bus*) walau dengan *confidence* yang cukup rendah.



Gambar 4.41 RTDETR-X *well-lit test confusion matrix*

Gambar 4.44 menunjukkan *confusion matrix* yang tidak dinormalisasi untuk hasil deteksi RTDETR-X pada kondisi pencahayaan cukup. Terlihat bahwa masih banyak objek yang salah deteksi dan terklasifikasi sebagai latar belakang (*background*). Perbedaan mayor terlihat pada nilai diagonal yang jauh lebih baik daripada RTDETR-L, dan juga klasifikasi cangkir (*Cup*) yang cukup drastis. Walau demikian, indikator-indikator ini masih saja belum cukup untuk mengungguli YOLOv9 secara keseluruhan.

Tabel 4. 32 Hasil evaluasi RTDETR-X pada *well-lit test*

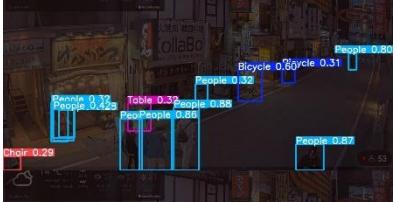
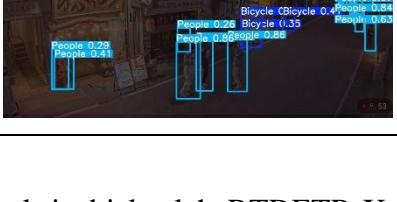
Model	mAP50	mAP50-95	Inference Time (ms)	Total Time (s)
RTDETR-X	67.8	59.5	25	1.25

RTDETR-X	0,4924	0,2531	2583,54	165,26
----------	--------	--------	---------	--------

Tabel 4.32 di atas, adalah tabel metrik hasil dari *testing* model RTDETR-X pada citra berpencahayaan cukup. Nilai-nilai metrik ini tercermin pula dari *confusion matrix* yang cukup baik sebelumnya. Metrik mAP50 dan mAP50-95 untuk kasus ini memiliki nilai lebih baik daripada RTDETR-L, yaitu 0,4924 (+ 0,1292) dan 0,2531 (+0,1045). Tetapi tentu saja efek samping dari model berparameter besar adalah mengenai *runtime* nya. RTDETR-X yang memiliki jumlah parameter lebih dari 60 juta ini membutuhkan 2583,54 ms untuk menginferensikan satu gambar atau kurang lebih 10 kali lipat daripada YOLOv9s. Performa seperti ini menghasilkan waktu total untuk mendekripsi 56 citra selama 165,26 detik.

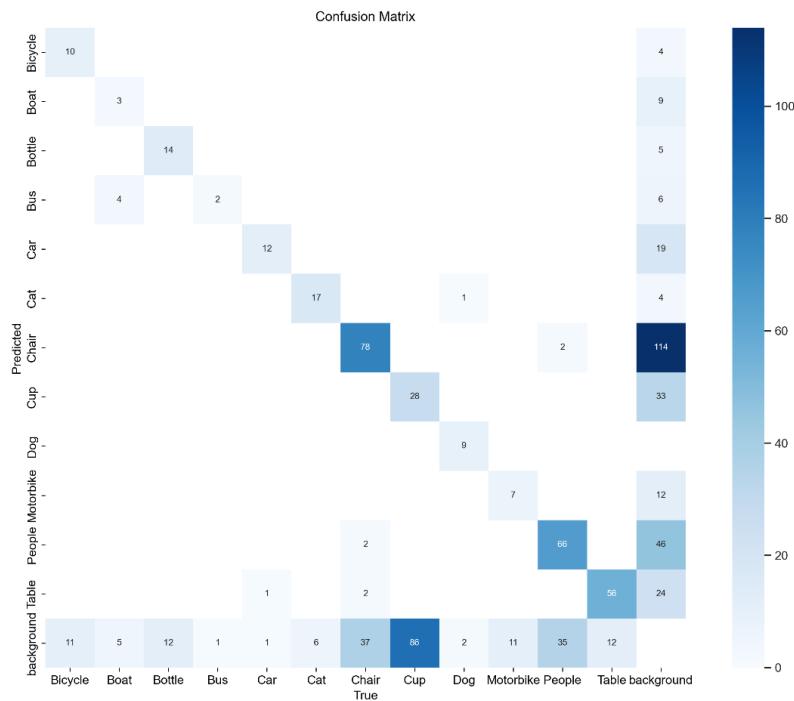
4.6.5.2 Low-light Detection

Tabel 4.33 Label vs Prediksi untuk RTDETR-X pada *low-light test*

<i>Ground Truth</i>	<i>Predicted</i>
30.jpg	
36.jpg	
40.jpg	
43.jpg	
44.jpg	
45.jpg	

Tabel 4.33 memperlihatkan contoh hasil deteksi objek oleh RTDETR-X yaitu model yang memiliki pendekatan Transformer dan berparameter paling besar di penelitian ini, pada enam *frame* rekaman CCTV dengan pencahayaan yang rendah. Secara umum, RTDETR-X berhasil mengenali objek-objek berskala cukup besar dan kecil seperti pada manusia (*People*) dan sepeda (*Bicycle*) yang jauh

maupun dekat secara sedikit lebih baik daripada kasus untuk cahaya cukup. Namun, terlihat bahwa model ini masih saja mengalami kebingungan terhadap objek yang seharusnya tidak ada seperti bis (*Bus*) dan objek kecil yang tiada seperti meja (*Table*) dan kursi (*Chair*) walau dengan *confidence* yang terbilang rendah.



Gambar 4.42 RTDETR-X *low-light test confusion matrix*

Gambar 4.42 menunjukkan *confusion matrix* yang tidak dinormalisasi untuk hasil deteksi RTDETR-X pada kondisi pencahayaan rendah. Terlihat bahwa masih saja banyak objek yang salah deteksi dan terklasifikasi sebagai latar belakang (*background*). Secara nilai diagonal, RTDETR-X pada kasus cahaya rendah ini menciptakan efek yang positif dikarenakan nilainya jauh lebih baik dibanding dengan kasus sebelumnya yaitu citra bercahaya cukup.

Tabel 4.34 Hasil evaluasi RTDETR-X pada *low-light test*

Model	mAP50	mAP50-95	Inference Time (ms)	Total Time (s)
RTDETR-X	0,5205	0,2796	2444,03	159,09

Tabel 4.34 di atas, adalah tabel metrik hasil dari *testing* model RTDETR-X pada citra berpencahayaan rendah. Nilai-nilai metrik ini tercermin pula dari *confusion matrix* yang lebih baik sebelumnya. Nilai-nilai metrik utama yaitu mAP50 dan mAP50-95 memiliki jumlah yang lebih besar dibanding dengan RTDETR-L yaitu 0,5205 (+0,0281) dan 0,2796 (+0,0265). Sama seperti kasus sebelumnya, waktu inferensi yang dibutuhkan juga terbilang sangat lama walaupun sedikit lebih cepat jika dibanding dengan RTDETR-L yaitu 2444,03 (-139,51 ms), yang menghasilkan total waktu untuk keseluruhan citra *test* selama 159,09 detik (- 6,17 detik).

4.6.6 YOLOv10s

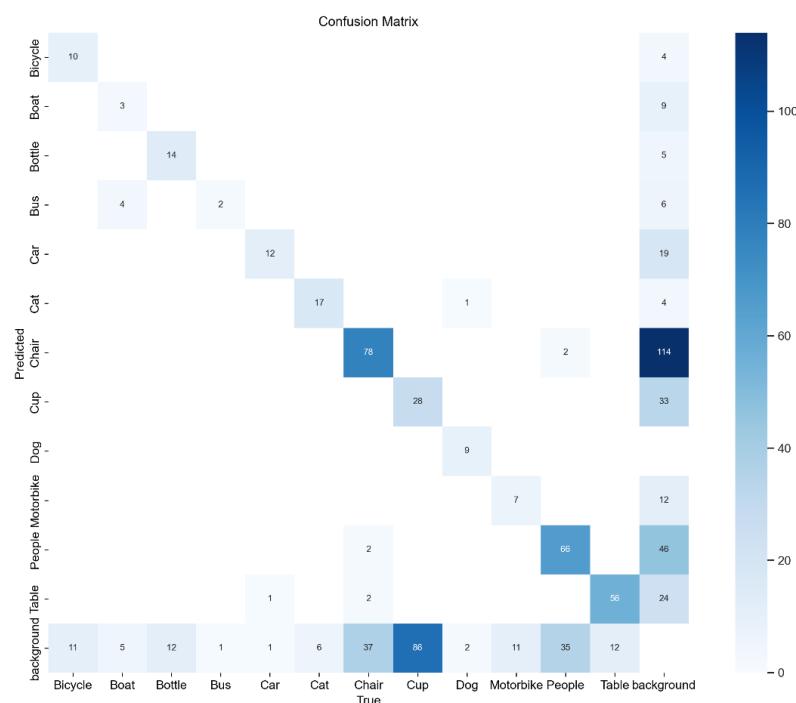
4.6.6.1 Well-lit Detection

Tabel 4.35 Label vs Prediksi untuk YOLOv10s pada *well-lit test*

<i>Ground Truth</i>	<i>Predicted</i>
30.jpg 	30.jpg 
36.jpg 	36.jpg 
40.jpg 	40.jpg 
43.jpg 	43.jpg 
44.jpg 	44.jpg 
45.jpg 	45.jpg 

Tabel 4.35 memperlihatkan contoh hasil deteksi objek oleh YOLOv10s yaitu model yang memiliki pendekatan hybrid dan berparameter paling kecil di keluarga YOLOv10, pada enam *frame* rekaman CCTV dengan pencahayaan yang cukup. Secara umum, YOLOv10s berhasil mengenali objek-objek berskala cukup

besar dan kecil seperti pada manusia (*People*) dan sepeda (*Bicycle*) yang jauh maupun dekat dengan cukup baik walau masih terdapat kesalahan deteksi pada objek yang tidak ada seperti mobil (*Car*). Terlihat juga model ini beberapa kali mendeteksi kursi (*Chair*) pada beberapa frame walau objek tersebut tidak pernah ada.



