FORMULIR LOOGBOOK BIMBINGAN DAN PENGAJUAN
SIDANG TUGAS AKHIR

Nama : Widya Pratiwi
NIM : 4212111018
Pembimbing : Dr. Budi Sugandi, S.T., M.Eng
Judul : Analisa Kinerja Sistem YOLOv4- tiny Pada Pembuatan Alat Deteksi Kecurangan Jual Beli Gas LPG 3Kg

No	Hari/Tgl	Rincian Kegiatan	TTD Pembimbing
1	20 September 2025	Perubahan judul untuk membedakan buku TA dari "Alat Pendeteksi Kecurangan Transaksi Jual Beli Gas LPG 3Kg pada Pangkalan Gas dengan Metode YOLOv4-Tiny" menjadi "Analisa Kinerja Sistem YOLOv4- tiny Pada Pembuatan Alat Deteksi Kecurangan Jual Beli Gas LPG 3Kg".	
2	7 Februari 2025	Melaporkan progres TA.	
3	22 Februari 2025	Melakukan penyesuaian Laporan TA BAB 3.	
4	28 Februari 2025	Diskusi mengenai pembagian fokus tugas akhir antara pengujian kinerja sistem dan keandalan sistem.	
5	8 Juni 2025	Melaporkan hasil Buku Tugas Akhir.	
6	10 juni 2025	Revisi Bab 4 terkait metode perihal anotasi, augmentasi, split rasio, dan pengaturan parameter.	
7	12 Juni 2025	Melaporkan hasil revisi BAB 4.	
8	13 Juni 2025	Revisi gambar hasil uji, perubahan model grafik Analisa pada BAB 4, dan perapihan penomoran pada BAB 5.	
9	23 Juni 2025	Finalisasi Laporan Tugas Akhir (Persiapan Sidang).	
10	24 Juni 2025	Finalisasi Laporan Tugas Akhir (Persiapan Sidang).	

Berdasarkan hasil bimbingan yang telah dilaksanakan selama 9 (Sembilan) bulan dan telah disetujui oleh dosen pembimbing, maka dengan ini saya mengajukan diri sebagai peserta Sidang Tugas Akhir.

Batam, 22 Juni 2025
Peserta

Widya Pratiwi
NIM: 4212111018

Perbaikan Proposal dan Buku Tugas Akhir (v0)

Program Studi Teknik Mekatronika
polibatam

Judul: Analisa Kinerja Sistem YOLOv4-Tiny Pada Pembuatan Alat Deteksi Kecurangan Jual Beli Gas LPG 3 kg

Nama Mahasiswa: Widya Pratiwi

NIM Mahasiswa: 4212111018

Pembimbing 1 (P1): Dr. Budi Sugandi, S.T., M.Eng

Pembimbing 2 (P2): -

Penguji 1 (E1): Rifqi Amalya Fatekha, S.ST., M.Tr.T

Penguji 2 (E2): Ridwan, S.ST., M.Tr.T

Tanggal Sidang: 18 Juli 2025

Tabel Daftar Perbaikan

No.	Komentar	Perbaikan	Halaman lama (s)	Halaman baru (s)
Komentar General				
1 E2	Kata-kata dalam Laporan Akhir perlu disesuaikan dan direvisi, karena masih ditemukan kalimat-kalimat yang berasal dari dokumen proposal atau seminar proposal sebelumnya.	Sudah dilakukan penyesuaian kata-kata laporan tugas akhir	-	-
Abstract				
Bab 1				
1 E1	Tujuan pada poin pertama harus diperjelas mengenai aspek yang dianalisis. Tujuan tersebut harus sesuai dengan pengujian dan analisis yang telah dilakukan. Jika terdapat tujuan yang tidak relevan karena tidak dilakukan pengujiannya, maka tujuan tersebut sebaiknya dihapus.	**1.3. Tujuan** Berdasarkan rumusan masalah yang telah dijabarkan di atas, maka didapatkan tujuan sebagai berikut: 1. Melakukan analisis terhadap faktor-faktor yang mempengaruhi kinerja, seperti rasio dataset dan pengaturan model, untuk meningkatkan kinerja model *deep learning* pada alat deteksi gas LPG 3 kg. 2. Mengevaluasi pengaruh pengaturan konfigurasi *hyperparameter* terhadap kinerja sistem deteksi untuk memahami hubungan antara pengaturan tersebut terhadap akurasi deteksi dan kestabilan deteksi. 3. Mengidentifikasi konfigurasi *hyperparameter* yang paling optimal untuk mencapai performa kinerja deteksi terbaik pada alat deteksi kecurangan jual beli gas LPG 3 kg.	2	2
Bab 2				
Bab 3				
2 E1	Teori pengujian harus disesuaikan dengan hasil analisis yang diperoleh dari pengujian.(Pengujian FPS)	dalam pemantauan *real time*. Pengujian dilakukan dengan memproses video berdurasi kurang lebih dua menit pada tiga kondisi yang berbeda dan menggunakan video baru bukan video yang digunakan sebagai dataset pelatihan, lalu dihitung nilai rata-rata FPS pada masing-masing kondisi. Hasil pengujian	26	28

<table>
<tr><td>3 E1</td><td>Pengujian rasio data harus disertai narasi penjelasan mengenai makna dari setiap nilai rasio yang digunakan.</td><td>3.8.1. Pengujian Rasio Dataset

Dalam pengujian rasio pembagian data, dataset dibagi menjadi tiga bagian secara terstruktur. Bagian pertama dialokasikan sebagai data pelatihan (training) yang digunakan untuk membangun dan melatih model dalam mengenali objek. Bagian kedua ditetapkan sebagai data validasi, yang berfungsi untuk mengevaluasi performa model selama proses pelatihan serta membantu dalam penyesuaian parameter agar model tidak mengalami overfitting. Bagian ketiga digunakan sebagai data pengujian (testing) untuk mengukur akurasi dan generalisasi model terhadap data yang belum pernah dipelajari sebelumnya. Pembagian ini memastikan bahwa setiap proses pelatihan, validasi, dan pengujian berjalan secara seimbang dan objektif.</td><td>24 - 25</td><td>27</td></tr>
<tr><td>2 E2</td><td>Harga barang pada Rencana Anggaran Biaya (RAB) disarankan untuk dihapus.</td><td>Tabel 6. Alat dan Bahan
<table><tr><th>No.</th><th>Alat/bahan</th><th>Jumlah</th></tr><tr><td>1</td><td>Laptop</td><td>1</td></tr><tr><td>2</td><td>Tiandy Bullet Camera TC-C34QN</td><td>1</td></tr><tr><td>3</td><td>Kabel LAN UTP RJ45 CAT 6 (10 Meter)</td><td>1</td></tr><tr><td>4</td><td>PoE Switch Zitech 6 Port Switch</td><td>1</td></tr><tr><td>5</td><td>Vention USB to LAN RJ45</td><td>1</td></tr></table></td><td>23</td><td>24</td></tr>
<tr><td colspan="5">Bab 4</td></tr>
<tr><td colspan="5">Bab 5</td></tr>
<tr><td>4 E1</td><td>Saran harus didasarkan pada hasil pengujian yang telah dilakukan. Hindari memberikan saran yang tidak relevan dengan pengujian yang telah dilaksanakan.</td><td>5.2. Saran

1. Pengaturan konfigurasi max_batches lebih tinggi dapat dipertimbangkan, karena berdasarkan pengujian menunjukkan adanya peningkatan signifikan pada nilai precision, recall, F1-Score sehingga berpotensi meningkatkan performa model secara keseluruhan.
2. Disarankan untuk mengeksplorasi lebih banyak parameter untuk di uji, seperti momentum, decay, subdivisions yang berpotensi mempengaruhi proses pembelajaran model.
3. Penelitian ini masih manual tuning disarankan pada penelitian selanjutnya menerapkan teknik hyperparameter tuning otomatis seperti Bayesian Optimization, Random Search, atau Genetic Algorithm untuk menemukan kombinasi lebih optimal.</td><td>59</td><td>62</td></tr>
<tr><td colspan="5">Lampiran</td></tr>
</table>

Batam, 22 Juli 2024

(Widya Pratiwi)

Analisa Kinerja Sistem YOLOv4-Tiny Pada Pembuatan Alat Deteksi Kecurangan Jual Beli Gas LPG 3 kg

Tugas Akhir

Oleh:
Widya Pratiwi (4212111018)

Program Studi Teknik Mekatronika
Jurusan Teknik Elektro
Politeknik Negeri Batam
2025

Pernyataan Keaslian Tugas Akhir

Saya yang bertandatangan di bawah ini menyatakan bahwa isi sebagian maupun keseluruhan Tugas Akhir saya yang berjudul : "Analisa Sistem YOLOv4-Tiny Pada Pembuatan Alat Deteksi Kecurangan Jual Beli Gas LPG 3 kg" adalah **hasil karya sendiri**, **diselesaikan tanpa menggunakan bahan-bahan yang tidak diizinkan**, dan **bukan merupakan karya pihak lain yang saya akui sebagai karya sendiri.** Semua referensi yang dikutip atau dirujuk telah ditulis secara lengkap pada daftar `pustaka. Apabila ternyata pernyataan saya ini tidak benar, saya bersedia menerima sanksi sesuai peraturan yang berlaku.

Batam, 06 Juni 2025

Widya Pratiwi
NIM: 4212111018

Lembar Pengesahan

Tugas Akhir disusun untuk memenuhi salah satu syarat memperoleh gelar
Sarjana Terapan Teknik (S.Tr.T)
di
Politeknik Negeri Batam

Oleh:
Widya Pratiwi (4212111018)

Tanggal Sidang: 18 Juli, 2025

Disetujui oleh :

1. Rifqi Amalya Fatekha, S.ST., M.Tr.T
NIK: 1131017

1. Dr. Budi Sugandi, S.T., M.Eng
NIK: 100002

2. Ridwan, S.ST., M.Tr.T
NIK: 113113

Analisa Kinerja Sistem YOLOv4-Tiny Pada Pembuatan Alat Deteksi Kecurangan Jual Beli Gas LPG 3 kg

Abstrak

Pangkalan gas LPG 3 kg memainkan peran penting dalam memenuhi kebutuhan gas domestik masyarakat. Namun, pelaku usaha pangkalan gas LPG sering menghadapi tantangan, seperti kecurangan konsumen yang berdampak pada kesalahan pencatatan dan kerugian finansial. Salah satu permasalahan yang umum terjadi adalah ketika konsumen membawa satu tabung gas kosong tetapi mengambil dua tabung isi. Untuk mengatasi masalah ini dan meningkatkan efisiensi operasional, diperlukan sistem deteksi visual yang andal. Penelitian ini mengembangkan sistem deteksi visual menggunakan metode *You Only Look Once* (YOLO) untuk mendeteksi gas yang dibawa oleh konsumen pada pangkalan gas LPG 3 kg, metode YOLO yang digunakan adalah YOLOv4-Tiny yang dipilih karena keunggulannya dalam kecepatan deteksi dan efisiensi komputasi, yang kemudian dioptimalkan melalui penyesuaian *hyperparameter*, seperti *learning rate*, *batch size*, serta rasio pembagian dataset. Berdasarkan hasil pengujian, model terbaik diperoleh dengan kombinasi *learning rate* 0,012 dan kombinasi *batch size* 64 yang menghasilkan akurasi deteksi sebesar 81,08%, dengan nilai *precision* 84%, *recall* 87%, F1-*Score* 86%, serta mAP@0.50 mencapai 91,70% selain itu, rata-rata waktu deteksi pada pengolahan citra sebesar 81.07 ms dan rata-rata kecepatan pemrosesan video mencapai 18,51 FPS.

Kata kunci: Deteksi objek, YOLOv4-*Tiny*, Optimasi Parameter

Performance Analysis of the YOLOv4-Tiny System in Making Fraud Detection Tools for Buying and Selling LPG 3 kg

Abstract

LPG 3 kg base plays a crucial role in fulfilling the domestic gas needs of the community. However, operators often face challenges such as consumer fraud, which can lead to recording errors and financial losses. For example, when a consumer brings in one empty gas cylinder but takes two filled ones. To address this problem and improve operational efficiency, a reliable visual detection system is required. This study develops a visual detection system using the You Only Look Once (YOLO) method to detect gas cylinders carried by consumers at a 3 kg LPG distribution point. The chosen method, YOLOv4-Tiny, was selected for its advantages in speed and computational efficiency. The system was further optimized by tuning hyperparameters, including learning rate, batch size, and dataset split ratio. Based on testing results, the best model was obtained using a learning rate of 0,012 and a batch size of 64, achieving a detection accuracy of 81,08%, with a precision of 84%, recall of 87% F1-Score of 86%, and mAP@0.50 of 91,70%. Additionally, the average image processing detection rate reached 81.07 ms *with an average video processing speed of 18,51 FPS.*

Keywords: Object Detection, YOLOv4-Tiny, Parameter Optimization

Kata Pengantar

Alhamdulillah, segala puji dan Syukur penulis panjatkan kepada Allah SWT atas nikmat ilmu, kekuatan, dan ketenangan hati yang telah dianugerahkan selama proses penyusunan tugas akhir ini. Tanpa campur tangan-Nya, barangkali lembar demi lembar ini hanya akan menjadi niat yang tak terselesaikan. Shalawat dan salam tak lupa penulis haturkan kepada Nabi Muhammad Shallallahu 'Alaihi Wasallam, yang telah menanamkan nilai-nilai pengasuhan yang menghidupkan jiwa manusia hingga hari ini.

Tugas Akhir yang berjudul "Analisa Kinerja Sistem YOLOv4-Tiny Pada Pembuatan Alat Deteksi Kecurangan Jual Beli Gas LPG 3 kg" ini berdasar dari keprihatinan penulis terhadap praktik kecurangan di pangkalan gas yang kerap dianggap sepele, namun berdampak nyata bagi pelaku usaha. Melalui pendekatan teknologi *Computer Vision* berbasi YOLOv4-Tiny, penulis berupaya merancang sebuah sistem sederhana yang diharapkan dapat memberikan kontribusi nyata dalam meningkat keadilan dan efiensi di pangkalan gas.

Dalam proses penyusunan tugas akhir ini, penulis tidak berjalan sendiri. Oleh karena itu, dengan penuh rasa hormat dan tulus dari hati, penulis ingin menyampaikan terima kasih sebesar-besarnya kepada:

1. Kedua orang tua dan keluarga yang telah memberikan do'a dan dukungannya untuk menyelesaikan tugas akhir.
2. Rekan seperjuangan yang menjadi teman bertukar pikiran di tengah proses penyelesaian tugas akhir.
3. Bapak Ir. Bambang Hendrawan, ST., MSM., CIPMP., CISCP. Selaku Direktur Politeknik Negeri Batam.
4. Bapak Indra Mulyadi, S.T., M.Eng. Selaku Ketua Jurusan Teknik Elektro Politek Negeri Batam.
5. Bapak Diono, S.Tr.T, M.Sc. Selaku ketua Program Studi Teknik Mekatronika dan Wali Dosen
6. Bapak Dr. Budi Sugandi, S.T., M.Eng. Selaku Dosen Pembimbing Tugas Akhir
7. Bapak Muhammad Prihadi Eko Wahyudi,S.T., M.T. Selaku Dosen Pembimbing Seminar Proposal
8. Ibu Nadrah Wivianus, S.Si., M.Si. Selaku Dosen Pengampu Mata Kuliah Tugas Akhir
9. Bapak Rifqi Amalya Fatekha, S.ST., M.Tr.T. dan Bapak Ridwan, S.ST., M.Tr.T. Selaku Dosen Penguji Tugas Akhir
10. Seluruh Dosen-Dosen Teknik Mekatronika Politeknik Negeri Batam
11. Teman seperjuangan Teknik Mekatronika Angkatan 2021 yang memberikan dukungan selama menyelesaikan tugas akhir.

Penulis menyadari bahwa tugas akhir ini masih jauh dari sempurna. Namun penulis berharap ini tidak hanya berhenti sebagai dokumen akademik, melainkan dapat menjadi pijakan bagi penelitian lanjutan dan penerapan di masyarakat. Akhir kata, semoga tugas akhir ini bermanfaat dan menjadi amal kebaikan yang diridhai oleh Allah SWT. Aamiin.

Batam, 06 Juni 2025

Widya Pratiwi
NIM: 4212111018

Daftar Isi

Daftar Gambar

Daftar Tabel

Bab 1. Pendahuluan

1.1. Latar Belakang

Pangkalan gas LPG 3 kg menjadi salah satu tempat vital bagi masyarakat dalam memenuhi kebutuhan gas domestik. Dengan adanya subsidi pemerintah, gas LPG 3 kg membantu untuk memenuhi kebutuhan energi sehari-hari bagi masyarakat menengah, sehingga gas LPG 3 kg harganya menjadi lebih terjangkau. Distribusi dan penjualan gas LPG 3 kg memainkan peran penting dalam menyuplai ketersediaan gas LPG 3 kg untuk masyarakat, terutama pada rumah tangga dan UMKM yang menggunakan LPG 3 kg sebagai bahan bakar utama untuk keperluan dapur atau produksi skala kecil. Namun, dalam proses pendistribusiannya, terdapat tantangan yang sering dihadapi oleh pelaku usaha pangkalan gas LPG 3 kg seperti tindakan kecurangan yang dilakukan oleh konsumen untuk mendapatkan keuntungan, misalnya dengan memanipulasi jumlah atau kondisi tabung gas. Kondisi ini dapat menyebabkan potensi kehilangan stok tanpa disadari, yang dapat menyebabkan kesalahan pencatatan atau bahkan kerugian finansial bagi pelaku usaha. Potensi kehilangan tabung gas LPG 3 kg harus dapat dideteksi sejak dini untuk memantau setiap tabung gas yang masuk dan tabung gas yang keluar sehingga dapat meminimalisir terjadinya kesalahan atau manipulasi yang tidak terdeteksi.

Untuk itu penelitian ini memberikan solusi sebuah sistem deteksi yang dapat diterapkan untuk mendeteksi tabung gas LPG 3 kg. Fokus utamanya adalah pada optimasi kinerja deteksi melalui pengujian metode dan konfigurasi tertentu untuk diterapkan pada alat deteksi kecurangan.