Gambar 4.43 YOLOv10s well-lit test confusion matrix

Gambar 4.43 menunjukkan *confusion matrix* yang tidak dinormalisasi untuk hasil deteksi YOLOv10s pada kondisi pencahayaan cukup. Dapat dipahami bahwa masih saja banyak objek yang terkласifikasi sebagai latar belakang (*background*). Uniknya model dengan parameter kecil ini mampu mendeteksi objek sangat kecil seperti cangkir (*Cup*) lebih akurat dibanding dengan RTDETR-L yang memiliki parameter berkali-kali lipat lebih besar. Memperhitungkan jumlah parameter, nilai diagonal pada YOLOv10s cukup bisa ditoleransi.

Tabel 4.36 Hasil evaluasi YOLOv10s pada *well-lit test*

Model	mAP50	mAP50-95	<i>Inference Time (ms)</i>	<i>Total Time (s)</i>
YOLOv10s	0,3960	0,2102	221,08	17,30

Tabel 4.36 di atas, adalah tabel metrik hasil dari *testing* model YOLOv10s pada citra berpencahayaan cukup. Nilai-nilai metrik ini tercermin pula dari *confusion matrix* yang sebelumnya. Dengan parameter kecil, model hybrid ini mampu meraih performa yang cukup kompetitif dengan YOLOv9s yaitu mAP50 sebesar 0,3960 (-0,0012) dan inferensi waktu 221,08 ms (-35,04 ms). Menghasilkan total waktu deteksi untuk 56 citra selama 17,30 detik saja.

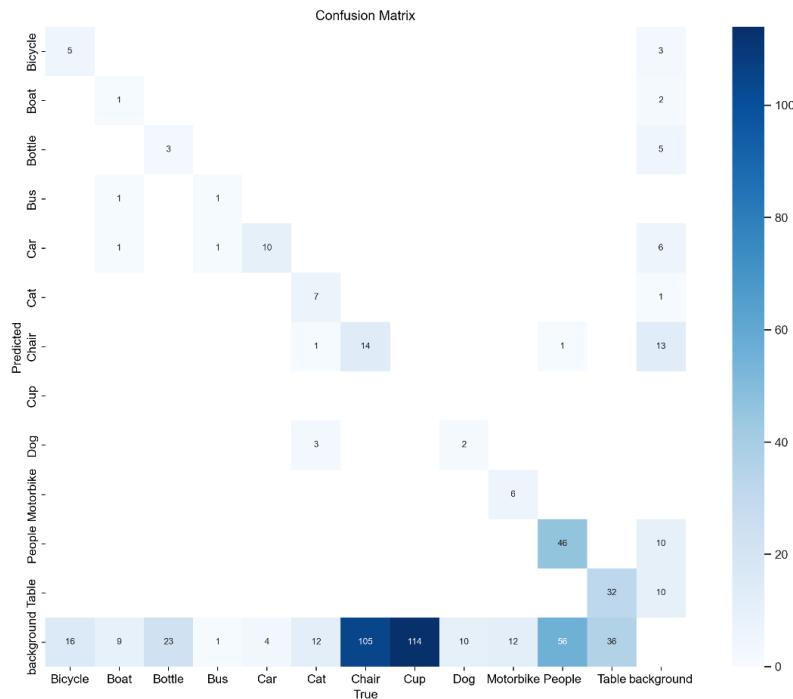
4.6.6.2 Low-light Detection

Tabel 4.37 Label vs Prediksi untuk YOLOv10s pada *low-light test*

<i>Ground Truth</i>	<i>Predicted</i>
30.jpg 	30.jpg
36.jpg 	36.jpg
40.jpg 	40.jpg
43.jpg 	43.jpg
44.jpg 	44.jpg
45.jpg 	45.jpg

Tabel 4.37 memperlihatkan contoh hasil deteksi objek oleh YOLOv10s yaitu model yang memiliki pendekatan hybrid dan berparameter paling kecil di keluarga YOLOv10, pada enam *frame* rekaman CCTV dengan pencahayaan yang rendah. Secara umum, YOLOv10s berhasil mengenali objek-objek berskala cukup besar dan kecil seperti pada manusia (*People*) dan sepeda (*Bicycle*) dengan lebih

akurat daripada kasus sebelumnya. Namun, terlihat bahwa model masih bingung dalam mendeteksi objek yang sebenarnya tidak ada seperti beberapa manusia pada sisi kiri.



Gambar 4.44 YOLOv10s *low-light test confusion matrix*

Gambar 4.44 menunjukkan *confusion matrix* yang tidak dinormalisasi untuk hasil deteksi YOLOv10s pada kondisi pencahayaan rendah. Gambar tersebut lebih banyak memberi informasi daripada tabel *groundtruth* sebelumnya karena *confusion matrix* ini menunjukkan bahwa nilai diagonal yang relatif buruk apalagi untuk objek sangat kecil seperti cangkir (*Cup*) yang tidak terdeteksi sama sekali.

Tabel 4.38 Hasil evaluasi YOLOv10s pada *low-light test*

Model	mAP50	mAP50-95	Inference Time (ms)	Total Time (s)
YOLOv10s	0,4144	0,2062	246,00	19,46

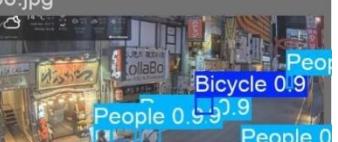
Tabel 4.38 di atas, adalah tabel metrik hasil dari *testing* model YOLOv10s pada citra berpencahayaan rendah. Nilai metrik utama mAP50 untuk YOLOv10s pada kasus citra berpencahayaan rendah lebih baik daripada kasus citra berpencahayaan cukup dengan

skor 0,4144 (+0,0184) tapi tidak dengan mAP50-95 yang turun sekitar 0,01 poin. Pendekatan hybrid ini juga mempengaruhi kecepatan inferensi yaitu 246,00 ms atau 10,07 ms lebih lama dibanding YOLOv9s yang menghasilkan total waktu 19,46 detik untuk 56 citra *test*.

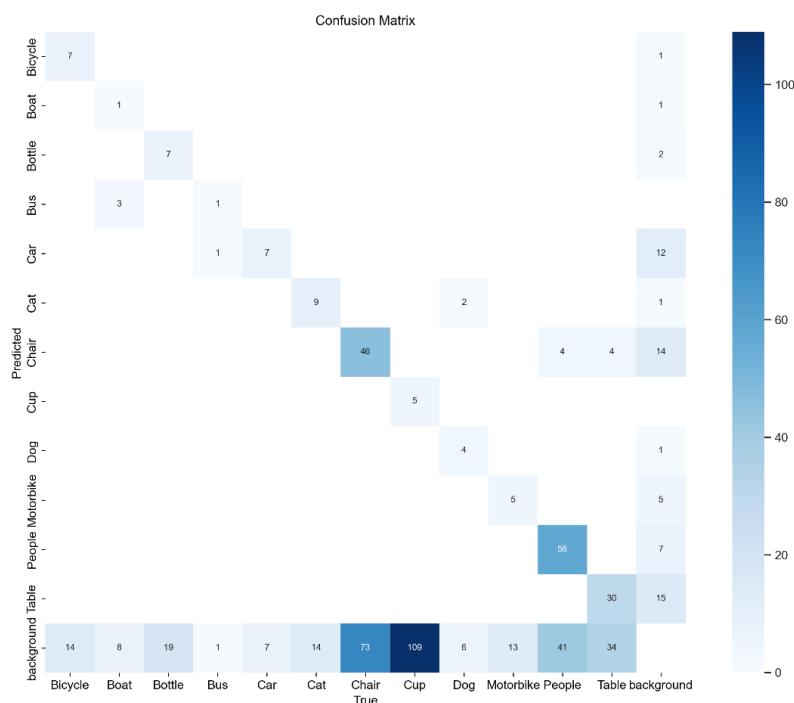
4.6.7 YOLOv10m

4.6.7.1 Well-lit Detection

Tabel 4.39 Label vs Prediksi untuk YOLOv10m pada *well-lit test*

<i>Ground Truth</i>	<i>Predicted</i>
30.jpg 	30.jpg 
36.jpg 	36.jpg 
40.jpg 	40.jpg 
43.jpg 	43.jpg 
44.jpg 	44.jpg 
45.jpg 	45.jpg 

Tabel 4.39 memperlihatkan contoh hasil deteksi objek oleh YOLOv10m, pada enam *frame* rekaman CCTV dengan pencahayaan yang cukup. Secara umum, YOLOv10m berhasil mengenali objek-objek berskala cukup besar dan kecil seperti pada manusia (*People*) dan sepeda (*Bicycle*) dengan jauh lebih akurat dari segi deteksi dan *bounding box* daripada YOLOv10s. Namun, terlihat bahwa model masih bingung dalam mendeteksi objek yang sebenarnya tidak ada seperti kursi (*Chair*) yang muncul pada satu frame awal.



Gambar 4.45 YOLOv10m *well-lit test confusion matrix*

Gambar 4.45 menunjukkan *confusion matrix* yang tidak dinormalisasi untuk hasil deteksi YOLOv10m pada kondisi pencahayaan cukup. Seperti pada kasus-kasus sebelumnya, masih saja banyak objek yang terklasifikasi sebagai latar belakang atau *background*. Nilai diagonal pada *confusion matrix* ini masih lebih baik daripada YOLOv10s, tetapi masih belum bisa mengungguli YOLOv9m.

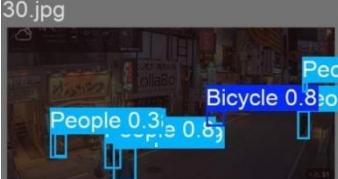
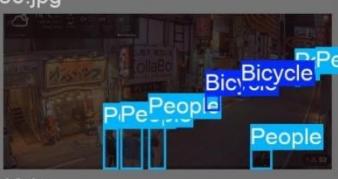
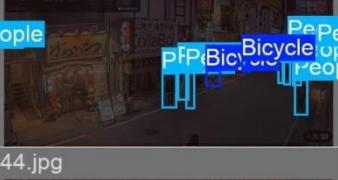
Tabel 4.40 Hasil evaluasi YOLOv10m pada *well-lit test*

Model	mAP50	mAP50-95	Inference Time (ms)	Total Time (s)
YOLOv10m	0,5047	0,2587	515,80	34,39

Tabel 4.40 di atas, adalah tabel metrik hasil dari *testing* model YOLOv10m pada citra berpencahayaan cukup. Nilai metrik utama mAP50 untuk YOLOv10m masih lebih rendah daripada YOLOv9m, yaitu 0,5047 (-0,0692). Di sisi lain, YOLOv10m mengungguli YOLOv9m dari segi inferensi waktu yaitu 515,80 ms atau 90,82 ms lebih cepat yang menghasilkan total waktu keseluruhan deteksi *test set* selama 34,39 detik.

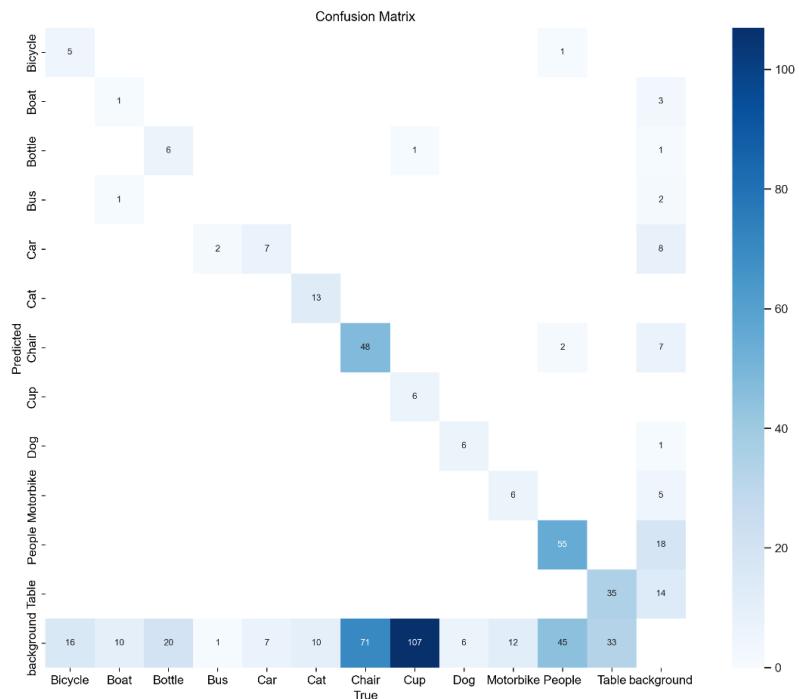
4.6.7.2 Low-light Detection

Tabel 4.41 Label vs Prediksi untuk YOLOv10m pada *low-light test*

<i>Ground Truth</i>	<i>Predicted</i>
30.jpg 	30.jpg 
36.jpg 	36.jpg 
40.jpg 	40.jpg 
43.jpg 	43.jpg 
44.jpg 	44.jpg 
45.jpg 	45.jpg 

Tabel 4.41 memperlihatkan contoh hasil deteksi objek oleh YOLOv10m, pada enam *frame* rekaman CCTV dengan pencahayaan yang rendah. Secara umum, YOLOv10m berhasil mengenali objek-objek berskala cukup besar dan kecil seperti pada manusia (*People*) dan sepeda (*Bicycle*) dengan lebih akurat daripada kasus sebelumnya walau dalam keadaan rendah cahaya. Kesalahan yang dilakukan oleh

YOLOv10m di kasus cahaya cukup adalah mendekksi kursi (*Chair*) yang sebenarnya tidak ada, namun tidak terlihat di kasus ini. Tetapi masih saja terlihat bahwa model mengalami kebingungan untuk mendekksi manusia (yang sebenarnya tidak ada) di sisi kiri walau dengan *confidence* yang rendah.



Gambar 4.46 YOLOv10m *low-light test confusion matrix*

Gambar 4.46 menunjukkan *confusion matrix* yang tidak dinormalisasi untuk hasil deteksi YOLOv10m pada kondisi pencahayaan rendah. Seperti pada kasus-kasus sebelumnya, masih saja banyak objek yang terklasifikasi sebagai latar belakang atau *background*. Nilai diagonal pada *confusion matrix* ini masih lebih baik daripada YOLOv10m untuk cahaya cukup, walau lagi-lagi belum dapat mengungguli YOLOv9m bahkan saat cahaya rendah sekalipun.

Tabel 4.42 Hasil evaluasi YOLOv10m pada *low-light test*

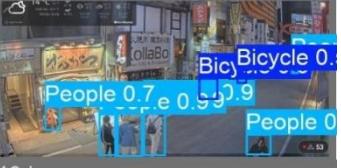
Model	mAP50	mAP50-95	Inference Time (ms)	Total Time (s)
YOLOv10m	0,5391	0,2677	574,25	37,67

Tabel 4.42 di atas, adalah tabel metrik hasil dari *testing* model YOLOv10m pada citra berpencahayaan rendah. Nilai metrik utama mAP50 untuk YOLOv10m masih lebih rendah daripada YOLOv9m di kasus yang sama, yaitu 0,5391 (-0,0316). Tetapi masih sama seperti kasus sebelumnya, YOLOv10m mengungguli YOLOv9m dari segi inferensi waktu yaitu 574,25 ms atau 14,2 ms lebih cepat yang menghasilkan total waktu keseluruhan deteksi *test set* selama 37,67 detik.

4.6.8 YOLOv10b

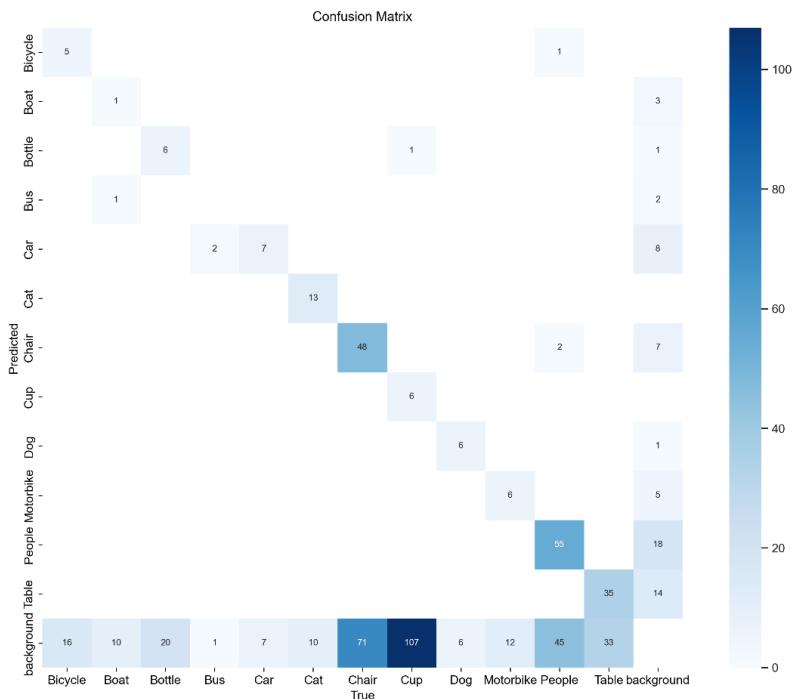
4.6.8.1 Well-lit Detection

Tabel 4.43 Label vs Prediksi untuk YOLOv10b pada *well-lit test*

<i>Ground Truth</i>	<i>Predicted</i>
30.jpg 	30.jpg 
36.jpg 	36.jpg 
40.jpg 	40.jpg 
43.jpg 	43.jpg 
44.jpg 	44.jpg 
45.jpg 	45.jpg 