Metode *You Only Look Once* (YOLO) menjadi pilihan model yang digunakan dalam penelitian ini. Penelitian terdahulu memperlihatkan keberhasilan YOLO dalam segi kecepatan deteksi, YOLO berhasil mencapai kecepatan deteksi hingga lebih dari 30 FPS dalam model dengan ukuran input 320 x 320 ketika dijalankan menggunakan CPU. Metode ini cocok untuk mendeteksi semua jenis rambu lalu lintas. Namun, hasil dari penelitian ini belum sepenuhnya memaksimalkan kinerja dari YOLO. Hal ini disebabkan oleh kurangnya variasi dalam dataset yang digunakan untuk mengoptimalkan kinerja dari metode YOLO [1]. Selain penelitian tersebut pemilihan YOLO dalam judul jurnal "Sistem Pemantauan Aktivitas Keseharian Lansia Berbasis Deteksi Objek Menggunakan Algoritma YOLO" yang menghasilkan performa model kinerja deteksi objek dengan nilai *Precision* sebesar 100%, *Recall* sebesar 100%, *F1-Score* sebesar 100%, *Average IoU* sebesar 87.26%, *Average Loss* sebesar 6.41%, dan *mAP* sebesar 100%. Performa yang optimal tersebut diperoleh melalui pengujian *hyperparameter* dengan konfigurasi rasio 90% : 10%, *Batchsize* sebesar 64, *Learning rate* sebesar 0.008, dan *Max batches* sebesar 4000 [2]. Hasil ini menunjukkan bahwa model memiliki akurasi yang

sangat tinggi dan menunjukkan keberhasilan pengaturan *hyperparameter* yang tepat dalam meningkatkan kinerja deteksi objek pada sistem monitoring lansia.

Berdasarkan dari penelitian sebelumnya, terlihat bahwa YOLO memiliki potensi yang besar dalam berbagai aplikasi deteksi objek, kinerja model dapat lebih ditingkatkan dengan variasi dataset dan optimasi pengaturan *hyperparameter* yang tepat. Oleh karena itu, penelitian ini mengajukan judul "Analisa Kinerja Sistem YOLOv4-Tiny Pada Pembuatan Alat Deteksi Kecurangan Jual Beli Gas LPG 3 kg". Penelitian ini diharapkan mampu menghasilkan konfigurasi terbaik yang dapat meningkatkan akurasi deteksi sistem YOLOv4-*Tiny* dalam mengenali tabung gas LPG 3 kg, sekaligus mengoptimalkan performa untuk mendukung deteksi kecurangan.

1.2. Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan di atas, maka didapatkan rumusan masalah sebagai berikut:

1. Bagaimana cara meningkatkan kinerja model *deep learning* pada alat deteksi gas LPG 3 kg?
2. Bagaimana pengaruh pengaturan konfigurasi *hyperparameter* terhadap kinerja sistem alat deteksi kecurangan jual beli gas LPG 3 kg?
3. Bagaimana menentukan konfigurasi *hyperparameter* yang paling optimal untuk mencapai performa kinerja deteksi terbaik pada alat deteksi kecurangan jual beli gas LPG 3 kg?

1.3. Tujuan

Berdasarkan rumusan masalah yang telah dijabarkan di atas, maka didapatkan tujuan sebagai berikut:

1. Melakukan analisis terhadap faktor-faktor yang mempengaruhi kinerja, seperti rasio dataset dan pengaturan model, untuk meningkatkan kinerja model *deep learning* pada alat deteksi gas LPG 3 kg.
2. Mengevaluasi pengaruh pengaturan konfigurasi *hyperparameter* terhadap kinerja sistem deteksi untuk memahami hubungan antara pengaturan tersebut terhadap akurasi deteksi dan kestabilan deteksi.
3. Mengidentifikasi konfigurasi *hyperparameter* yang paling optimal untuk mencapai performa kinerja deteksi terbaik pada alat deteksi kecurangan jual beli gas LPG 3 kg.

1.4. Manfaat

Manfaat dari penelitian ini adalah memberikan pengembangan sistem deteksi objek menggunakan YOLOv4-*Tiny.* Penelitian ini diharapkan dapat menghasilkan konfigurasi *hyperparameter* yang optimal untuk memaksimalkan performa deteksi sehingga mampu meminimalisir kesalahan dan manipulasi yang tidak terdeteksi dalam transaksi jual beli gas di pangkalan. Selain itu, penelitian ini juga dapat menjadi referensi untuk pengembangan sistem deteksi *real time* yang efisien.

1.5. Batasan

Batasan masalah dalam pelaksanaan penelitian ini adalah sebagi berikut:

1. Penelitian ini menggunakan metode YOLOv4-*Tiny* untuk mendeteksi jumlah tabung gas LPG 3 kg dan hanya berfokus pada optimasi *hyperparameter* untuk meningkatkan performa deteksi objek.
2. Dataset yang digunakan merupakan *custom* dataset dengan satu kelas objek, yaitu gas.
3. Pengujian dilakukan menggunakan *Google Colaboratory* (berbasis *cloud*) karena keterbatasan perangkat keras lokal.
4. Optimasi *hyperparameter* yang dilakukan mencakup pemilihan rasio pembagian data, *learning* rate, dan *batch size* hanya diterapkan pada model YOLOv4-*Tiny*.
5. Penelitian ini hanya melakukan pengujian kinerja model menggunakan data berupa gambar statis dan video rekaman, serta tidak mencakup pengujian menggunakan *realtime* langsung dari kamera IP (CCTV).

1.6. *Work Breakdown Structure*

Berikut terlampir *Work Breakdown Structure* (WBS) pelaksanaan TA ini:

Tabel 1. *Work Breakdown Structure* (WBS)

No	Nama	Tugas dan Tanggung Jawab dalam Tim
1	Mohamad Romi Fakhri Wildandi	1. Mendesain logika *counting*. 2. Pengembangan program *counting* berbasis pelacakan objek (*DeepSORT*). 3. Melakukan pengujian dan analisa keandalan dengan berbagai kondisi lingkungan.

No	Nama	Tugas dan Tanggung Jawab dalam Tim
2	Widya Pratiwi	1. Melatih dan mengoptimalkan model YOLOv4-*Tiny* melalui optimasi *hyperparameter*. 2. Melakukan pengujian performa model berdasarkan metriks evaluasi. 3. Menganalisis pengaruh parameter terhadap performa model dan menentukan konfigurasi terbaik untuk sistem deteksi.

Bab 2. Tinjauan Pustaka

2.1. Penelitian Terkait

Tabel 2. Penelitian Terkait

No	Tahun	Penulis/tahun	Jurnal	Metode	Kelemahan
1	2023	Raihan Digo Saputra	Pengembangan Sistem Deteksi Objek Pada Produk Retail dengan Arsitektur YOLOV4-Tiny	YOLOv4-Tiny	Menggunakan dataset dengan resolusi rendah sehingga mempengaruhi performa yang dihasilkan dari model.
2	2023	Muhamad Addin Al Haadi, Casi Setianingsih, Tito Waluyo Purboyo	Sistem Pemantauan Aktivitas Keseharian Lansia Berbasis Deteksi Objek Menggunakan Algoritma YOLO	YOLOv4	Keterbatasan perangkat keras saat penelitian, sistem belum mendukung multi kamera, FPS tidak stabil.
3	2024	Hadi Supriyanto, Sarosa Castrena Abadi, Aliffa Shalsabilah	Deteksi Helm Keselamatan Menggunakan Jetson Nano dan YOLOv7	YOLOv7	FPS rata-rata rendah hanya 5,723 FPS, Algoritma deteksi objek belum optimal, Akurasi dan kecepatan deteksi masih kurang optimal.
4	2023	Edy Salim, Suharjito	*Hyperparameter optimization of YOLOv4 tiny for palm oil fresh fruit bunches maturity detection using genetics algorithms*	YOLOv4-Tiny	Peningkatan mAP terbatas hanya 0,1% dan 0,5%, penelitian hanya fokus pada optimasi *learning rate*, belum mencoba algoritma optimasi lain.
5	2024	Yunus Fadhillah, Budi Berlinton, Supriyanto Karya, Samin	*Advancing Vehicle Detection with YOLOv9: Integrating Programmable Gradient Information and Efficient Layer Aggregation*	YOLOv9	FPS masih kurang cukup untuk *game multiplayer*, sistem deteksi masih kurang akurat untuk mendeteksi objek dengan ukuran kecil atau posisi yang jauh.

No	Tahun	Penulis/tahun	Jurnal	Metode	Kelemahan
6	2023	Dany Alfiyandi Abdurrafi, M Taqijjuddin Alawiy, dan Bambang Minto Basuki	Deteksi Klasifikasi dan Menghitung Kendaraan Berbasis Algoritma You Only Look Once (YOLO) Menggunakan Kamera CCTV	YOLO	Hasil deteksi *real time* masih belum ditampilkan secara *User-friendly.*
7	2021	Lusiana Rahma, Hadi Syaputra, A. Haidar Mirza, Susan Dian Purnamasari	Objek Deteksi Makanan Khas Palembang Menggunakan Algoritma YOLO (*You Only Look Once*)	YOLO	Belum diterapkan secara *real time*.
8	2023	Nadia Khairunnisa	Detektor angka *real time* berbasis mikrokontroler esp32 cam dengan pengolahan data menggunakan aloritma yolo	YOLOv4-Tiny	Dataset kurang bervariasi sehingga akurasi pada sistem deteksi kurang akurat.
9	2022	Muhammad Farishanif Widyono, Fauzan Fadhlurrahman Pratama	Rancang Bangun Sistem Live Stream Video dan Pengenalan Penyakit Padi	YOLO	Nilai akurasi yang rendah karena kualitas gambar kurang baik, memiliki nilai FPS yang sangat kecil.
10	2024	M.Rizky Pramana, Haida Dafitri, Sumi Khairani	Sistem Deteksi Jenis Kendaraan Metode YOLOv4 Untuk Mendukung Transportasi Cerdas Kota Medan	YOLOv4	Proses pelabelan data kelas kendaraan yang tidak konsisten sehingga terjadi kesalahan deteksi.

Berdasarkan penelitian terkait di atas, terdapat beberapa sistem deteksi objek yang telah dikembangkan dengan berbagai pendekatan dan metode. Oleh karena itu penelitian ini fokus pada analisis kinerja model YOLOv4-*Tiny* menggunakan gambar statis dan video rekaman sebagai langkah awal dalam pengujian performa model.

2.2. Computer Vision

Computer vision merupakan kemampuan dalam memahami dan menginterprestasikan informasi visual. Bagi manusia, persepsi visual terjadi melalui pengamatan pola, objek, dan lingkungan yang ditangkap oleh mata. Sistem *computer vision* dirancang untuk meniru kemampuan visual manusia, yang melibatkan dua komponen utama yaitu *sensing device* untuk meniru fungsi mata dan *interpreting device* untuk meniru fungsi otak [3]. Oleh karena itu, tujuan *computer vision* bukan hanya dapat melihat, tetapi juga untuk menganalisis dan menghasilkan informasi yang berguna dari pengamatan tersebut sehingga menjadikan komputer untuk melakukan tugas-tugas yang serupa manusia dengan efisiensi yang tinggi [4].

Para ilmuwan terinspirasi dari cara kerja otak manusia dan mengembangkan model yang meniru sistem saraf otak yang dikenal sebagai *Artificial Neural Network* (ANN). ANN merupakan model matematis yang terdiri dari serangkaian "neuron" atau "node" yang terhubung satu sama lain dan dapat belajar dari data. Melalui ANN, computer mampu belajar mengenai pola, objek, dan fenomena kompleks dalam data visual, mirip dengan cara kerja otak manusia memproses informasi. Ini mendukung perkembangan sistem komputer yang semakin cerdas dan dapat menangani tugas visual yang semakin rumit.

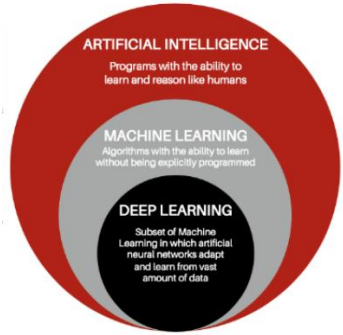
Gambar 1. *Computer Vision Works*

2.3. *Object Detection*

Object detection adalah prosedur dalam bidang *computer vision* yang berfungsi untuk mengidentifikasi kelas dari setiap objek dan memperkirakan lokasi objek dengan memberikan *bounding box* disekeliling objek [5]. Proses ini menggunakan citra sebagai *input* yang memiliki satu objek atau lebih lalu menghasilkan *output* berupa prediksi lokasi objek melalui *bounding box* serta klasifikasi objek pada setiap *bounding box* [6]. *Object detection* membantu untuk menemukan lokasi dan keberadaan objek dalam suatu ruangan tertentu dan dibagi menjadi dua jenis deteksi, yaitu *soft detection* dan *hard detection*. Pada *soft detection*, objek dikenali tanpa detail spesifik tentang lokasinya, sedangkan *hard detection* mengidentifikasi objek beserta informasi lokasinya secara detail [7]. Beberapa model *object detection* yang memiliki kinerja yang baik yaitu SSD (*Single Shot MultiBox Detector*), Faster R-CNN (*Region-based Convolutional Neural Network*), dan YOLO (*You Only Look Once*). SSD dikenal dengan kecepatan inferensinya dan kemampuan mendeteksi objek dalam berbagai ukuran, Faster R-CNN unggul dalam akurasi lokalisasi objek, sedangkan YOLO unggul dengan kemampuan deteksi objek *real time* serta waktu inferensi yang cepat [8].

2.4. *Deep Learning*

Deep learning adalah teknik dalam *Artificial Intelligence* (AI), yaitu kecerdasan buatan yang diterapkan pada mesin agar dapat memahami informasi sebagaimana manusia lakukan [3]. Gambar 2 menunjukkan bahwa *deep learning* merupakan sub bidang khusus dari *Machine Learning* (ML) yang berfokus pada metode pembelajaran representasi data melalui susunan lapisan (l*ayer*). *Deep learning* sering melibatkan puluhan hingga ratusan l*ayer* berurutan yang dilatih secara otomatis berdasarkan data pelatihan.



Gambar 2. Hubungan antara *Deep Learning* dengan *Machine Learning*

Deep learning terinspirasi dari cara kerja sel-sel otak manusia yang disebut neuron. Konsep ini melahirkan *Neural Network* (NN), yaitu algoritma dalam *deep learning* yang memanfaatkan struktur mirip neuron untuk menyelesaikan masalah-masalah kompleks. *Neural network* memungkinkan model *deep learning* dalam mengenali pola rumit dalam data visual, teks, suara, dan jenis data lainnya, sehingga dapat menghasilkan prediksi yang akurat [9]

Gambar 3. *Deep Learning Flow*

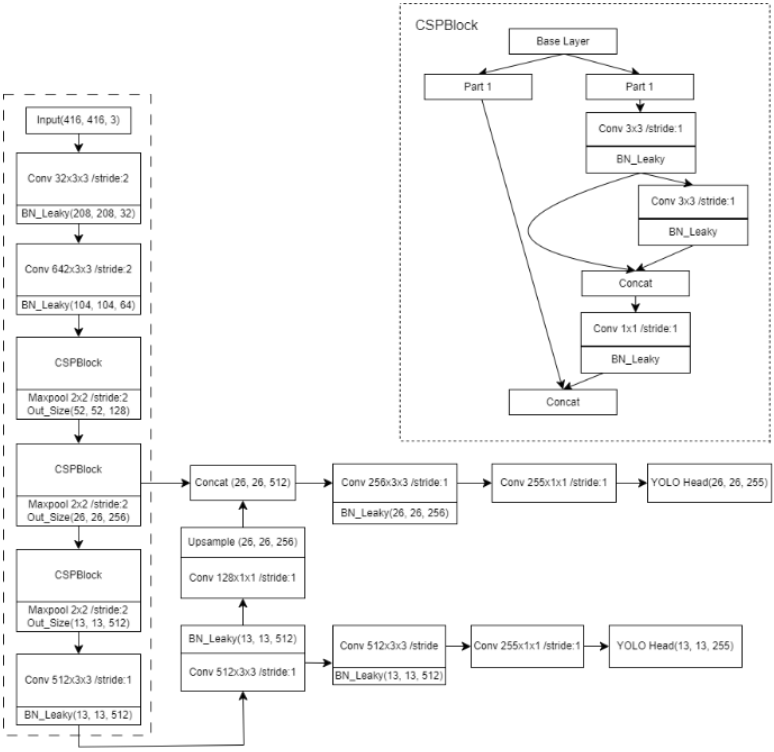
2.5. YOLOv4-Tiny

YOLO (*You Only Look Once*) adalah model deteksi objek berbasis *deep learning* yang pertama kali dikembangkan oleh Joseph Redmon. Versi YOLOv4 dikembangkan oleh tiga penulis, yaitu Alexey Bochkovskiy, Chien-Yao Wang, dan Hong-Yuan Mark Liao, menawarkan peningkatan signifikan dibandingan dengan YOLOv3, dengan kenaikan *Average Precision* (AP) sebesar 100% dan peningkatan *Frame per Second* (FPS) sebesar 12% [3].

YOLOv4-Tiny adalah versi kompresi dari YOLOv4, dirancang dengan struktur jaringan yang lebih sederhana dan parameter yang lebih sedikit sehingga dapat diimplementasikan pada perangkat seluler dan *embedded device* . YOLOv4 sendiri menunjukkan akurasi yang lebih baik daripada YOLOv3, sehingga penggunaan YOLOv4-*Tiny* memberikan keunggulan akurasi dibandingkan dengan YOLOv3-*Tiny* [10].

Kecepatan YOLOv4-*Tiny* dalam *Frame per Second* (FPS) mencapai delapan kali lipat dari YOLOv4, meskipun akurasinya sekitar 2/3 dari YOLOv4 saat diuji pada dataset MS COCO. YOLOv4-*Tiny* merupakan pilihan yang tepat karena memiliki waktu inferensi yang lebih cepat [11]. Sistem yang dikembangkan saat ini menjadi pilihan yang tepat karena kecepatan inferensi yang tinggi memungkinkan proses analisis video berjalan lebih cepat dan efisien. Dengan pengurangan jumlah lapisan konvolusi dari 137 lapisan menjadi 29 lapisan, YOLOv4-*Tiny* mampu memproses lebih banyak *frame* per detik (FPS), sehingga mempercepat hasil deteksi pada rekaman video. Arsitektur YOLOv4-*Tiny* Menggunakan *CSPDarknet53-Tiny* sebagai *backbone* yang memanfaatkan CSPBlock untuk membagi *feature map* menjadi dua bagian dan menggabungkannya kembali melalui *cross-stage residual edge*. Penggunaan CSPBlock terbukti meningkatkan kemampuan pembelajaran jaringan konvolusi. Pada bagian *neck*, arsitektur ini mengadopsi struktur FPN yang mengintegrasikan fitur dari berbagai skala. Di bagian *head*, arsitektur menghasilkan dua skala prediksi, yaitu 26x26 dan 13x13,

yang diperoleh dari *feature map* yang dihasilkan FPN. Proses prediksi YOLOv4-*Tiny* ini serupa dengan yang ada pada YOLOv4 [12].