Tabel 4.43 memperlihatkan contoh hasil deteksi objek oleh YOLOv10b yaitu model dari keluarga YOLOv10 dengan parameter terbesar di penelitian ini, pada enam *frame* rekaman CCTV dengan pencahayaan yang cukup. Jika hanya berdasarkan pada tabel 4.43, YOLOv10b berhasil mengenali objek-objek berskala cukup besar dan kecil seperti pada manusia (*People*) dan sepeda (*Bicycle*) dengan

sangat baik melebihi YOLOv10m. Tetapi satu sampel dari hasil ini saja tidak dapat membuktikan performa model secara keseluruhan.



Gambar 4.47 YOLOv10b well-lit test confusion matrix

Gambar 4.47 menunjukkan *confusion matrix* yang tidak dinormalisasi untuk hasil deteksi YOLOv10b pada kondisi pencahayaan cukup. Terlihat bahwa nilai diagonal dari model ini tidak jauh berbeda, atau bahan cenderung sedikit lebih buruk daripada YOLOv10m. Keakuratan model YOLOv10b ini memiliki performa yang jauh lebih buruk pula jika dibandingkan dengan YOLOv9c.

Tabel 4.44 Hasil evaluasi YOLOv10b pada *well-lit test*

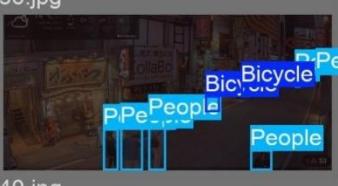
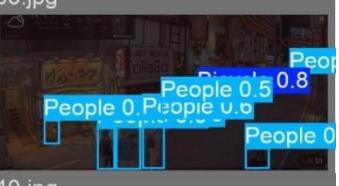
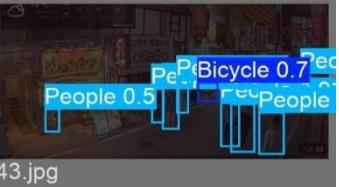
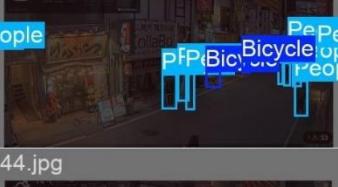
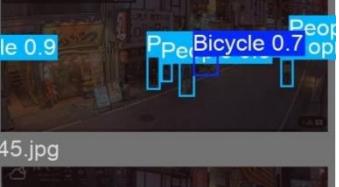
Model	mAP50	mAP50-95	Inference Time (ms)	Total Time (s)
YOLOv10b	0,4900	0,2251	770,25	49,14

Tabel 4.44 di atas, adalah tabel metrik hasil dari *testing* model YOLOv10b pada citra berpencahayaan cukup. Nilai-nilai pada tabel ini juga dicerminkan dari *confusion matrix* sebelumnya. Jika dibandingkan dengan YOLOv10m di kasus yang sama saja, model

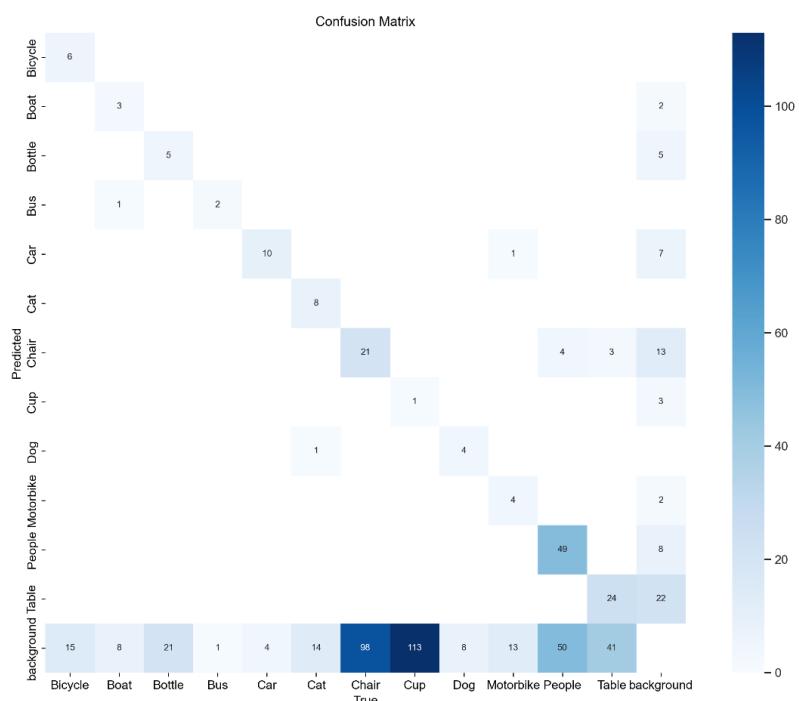
ini masih memiliki nilai mAP50 lebih buruk, yaitu 0,4900 atau 0,0424 lebih rendah. Terlebih lagi, model ini hanya unggul terhadap YOLOv9c dari segi inferensi waktu saja, yaitu 770,25 ms untuk setiap gambar, atau 90,29 ms lebih cepat. Menghasilkan total waktu deteksi untuk keseluruhan *test set* selama 49,14 detik saja.

4.6.8.2 Low-light Detection

Tabel 4.45 Label vs Prediksi untuk YOLOv10b pada *low-light test*

<i>Ground Truth</i>	<i>Predicted</i>
30.jpg 	30.jpg 
36.jpg 	36.jpg 
40.jpg 	40.jpg 
43.jpg 	43.jpg 
44.jpg 	44.jpg 
45.jpg 	45.jpg 

Tabel 4.45 memperlihatkan contoh hasil deteksi objek oleh YOLOv10b yaitu model dari keluarga YOLOv10 dengan parameter terbesar di penelitian ini, pada enam *frame* rekaman CCTV dengan pencahayaan yang rendah. Jika hanya berdasarkan pada Tabel 4.45, kasus kali ini yaitu deteksi pada cahaya rendah justru membuat model mengalami kesulitan deteksi walau sebagian besar objek manusia (*People*) dan sepeda (*Bicycle*) dapat terdeteksi dengan benar. Kebingungan model masih sama seperti kasus-kasus sebelumnya, dimana model menganggap sebuah latar belakang sebagai manusia ataupun kursi (*Chair*).



Gambar 4.48 YOLOv10b *low-light test confusion matrix*

Gambar 4.48 menunjukkan *confusion matrix* yang tidak dinormalisasi untuk hasil deteksi YOLOv10b pada kondisi pencahayaan rendah. Terlihat bahwa nilai diagonal dari kasus cahaya rendah ini cenderung lebih buruk jika dibandingkan dengan YOLOv10b pada cahaya cukup. Walau demikian, model ini masih dapat mengungguli RTDETR-L di kasus yang sama.

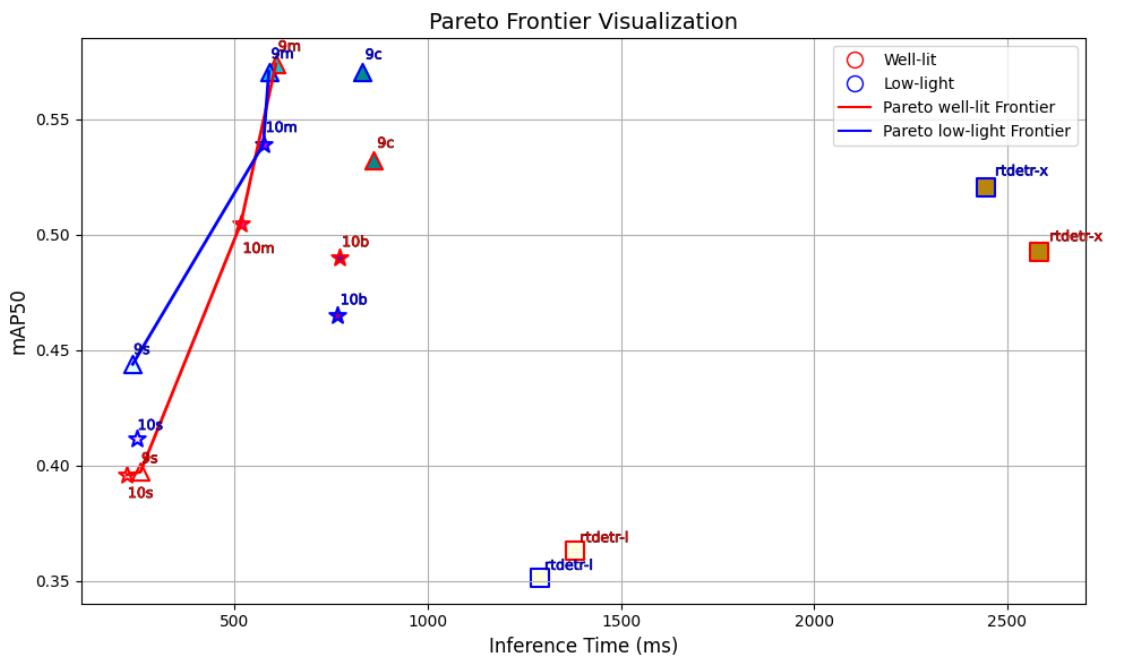
Tabel 4. 46 Hasil evaluasi YOLOv10b pada *low-light test*

Model	mAP50	mAP50-95	Inference Time	Total Time
YOLOv10b	0,4652	0,2136	765,94	49,52

Tabel 4.46 di atas, adalah tabel metrik hasil dari *testing* model YOLOv10b pada citra berpencahayaan cukup. Nilai-nilai pada tabel ini juga dicerminkan dari *confusion matrix* sebelumnya. Jika dibandingkan dengan YOLOv9c di kasus yang sama saja, model ini masih memiliki nilai mAP50 yang jauh lebih buruk, yaitu 0,4652 atau 0,1052 lebih rendah. Sama seperti kasus sebelumnya, model hybrid ini mampu meraih waktu inferensi yang lebih cepat jika dibandingkan dengan model murni CNN, yaitu 765,94 ms per gambar, atau 63,79 ms lebih cepat dibanding YOLOv9c. Menghasilkan total waktu deteksi selama 49,52 detik untuk keseluruhan *test set*.

4.7 Pareto Frontier Analysis

Setelah seluruh model selesai melalui tahap pelatihan dan evaluasi, metrik utama yang digunakan untuk menilai performa model adalah *mean Average Precision* (mAP50) dan *inference time*. Visualisasi hasil evaluasi terhadap kedua metrik ini ditunjukkan dalam bentuk kurva *Pareto Frontier* berikut:



Gambar 4.48 Pareto Frontier

Visualisasi pada Gambar 4.47 memperlihatkan perbedaan kinerja yang signifikan antara model CNN, Transformer, dan Hybrid. Secara umum, ketiga arsitektur tersebut menunjukkan performa yang lebih optimal ketika diuji dalam kondisi pencahayaan rendah. Hal ini ditunjukkan oleh posisi awal kurva Pareto pada kondisi *low-light* yang berada jauh di atas kurva pada kondisi *well-lit*. Selain itu, model-model yang mendominasi kurva Pareto ini mayoritas memiliki komponen arsitektur CNN, dengan keluarga YOLOv9 menjadi kelompok yang paling banyak berkontribusi.

Meskipun sebagian besar model menunjukkan keunggulan masing-masing pada kondisi pencahayaan cukup maupun rendah, *trade-off* terbaik pada kedua skenario pencahayaan tersebut secara konsisten dicapai oleh YOLOv9m. Model ini merupakan varian murni CNN dengan jumlah parameter yang berada di antara YOLOv9s dan YOLOv9c. YOLOv9m mencatatkan nilai mAP50 sebesar 0,5739 pada kondisi pencahayaan terang, dan 0,5707 pada kondisi pencahayaan rendah. Dari segi efisiensi waktu,

model ini juga tergolong cepat, yaitu dengan waktu inferensi sebesar 606,62 ms untuk kondisi cukup cahaya dan 588,45 ms untuk kondisi rendah cahaya.

4.8 Diskusi

Hasil penelitian menunjukkan adanya variasi kinerja yang signifikan di antara model-model yang diuji. Pada kondisi cahaya cukup, rentang nilai mAP50 berkisar antara 0,3632 (RTDETR-L) hingga 0,5739 (YOLOv9m), sementara waktu inferensi bervariasi dari 221,08 ms (YOLOv10s) hingga 2583,54 ms (RTDETR-X). Performa pada kondisi rendah cahaya juga menunjukkan tren serupa, dengan nilai mAP50 antara 0,3513 (RTDETR-L) dan 0,5707 (YOLOv9m), dan waktu inferensi antara 236,07 ms (YOLOv9s) dan 2444,03 (RTDETR-X).

Analisis lebih lanjut menggunakan konsep Pareto Frontier mengidentifikasi YOLOv9m sebagai model yang paling unggul. Keunggulan ini didasarkan pada kemampuannya untuk mencapai nilai map50 yang relatif tinggi pada kondisi cukup cahaya (0,5739) maupun rendah cahaya (0,5707), sambil mempertahankan waktu inferensi yang kompetitif (606,62 ms pada cahaya cukup dan 588,45 pada cahaya rendah) dibandingkan dengan model lain yang diuji. Selainnya, tipe model dengan parameter *medium* seringkali memberikan *trade-off* optimal. Parameter yang terlalu besar (tipe c / x) memperlambat inferensi dan juga belum tentu menaikkan akurasi, sementara parameter terlalu kecil (tipe s) menurunkan akurasi.

Dengan demikian, penelitian ini secara empiris menunjukkan bahwa untuk dataset dan konfigurasi eksperimen yang digunakan, YOLOv9m menawarkan keseimbangan terbaik antara mAP dan kecepatan dalam mendeteksi objek, terlepas dari kondisi pencahayaan. Temuan ini menjadi titik awal yang penting untuk membandingkan kinerja model-model ini dengan hasil yang dilaporkan dalam literatur ilmiah yang lebih luas.

4.8.1 Studi yang Konsisten dengan Temuan Penelitian

Beberapa penelitian dalam literatur mendukung gagasan bahwa model berbasis CNN terutama varian YOLO, menunjukkan kinerja yang baik dalam tugas deteksi objek, termasuk dalam kondisi yang menantang seperti pencahayaan rendah. Meskipun tidak semua studi secara spesifik fokus pada perbandingan langsung dengan model berbasis Transformer dalam kondisi *low-light*, beberapa temuan memberikan dukungan kontekstual terhadap keunggulan YOLOv9m yang diamati dalam penelitian ini.

Sebagai contoh, studi mengenai arsitektur hybrid CNN-Transformer oleh Tan dkk. (2023), mencatat bahwa *Convolutional Neural Networks* (CNN) tetap menjadi pilihan yang lebih disukai untuk tugas visi dunia nyata karena kecepatan inferensinya yang umumnya lebih tinggi dibandingkan dengan *Vision Transformers* (ViTs). Penelitian ini menyatakan bahwa meskipun ViT telah mencapai keberhasilan dalam berbagai aplikasi visi, termasuk deteksi objek, kinerja inferensinya seringkali lebih lambat daripada CNN tradisional seperti ResNet. Temuan ini secara tidak langsung mendukung mengapa YOLOv9m dalam penelitian ini menunjukkan waktu inferensi yang lebih kompetitif dibandingkan dengan model RT-DETR berbasis Transformer. Preferensi terhadap CNN untuk aplikasi praktis karena efisiensi sejalan dengan keunggulan YOLOv9m dalam hal *trade-off* akurasi dan kecepatan. Lebih lanjut, penelitian yang mengevaluasi varian YOLO untuk tugas deteksi spesifik, seperti EFA-YOLO untuk deteksi kebakaran oleh Pan dkk. (2024) menemukan bahwa model YOLO yang dioptimalkan dapat mencapai peningkatan signifikan dalam mAP dan kecepatan inferensi dibandingkan dengan versi-versi YOLO sebelumnya. Meskipun konteks deteksi kebakaran berbeda dengan deteksi objek umum dalam kondisi *low-light*, keberhasilan EFA-YOLO menunjukkan potensi inheren dari arsitektur YOLO untuk mencapai kinerja tinggi dan efisiensi komputasi. Peningkatan mAP dan pengurangan waktu inferensi yang dilaporkan dalam studi tersebut memperkuat gagasan bahwa keluarga model YOLO mampu memberikan *trade off* yang menguntungkan antara mAP dan kecepatan, yang konsisten dengan temuan penelitian ini mengenai YOLOv9m..

Dengan demikian, literatur yang tersedia memberikan dukungan tidak langsung terhadap keunggulan YOLO dalam hal kecepatan inferensi dan potensi untuk mencapai mAP tinggi melalui optimasi arsitektur. Kurangnya studi langsung yang membandingkan YOLO dan Transformer dalam deteksi objek *low-light* dalam literatur yang ditinjau menyoroti potensi kontribusi penelitian ini dalam engisi kesenjangan tersebut.