Gambar 4. Arsitektur YOLOv4-*Tiny*

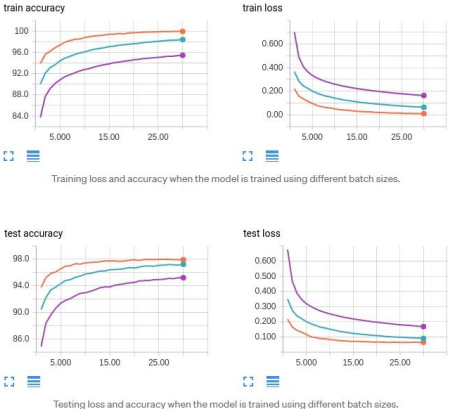
2.6. Optimasi *Hyperparameter*

Hyperparameter adalah parameter dalam model *machine learning* dan *deep learning* yang ditetapkan sebelum proses pelatihan dimulai. *Hyperparameter* berfungsi untuk mengatur konfigurasi model *neural network* agar dapat mencapai hasil yang optimal. Proses pengaturan ini dikenal dengan istilah *hyperparameter tuning* dan merupakan faktor penting dalam membangun model *deep learning* yang efisien dan efektif. Optimasi *hyperparameter* adalah proses mengatur *learning rate*, *batch size,* dan lain-lain. *Hyperparameter* ini perlu disesuaikan selama proses pelatihan guna memperoleh performa model yang optimal. Adapun *hyperparameter* yang diatur sebagai berikut:

2.7. *Configuration Hyperparameter*

2.7.1. Batch Size

Batch size adalah variabel yang menentukan jumlah *image* atau data *training* yang dimasukkan dalam setiap iterasi *training*. Dengan *batch size* yang lebih kecil, proses *training* akan berjalan lebih cepat. Sebaliknya, *batch size* yang lebih besar membutuhkan waktu *training* yang lebih lama karena memerlukan kapasitas penyimpanan yang lebih besar [13]. Penggunaan *batch* kecil cenderung menghasilkan pembaruan bobot yang fluktuatif (*noise*), yang menyebabkan pelatihan kurang stabil tetapi kadang membantu model keluar dari *local* minima. Sementara itu, *batch size* yang besar dapat meningkatkan stabilitas model dan kualitas pembelajaran model karena mampu menjadikan model dalam menangkap pola dari kumpulan data yang lebih besar dalam satu kali proses, sehingga menghasilkan pembaruan bobot yang lebih akurat.



Gambar 5. Visualisasi Pengaruh Batch Size terhadap Akurasi dan Loss

Gambar 5 merupakan grafik perbandingan akurasi dan *loss* selama proses pelatihan dan pengujian model berdasarkan variasi *batch size*. Sumbu x menunjukkan jumlah *maximum batch steps*, sedangkan sumbu y menunjukkan nilai akurasi dan *loss*.

Keterangan:

- Grafik *Train Accuracy* (Kanan Atas) menunjukkan peningkatan akurasi selama proses pelatihan untuk berbagai ukuran *batch.* Terlihat bahwa ukuran *batch* tertentu memiliki akurasi yang lebih tinggi dan stabil daripada yang lain, menunjukan bahwa ukuran *batch* mempengaruhi kecepatan dan kualitas pembelajaran model.

- Grafik *Train Loss* (Kiri atas) menunjukkan penurunan nilai *loss* selama pelatihan. Performa model lebih baik dengan nilai kehilangan yang lebih kecil. Ukuran *batch* yang lebih optimal menunjukkan penurunan kehilangan yang lebih cepat dan stabil.
- Grafik *Test Accuracy* (Kiri bawah) menunjukkan akurasi model pada data pengujian yang tidak digunakan saat pelatihan. Akurasi yang tinggi mengindikasikan bahwa model memiliki kemampuan generalisasi daripada hanya menghafal data latih.
- Grafik *Test Loss* (Bawah kanan) menunjukkan nilai *loss* pada data pengujian, jika *test loss* meningkat saat *train loss* menurun merupakan indikasi bahwa model *overfitting*, yaitu saat model terlalu menyesuaikan dengan data latih dan tidak dapat memahami pola baru pada data uji.

2.7.2. *Max Batches*

Max batches adalah jumlah iterasi selama proses pelatihan data. Semakin besar nilai *Max batches*, maka semakin banyak sistem mempelajari data pelatihan. Jumlah data pelatihan tidak boleh melebihi *max batches* dan nilainya harus disesuaikan dengan jumlah kelas objek yang akan dideteksi [13]. YOLO mempunyai rekomendasi pengaturan nilai *Max batches* berdasarkan jumlah kelas (jumlah kategori objek yang dideteksi:

$$\textit{Max Batches} = \text{Jumlah kelas} \times 2000 \quad (1)$$

2.7.3. *Learning Rate*

Learning rate adalah parameter yang nilainya berpengaruh pada kecepatan dan stabilitas pelatihan. *Learning rate* yang rendah dapat memperlambat proses pelatihan, tetapi meningkatkan peluang model untuk mencapai bobot yang lebih optimal. Sebaliknya, *learning rate* yang tinggi dapat mempercepat pelatihan, namun beresiko mengurangi akurasi dalam menentukan bobot yang optimal [14]. Dalam pelatihan model *deep learning*, nilai *learning rate* yang ideal tidak bersifat tetap, melainkan sangat bergantung pada jenis permasalahan, karakteristik dataset, arsitektur model, serta algoritma optimasi yang digunakan. Oleh karena itu, penentuan *learning rate* optimal memerlukan eksperimen sebagai bagian dari proses *hyperparameter tuning*. Dalam penelitian ini, penentuan nilai *learning rate* dilakukan melalui serangkaian pengujian dengan pendekatan sistematis untuk mengetahui pengaruhnya terhadap performa deteksi model YOLOv4-*Tiny*.

2.8. Parameter Performansi

2.8.1. Akurasi

Akurasi mengukur persentase prediksi yang benar terhadap keseluruhan prediksi yang dilakukan model. Metriks ini memberikan gambaran keseluruhan tentang kemampuan model dalam mengidentifikasi kelas dengan tepat [15]. Nilai ini dapat dihitung dengan rumus:

$$Accuracy = \frac{True\ Positive\ (\mathrm{TP}) + True\ Negative\ (\mathrm{TN})}{TP + \mathrm{FP} + FN + TN} \qquad (2)$$

Keterangan:

- TP (*True Positive*) adalah model berhasil mendeteksi keberadaan objek dengan benar sesuai kenyataannya.
- TN (*True Negative*) adalah model tidak memberikan deteksi karena memang tidak ada objek.
- FP (*False Positive*) adalah model keliru memberikan deteksi objek, padahal tidak ada objek yang muncul.
- FN (*False Negative*) adalah model gagal mendeteksi objek yang sebenernya ada.

2.8.2. *Precision*

Precision merupakan salah satu metriks evaluasi performa model yang digunakan untuk mengukur tingkat ketepatan model dalam mengklasifikasikan data ke dalam kelas positif. Nilai presisi adalah rasio antara jumlah prediksi positif yang benar (*true positive*) dengan total prediksi positif yang dihasilkan oleh model. Presisi yang tinggi menunjukkan bahwa prediksi positif yang dihasilkan model sebagian besar benar (jumlah *False Positive* rendah) [16]. Nilai ini dapat dihitung menggunakan rumus sebagai berikut:

$$Precision = \frac{True\ Positive\ (\mathrm{TP})}{True\ Positive\ (\mathrm{TP}) + False\ Positive\ (FP)} \text{ x } 100\% \qquad (3)$$

Dalam penerapan deteksi objek menggunakan model YOLO, *precision* membantu mengevaluasi apakah model sering salah mendeteksi objek yang sebenernya tidak ada. Metriks ini menjadi penting terutama jika digunakan bersamaan dengan *recall* untuk memberikan penilaian yang lebih lebih menyeluruh terhadap kinerja model.

2.8.3. *Recall*

Recall adalah ukuran kemampuan model dalam mendeteksi semua kasus relevan (kotak pembatas yang sesuai dengan *ground truth*). Nilainya menunjukkan persentase prediksi positif yang benar terhadap seluruh *ground truth* yang tersedia.

$$Recall = \frac{True\ Positive\ (\mathrm{TP})}{True\ Positive\ (\mathrm{TP}) + False\ Negative\ (\mathrm{FN})} \text{ x } 100\% \qquad (4)$$

Recall yang tinggi menunjukkan model mampu mendeteksi Sebagian besar objek yang telah di latih dan belum tentu menunjukkan kinerja model yang ideal jika *precision* rendah, karena bisa jadi model mungkin menghasilkan banyak deteksi yang keliru. Oleh karena itu, *recall* biasanya dianalisis bersama dengan *precision* untuk mendapatkan keseimbangan antara tingkat keberhasilan dalam mendeteksi objek dan jumlah kesalahan prediksi yang dilakukan model.

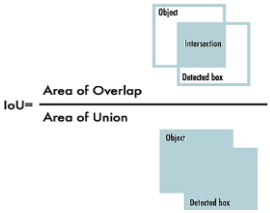
2.8.4. F1-Score

F1-*Score* merupakan rata-rata harmonik antara *presicion* dan *recall* yang memberikan keseimbangan antara kedua metriks tersebut. Metriks ini digunakan untuk memberikan evaluasi yang lebih menyeluruh terhadap performa model, khususnya saat menghadapi ketidakseimbangan antara kelas normal dan anomali [15]. Perhitungan F1-*Score* dilakukan menggunakan rumus berikut:

$$F1\text{-}Score = 2 \text{ x } \frac{Recall \text{ x } Precision}{Recall + Precision} \qquad (5)$$

2.8.5. Intersection over Union (IoU)

IoU adalah metriks yang digunakan untuk mengevaluasi seberapa baik *bounding box* (kotak pembatas) yang dideteksi oleh model mencocokkan objek sebenarnya (*ground truth*) di dalam gambar. IoU sering digunakan sebagai acuan nilai *threshold*. Terdapat dua nilai *threshold* yang umum dipakai yaitu 0,5 dan 0,75. [13].



Gambar 6. IoU

Intersection (*Area of Overlap*) adalah area yang menjadi irisan antara *bounding box* dideteksi (*detected box*) dan *ground truth* (*object*). Bagian ini menunjukkan wilayah yang benar-benar diprediksi dengan tepat oleh model. *Union* (*Area of Union*) adalah gabungan antara *bounding box* yang dideteksi dan *ground truth*.

2.8.6. Mean Average Precision (mAP)

mAP adalah metriks utama dalam deteksi objek yang mengukur seberapa baik model mendeteksi dan mengklasifikasikan objek. mAP merupakan kemampuan model untuk mendeteksi objek secara akurat tanpa menghasilkan banyak kesalahan. Semakin tinggi mAP, semakin baik model tersebut. Average Precision (AP) merupakan salah satu metode yang digunakan untuk menilai performa setiap model. AP menghitung rata-rata dari nilai precision pada berbagai nilai recall dan diukur dalam rentang 0 hingga 1[17].

2.8.7. Confidence Score

Confidence Score adalah nilai yang menampilkan tingkat keyakinan dari model terhadap keberadaan suatu objek pada Lokasi tertentu di dalam gambar. Perhitungan *confidence score* dapat dihitung menggunakan rumus berikut:

$$\textit{Confidence Score} = P(\textit{Object}) \times \text{IOU} \qquad (6)$$

Dimana, *P*(*Object*) adalah peluang keberadaan suatu objek di dalam *bounding box.*[18]. Semakin tinggi nilai *confidence*, semakin besar tingkat kepastian bahwa prediksi model sesuai dengan keberadaan objek yang sebenernya. Nilai *confidence* menentukan apakah suatu deteksi akan ditampilkan atau tidak. Dalam implementasi, hanya prediksi dengan *confidence score* dengan nilai ambang tertentu (*threshold*) yang di proses lebih lanjut. Hal ini bertujuan untuk menghindari deteksi palsu atau prediksi yang tidak meyakinkan. Oleh karena itu, *confidence score* tidak hanya mencerminkan keyakinan model, tetapi juga menjadi dasar dalam proses seleksi hasil deteksi.

2.8.8. Kecepatan

Pengukuran kecepatan merupakan indikator penting untuk mengevaluasi kinerja sebuah model. Kecepatan dapat menentukan kelayakan pada saat penerapan model di area deteksi. Kecepatan merujuk pada seberapa cepat model dapat mendeteksi objek. Untuk pengujian menggunakan video, kecepatan diukur dalam satuan *frame per second* (FPS), yaitu jumlah *frame* yang dapat diproses per detik. Sedangkan kecepatan saat deteksi objek pada sebuah citra adalah seberapa

lama waktu yang dibutuhkan untuk mendapatkan hasil deteksi (*mili-second*). Nilai FPS yang lebih tinggi menunjukkan bahwa model bekerja lebih cepat dalam memproses *frame* video. Salah satu faktor utama yang mempengaruhi kecepatan FPS adalah CPU dan GPU. GPU mampu meningkatkan performa deteksi karena memiliki arsitektur yang mendukung pemrosesan paralel secara optimal, berbeda dengan CPU yang lebih terbatas [3].

2.9. *Roboflow*

Roboflow adalah *platform* yang menyediakan berbagai alat dan layanan untuk mempermudah proses pelabelan data hingga pelatihan model. *Platform* ini mendukung integrasi dengan sejumlah *framework machine learning*, seperti *TensorFlow*, *PyTorch*, dan *Darknet*. *Roboflow* digunakan untuk melakukan anotasi gambar, memeriksa pelabelan, mengonversi format anotasi, melakukan pra-pemrosesan gambar, menerapkan teknik augmentasi yang beragam, serta mengekspor dataset dengan mudah ke format yang kompatibel untuk melanjutkan proses pelatihan model. Berikut ini adalah alur kerja untuk *machine vision* di *Roboflow*:

Gambar 7. Alur Kerja pada *Roboflow*

Platform *end-to-end* ini dimulai dengan mengunggah *raw* dataset lalu diolah oleh *Roboflow* agar data siap untuk dianotasi atau diberi *bounding box*. Setelah itu, data tersebut dapat digunakan sebagai *input* untuk melatih model. Model yang sudah terlatih dapat digunakan dan hasil deteksinya dapat ditampilkan [19].

2.10. *Google Colaboratory*

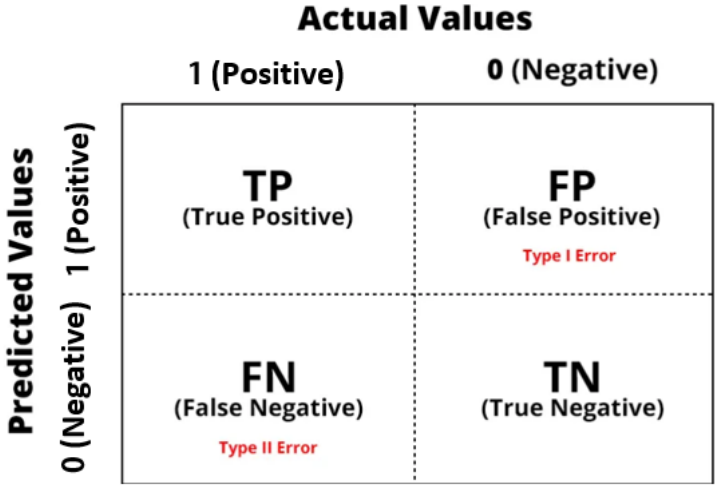
*Google Colaborator*y atau sering disebut "*Google Colab*" atau "*Colab*" adalah produk dari *Google Research* yang menyediakan layanan *jupyter notebook* terhosting tanpa memerlukan pengaturan khusus. *Colab* menawarkan akses gratis ke sumber daya komputasi, termasuk GPU untuk mempermudah pengguna dalam menjalakan kode yang dapat *executable code* dan *rich text* dalam satu dokumen, yang mendukung penambahan gambar, HTML, LaTex, dan lainnya. *Colab* menggunakan Bahasa Pemrograman *Python*, dengan dukungan untuk versi *Python* 3 dan hasil dari setiap *colab notebook* akan otomatis tersimpan di *google drive*

pengguna. *Colab* juga mendukung fitur pengguna berbagi *notebook* untuk kolaborasi bersama [5].

Colab menjalankan *jupyter notebook* yang terhubung ke *Virtual Machine* (VM) dengan batas waktu penggunaan maksimal 12 jam. Setelah waktu ini tercapai VM akan otomatis di reset oleh *Colab*. Batasan ini sering dihadapi saat pelatihan model yang membutuhkan waktu lama. Oleh karena itu, disarankan untuk secara berkala menyimpan *model's checkpoint* dengan berkala, proses pelatihan dapat dilanjutkan dari titik terakhir setelah *reset* terjadi.

2.11. Confusion Matrix

Confusion matrix adalah alat untuk mengevaluasi performa model dalam klasifikasi *machine learning. Confusion matrix* dikenal sebagai *error matrix* dimana sebuah tabel yang memberikan gambaran tentang perbandingan antara hasil klasifikasi yang dilakukan oleh sistem dengan hasil klasifikasi yang sebenarnya [20]. Gambar di bawah ini menunjukkan contoh *confusion matrix* yang terdiri dari empat kombinasi nilai prediksi:



Gambar 8. *Confusion Matrix*

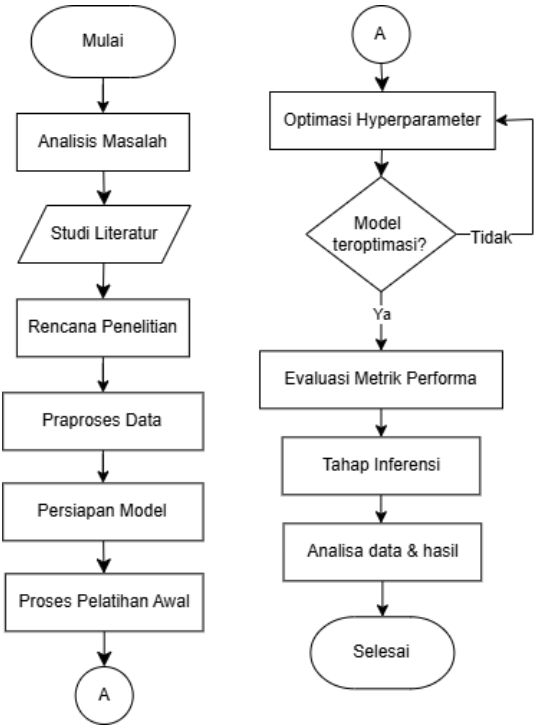
Terdapat empat representasi hasil proses klasifikasi pada *confusion matrix,* yaitu *True Positive* (TP), *True Negative* (TN), *False Positive* (FP) – *Type I Error*, dan *False Negative* (FN) – *Type II Error*. TP merupakan hasil yang dideteksi benar, sebagai contoh sistem mendeteksi benar sebagai gas. FP merupakan klasifikasi

positif tetapi kenyataannya negatif, sebagai contoh sistem mendeteksi adanya keberadaan gas kenyataannya tidak ada gas. TN merupakan hasil klasifikasi negatif dan prediksi tersebut benar negatif, sebagai contoh sistem dengan benar mengidentifikasi bahwa tidak ada gas. FN merupakan hasil deteksi negatif tetapi kenyataan positif, sebagai contoh sistem tidak mendeteksi keberadaan gas padahal sebenernya ada gas.

Bab 3. Metodologi Penelitian / Metode Pelaksanaan

3.1. Perancangan Pelaksanaan Penelitian

Gambar 9 merupakan tahapan metode pelaksanaan pada sistem. Diagram penelitian ini digunakan untuk menjelaskan proses penelitian yang dijalankan mulai dari menganalisis masalah sampai dengan analisa data dan hasil.



Gambar 9. Diagram Penelitian

Tahap awal dalam penelitian ini adalah menganalisis masalah yang dihadapi oleh pangkalan gas LPG kg. Melakukan identifikasi tantangan seperti kecurangan konsumen dalam transaksi jual beli gas. Setelah memahami masalah yang dihadapi, peneliti melakukan studi literatur untuk mengumpulkan informasi

tentang penggunaan teknologi deteksi visual, khususnya metode YOLO (*You Only Look* Once). Selanjutnya, penyusunan rencana penelitian sebagai dasar dalam pelaksanaan penelitian. Setelah itu proses pengumpulan data berupa citra tabung gas dari pangkalan dan melakukan pra proses data dengan melakukan anotasi dan augmentasi gambar. Setelah itu persiapan model seperti pemilihan konfigurasi parameter awal. Setelah itu, dilakukan proses pelatihan awal untuk melihat performa awal model.

Kemudian, ke tahap optimasi *hyperparameter*, yaitu penyesuain parameter-parameter model untuk meningkatkan performa. Jika model belum teroptimasi, maka dilakukan proses optimasi kembali dengan mengatur konfigurasi parameter, jika model sudah teroptimasi, maka dilanjutkan ke tahap evaluasi metrik performa untuk menilai keakuratan dan efektivitas model. Penelitian diakhiri dengan tahap inferensi, di mana model diterapkan pada data baru. Selanjutnya pada tahap analisis data dan hasil, yaitu menganalisis data dan hasilnya yang mencakup keakuratan sistem membentuk interprestasi akhir dari kinerja model.

3.2. Perancangan Alat

Gambar 10. Perancangan Alat

Pada gambar 10 menunjukkan perancangan alat pada penelitian ini. Gambar di atas menjelaskan bahwa terdapat integrasi antara kamera IP, perangkat jaringan, dan perangkat pemrosesan sebagai bagian utama dari sistem deteksi objek berbasis YOLOv4-*Tiny*. Kamera IP merek Tiandy dihubungkan ke perangkat PoE *Switch* Zitech, yang berfungsi sebagai sumber daya dan jalur transmisi data secara bersamaan melalui kabel *ethernet*. PoE *switch* mendapat suplai daya dari adaptor dengan sumber Listrik 220V AC. Setelah itu, PoE *switch* juga terhubung ke laptop sebagai pusat kendali pemrosesan data. Perancangan ini dirancang agar sistem mampu mendeteksi apakah sistem dapat mendeteksi objek gas LPG 3 kg dengan akurat serta mendukung pengembangan sistem dalam menghitung jumlah tabung gas LPG 3kg dan mendukung pengujian keandalan sistem nantinya.

3.3. Perancangan Lokasi Deteksi

Tabel 3. Perancangan Lokasi Deteksi

***Point of View* Konsumen**	***Point of View* Kamera IP**
Jalur Masuk ke Pangkalan 	

Pada tabel di atas menunjukkan tahapan pada rencana penelitian dalam memvalidasi kesesuaian sistem dengan kondisi lingkungan sebenarnya, pada tahap ini dilakukan perancangan lokasi deteksi sebagai bagian penting dalam mendukung pengujian sistem deteksi berbasis YOLOv4-*Tiny*. Pada PoV konsumen menunjukkan jalur masuk ke pangkalan gas yang telah dilengkapi dengan pembatas berupa tali serta rambu petunjuk arah untuk memisahkan jalur gas isi dan gas kosong. Penempatan kamera IP dirancang pada sudut lurus terhadap pagar pangkalan agar dapat menangkap keseluruhan aktivitas konsumen ketika keluar dan masuk pangkalan. Penataan lokasi bertujuan untuk menciptakan alur distribusi yang sistematis yang menjadi dasar pengembangan lebih lanjut oleh rekan peneliti yang berfokus pada sistem *counting* dan klasifikasi antara pembeda gas isi dan gas kosong berdasarkan arah jalur.

3.4. Perancangan Pengaturan Parameter Optimasi

Tabel 4. Perancangan Pengaturan Parameter Optimasi

Parameter	Variasi Nilai
Rasio Data	• 80% : 10% : 10% • 70% : 20% : 10% • 70% : 15% : 15% • 60% : 30% : 10% • 60% : 20% : 20%
Learning Rate	• 0,008 • 0,009

	• 0,010 • 0,011 • 0,012
Batch Size	• 16 • 32 • 64 • 128

Pada tabel di atas merupakan parameter penelitian untuk dioptimasi. Perancangan ini mencakup penentuan kombinasi beberapa parameter utama (variabel), yaitu rasio data pelatihan/validasi/pengujian, *learning rate*, dan *batch size*. Tujuan dari variasi parameter ini adalah untuk menguji pengaruh masing-masing konfigurasi terhadap performa model dan memperoleh kofigurasi terbaik terhadap hasil deteksi yang paling akurat dan stabil. Tabel perancangan parameter disusun untuk menggambarkan kombinasi skenario uji coba yang diterapkan. Misalnya beberapa rasio data diuji bersamaan lalu hasil mAP terbaik dipakai sebagai *input* pada pengujian selanjutnya yaitu dengan variasi *learning rate* mulai dari 0,008 hingga 0,012. Begitu juga dengan pengujian *batch size* mulai dari 16 hingga 128 yang *input* nya merupakan hasil pengujian terbaik dari *learning rate*.

3.5. Perancangan Analisis Kinerja Model

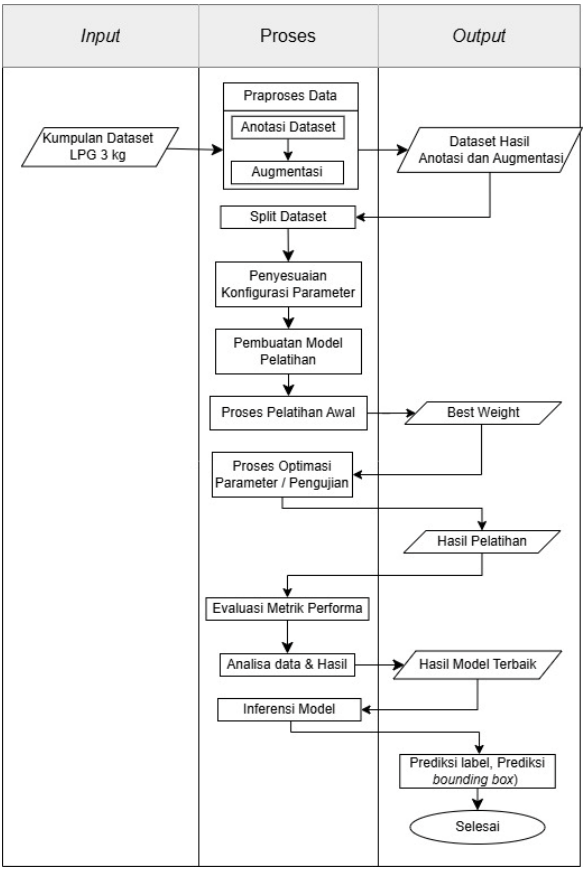
Tabel 5. Parameter Penilaian Kinerja Model

Metriks	Rumus	Rentang Nilai Baik
Precision	TP / (TP + FP)	≥ 85% dianggap baik
Recall	TP / (TP + FN)	≥ 85% dianggap baik
F1-Score	2 x (Precision x Recall) / (Precision x Recall)	≥ 85% dianggap baik
mAP (mean Average Precision)	Rata-rata AP kelas	≥ 50% (baik); ≥ 70% (baik)
IoU (Intersection over Union)	Area tumpang tindih prediksi dan ground truth / Area gabungan	≥ 50% minimum (default); ≥ 75% (tinggi)
FPS (*Frame Per Second*)	Kecepatan deteksi per detik	≥ 15 FPS (cukup baik untuk real-time)

Dalam penelitian ini analisis kinerja model YOLOv4-*Tiny* dilakukan berdasarkan metriks-metriks standar yang umum digunakan dalam deteksi objek. Setiap metriks memiliki batas minimum nilai yang menunjukkan apakah sistem dapat dikategorikan memiliki kinerja yang baik. Berdasarkan referensi dari Padilla et

al.(2020), Redmon et al.(2016), dan standar evaluasi dari COCO, PASCAL VOC, *precision, recall,* dan F1-*Score* dikatakan baik jika ≥ 85%. Nilai *mean Average Precision* (mAP) dinyatakan baik apabila ≥ 50% dan sangat baik jika ≥ 70%. Sedangkan nilai IoU yang umum digunakan untuk menentukan kebenaran deteksi adalah 50%. Lalu untuk aplikasi *real time*, sistem deteksi dianggap cukup responsif apabila dapat menghasilkan minimal 15 *Frame Per Second*. Dengan acuan tersebut, penilaian hasil deteksi yang dilakukan pada penelitian ini mengacu pada *range* dan batasan yang telah ditetapkan.

3.6. Perancangan Alur Kerja Sistem



Gambar 11. Perancangan Alur Kerja Sistem

Pada gambar 11 di atas menjelaskan tahapan perancangan alur kerja sistem untuk mendeteksi objek gas LPG 3 kg. Proses dimulai dari tahap pengumpulan dataset LPG 3 kg, yang kemudian dilakukan praproses data seperti anotasi dataset dan augmentasi data. Dataset yang sudah diproses ini kemudian dipecah menjadi bagian-bagian dataset yaitu *train, valid, test* melalui tahapan *split* dataset. Setelah dataset siap, proses dilanjutkan dengan penyesuaian konfigurasi parameter dan pembuatan model pelatihan. Kemudian model dilatih melalui tahap proses pelatihan awal yang menghasilkan bobot terbaik (*best weight*) sebagai hasil uji sementara. Tahap berikutnya adalah optimasi parameter atau pengujian model dari model sebelumnya, hasil pelatihan ini dievaluasi menggunakan metriks performa untuk menilai akurasi dan metriks model yang dianalisis lebih lanjut untuk mendapatkan hasil model terbaik. Saat model terbaik diperoleh, proses selanjutnya adalah inferensi model menggunakan gambar yang belum pernah digunakan dalam proses pelatihan. Inferensi menghasilkan *output* berupa prediksi label objek dan prediksi *bounding box* sebagai indikator keberhasilan proses deteksi objek gas LPG 3 kg.

3.7. Alat dan Bahan

Pada tugas akhir ini terdapat berbagai alat, bahan, perangkat pendukung yang digunakan, dan lokasi penelitian. Proses pengambilan data dilakukan di lingkungan pangkalan gas milik penulis yang merupakan rumah pribadi. Lalu menggunakan perangkat keras dan perangkat lunak yang mendukung pelatihan dan implementasi sistem. Berikut rincian alat, bahan, perangkat keras, dan perangkat lunak yang digunakan selama penelitian ini:

Tabel 6. Alat dan Bahan

No.	Alat/bahan	Jumlah
1	Laptop	1
2	Tiandy *Bullet Camera* TC-C34QN	1
3	Kabel LAN UTP RJ45 CAT 6 (10 Meter)	1
4	PoE *Switch* Zitech 6 *Port Switch*	1
5	Vention USB to LAN RJ45	1

Penelitian ini menggunakan laptop sebagai pusat pemrosesan data, kamera IP sebagai perangkat pengumpulan citra untuk dataset, kamera dihubungkan menggunakan kabel LAN UTP RJ4R5 CAT 6 dan sumbernya melalui PoE *Switch*. Karena laptop tidak memiliki port LAN, digunakan adaptor Vention USB to LAN RJ45 sebagai konektor.

Tabel 7. Spesifikasi Laptop

Aspek	Spesifikasi
Processor	AMD Ryzen 5 5500U
RAM	16 GB DDR4 (3200 MHz)
Storage	512 GB NVMe SSD
Graphics Card	AMD Radeon Graphics (integrated)
Operating	Windows 11 Home Single Language 64-bit
System	HP Pavilion Laptop 14-ec0xxx

Laptop yang digunakan dalam penelitian ini adalah HP Pavilion 14-ec0xxx dengan spesifikasi yang sudah dicantumkan pada tabel diatas. Meskipun tidak menggunakan GPU eksternal, laptop ini cukup untuk menjalankan kebutuhan pengujian dan pemantauan sistem, serta digunakan untuk proses pelatihan dan inferensi YOLOv4-*Tiny* melalui *google colab* yang memanfaatkan GPU *cloud*.

Tabel 8. Spesifikasi *Software*

Aspek	Spesifikasi
Bahasa Pemrograman	*Python* 3
Software Training Data	*Google Colab*
Framework Deteksi Objek	*Darknet* (YOLOv4-*Tiny*)
Tools Anotasi Data	*Roboflow*

Python 3 digunakan sebagai bahasa pemrograman utama dalam penelitian ini untuk membuat skrip pengolahan data, mengatur pelatihan model, dan menerapkan inferensi deteksi. *Google colab* digunakan sebagai *platform cloud computing* untuk proses pelatihan model YOLOv4-*Tiny*. *Framework* deteksi objek menggunakan *darknet*, yang dipilih karena ringan namun cukup akurat ketika digunakan pada dataset tabung gas. Lalu untuk *tools* anotasi data menggunakan *roboflow,* yaitu sebuah *tools* berbasis web yang mempermudah *labelling* gambar, mengelola dataset, dan melakukan augmentasi.

3.8. Pengujian *Optimasi Hyperparameter*

Pada penelitian ini konfigurasi parameter yang digunakan dalam pelatihan model terdiri atas dua kategori, yaitu parameter tetap dan parameter variabel. Parameter tetap merupakan parameter yang nilainya tidak mengalami perubahan selama seluruh proses pengujian, parameter ini mencakup nilai-nilai seperti *subdivisions*, *input image*, jumlah *channels*, nilai *max batches*, *steps*, jumlah *filters*, dan jumlah *classes*. Sedangkan parameter variabel adalah parameter yang nilainya diubah/diuji untuk melihat pengaruhnya terhadap kinerja model. Dalam penelitian ini, parameter variabel tersebut meliputi rasio pembagian dataset, *learning rate*, dan *batch size.* Ketiga parameter ini dipilih karena dapat mempengaruhi performa model dalam mendeteksi objek dari segi akurasi maupun stabilitas selama pelatihan. Oleh karena itu, pengujian dilakukan secara sistematis sehingga parameter dapat dianalisis dengan lebih objektif. Berikut parameter tetap yang di konfigurasi pada *file* .cfg selama proses pengujian:

Tabel 9. Pengaturan Parameter Tetap *File* .cfg

Kategori	Parameter	Nilai
#*Training*	*Subdivisions*	16
	Widht	416
	Height	416
	Channels	3
	Max_batches	6000
	Steps	4800, 5400
[*convolutional*]	*Filters*	18
[yolo]	*Classes*	1

Tabel 9 menunjukkan parameter tetap yang digunakan dalam *file* .cfg selama pelatihan, yang tidak berubah selama proses pengujian. Parameter ini meliputi ukuran *input* gambar (*width* dan *height*) sebesar 416x416 piksel, jumlah saluran warna (*channelsI*) sebanyak 3 (RGB), serta nilai *subdivisions* sebesar 16 untuk mengatur efisiensi memori. Selain itu, pelatihan dibatasi hingga 6000 iterasi (*Max_batches)* dengan penurunan *learning rate* pada iterasi ke 4800, 5400 (*Steps*). Pada bagian konvolusional, jumlah *Filters* disesuaikan menjadi 18 sesuai dengan jumlah kelas, dan pada bagian YOLO jumlah *Classes* yang dideteksi adalah 1 yaitu gas.

3.8.1. Pengujian Rasio Dataset

Dalam pengujian rasio pembagian data, dataset dibagi menjadi tiga bagian secara terstruktur. Bagian pertama dialokasikan sebagai data pelatihan (*training*) yang digunakan untuk membangun dan melatih model dalam mengenali objek. Bagian kedua ditetapkan sebagai data validasi, yang berfungsi untuk mengevaluasi performa model selama proses pelatihan serta membantu dalam penyesuaian parameter agar model tidak mengalami *overfitting*. Bagian ketiga digunakan sebagai data pengujian (*testing*) untuk mengukur akurasi dan generalisasi model terhadap data yang belum pernah dipelajari sebelumnya. Pembagian ini memastikan bahwa setiap proses pelatihan, validasi, dan pengujian berjalan secara seimbang dan objektif.

3.8.2. Pengujian *Learning Rate*

Pengujian *learning rate* adalah pengujian yang bertujuan untuk mengetahui seberapa pengaruh nilai *learning rate* terhadap kinerja model termasuk akurasi deteksi, tingkat konvergensi, dan stabilitas selama pelatihan. Dalam penelitian ini, pengujian *learning rate* dilakukan menggunakan *input* dari hasil terbaik pengujian rasio pembagian data sebelumnya. Penentuan nilai awal *learning rate* 0,008 dipilih berdasarkan referensi dari jurnal *Sistem Pemantauan Aktivitas Keseharian Lansia Berbasis Deteksi Objek Menggunakan Algoritma YOLO*, yang menunjukkan bahwa nilai tersebut memberikan nilai performa terbaik. Pengujian ini dilakukan sebanyak lima kali pengujian dengan variasi nilai *learning rate* yang diuji mulai dari 0,008 hingga 0,012 dengan inkremen 0,001. Rentang nilai ini dipilih untuk mengeksplorasi apakah terdapat nilai *learning rate* lain dapat memberikan nilai performa yang lebih optimal.

3.8.3. Pengujian *Batch Size*

Pengujian *batch size* adalah pengujian untuk mengetahui apakah pengaturan *batch size* dapat mempengaruhi kinerja model. Pemilihan parameter *batch size* mempengaruhi kecepatan pelatihan dan performa model. Pada penelitian ini, pengujian *batch size* dilakukan menggunakan *input* dari hasil terbaik pengujian *learning rate* sebelumnya,variasi *batch size* yang di uji terdiri dari empat variasi *batch size* mulai dari nilai *batch size* 16 hingga 128.