4.8.2 Studi yang Bertentangan dengan Temuan Penelitian

Meskipun penelitian ini menunjukkan keunggulan YOLOv9m, literatur juga mencatat keberhasilan model berbasis Transformer dalam berbagai tugas visi, termasuk deteksi objek (Tan *et al.*, 2023). Beberapa studi mungkin menunjukkan bahwa model berbasis Transformer, seperti DETR dan variannya, dapat mencapai akurasi yang lebih tinggi dalam deteksi objek, bahkan dalam kondisi cahaya rendah, meskipun mungkin dengan mengorbankan waktu inferensi yang lebih lambat.

Penelitian tentang arsitektur hybrid CNN-Transformer mengakui bahwa Vision Transformers (ViTs) telah berhasil dalam berbagai aplikasi visi komputer, termasuk deteksi objek. Ini mengimplikasikan bahwa Transformer memiliki kemampuan yang signifikan dalam memproses informasi visual dan berpotensi unggul dalam aspek-aspek tertentu dari kinerja, seperti mAP, dibandingkan dengan model berbasis CNN dalam kondisi tertentu. Meskipun fokus utama dari studi-studi ini seringkali adalah pada tantangan efisiensi ViT, keberhasilan mereka dalam deteksi objek secara umum menunjukkan bahwa mungkin ada studi lain yang secara spesifik menunjukkan kinerja lebih baik dari Transformer dalam kondisi *low-light*.

Selain itu, kemunculan arsitektur baru seperti Mamba, yang didasarkan pada *State Space Models* (SSMs) oleh Wang dkk. (2024) diklaim lebih efisien daripada Transformer, juga perlu dipertimbangkan. Jika penelitian di masa depan menunjukkan bahwa arsitektur seperti Mamba mampu mengungguli model YOLO dalam deteksi objek *low-light* dengan *trade-off* mAP dan kecepatan yang lebih baik, ini akan menjadi temuan yang bertentangan dengan hasil penelitian ini. Namun,

saat ini, literatur yang tersedia tidak memberikan informasi spesifik tentang kinerja Mamba dalam konteks deteksi objek *low-light*.

Oleh karena itu, meskipun literatur-literatur yang ditinjau ini tidak secara eksplisit menyajikan studi yang menyajikan studi yang menunjukkan model berbasis Transformer mengungguli YOLO dalam deteksi objek *low-light*, pengakuan terhadap kemampuan Transformer dalam tugas deteksi objek secara umum menunjukkan bahwa kemungkinan adanya studi semacam itu dalam literatur yang lebih luas. Penelitian lebih lanjut diperlukan untuk mengidentifikasi studi-studi yang secara spesifik mengevaluasi kinerja model berbasis Transformer pada dataset *low-light* dan membandingkannya secara langsung dengan model berbasis CNN seperti YOLO.

4.8.3 Analisis Penyebab Perbedaan

Perbedaan dalam kinerja antara model berbasis CNN dan Transformer dalam tugas deteksi objek pada kondisi *low-light* dapat disebabkan oleh beberapa faktor mendasar, termasuk arsitektur model, karakteristik dataset yang digunakan, metrik kerja yang dievaluasi, pendekatan peningkatan *low-light* yang diterapkan, serta ukuran dan kompleksitas model.

Arsitektur CNN, dengan lapisan konvolusi yang dirancang untuk mengekstrak fitur lokal dan hierarkis, mungkin lebih tahan terhadap *noise* yang seringkali hadir dalam gambar *low-light* (Tan *et al.*, 2023). Kemampuan CNN untuk mempelajari pola spasial lokal secara efektif dapat membantu mengidentifikasi objek meskipun detail visualnya kurang jelas karena pencahayaan rendah. Di sisi lain, arsitektur Transformer, yang mengandalkan mekanisme *self-attention* untuk menangkap hubungan global antar bagian gambar, mungkin akan lebih unggul dalam memahami konteks keseluruhan citra, yang dapat membantu dalam mendeteksi objek meskipun visibilitasnya buruk (Tan *et al.*, 2023). Namun, mekanisme *self-attention* pada Transformer juga bisa lebih rentan terhadap *noise* karena setiap bagian gambar berinteraksi dengan semua bagian lainnya.

Karakteristik dataset yang digunakan untuk evaluasi juga memainkan peran penting dalam menentukan kinerja model. Dataset deteksi objek *low-light* dapat bervariasi secara signifikan dalam hal tingkat pencahayaan, jenis citra yang direkam, variasi objek yang ada, dan kualitas anotasi (Shi *et al.*, 2024). Model yang berkinerja baik pada satu dataset *low-light* mungkin tidak menunjukkan kinerja yang sama pada dataset lain dengan karakteristik yang bebeda. Oleh karena itu, penting untuk mempertimbangkan kesamaan antara dataset yang digunakan dalam penelitian ini dan dataset yang digunakan dalam studi literatur untuk membuat perbandingan yang valid.

Metrik kinerja yang digunakan untuk evaluasi juga harus diperhatikan dengan baik. Penelitian ini menggunakan mAP50 dan inferensi sebagai metrik utama. Studi lain mungkin menggunakan metrik yang berbeda, seperti mAP pada IoU yang bervariasi (mAP50-95) atau bahkan *Frames Per Second* (FPS) untuk mengukur kecepatan inferensi. Perbedaan dalam metrik dapat mempersulit perbandingan langsung antar studi.

Pendekatan yang digunakan untuk meningkatkan kinerja deteksi objek dalam kondisi *low-light* juga dapat menjadi faktor penyebab perbedaan hasil. Beberapa studi mungkin menggunakan teknik *preprocessing* gambar khusus, seperti peningkatan kontras, penyesuaian histogram, atau *denoization*, sebelum memasukkan gambar ke dalam model deteksi (Shi *et al.*, 2024). Studi lain mungkin memodifikasi arsitektur model itu sendiri untuk membuatnya lebih cocok untuk kondisi *low-light*.

Terakhir, ukuran dan kompleksitas model deteksi objek juga dapat mempengaruhi kinerjanya. Model yang lebih besar dan lebih kompleks dengan lebih banyak parameter cenderung memiliki kapasitas yang lebih besar untuk mempelajari pola yang rumit dalam data, yang berpotensi menghasilkan mAP yang lebih tinggi. Namun, model yang lebih besar juga membutuhkan lebih banyak sumber daya komputasi dan cenderung memiliki waktu inferensi yang lebih lama (Tan *et al.*, 2023).

YOLOv9m unggul dalam *trade-off* antara mAP dan kecepatan inferensi karena ia menggabungkan dua inovasi arsitektural yang saling melengkapi.

Pertama, *Programmable Gradient Information (PGI)* yang memperkenalkan cabang *reversible* yang mempertahankan aliran gradien penuh ke lapisan terdalam, yang dapat mengatasi *bottleneck* informasi yang sering terjadi pada arsitektur CNN tradisional dan menungkinkan model menangkap fitur semantik dengan baik. Kedua, *Generalized Efficient Layer Aggregation Network (GELAN)* memanfaatkan blok konvolusi ringan (seperti CSPblock dan Resblock) untuk mengagregasi fitur *multiscale* tanpa menambah beban parameter signifikan. Kombinasi ini menghasilkan model yang relatif ringkas namun mencapai mAP50 sekitar 0,57 sambil mempertahankan waktu inferensi yang cukup (600 ms), sehingga konsisten dominan pada kurva Pareto untuk kondisi *low-light* maupun *well-lit*.

Sebaliknya, RT-DETR menonjol pada sisi akurasi berkat desain *Efficient Hybrid Encoder* dan strategi *uncertainty-minimal query selection*, namun mengembangkan biaya waktu inferensi yang jauh lebih tinggi. *Hybrid encoder* RT-DETR memisahkan *Attention-based Intra-scale Feature Interaction (AIFI)* yang memperkuat pemrosesan fitur dalam setiap skala dengan *CNN-based Cross-scale Feature Fusion (CCFF)* untuk efisiensi komputasi. Sementara itu, mekanisme *query selection* memastikan *decoder Transformer* menerima *query* dengan ketepatan tinggi, meningkatkan kualitas prediksi *bounding box*. Namun, kompleksitas *self-attention* dan decoding menghasilkan waktu inferensi antara 1.300 – 2.400 ms per citra, sehingga RTDETR meski kompetitif di mAP (0,36 – 0,52) tidak seefisien YOLOv9m dalam aplikasi *real-time*.

YOLOv10 menawarkan pendekatan *hybrid* yang paling cpeat pada tipe terkecil yaitu YOLOv10s, berkat *dual label assignments* pengganti *Non-Maximum Suppression (NMS)* guna memangkas waktu. Pada tipe yang besar, akurasi didongkrak oleh *large-kernel convolutions*, yang memperluas *receptive field* untuk menangani objek beragam skala dan *partial self-attention* yang menyuntikkan konteks global secara selektif tanpa menambah komputasi yang berlebih. Hasilnya, YOLOv10s mencapai inferensi tercepat (~221 ms) dengan mAP lebih rendah (~0,40), sedangkan YOLOv10b memperbaiki mAP hingga ~0,49-0,46 meski dengan waktu infrensi menengah (~760 ms). Komponen-komponen dalam

arsitektur inilah mengapa YOLOv10 cocok untuk skenario kecepatan ekstrem, namun belum mengungguli keseimbangan performa-komputasi YOLOv9m.