3.8.4. Pengujian *Accuracy* Dataset

Pengujian ini dilakukan untuk mengevaluasi pengaruh variasi *batch size* terhadap akurasi deteksi model dengan mengambil nilai *True Positive* (TP), *False*

Positive (FP), dan *False Negative* (FN). Hasil pengujian ini menentukan hasil uji mana yang paling optimal sebelum model diterapkan.

3.8.5. Pengujian Performa Inferensi Model

Pengujian ini bertujuan untuk mengukur kecepatan inferensi (waktu deteksi) dari model YOLOv4-*Tiny* dalam merespon *input* berupa gambar. Pengujian ini dilakukan dengan menggunakan 30 gambar uji dengan ukuran input 416 x 416 yang merupakan hasil tangkapan layar dari rekaman video secara *real time.* Gambar-gambar tersebut disimpan dalam folder "UJIPREDIKSI" yang berada di *google drive*, lalu diproses melalui perintah *darknet detector test* pada *google colab.* Waktu deteksi (dalam satuan ms) diperoleh dari *output* skrip sebagai metriks inferensi untuk masing-masing gambar. Pengujian ini hanya dilakukan pada model terbaik yang telah dipilih berdasarkan hasil pengujian optimasi *hyperparameter* dan pengujian akurasi sebelumnya.

3.8.6. Pengujian *Frame per Second* (FPS)

Pengujian FPS bertujuan untuk mengetahui seberapa cepat sistem deteksi objek dalam menjalankan proses inferensi terhadap video *input* dalam satuan *frame per second*. Pengujian ini juga dapat melihat gambaran kelayakan sistem dalam pemantauan *real time*. Pengujian dilakukan dengan memproses video berdurasi kurang lebih dua menit pada tiga kondisi yang berbeda dan menggunakan video baru bukan video yang digunakan sebagai dataset pelatihan, lalu dihitung nilai rata-rata FPS pada masing-masing kondisi. Hasil pengujian kemudian di analisa untuk membandingkan performa sistem di tiap kondisi, serta menilai apakah FPS yang dihasilkan sudah berada dalam rentang yang baik.

Bab 4. Hasil dan Pembahasan

Bab ini membahas hasil dari pengujian sistem deteksi jual beli gas LPG 3 Kg. Setelah dilakukan pengujian, data yang diperoleh dianalisis untuk mengetahui tingkat keberhasilan sistem.

4.1. Hasil Penerapan Metode

4.1.1. Tahapan Praproses Data

- **Jumlah Dataset Awal**

Langkah pertama dalam penelitian ini adalah mengumpulkan data awal yaitu dataset. Dalam penelitian ini, perangkat keras Tiandy *Bullet Camera* TC-C34QN digunakan sebagai sumber *input* data dengan mengumpulkan gambar tabung gas LPG 3kg, gambar dataset mencakup berbagai kondisi waktu yaitu pagi, siang, sore, dan malam yang diolah menggunakan *Roboflow* untuk melakukan proses anotasi (*labelling*) dan augmentasi data. Terdapat tabel jumlah dataset sesuai dengan kondisi waktu seperti di bawah ini:

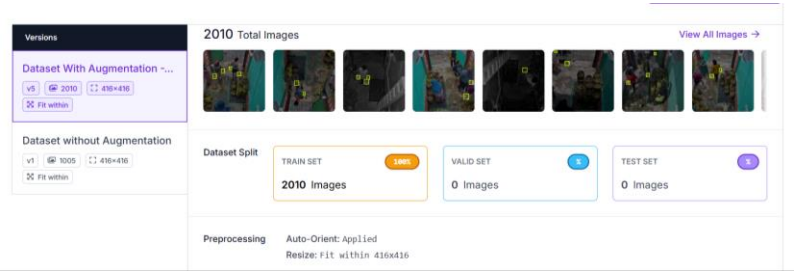
Tabel 10. Jumlah Dataset per Kondisi Waktu

Kondisi Waktu	Pagi	Siang	Sore	Malam
Jumlah gambar	251	259	200	295
Total	**1005**			

Tabel 10 menunjukkan pembagian jumlah gambar berdasarkan kondisi waktu pengambilan, yaitu pagi, siang, sore, dan malam. Dari total 1005 gambar, terdapat jumlah gambar dari kondisi pagi sebanyak 251 gambar, siang sebanyak 259 gambar, sore sebanyak 200 gambar, dan malam sebanyak 295 gambar. Pengambilan dataset diambil berdasarkan perbedaan kondisi waktu untuk keberagaman dataset yang akan digunakan saat pelatihan, sehingga dapat meningkatkan kemampuan deteksi objek pada berbagai kondisi pencahayaan nyata.

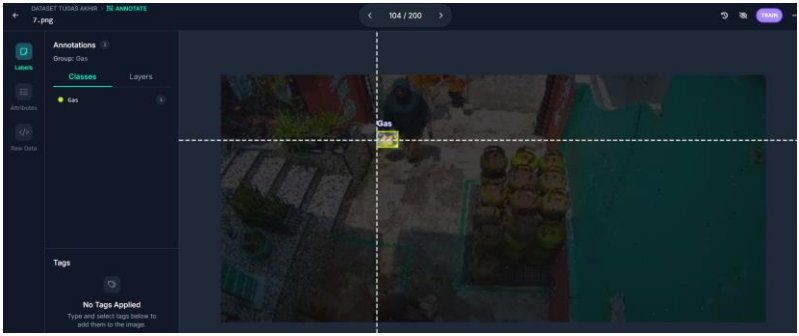
- **Proses Anotasi Gambar**

Pada penelitian ini seluruh dataset pada *roboflow* tidak dibagi langsung ke dalam tiga bagian, gambar dijadikan sebagai data pelatihan 100% *train set.* Untuk pembagian dataset menjadi tiga bagian, dilakukan menggunakan *script* tambahan yang dijelaskan pada hasil *split* dataset.



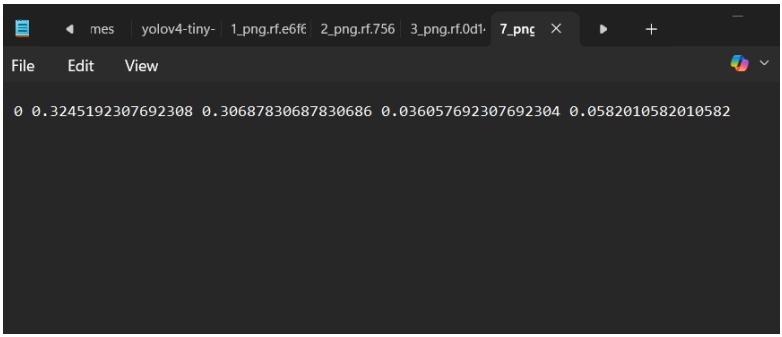
Gambar 12. Tampilan Dataset pada *Roboflow*

Terdapat tahapan prapemrosesan data agar data yang digunakan sesuai dengan tujuan sistem deteksi. Salah satu tahapan ini adalah proses anotasi gambar yang dilakukan dengan menandai tabung gas menggunakan *bounding box* berwarna kuning pada *Roboflow*.



Gambar 13. Proses Anotasi gambar

Proses anotasi gambar dilakukan untuk memberikan label atau informasi tertentu pada objek yang ingin dikenali oleh sistem. Dalam penelitian ini, anotasi gambar berfungsi sebagai dasar pembelajaran bagi model, yang mana model dilatih menggunakan data gambar tabung gas yang telah diberi label sebelumnya. Sehingga setelah melakukan anotasi gambar, model dapat mengenali objek tabung gas secara otomatis pada kegiatan jual beli di pangkalan gas LPG 3 Kg. Hasil dari proses anotasi ini disimpan dalam format *file* teks (.txt) dengan struktur tertentu. Berikut contoh tampilan hasil anotasi gambar:



Gambar 14. Contoh *file* teks (.txt)

Berdasarkan pada gambar di atas, setiap baris berisi informasi mengenai satu objek yang terdeteksi. Struktur baris anotasi pada YOLO menyimpan informasi penting terkait posisi dan ukuran objek dalam gambar. Berikut adalah penjelasan struktur baris anotasi pada YOLO:

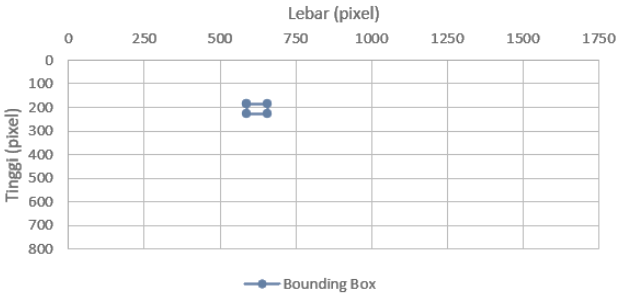
```
<class_id> <x_center> <y_center> <width> <height>
```

Gambar 15. Struktur Baris Anotasi pada YOLO

Penjelasan:

1. <*class_id*> menunjukkan kelas objek yang bernilai angka. Angka ini merepresentasikan urutan kelas dalam daftar *file obj.names* yang dimulai dari indeks ke-0, karena pada penelitian ini hanya memiliki satu kelas yaitu "Gas" maka nilai <*class_id*> untuk "Gas" adalah 0.
2. <*x_center*> dan <*y_center*> merupakan koordinat titik tengah dari *bounding box* objek yang dinyatakan dalam bentuk nilai relatif terhadap ukuran gambar. Nilai ini berada dalam rentang 0 hingga 1 dimana 0 berarti posisi paling kiri (untuk <*x_center*>) atau atas gambar (untuk <*y_center*>), dan 1 berarti posisi paling kanan atau bawah.
   Pada penelitian ini:
   - <*x_center*> bernilai 0.3245. jika dikalikan dengan lebar asli gambar (1918 piksel) maka diperoleh:
     - 0.3245 x 1918 = 622 piksel, yang artinya titik tengah *bounding box* terletak sekitar 622 piksel dari sisi kiri gambar.
   - Sedangkan <*y_center*> bernilai 0.3069. Jika dikalikan dengan tinggi gambar (872 piksel) maka diperoleh:
     - 0.3069 x 872 = 267 piksel, yang artinya titik tengah *bounding box* terletak sekitar 267 piksel dari sisi atas gambar.

3. <*width*> dan <*height*> merupakan ukuran lebar dan tinggi *bounding box* dalam bentuk nilai relatif terhadap dimensi gambar.
   Pada penelitian ini:
   - <*width*> bernilai 0.0360 yang artinya lebar *bounding box* adalah sekitar 3,6% dari lebar gambar atau setara dengan 0.0360 x 1918 = 69 piksel.
   - <*height*> bernilai 0.0582 yang artinya tinggi *bounding box* adalah sekitar 5,82% dari tinggi gambar atau setara dengan 0.0582 x 872 = 50 piksel.



Gambar 16. Visualisasi *Bounding Box* YOLO dalam piksel

Gambar di atas merupakan visualisasi *bounding box* berdasarkan struktur baris anotasi pada gambar 14. Visualisasi ini menunjukkan posisi dan ukuran b*ounding box* yang dapat dibandingkan dengan posisi objek tabung gas pada gambar 13.

- **Pengaturan Augmentasi Dataset**

Setelah melakukan anotasi gambar dilakukan pengaturan augmentasi meliputi:

Tabel 11. Pengaturan Augmentasi

Jenis Augmentasi	Parameter
Flip	Horizontal, Vertikal
Crop (Zoom)	Min Zoom: 0% Max Zoom: 30%
Saturation	-30% sampai +30%
Brightness	-20% sampai +20%
Noise	Hingga 0.1% pixel
***Outputs per training example*: 2**	

Tabel di atas menjelaskan setiap gambar asli menghasilkan dua gambar keluaran yaitu satu gambar asli dan satu gambar hasil augmentasi. Dengan jumlah dataset asli sebanyak 1005 gambar setelah menjalankan proses augmentasi menghasilkan total dataset sebanyak 2010 gambar untuk di *training*. Berikut merupakan contoh hasil augmentasi berdasarkan jenis augmentasi yang disesuaikan pada *roboflow*:

Tabel 12. Contoh Hasil Augmentasi

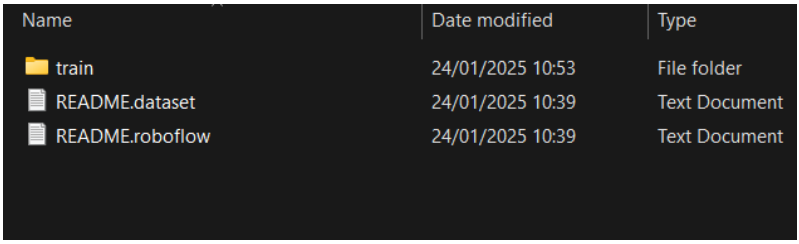
Gambar Asli	Gambar Hasil Augmentasi	Jenis Augmentasi
		*Flip* Horizontal
		*Flip* Vertikal
		*Crop Min Zoom* 0%

Gambar Asli	Gambar Hasil Augmentasi	Jenis Augmentasi
		*Crop Max Zoom* 30%
		*Saturation* -30%
		*Saturation* +30%
		*Brighten* 20%

Gambar Asli	Gambar Hasil Augmentasi	Jenis Augmentasi
		*Darken* 20%
		*Noise* 0.1%

Pada tabel 12 menunjukkan terdapat proses augmentasi data dengan berbagai jenis transformasi untuk meningkatkan kemampuan generalisasi model terhadap variasi kondisi di dunia nyata. Augmentasi yang digunakan meliputi *flip* dengan dua arah, yaitu *horizontal* dan *vertical,* untuk melatih model mengenali objek dari orientasi berbeda. Selain itu, dilakukan *crop* (*zoom*) dengan parameter *min zoom* 0% dan *max zoom* 30% untuk mensimulasikan objek yang dekat atau sebagian terpotong dalam bingkai kamera. *Saturation* disesuaikan dalam rentang -30% hingga +30% untuk menyesuaikan variasi tingkat kejenuhan warna, sementara *brightness* diatur antara -20% hingga +20% agar model dapat mendeteksi objek dalam kondisi pencahayaan rendah maupun tinggi. Lalu proses ditambahkan *noise* hingga 0,1% *pixel* untuk menciptakan gangguan visual ringan seperti kondisi buram atau kamera kualitas rendah. Setiap gambar pelatihan menghasilkan dua *output* hasil augmentasi, sehingga memperkaya variasi dataset tanpa perlu menambah data asli.

Seluruh dataset yang telah di anotasi dan di augmentasi akan di *download* dan disimpan dalam satu folder "*train*" seperti yang ditampilkan pada gambar di bawah ini:



Gambar 17. Hasil Folder Dataset yang Tersimpan

Gambar 17 menunjukkan tampilan folder "*train"* yang berisi seluruh dataset yang telah melalui proses anotasi dan augmentasi. Folder ini digunakan sebagai direktori utama yang selanjutnya di *split* menjadi tiga data yaitu data *train,* data *valid,* dan *test* sebelum digunakan ke dalam proses *training* model di *google colab.*

- **Pengaturan *Split* Dataset**

Pada tahap ini, pembagian dataset menjadi *train*, *valid*, dan *test* dilakukan menggunakan skrip Python tambahan yaitu *yolosplitter.py*. Pembagian dataset tidak dilakukan menggunakan fitur *split* bawaan dari *roboflow*. Hal ini dikarenakan menurut pengalaman penulis pembagian dataset manual memberikan hasil yang lebih jelas dan terstruktur. Pembagian dataset dapat disesuaikan berdasarkan rasio proporsi yang ingin di teliti, seperti pada penelitian ini untuk pelatihan awal, data dibagi dengan proporsi sebagai berikut:

```
14  # Proporsi split
15  train_ratio = 0.7
16  valid_ratio = 0.2
17  test_ratio = 0.1
```

Gambar 18. Pembagian Dataset Awal

Berdasarkan gambar di atas, dataset dibagi dengan proporsi 70% data digunakan untuk pelatihan (*train*), 20% untuk validasi (*valid*), dan 10% untuk pengujian (*test*). Rasio ini dipilih untuk memastikan model memiliki cukup data untuk belajar dan diuji secara objektif selama proses pelatihan dan validasi awal. Selain itu, beberapa kombinasi rasio juga diuji untuk mencari konfigurasi dataset yang paling optimal dalam meningkatkan performa model. Proses ini mendukung optimasi *hyperparameter* dan penentuan konfigurasi terbaik untuk sistem deteksi. Hasil dari eksekusi skrip ini adalah terbentuknya file *train.*txt, *valid*.txt, dan *test*.txt yang masing-masing berisi jumlah gambar sesuai pembagian dataset.

4.1.2. Hasil Konfigurasi Model YOLOv4-*Tiny*

Setelah menyelesaikan proses anotasi, augmentasi data, dan *split* dataset, tahap selanjutnya adalah melakukan konfigurasi awal parameter dalam model YOLOv4-*Tiny*. Konfigurasi dilakukan untuk menyesuaikan arsitektur model dengan jumlah kelas dan pengaturan lain yang mempengaruhi proses pelatihan, termasuk beberapa parameter penting yang perlu disesuaikan.

- **Pengaturan *File* Konfigurasi**

Pengaturan parameter pada penelitian ini dilakukan untuk proses *training* tahap awal sebelum dilakukannya pencarian parameter terbaik. Berikut adalah parameter yang disesuaikan pada *file* .cfg YOLOv4-*Tiny*.

Tabel 13. Pengaturan Parameter *File* .cfg Awal

Kategori	Parameter	Nilai
#*Training*	*Batch*	64
	Subdivisions	16
	Widht	416
	Height	416
	Channels	3
	Max_batches	2000
	Steps	1600, 1800
[*convolutional*]	*Filters*	18
[yolo]	*Classes*	1

Berdasarkan tabel 13, dilakukan penyesuaian parameter pada *file* konfigurasi YOLOv4-*Tiny* agar sesuai dengan kebutuhan model. Parameter *batch* diatur menjadi 64 sebagai konfigurasi awal karena nilai ini umum digunakan pada arsitektur YOLOv4-*Tiny*, nilai ini kemudian dievaluasi lebih lanjut dalam eksperimen variasi *batch size* untuk menentukan konfigurasi terbaik. Untuk mengurangi beban memori pada tahap pelatihan awal, parameter *subdivisions* disesuaikan menjadi 16. Ukuran *input* gambar diatur dengan nilai *width* dan *height* masing-masing menjadi 416 piksel, sesuai dengan ukuran yang biasanya digunakan pada saat mengatur parameter pada YOLOv4-*Tiny*. Setelah itu pengaturan nilai pada *max_batches* menjadi 2000 karena dalam penelitian ini hanya ada 1 kelas yaitu gas. Rumusnya adalah *max_batches* = (jumlah kelas) x 2000. Parameter *steps* disesuaikan menjadi 1600 dan 1800, yang merupakan 80% dan 90% dari total *max_batches.* Setelah itu, penyesuaian nilai pada bagian [*convolutional*] dan [yolo]. Nilai *filters* diatur menjadi 18 yang diperoleh dari rumus (jumlah kelas + 5) x 3, maka nilai *filters* disesuaikan menjadi 18. Lalu nilai *classes*

disesuaikan menjadi 1 sesuai dengan jumlah kelas yang digunakan pada penelitian ini.

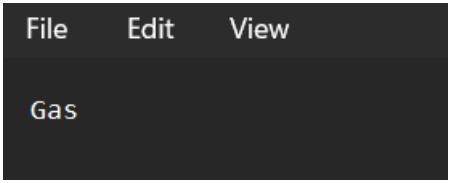
- **Pengaturan *File Obj.data* dan *Obj.names***

Setelah melakukan konfigurasi parameter pada *file* .cfg, tahap selanjutnya adalah menyiapkan struktur *file* pendukung untuk proses *training* YOLOv4-*Tiny* yang dilakukan menggunakan *google colaboratory*. *File* itu adalah *obj.data* dan *obj.names*. *File obj.data* memuat mengenai jumlah kelas yang digunakan dalam proses *training*, *path* untuk menuju *file train.txt*, *valid.txt*, *test.txt, obj.names*, dan tempat penyimpanan hasil model (*weight*) setelah proses pelatihan awal.

```
classes= 1
train  = /content/darknet/data/train.txt
valid  = /content/darknet/data/valid.txt
test   = /content/darknet/data/test.txt
names = /content/darknet/data/obj.names
backup = /content/gdrive/MyDrive/withAUG/training
```

Gambar 19. Hasil Penyesuaian *File obj.data*

Setelah melakukan *file obj.data* seselanjutnya menyesuaikan *file obj.names* yang hanya berisi nama objek sesuai dengan nama objek yang ada pada proses anotasi gambar. Seperti gambar di bawah ini:

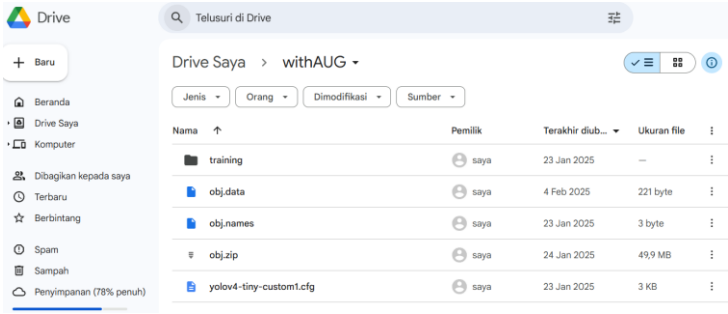


Gambar 20. Hasil Penyesuaian *File obj.names*

Gambar 20 merupakan tampilan hasil dari *file obj.names* yang isinya hanya ada 1 kelas objek yaitu “Gas”. *File* ini digunakan sebagai referensi label kelas oleh model selama proses pelatihan dan inferensi, sehingga model hanya akan mendeteksi objek gas dengan label “Gas”.

4.1.3. Hasil Struktur Folder di Google Drive

Setelah seluruh *file* pendukung selesai disiapkan, *file-file* tersebut diunggah ke *google drive*.



Gambar 21. *File* Pendukung Pelatihan Awal

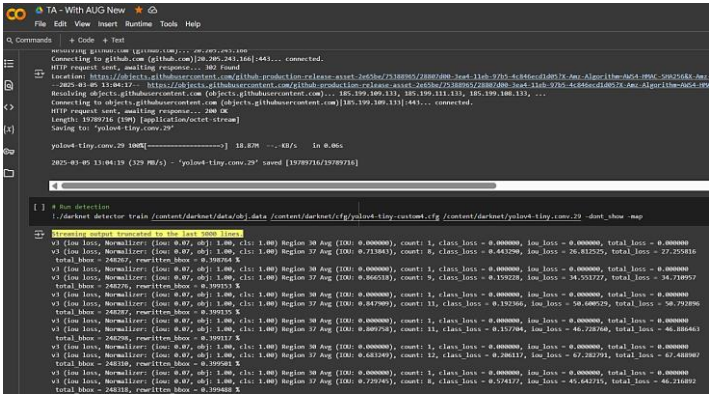
Seperti yang ditunjukkan pada gambar 21. Struktur folder yang terorganisasi dapat menghindari *error* saat pemanggilan *file* pada saat proses pelatihan awal. Selain itu, untuk penyimpanan hasil model berupa *weight* setelah proses pelatihan awal, diperlukan pembuatan folder khusus yang terpisah. Pada penelitian ini, folder tersebut dinamai "*training*". Lalu ada *file obj.zip* yang diunggah pada *google drive. File* tersebut merupakan seluruh gambar dataset yang telah di kompresi menjadi format .zip

4.1.4. Proses Pelatihan Awal Model

Proses pelatihan awal bertujuan untuk melihat kinerja dasar pada model terhadap dataset tabung gas LPG 3 Kg yang telah disiapkan. Tahapan ini dilakukan sebelum proses optimasi parameter untuk mengetahui sejauh mana pengaruh parameter terhadap performa model.

Langkah awal adalah menghubungkan *google drive* dengan *google colaboratory* agar *file* pendukung dapat diakses dan digunakan langsung di dalam sesi *colab.* Setelah itu, dilakukan konfigurasi *file Makefile* yang dapat mengaktifkan penggunaan GPU dan pustaka tambahan seperti CUDNN untuk mempercepat proses pelatihan. Lalu pada *colab*, *file* yang terdapat pada folder di *google drive* disalin ke direktori *darknet.* Untuk mempermudah data dan menghindari konflik *file*, direktori cfg dibersihkan dan dibuat ulang, sedangkan folder data dibersihkan kecuali folder *labels* yang masih dibutuhkan dalam proses *training*. Selanjutnya, *file* dataset dalam bentuk obj.zip disalin dari *google drive* ke dalam direktori kerja lalu diekstrak ke dalam folder data. *File* konfigurasi model dan dataset juga dipindahkan ke direktori yang sesuai agar dapat digunkan pada tahap pelatihan awal pada *colab.* Selanjutnya adalah menyalin *file* konfigurasi *yolov4-tiny-custom.cfg* ke dalam direktori /darknet/cfg/, *file* ini merupakan konfigurasi utama yang digunakan dalam pelatihan model. Kemudian *file obj.names* dan *obj.data* juga disalin ke direktori /darknet/data/.

Tahap selanjutnya adalah mengunduh *pre-trained weight* dari YOLOv4-*Tiny,* yaitu *file yolov4-tiny.conv.29* yang telah dilatih sebelumnya hingga lapisan konvolusi ke-29 atau disebut sebagai *transfer learning* yang artinya model tidak dilatih dari awal melainkan melanjutkan bobot yang telah ada agar hasil pelatihan lebih efisien. Setelah semua *file* telah tersedia dan konfigurasi telah selesai dilakukan, proses pelatihan awal dapat dimulai menggunakan perintah pelatihan pada *darknet.*



Gambar 22. Proses Pelatihan Awal pada *Colab*

Pada gambar 22 menunjukkan bahwa saat pelatihan berlangsung, *log output* menampilkan nilai *Mean Average Precision* (*mAP*) secara berkala. Nilai *mAP* untuk mengevaluasi kinerja model dalam mendeteksi objek.

4.1.5. Hasil *Output* Pelatihan Awal Model

Pada hasil pelatihan awal, diperoleh nilai metriks evaluasi beserta grafik hasil uji yang ditampilkan pada tabel di bawah ini:

Tabel 14. Hasil Nilai Metriks dan Grafik Pelatihan Awal

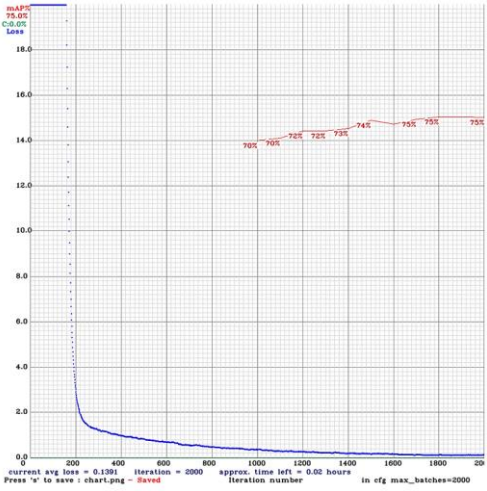
Max Batches	Precision	Recall	F1-Score	Average IoU	mAP@0.50	Loss
2000	78%	72%	75%	59,03%	75%	0,2

Tabel 14 menunjukkan hasil evaluasi pelatihan awal model dengan beberapa metrik yang menggambarkan kinerja deteksi objek. Dengan nilai *max batches*

2000, model menghasilkan *precision* sebesar 78%, *recall* sebesar 72%, Nilai F1-*Score* sebesar 75%, rata-rata IoU sebesar 59.03%, nilai mAP@0.50 sebesar 75% pada ambang batas IoU 0.50, *loss* sebesar 0,2.

```
 (next mAP calculation at 2000 iterations)

 Tensor Cores are disabled until the first 3000 iterations are reached.
 Last accuracy mAP@0.50 = 75.13 %, best = 75.16 % H2000/2000: loss=0.2 map=0.75 best=0.75 hours left=0.0
 2000: 0.200744, 0.139061 avg loss, 0.000026 rate, 0.651533 seconds, 128000 images, 0.020089 hours left

 calculation mAP (mean average precision)...
 Detection layer: 30 - type = 28
 Detection layer: 37 - type = 28
568
 detections_count = 2772, unique_truth_count = 1419
class_id = 0, name = Gas, ap = 75.02%          (TP = 1024, FP = 289)

 for conf_thresh = 0.25, precision = 0.78, recall = 0.72, F1-score = 0.75
 for conf_thresh = 0.25, TP = 1024, FP = 289, FN = 395, average IoU = 59.03 %

 IoU threshold = 50 %, used Area-Under-Curve for each unique Recall
 mean average precision (mAP@0.50) = 0.750211, or 75.02 %
Total Detection Time: 5 Seconds

Set -points flag:
 `-points 101` for MS COCO
 `-points 11` for PascalVOC 2007 (uncomment `difficult` in voc.data)
 `-points 0` (AUC) for ImageNet, PascalVOC 2010-2012, your custom dataset

 mean_average_precision (mAP@0.50) = 0.750211
Saving weights to /content/gdrive/MyDrive/WithAUG/training/yolov4-tiny-custom1_2000.weights
Saving weights to /content/gdrive/MyDrive/WithAUG/training/yolov4-tiny-custom1_last.weights
Saving weights to /content/gdrive/MyDrive/WithAUG/training/yolov4-tiny-custom1_final.weights
If you want to train from the beginning, then use flag in the end of training command: -clear
```

Gambar 23. Hasil Pelatihan Awal



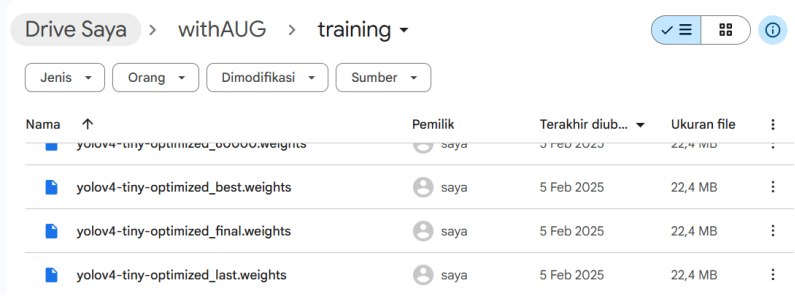
Gambar 24. Grafik Hasil Pelatihan Awal

Gambar 23 dan 24 menunjukkan hasil pelatihan awal model yang ditampilkan melalui *log output* di *google colab* serta grafik pelatihan yang merekam tren metrik selama proses *training.* Grafik pada gambar 24 menampilkan perubahan nilai *loss*

secara bertahap hingga mencapai titik konvergensi yang menunjukkan bahwa model mulai belajar secara stabil.

Selain itu pelatihan ini juga menghasilkan sejumlah *file weights* yang disimpan pada *google drive*, dengan iterasi mulai dari 1000 *weights* hingga 2000 *weights*. Lalu, terdapat tiga *output* utama dari hasil pelatihan, yaitu:

- *Best weights* = bobot model dengan performa terbaik selama *training*, yang di validasi dari nilai mAP.
- *Final weights* = bobot pada iterasi terakhir dari proses pelatihan.
- *Last weights* = bobot terakhir yang tersimpan pada proses pelatihan dihentikan.



Gambar 25. Hasil *Weight* Pelatihan Awal

Berdasarkan hasil pelatihan awal pada tabel 14 menunjukkan performa model yang masih tergolong cukup rendah, dengan *precision* sebesar 78%, *recall* 72%, dan mAP@0.50 hanya mencapai 75%. Nilai-nilai ini menunjukkan bahwa kemampuan model dalam mendeteksi objek pada dataset masih belum optimal, sehingga perlu dilakukan upaya meningkatkan performasi deteksi.

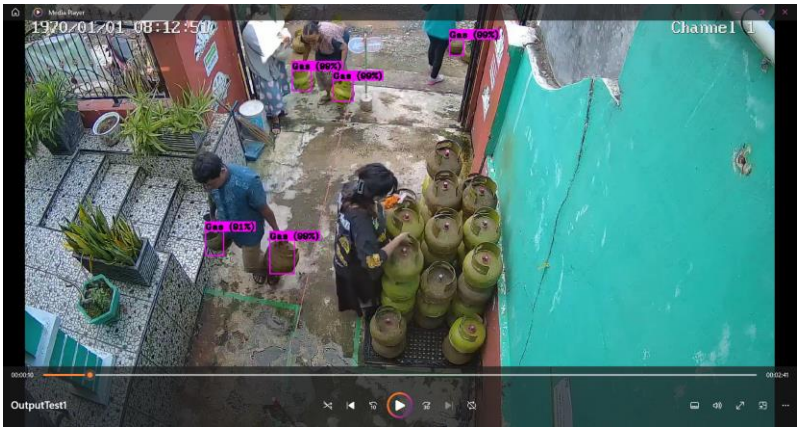
Untuk itu, penelitian ini melakukan optimasi *hyperparameter* secara manual (manual *tuning*), yaitu mengatur parameter variabel secara manual berdasarkan hasil pengamatan performa model sebelumnya.

4.1.6. Hasil Uji Coba Video Detektor

Pada tahap ini dilakukan untuk uji coba apakah model sudah dapat mengenal objek. Proses uji coba ini di uji pada video yang menunjukkan pendeteksian objek menggunakan YOLOv4-*Tiny* yang telah dilatih sebelumnya. Video yang diuji berisi aktivitas transaksi jual beli gas LPG 3Kg. Pengujian dilakukan pada *platform google colab* menggunakan GPU Tesla T4 dengan *driver* NVIDIA versi 535.104.05 dan versi CUDA 12.2 .

Gambar 26. Tampilan Program *Test Detector* pada *Google Colab*

Gambar 26 menunjukkan tampilan program pengujian (*test detector*) yang dijalankan pada *platform google colab*. Dalam tahap ini model diuji menggunakan perintah *darknet detector demo* untuk mendeteksi objek pada video uji.



Gambar 27. Hasil Uji Detektor Video

Gambar 27 menunjukkan hasil uji detektor video menggunakan model YOLOv4-*Tiny* terhadap objek tabung gas LPG 3Kg. Objek yang terdeteksi ditandai dengan *bounding box* berwarna magenta dengan label "Gas" dan tingkat kepercayaan (*confidence score*) yang tinggi sekitar 91% hingga 99%. Hasil ini menggambarkan kemampuan model dalam mendeteksi objek pada video uji dan menjadi dasar untuk mengevaluasi performa model dalam kondisi nyata.