Mengingat berbagai faktor ini, perbedaan dalam hasil kinerja antara penelitian ini dan studi literatur dapat disebabkan oleh kombinasi dari perbedaan arsitektur model, dataset yang digunakan, dan ukuran serta kompleksitas model.

4.8.4 Implikasi Temuan

Temuan penelitian ini, yang menunjukkan keunggulan YOLOv9m dalam hal *trade-off* antara mAP dan kecepatan inferensi untuk deteksi objek dalam kondisi cukup cahaya dan rendah cahaya, memiliki implikasi signifikan untuk pengembangan dan penerapan sistem CCTV yang beroperasi selama 24 jam, Sistem CCTV modern diharapkan dapat memberikan kinerja yang baik di berbagai kondisi pencahayaan dari siang hari yang terang hingga malam hari yang gelap.

Persyaratan kinerja untuk aplikasi CCTV dapat bervariasi tergantung pada skenario penggunaan spesifik. Dalam beberapa kasus, deteksi *real-time* dengan waktu inferensi rendah mungkin menjadi prioritas utama, terutama untuk aplikasi yang memerlukan respons cepat terhadap peristiwa yang terdeteksi. Dalam kasus lain, akurasi deteksi yang tinggi mungkin lebih tinggi, terutama dalam skenario keamanan kritis dimana kegagalan untuk mendeteksi objek dapat memiliki konsekuensi yang serius.

Pemilihan model deteksi objek untuk aplikasi CCTV 24 jam juga harus mempertimbangkan sumber daya komputasi yang tersedia untuk sistem. Model dengan waktu inferensi yang lebih rendah, seperti varian YOLO yang lebih kecil mungkin lebih cocok untuk sistem dengan daya komputasi terbatas, meskipun mungkin dengan mengorbankan sedikit akurasi. Sebaliknya, model yang lebih besar dan akurat mungkin memerlukan *hardware* yang lebih kuat. YOLOv9m, dengan *trade-off* kinerjanya menawarkan pilihan yang baik untuk banyak aplikasi CCTV, tetapi pemilihan model akhir harus didasarkan pada persyaratan kinerja spesifik dan batasan sumber daya dari sistem yang diterapkan.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil pengujian dan analisis Pareto Frontier terhadap tiga keluarga model (YOLOv9 sebagai *CNN-based*, RT-DETR sebagai *Transformer-based*, dan YOLOv10 sebagai *CNN-Transformer-based* atau *Hybrid-based*) pada akondisi pencahayaan rendah, dapat ditarik beberapa kesimpulan utama:

1. Performa *trade-off* terbaik dicapai oleh YOLOv9m.

YOLOv9m menempati kurva Pareto Front sebagai model dengan kombinasi mAP50 dan inferensi yang paling seimbang. Pada kondisi pencahayaan cukup, YOLOv9m mencapai mAP50 sebesar 0,5739 dan waktu inferensi 606,62 ms. Sedangkan pada kondisi pencahayaan rendah memperoleh mAP50 sebesar 0,5707 dengan waktu inferensi 588,45 ms.

2. *CNN-based* umumnya lebih cepat dibanding *Transformer-based*.

Varian YOLO (CNN) seperti YOLOv9s, YOLOv9m, dan YOLOv9c menunjukkan waktu inferensi yang jauh lebih singkat (606 – 700 ms) dibandingkan RTDETR-L dan RTDETR-X yang memerlukan 1.289 ms – 2.444 ms per gambar.

3. *Hybrid-based* menawarkan keseimbangan, tetapi belum mengungguli CNN murni.

YOLOv10s adalah model tercepat di antara semua model (221,08 ms per gambar) namun hanya mencapai mAP50 sebesar 0,3960 pada citra cahaya cukup. Sedangkan varian terbesar YOLOv10b memperoleh mAP50 sebesar 0,4652 (*low-light*) dengan inferensi 765,94 ms , masih di bawah performa *trade-off* YOLOv9m.

5.2 Saran

Berdasarkan kesimpulan di atas, berikut beberapa rekomendasi untuk penelitian dan penerapan di masa mendatang:

1. Penelitian ini hanya menguji pada varian YOLOv10 standard, tidak bereksperimen dengan arsitektur *hybrid custom* yang menyeimbangkan *depth CNN* dan kompleksitas *Transformer* secara adaptif. Meneliti desain *hybrid* baru, misalnya dengan *partial attention* hanya pada lapisan atas atau modul *self-attention* yang dipangkas, bisa saja menurunkan beban komputasi tanpa mengorbankan akurasi.
2. Tuning yang dilakukan oleh penelitian ini juga terbatas pada *learning rate* dan *batch size* yang total berjumlah 4 kombinasi saja. Menambahkan *hyperparameter tuning* seperti *scheduler LR* dan *Optimizer* untuk berbagai tingkat dan juga menggunakan *GAN-based augmentation* bisa saja dapat memperluas ruang pencarian.
3. Teknik *contrast enhancement* yang diterapkan pada penelitian ini hanya CLAHE dan tidak ada uji coba modul *enhancement* berbasis CNN/Transformer. *Deep-learning-based enhancement* misalnya *Low-light Image Enhancement* (LLIE) atau *ISP-aware modules* bisa saja memperkaya fitur sebelum inferensi.

DAFTAR PUSTAKA

- Alif, M. A. R., & Hussain, M. (2024). YOLOv1 to YOLOv10: A comprehensive review of YOLO variants and their application in the agricultural domain. *arXiv preprint arXiv:2406.10139*.
- Baik, S., & Kim, E. (2025). *Detection of Human Traffic Controllers Wearing Construction Workwear via Synthetic Data Generation*. Sensors (Basel, Switzerland), 25(3), 816.
- Benmeziane, H., Niar, S., Ouarnoughi, H., & El Maghraoui, K. (2022, May). Pareto rank surrogate model for hardware-aware neural architecture search. In 2022 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS) (pp. 267-276). IEEE.
- Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of machine learning research*, 13(2).
- Cacciola, M., Frangioni, A., Asgharian, M., Ghaffari, A., & Nia, V. P. (2023). On the convergence of stochastic gradient descent in low-precision number formats. *arXiv preprint arXiv:2301.01651*.
- Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., & Zagoruyko, S. (2020, August). End-to-end object detection with transformers. In *European conference on computer vision* (pp. 213-229). Cham: Springer International Publishing.
- Chen, C., Chen, Q., Xu, J., & Koltun, V. (2018). Learning to see in the dark. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3291-3300).
- Chen, W., & Shah, T. (2021). Exploring low-light object detection techniques. *arXiv preprint arXiv:2107.14382*.
- Chicco, D., & Jurman, G. (2020). The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. *BMC genomics*, 21, 1-13.
- Cui, Z., Li, K., Gu, L., Su, S., Gao, P., Jiang, Z., ... & Harada, T. (2022). You only need 90k parameters to adapt light: a light weight transformer for image enhancement and exposure correction. *arXiv preprint arXiv:2205.14871*.
- Cui, Z., Qi, G. J., Gu, L., You, S., Zhang, Z., & Harada, T. (2021). Multitask aet with orthogonal tangent regularity for dark object detection. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 2553-2562).
- Deb, K. (2001). *Multiobjective Optimization Using Evolutionary Algorithms*. Wiley, New York.

- Dhillon, A., & Verma, G. K. (2019). *Convolutional neural network: a review of models, methodologies and applications to object detection*. Progress in Artificial Intelligence, 9(2), 85–112. <https://doi.org/10.1007/s13748-019-00203-0>.
- Diwan, T., Anirudh, G., & Tembhurne, J. V. (2022). *Object detection using YOLO: challenges, architectural successors, datasets and applications*. Multimedia Tools and Applications, 82(6), 9243–9275. <https://doi.org/10.1007/s11042-022-13644-y>.
- Everingham, M., Van Gool, L., Williams, C. K., Winn, J., & Zisserman, A. (2010). *The pascal visual object classes (voc) challenge*. International journal of computer vision, 88, 303–338.
- Gao, C., Zhang, Q., Tan, Z., Zhao, G., Gao, S., Kim, E., & Shen, T. (2024). *Applying optimized YOLOv8 for heritage conservation: enhanced object detection in Jiangnan traditional private gardens*. Heritage Science, 12.
- Guemas, E., Routier, B., Ghelfenstein-Ferreira, T., Cordier, C., Hartuis, S., Marion, B., Bertout, S., Varlet-Marie, E., Costa, D., & Pasquier, G. (2024). *Automatic patient-level recognition of four Plasmodium species on thin blood smear by a real-time detection transformer (RT-DETR) object detection algorithm: a proof-of-concept and evaluation*. Microbiology Spectrum, 12(2), e01440-23.
- Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. In *MIT Press eBooks*. <https://dl.acm.org/citation.cfm?id=3086952>.
- Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M., & Tang, P. T. P. (2016). *On large-batch training for deep learning: Generalization gap and sharp minima*. arXiv preprint arXiv:1609.04836.
- LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). *Backpropagation applied to handwritten zip code recognition*. Neural computation, 1(4), 541–551.
- Lee, J., Bang, J., & Yang, S. (2017). *Object detection with sliding window in images including multiple similar objects*. 2017 International Conference on Information and Communication Technology Convergence (ICTC), Pp. 803–806. <https://doi.org/10.1109/ictc.2017.8190786>.
- Li, J., Chen, C., Huang, W., Lang, Z., Song, F., Yan, Y., & Xiong, Z. (2023). *Learning steerable function for efficient image resampling*. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 5866–5875).
- Li, J., Feng, Y., Shao, Y., & Liu, F. (2024). *IDP-YOLOV9: Improvement of Object Detection Model in Severe Weather Scenarios from Drone Perspective*. Applied Sciences, 14(12), 5277.