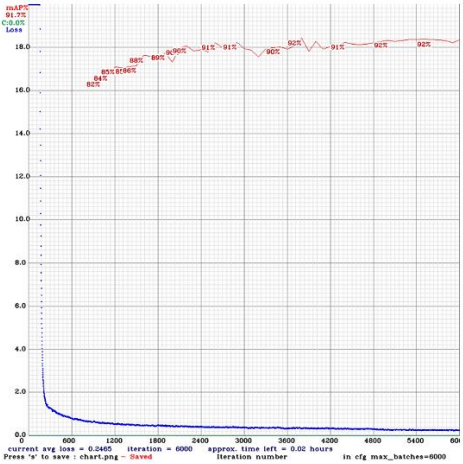
4.2. Hasil Pengujian Optimasi *Hyperparameter*

4.2.1. Hasil Pengujian Pengaruh Rasio Dataset

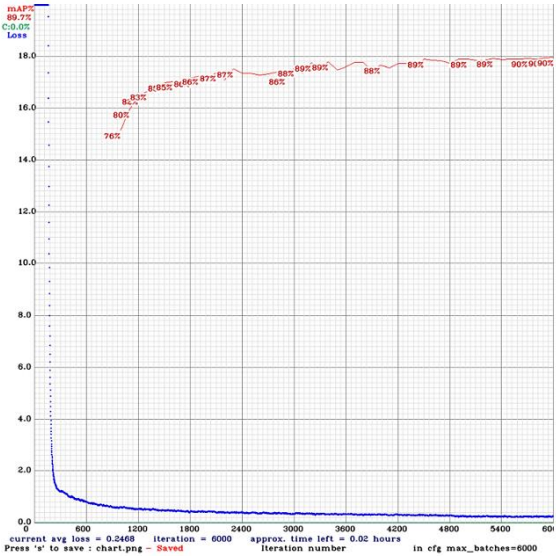
Tabel 15. Hasil Pengujian Pengaruh Rasio

Data Hasil Output Pelatihan YOLO							
No	**Rasio** (*train, valid, test*)	**Metriks**					
		Precision	**Recall**	**F1-Score**	**Average IoU**	**mAP@0.50**	**Loss**
1	80:10:10	84%	87%	86%	66,97%	91,70%	0,3
2	70:20:10	83%	87%	85%	65,16%	89,70%	0,3
3	70:15:15	82%	86%	84%	64,94%	88,94%	0,2
4	60:30:10	82%	85%	84%	64,25%	88,27%	0,2
5	60:20:20	80%	84%	82%	62,60%	86,42%	0,2

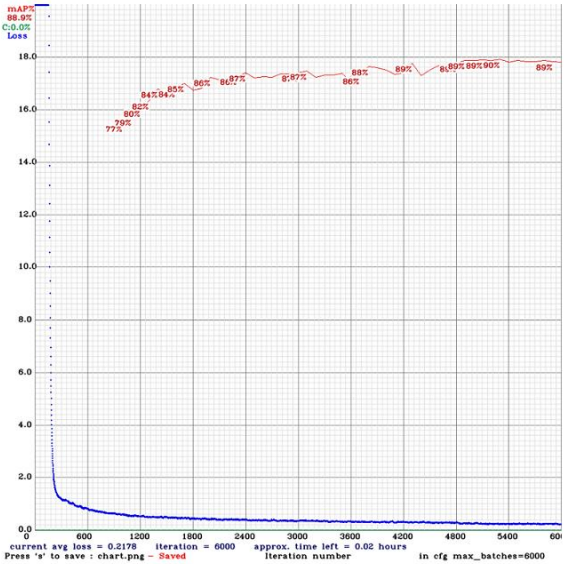
Tabel 15 menunjukkan variasi rasio dataset menghasilkan nilai metriks performa yang berbeda-beda, hasil terbaik diperoleh pada rasio 80:10:10 dengan nilai *precision* sebesar 84%, *recall* 87%, F1-*Score* sebesar 86%, *Average IoU* 66,97%, *mAP@0.50* 91,70%, dan nilai *Loss* sebesar 0,3. Sementara itu, variasi rasio lainnya menunjukkan nilai metriks yang cenderung lebih rendah, baik dalam aspek kemampuan deteksi objek (*precision* dan *recall*) maupun akurasi penempatan *bounding box* (ditunjukkan oleh *IoU dan mAP*). Selanjutnya rasio dengan performa terbaik tersebut menjadi acuan dalam proses analisis dan pengujian parameter selanjutnya, sehingga evaluasi parameter berikutnya, seperti *learning rate*, *batch size*, menjadi lebih terukur. Berikut gambar hasil pengujian rasio:



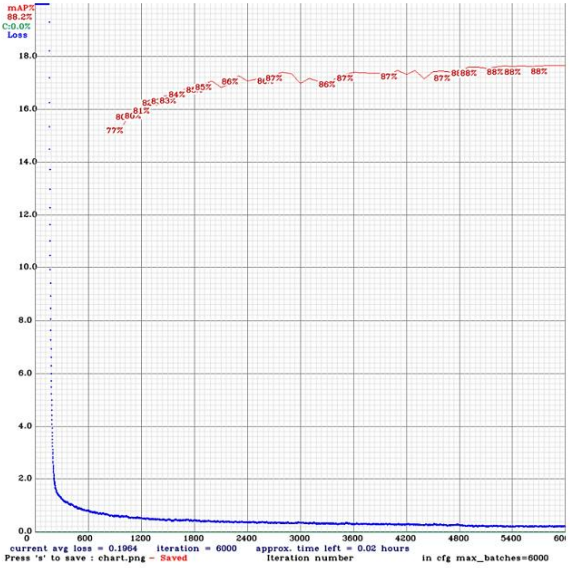
Gambar 28. Grafik Hasil Pelatihan Rasio 80:10:10



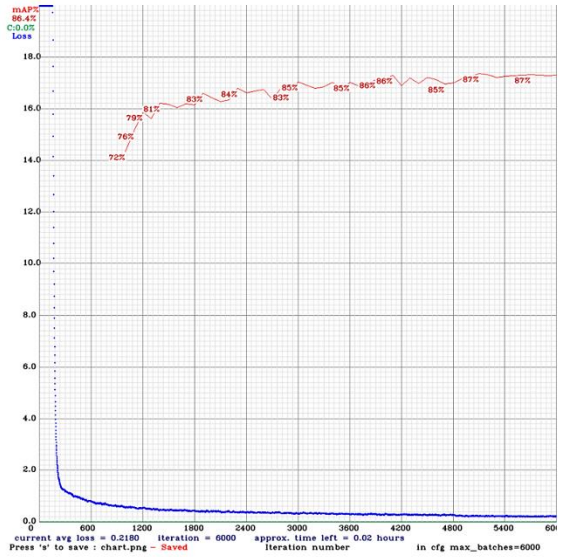
Gambar 29. Grafik Hasil Pelatihan Rasio 70:20:10



Gambar 30. Grafik Hasil Pelatihan Rasio 70:15:15



Gambar 31. Grafik Hasil Pelatihan Rasio 60:30:10



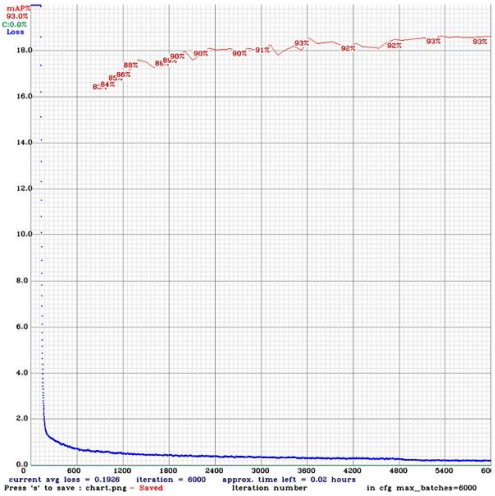
Gambar 32. Grafik Hasil Pelatihan Rasio 60:20:20

4.2.2. Hasil Pengujian *Learning Rate*

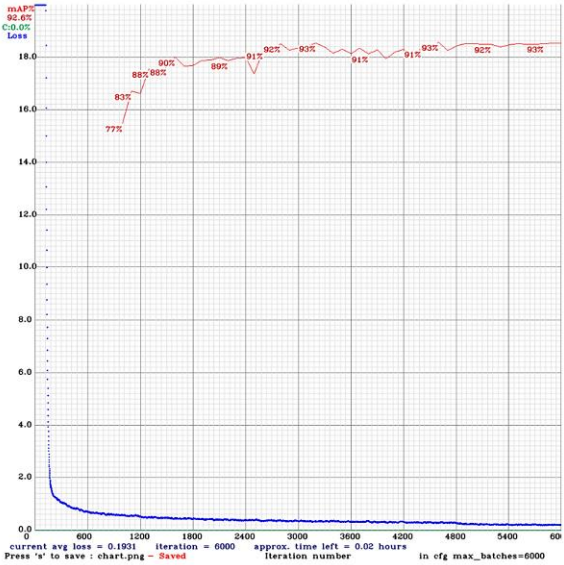
Tabel 16. Hasil Pengujian *Learning Rate*

Data Hasil Pengujian Variasi *Learning Rate*							
No	**Learning Rate**	**Metriks**					
		Precision	**Recall**	**F1-Score**	**Average IoU**	**mAP@0.50**	**Loss**
1	0,008	87%	89%	88%	68,85%	93,03%	0,2
2	0,009	89%	90%	90%	70,56%	92,60%	0,2
3	0,010	86%	90%	88%	68,37%	92,50%	0,2
4	0,011	88%	89%	89%	69,68%	92,40%	0,2
5	0,012	88%	91%	90%	70,02%	93,60%	0,3

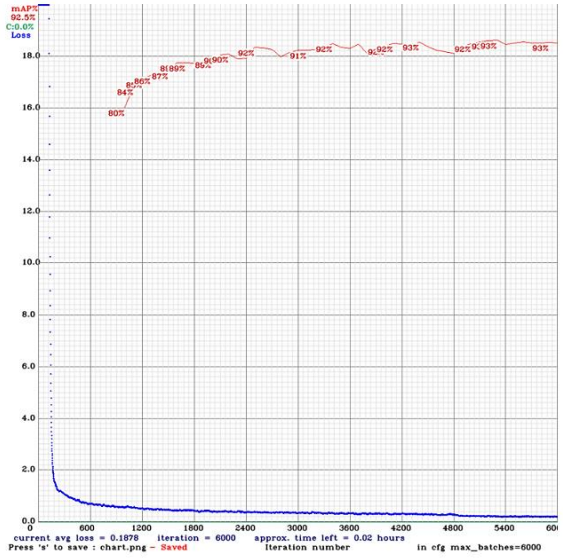
Tabel 16 menunjukkan setiap variasi *learning rate* memberikan hasil metriks performa yang relatif tinggi namun penelitian ini mengambil nilai terbaik dari *learning rate.* Diperoleh pada *learning rate* 0,012 dengan *precision* sebesar 88%, *recall* 91%, F1-*Score* sebesar 90%, *Average IoU* sebesar 70,02%, *mAP@0.50* sebesar 93,60%, dan nilai *Loss* sebesar 0,3. Selanjutnya nilai performa terbaik dari variasi *learning rate* dibandingkan dengan hasil pengujian optimasi sebelumnya untuk mengamati sejauh mana pengaruh perubahan *learning rate* terhadap hasil pelatihan model YOLOv4-*Tiny*. Berikut gambar hasil pengujian *learning rate:*



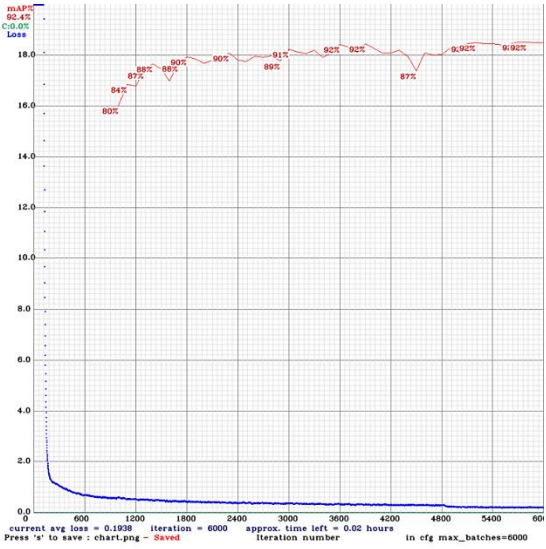
Gambar 33. Grafik Hasil Pelatihan *Learning Rate* 0,008



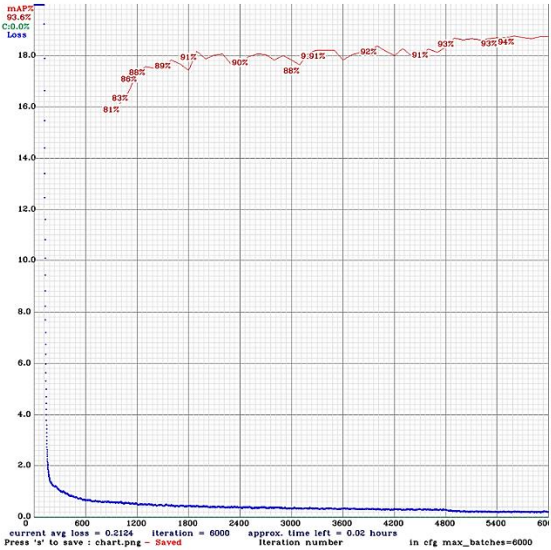
Gambar 34. Grafik Hasil Pelatihan *Learning Rate* 0,009



Gambar 35. Grafik Hasil Pelatihan *Learning Rate* 0,010



Gambar 36. Grafik Hasil Pelatihan *Learning Rate* 0,011



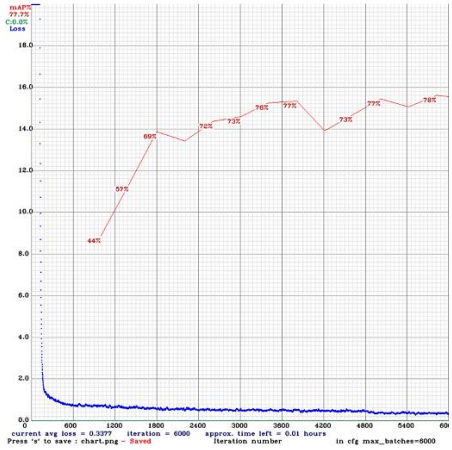
Gambar 37. Grafik Hasil Pelatihan *Learning Rate* 0,012

4.2.3. Hasil Pengujian *Batch Size*

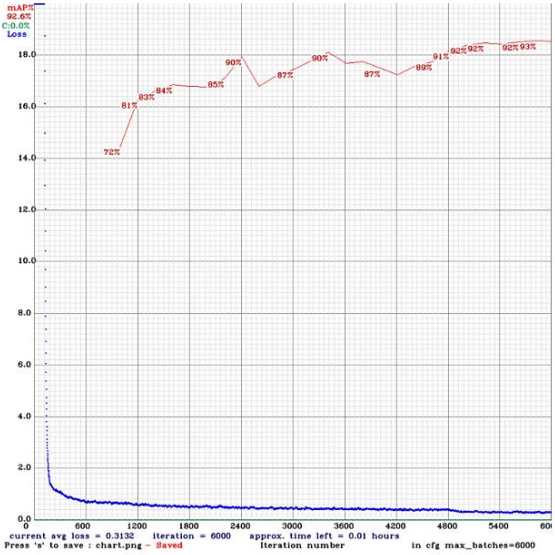
Tabel 17. Hasil Uji Berdasarkan *Batch Size*

Data Hasil Pengujian Variasi *Batch Size*							
No	**Batch Size**	**Metriks**					
		Precision	**Recall**	**F1-Score**	**Average IoU**	**mAP@0.50**	**Loss**
1	16	86%	66%	75%	66,26%	77,70%	0,4
2	32	84%	90%	87%	66,60%	92,60%	0,4
3	64	88%	91%	90%	70,02%	93,60%	0,3
4	128	89%	88%	88%	71,02%	92,00%	0,1

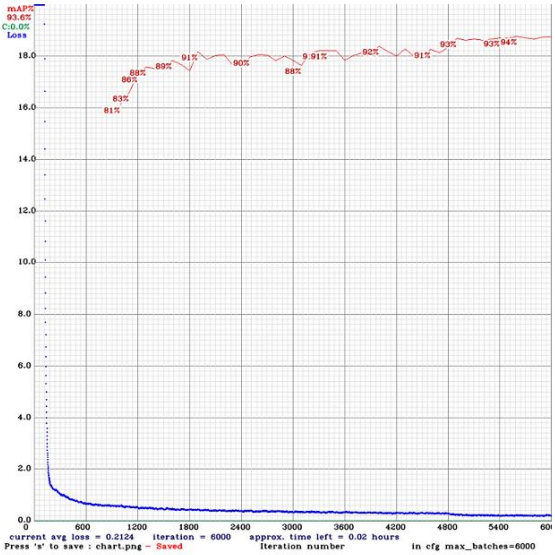
Berdasarkan hasil pengujian terhadap variasi *batch size*, diperoleh bahwa konfigurasi *batch size* 64 menghasilkan performa model terbaik dengan capaian *precision* sebesar 88%, *recall* sebesar 91%, F1-*Score* sebesar 90%, *Average* IoU sebesar 70,02% dan mAP@0.50 mencapai 93,60%. Pada *batch size* 16 dan 32 performa model cenderung rendah dan belum optimal dibandingkan *batch size* 64. Sedangkan pada *batch size* 128, meskipun *precision* mencapai nilai tertinggi yaitu 89%, namun terdapat penurunan *recall* dan mAP dibandingkan *batch size* 64, yang mengindikasikan adanya *overfitting* atau ketidakstabilan proses pembelajaran model. Berikut lampiran hasil pelatihan pengujian *batch size*:



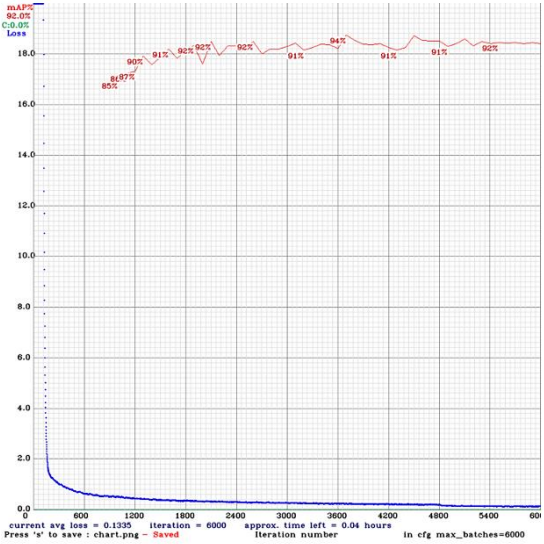
Gambar 38. Grafik Hasil Pelatihan *Batch Size* 16



Gambar 39. Grafik Hasil Pelatihan *Batch Size* 32



Gambar 40. Grafik Hasil Pelatihan *Batch Size* 64



Gambar 41. Grafik Hasil Pelatihan *Batch Size* 128

4.1.10. Hasil Pengujian *Accuracy* Dataset

Tabel 18. Hasil Pengujian Akurasi berdasarkan Variasi *Batch Size*

Parameter (*Batch Size)*	TP	FP	FN	Accuracy
16	328	54	167	59,74%
32	446	82	49	77,29%
64	450	60	45	81,08%
128	436	55	59	79,27%

Berdasarkan hasil pengujian tabel 18 terlihat bahwa *batch size* 64 menghasilkan akurasi tertinggi sebesar 81,08% yang merupakan kombinasi terbaik antara TP yang tinggi dan nilai FP serta FN yang rendah. Sehingga konfigurasi ini menjadi pilihan paling optimal dan digunakan sebagai model akhir dalam pengujian lanjutan yaitu implementasi pada sistem *counting*. Pemilihan ini berdasarkan pada hasil pengujian akurasi sehingga memperkuat keandalan model dalam berbagai kondisi.