- Li, Y., & Liang, Y. (2018). *Learning overparameterized neural networks via stochastic gradient descent on structured data*. Advances in neural information processing systems, 31.
- Li, X., Wang, W., Wu, L., Chen, S., Hu, X., Li, J., Tang, J., & Yang, J. (2020). *Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection*. Advances in Neural Information Processing Systems, 33, 21002-21012.
- Li, Z., Liu, F., Yang, W., Peng, S., & Zhou, J. (2021). *A survey of convolutional neural networks: analysis, applications, and prospects*. IEEE transactions on neural networks and learning systems, 33(12), 6999-7019.
- Lin, T. Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). *Focal loss for dense object detection*. In Proceedings of the IEEE international conference on computer vision (pp. 2980-2988).
- Loh, Y. P., & Chan, C. S. (2019). *Getting to know low-light images with the exclusively dark dataset*. Computer Vision and Image Understanding, 178, 30-42.
- Manongga, W. E., Chen, R. C., & Jiang, X. *Enhancing road marking sign detection in low-light conditions with YOLOv7 and contrast enhancement techniques*.
- Masters, D., & Luschi, C. (2018). *Revisiting small batch training for deep neural networks*. arXiv preprint arXiv:1804.07612.
- Mottola, M. (2018). *Reproducibility of CT-based radiomic features against image resampling and perturbations for tumour and healthy kidney in renal cancer patients*. Sci Rep 11, 11542 (2021). 4. Rizzo, S. et al. Radiomics: the facts and the challenges of image analysis. *Eur Radiol Exp*, 2, 36.
- Pan, W., Wang, X., & Huan, W. (2024). *EFA-YOLO: An Efficient Feature Attention Model for Fire and Flame Detection*. arXiv preprint arXiv:2409.12635.
- Paul, C., & Godambe, M. (2021). *Image Downsampling & Upsampling*.
- Powers, D. M. (2020). *Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation*. arXiv preprint arXiv:2010.16061.
- Putri, G. R. C. K. B. (2021). *Peran Kamera Pengawas Closed-Circuit Television (CCTV) dalam Kontra Terorisme*. *Jurnal Lemhannas RI*, 9(4), 100–116. <https://doi.org/10.55960/jlri.v9i4.418>
- Prechelt, L. (2002). *Early stopping-but when?*. In *Neural Networks: Tricks of the trade* (pp. 55-69). Berlin, Heidelberg: Springer Berlin Heidelberg.
- Redmon, J., & Farhadi, A. (2018). *YOLOV3: an incremental improvement*. arXiv (Cornell University). <https://doi.org/10.48550/arxiv.1804.02767>.

- Redmon, J., Santosh, D. H. H., Ross, G., & Farhadi, A. (2015). *You only look once: Unified, Real-Time Object Detection*. arXiv (Cornell University). <https://doi.org/10.48550/arxiv.1506.02640>.
- Rezatofighi, H., Tsoi, N., Gwak, J., Sadeghian, A., Reid, I., & Savarese, S. (2019). *Generalized intersection over union: A metric and a loss for bounding box regression*. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 658-666).
- Rouhbakhshmeghrazi, A., & Alizadeh, G. *Instance Segmentation of Messier Objects: YOLO vs. Mask R-CNN*.
- Ruby, U., & Yendapalli, V. (2020). *Binary cross entropy with deep learning technique for image classification*. *Int. J. Adv. Trends Comput. Sci. Eng.*, 9(10).
- Saito, T., & Rehmsmeier, M. (2015). *The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets*. *PLoS one*, 10(3), e0118432.
- Sandy, C. L. M., Husna, A., & Rizal, R. A. (2024). Real time chicken egg size classification using Yolov4 algorithm. *Brilliance Research of Artificial Intelligence*, 4(2), 577–584. <https://doi.org/10.47709/brilliance.v4i2.4496>
- Sheibanifard, A., & Yu, H. (2023). *A novel implicit neural representation for volume data*. *Applied Sciences*, 13(5), 3242.
- Shi, K., He, S., Shi, Z., Chen, A., Xiong, Z., Chen, J., & Luo, J. (2024). *Radar and Camera Fusion for Object Detection and Tracking: A Comprehensive Survey*. *arXiv preprint arXiv:2410.19872*.
- Shorten, C., & Khoshgoftaar, T. M. (2019). *A survey on Image Data Augmentation for Deep Learning*. *Journal of Big Data*, 6(1), 60.
- Smith, L. N. (2017, March). *Cyclical learning rates for training neural networks*. In *2017 IEEE winter conference on applications of computer vision (WACV)* (pp. 464-472). IEEE.
- Tan, W., Geng, Y., & Xie, X. (2024). *FMViT: A multiple-frequency mixing Vision Transformer*. In *ECAI 2024* (pp. 97-104). IOS Press.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). *Attention is All you Need*. arXiv (Cornell University), 30, 5998–6008. <https://arxiv.org/pdf/1706.03762v5.pdf>.
- Voulodimos, A., Doula, A., & Protopapadakis, E. (2018). *Deep Learning for Computer Vision: A Brief review*. *Computational Intelligence and Neuroscience*, 2018, 1–13. <https://doi.org/10.1155/2018/7068349>.
- Wang, A., Chen, H., Liu, L., Chen, K., Lin, Z., Han, J., & Ding, G. (2024). *Yolov10: Real-time end-to-end object detection*. *arXiv preprint arXiv:2405.14458..*

- Wang, C. Y., & Liao, H. Y. M. (2024). YOLOv1 to YOLOv10: *The fastest and most accurate real-time object detection systems*. arXiv preprint arXiv:2408.09332.
- Wang, C. Y., Yeh, I. H., & Liao, H. Y. M. (2024). *Yolov9: Learning what you want to learn using programmable gradient information*. arXiv preprint arXiv:2402.13616.
- Wang, C., Zheng, W., Zhou, J., & Lu, J. (2024). *GlobalMamba: Global Image Serialization for Vision Mamba*. arXiv preprint arXiv:2410.10316.
- Wang, M., & Wu, L. (2023). *A theoretical analysis of noise geometry in stochastic gradient descent*. arXiv preprint arXiv:2310.00692.
- Wiley, V., & Lucas, T. W. (2018). *Computer Vision and Image Processing: A paper review*. International Journal of Artificial Intelligence Research, 2(1), 22. <https://doi.org/10.29099/ijair.v2i1.42>.
- Yu, J., Jiang, Y., Wang, Z., Cao, Z., & Huang, T. (2016, October). *Unitbox: An advanced object detection network*. In Proceedings of the 24th ACM international conference on Multimedia (pp. 516-520).
- Yaghini, M., Liu, P., Boenisch, F., & Papernot, N. (2023). *Learning with impartiality to walk on the pareto frontier of fairness, privacy, and utility*. arXiv preprint arXiv:2302.09183.
- Ye, R., Chen, L., Liao, W., Zhang, J., & Ishibuchi, H. (2024). *Data-Driven Preference Sampling for Pareto Front Learning*. arXiv.
- Yin, X., Yu, Z., Fei, Z., Lv, W., & Gao, X. (2023, September). *Pe-yolo: Pyramid enhancement network for dark object detection*. In *International Conference on Artificial Neural Networks* (pp. 163-174). Cham: Springer Nature Switzerland.
- Yusuf, M. O., Hanzla, M., Al Mudawi, N., Sadiq, T., Alabdullah, B., Rahman, H., & Algarni, A. (2024). *Target Detection and Classification via EfficientDet and CNN over Unmanned Aerial Vehicles*. Frontiers in Neurorobotics, 18, 1448538.
- Zhao, Y., Lv, W., Xu, S., Wei, J., Wang, G., Dang, Q., Liu, Y., & Chen, J. (2024). *Detrs beat yolos on real-time object detection*. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 16965-16974).
- Zhao, Z. Q., Zheng, P., Xu, S. T., & Wu, X. (2019). *Object detection with deep learning: A review*. IEEE transactions on neural networks and learning systems, 30(11), 3212-3232.
- Zheng, S., & Gupta, G. (2022). *Semantic-guided zero-shot learning for low-light image/video enhancement*. In *Proceedings of the IEEE/CVF Winter conference on applications of computer vision* (pp. 581-590).