4.1.11. Hasil Pengujian Performa Inferensi Model

```
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y:
nms_kind: greedynms (1), beta = 0.600000
Total BFLOPS 6.787
avg_outputs = 299663
 Allocate additional workspace_size = 26.22 MB
Loading weights from /content/gdrive/MyDrive/UJIPREDIKSI/yolov4-tiny-custom19_best.weights...
 seen 64, trained: 358 K-images (5 Kilo-batches_64)
Done! Loaded 38 layers from weights-file
 Detection layer: 30 - type = 28
 Detection layer: 37 - type = 28
/content/gdrive/MyDrive/UJIPREDIKSI/hasil_resize/Inferensi8_416x416.jpg: Predicted in 82.055000 milli-seconds.
Gas : 100%      (left_x:   89   top_y:  195   width:   18   height:   33)
Gas : 100%      (left_x:  127   top_y:  186   width:   16   height:   37)
Gas : 100%      (left_x:  177   top_y:   60   width:   14   height:   30)
Gas : 100%      (left_x:  211   top_y:   88   width:   14   height:   26)
```

Gambar 42. *Output* Pengujian Inferensi8_416x416

```
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1
nms_kind: greedynms (1), beta = 0.600000
Total BFLOPS 6.787
avg_outputs = 299663
 Allocate additional workspace_size = 26.22 MB
Loading weights from /content/gdrive/MyDrive/UJIPREDIKSI/yolov4-tiny-custom19_best.weights...
 seen 64, trained: 358 K-images (5 Kilo-batches_64)
Done! Loaded 38 layers from weights-file
 Detection layer: 30 - type = 28
 Detection layer: 37 - type = 28
/content/gdrive/MyDrive/UJIPREDIKSI/hasil_resize/Inferensi24_416x416.jpg: Predicted in 81.213000 milli-seconds.
Gas : 97%       (left_x:  153   top_y:  169   width:   12   height:   27)
Gas : 76%       (left_x:  159   top_y:  316   width:   13   height:   32)
Gas : 86%       (left_x:  196   top_y:  159   width:   12   height:   30)
Gas : 99%       (left_x:  228   top_y:  186   width:   14   height:   26)
```

Gambar 43. *Output* Pengujian Inferensi24_416x416

Pada gambar 42 yaitu uji Inferensi8_416x416 semua objek terdeteksi dengan *confidence* sebesar 100% dengan waktu prediksi sebesar 82.055000 *milli-seconds*. Sedangkan gambar 43 yaitu uji Inferensi24_416x416 nilai *confidence* berfluktuatif yaitu 97%, 76%, 86%, dan 99% dengan waktu deteksi sebesar 81.213000 *milli-seconds*.

Gambar 44. Hasil Pengujian Inferensi8_416x416 dan Inferensi24_416x416

Gambar 44 merupakan perbandingan hasil pengujian antara inferensi8_416x416 dan inferensi24_416x416. Gambar ini memperlihatkan visual deteksi objek secara berdampingan, yang bertujuan untuk menunjukkan perbedaan tingkat *confidence* antar dua uji coba dalam kondisi yang berbeda.

Tabel 19. Hasil Pengujian Kecepatan Deteksi (ms)

Uji	Nama File	Resolusi *Input*	Waktu Deteksi (ms)
1	Inferensi_416x416	416 x 416	80.68 ms
2	Inferensi2_416x416	416 x 416	81.20 ms
3	Inferensi3_416x416	416 x 416	83.66 ms
4	Inferensi4_416x416	416 x 416	82.77 ms
5	Inferensi5_416x416	416 x 416	83.29 ms
6	Inferensi6_416x416	416 x 416	84.12 ms
7	Inferensi7_416x416	416 x 416	85.09 ms
8	Inferensi8_416x416	416 x 416	82.05 ms
9	Inferensi9_416x416	416 x 416	83.67 ms
10	Inferensi10_416x416	416 x 416	82.06 ms
11	Inferensi11_416x416	416 x 416	78.33 ms
12	Inferensi12_416x416	416 x 416	77.92 ms
13	Inferensi13_416x416	416 x 416	78.99 ms
14	Inferensi14_416x416	416 x 416	78.16 ms
15	Inferensi15_416x416	416 x 416	79.34 ms
16	Inferensi16_416x416	416 x 416	80.78 ms
17	Inferensi17_416x416	416 x 416	81.41 ms
18	Inferensi18_416x416	416 x 416	79.32 ms
19	Inferensi19_416x416	416 x 416	80.17 ms
20	Inferensi20_416x416	416 x 416	82.16 ms
21	Inferensi21_416x416	416 x 416	80.38 ms
22	Inferensi22_416x416	416 x 416	83.93 ms
23	Inferensi23_416x416	416 x 416	82.07 ms
24	Inferensi24_416x416	416 x 416	81.21 ms
25	Inferensi25_416x416	416 x 416	77.59 ms
26	Inferensi26_416x416	416 x 416	81.65 ms
27	Inferensi27_416x416	416 x 416	80.25 ms
28	Inferensi28_416x416	416 x 416	80.55 ms
29	Inferensi29_416x416	416 x 416	78.41 ms
30	Inferensi30_416x416	416 x 416	81.11 ms
Rata – rata waktu deteksi			**81.07 ms**

Berdasarkan tabel pengujian di atas, terdapat data pengujian waktu deteksi tiap gambar dimulai dari 77.59 ms hingga 85 ms. Dari keseluruhan data uji, diperoleh rata-rata waktu deteksi sebesar 81.07 ms.

4.1.12. Hasil Pengujian FPS

Tabel 20. Pengujian FPS

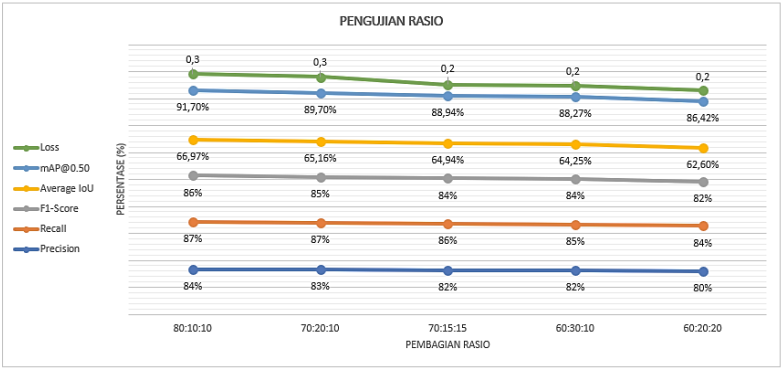
No	Video	Durasi Video	FPS	Avg FPS
1	Kondisi 1	2 menit 35 detik	21,7	18,26
2			16,8	
3			20,2	
4			20,4	
5			20,1	
6			18,9	
7			12,5	
8			22,2	
9			14,7	
10			15,1	
11	Kondisi 2	2 menit 48 detik	14,1	16,4
12			18,2	
13			12,5	
14			14,5	
15			16,8	
16			13,3	
17			19,6	
18			18,8	
19			19,2	
20			13,4	
21	Kondisi 3	2 menit 38 detik	16,2	18,51
22			19,2	
23			18,0	
24			20,2	
25			14,5	
26			19,9	
27			17,4	

28			18,9	
29			20,8	
30			20,0	

Berdasarkan tabel hasil pengujian di atas rata-rata FPS yang dihasilkan pada video kondisi 1 adalah 18,26 FPS, pada video kondisi 2 adalah 16,4 FPS, dan pada kondisi 3 adalah 18,51 FPS.

4.3.Analisis Hasil Pengujian

4.3.1. Analisis Pengaruh Rasio Dataset

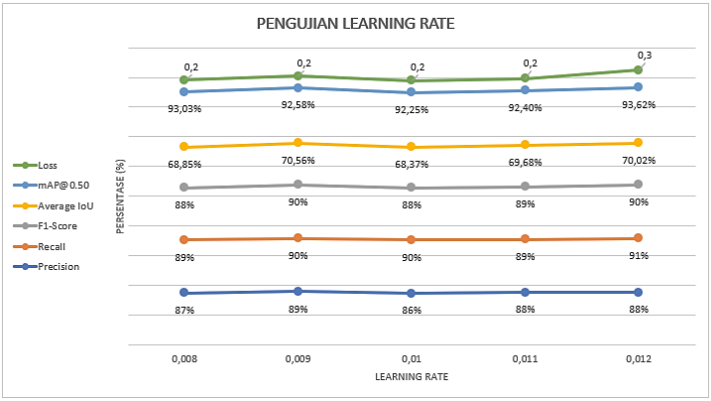


Gambar 45. Grafik Hasil Pengujian Rasio

Hasil pengujian menunjukkan bahwa pengujian variasi rasio pembagian data *train*, *valid*, *test* memberikan peningkatan performa pada model. Pada pelatihan awal dengan *max batches* 2000 performa model cukup rendah, berdasarkan kondisi tersebut, dilakukan penyesuaian dengan meningkatkan nilai *max batches* menjadi 6000 agar model memiliki waktu pelatihan yang lebih panjang dan mempelajari fitur objek secara menyeluruh sehingga sudah pasti terjadi peningkatan performa model. Setelah itu, dilakukan analisa berdasarkan grafik "Pengujian Rasio", terlihat bahwa perbedaan proporsi data *train*, *valid*, dan *test* secara langsung mempengaruhi performa model YOLOv4-*Tiny* pada berbagai metriks evaluasi, seperti *precision*, *recall*, F1-*Score*, *Average IoU*, dan mAP@0.50. Rasio 80 data *train*, 10 data valid, dan 10 data *test* atau bisa disebut 80:10:10 menghasilkan performa tertinggi dengan mAP sebesar 91,70%, *precision* sebesar 84%, dan F1-*Score* sebesar 86%, sedangkan rasio lainnya menunjukkan penurunan metriks secara bertahap.

Hal ini menunjukkan semakin banyak data pelatihan yang tersedia, semakin baik pula model dalam mempelajari pola dalam dataset, sehingga menghasilkan hasil prediksi yang lebih akurat. Sebaliknya ketika porsi data pelatihan dikurangi seperti pada rasio 60:30:10 atau 60:20:20 terjadi penurunan pada hampir semua metriks evaluasi. Hal ini karena model memiliki lebih sedikit informasi untuk belajar yang berpotensi model kurang mampu mengenali objek secara optimal.

4.3.2. Analisis Pengaruh Learning Rate



Gambar 46. Grafik Hasil Pengujian *Learning Rate*

Tabel 21. Perbandingan Hasil Optimasi Rasio dan Optimasi LR

Metriks	Optimasi (Rasio 80:10:10)	Optimasi (LR = 0,012)	Perubahan Performa (Meningkat/Menurun)
Precision	84,00%	88,00%	+ 4,76%
Recall	87,00%	91,00%	+ 4,59%
F1-Score	86,00%	90,00%	+ 4,65%
IoU	66,97%	70,02%	+ 4,55%
mAP@0.50	91,70%	93,62%	+ 2,09%
Loss	0,3	0,3	-

- *Persentase Precision* = $\frac{(88{,}00-84{,}00)}{84{,}00}$ x 100%
  = 4,76% (Meningkat)

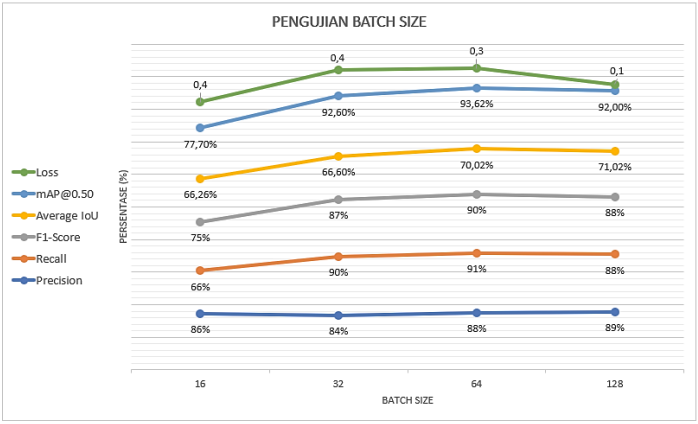
- *Persentase Recall* = $\frac{(91{,}00-87{,}00)}{87{,}00}$ x 100%
  = 4,59% (Meningkat)

- *Persentase F1-Score* $= \frac{(90{,}00-86{,}00)}{86{,}00}$ x 100%
  = 4,65% (Meningkat)

- *Persentase IoU* $= \frac{(70{,}02-66{,}97)}{66{,}97}$ x 100%
  = 4,55% (Meningkat)
- *Persentase mAP* $= \frac{(93{,}62-91{,}70)}{91{,}70}$ x 100%
  = 2,09% (Meningkat)

Berdasarkan tabel 21, optimasi *learning rate* sebesar 0,012 terbukti meningkatkan performa model dibandingkan dengan melakukan optimasi rasio dataset. Terlihat dari peningkatan pada seluruh metriks evaluasi, seperti *precision* dari 84,00% menjadi 88,00%, *recall* dari 87,00% menjadi 91,00%, F1-*Score* dari 86,00% menjadi 90,00%, IoU dari 66,97% menjadi 70,02%, nilai mAP@0.50 yang naik dari 91,70% menjadi 93,62%, dan nilai *loss* tetap stabil hal ini menandakan bahwa peningkatan performa tidak terjadi *overfitting*. Dengan demikian, pengujian *learning rate* mampu meningkatkan akurasi dan konsistensi model dalam mendeteksi objek gas.

4.3.3. Analisis Pengaruh Batch Size



Gambar 47. Grafik Hasil Pengujian *Batch Size (BS)*

Tabel 22. Analisis Performa Sebelum dan Sesudah Optimasi

Metriks	Sebelum Optimasi	Sesudah Optimasi (BS)	Perubahan Performa (Meningkat/Menurun)
Precision	78,00%	88,00%	+ 12,82%
Recall	72,00%	91,00%	+ 26,38%
F1-Score	75,00%	90,00%	+ 20,00%
IoU	59,03%	70,02%	+ 18,61%
mAP@0.50	75,00%	93,62%	+ 24,82%
Loss	0,2	0,3	Performa menurun

- *Persentase Precision* = $\frac{(88{,}00-78{,}00)}{78{,}00}$ x 100%
  = 12,82% (Meningkat)

- *Persentase Recall* = $\frac{(91{,}00-72{,}00)}{72{,}00}$ x 100%
  = 26,38% (Meningkat)

- *Persentase F1-Score* = $\frac{(90{,}00-75{,}00)}{75{,}00}$ x 100%
  = 20,00% (Meningkat)

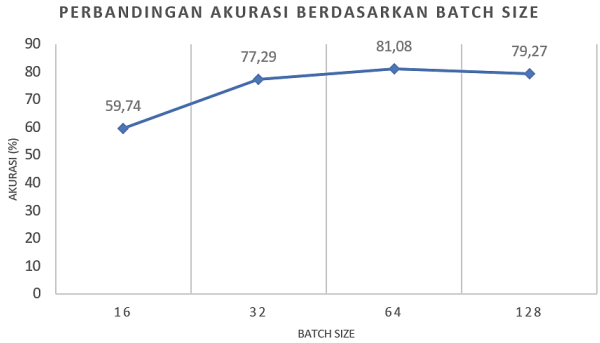
- *Persentase IoU* = $\frac{(70{,}02-59{,}03)}{59{,}03}$ x 100%
  = 18,61% (Meningkat)

- *Persentase mAP* = $\frac{(93{,}62-75{,}00)}{75{,}00}$ x 100%
  = 24,82% (Meningkat)

- *Persentase Loss* = $\frac{(0{,}2-0{,}3)}{0{,}3}$ x 100%
  = 33,33% (Meningkat)

Berdasarkan data sebelum dan sesudah optimasi *batch size* data menunjukkan peningkatan signifikan pada seluruh metriks evaluasi. *Precision* meningkat dari 78,00% menjadi 88,00%, *recall* dari 72,00% menjadi 91,00% dan F1-*Score* dari 75,00% menjadi 90,00%, nilai IoU naik dari 59,03% ke 70,02%, dan nilai mAP@0.50 meningkat dari 75,00% menjadi 93,62%. Peningkatan ini menunjukkan bahwa optimasi *batch size* tidak hanya berdampak pada kecepatan pelatihan, tetapi secara langsung memperbaiki kualitas prediksi akhir. Oleh karena itu, hasil ini menunjukkan keberhasilan optimasi dan menetapkan konfigurasi akhir yang optimal untuk digunakan pada tahap implementasi sistem.

4.3.4. Analisis *Accuracy* Dataset



Gambar 48. Grafik Hasil Perbandingan Akurasi Berdasarkan BS

Pada pengujian ini *batch size* 64 menghasilkan akurasi tertinggi yaitu 81,08%, sedangkan *batch size* 16 memberikan hasil akurasi terendah yaitu 59,74%. Berdasarkan referensi jurnal, *batch size* yang kecil dapat memberikan gradien bervariasi dan membantu generalisasi model. Namun, pada penelitian ini dataset yang digunakan merupakan dataset *custom* dengan objek gas dan hanya terdiri dari satu kelas sehingga model tidak memiliki beban untuk membedakan berbagai objek dan model tidak memerlukan generalisasi kompleks antar kelas. Oleh karena itu, *batch size* besar seperti 64 justru mampu menangkap pola secara lebih stabil dan efisien karena gradien yang dihasilkan dari satu *batch* sudah cukup representatif untuk seluruh dataset.

4.3.5. Analisis Performa Inferensi Model

Pengujian inferensi model menghasilkan rata-rata waktu deteksi sebesar 81.07 ms menunjukkan performa model YOLOv4-*Tiny* cukup efisien dalam memproses gambar dengan resolusi 416 x 416. Pengujian ini dilakukan dengan menyesuaikan data uji terhadap arsitektur model yang digunakan, sesuai dengan konfigurasi *width* dan *height* pada *file* .cfg saat proses pelatihan. Hasil inferensi menghasilkan *output* berupa nilai *confidence* terhadap objek yang terdeteksi pada setiap gambar uji. Berdasarkan hasil pengujian ini terdapat dua kondisi yang dapat dijadikan perbandingan. Pertama, pada gambar Inferensi8_416x416, model mendeteksi empat objek gas dengan nilai *confidence* sebesar 100% yang menunjukkan bahwa model mampu mengenali objek dengan tingkat keyakinan tinggi, faktor yang mempengaruhi hal tersebut yaitu kondisi lingkungan, posisi objek yang terlihat jelas atau tidak ada visual yang menghalangi proses deteksi. Hal ini berbeda dengan gambar uji Inferensi24_416x416 yang mendapatkan nilai

confidence berfluktuasi, mulai dari 0,76 hingga 0,99. Terlihat dari gambar tersebut terdapat adanya pergerakan yang menutupi sebagian objek sehingga mempengaruhi tingkat kepercayaan pada uji inferensi model. Meskipun secara jumlah objek yang terdapat di gambar uji adalah sama, namun nilai *confidence* sangat berpengaruh dari faktor eksternal yang dapat mempengaruhi kinerja model dalam mengenali objek.

4.3.6. Analisis Pengujian FPS

Berdasarkan pengujian FPS pada tiga kondisi berbeda, diperoleh nilai rata-rata FPS dalam rentang 16,4 hingga 18,51 FPS. Kondisi 3 menunjukkan performa terbaik dengan rata-rata FPS 18,51. Hal ini mungkin dipengaruhi oleh tingkat kompleksitas visual yang lebih rendah dan pencahayaan yang lebih baik dibandingkan kondisi 1 maupun kondisi 2. Sementara itu, kondisi 1 mendapat nilai rata-rata FPS sebesar 18,26 dan kondisi 2 mendapatkan nilai FPS terendah dengan rata-rata FPS 16,4. Meskipun terdapat perbedaan nilai FPS antar kondisi, hasil yang diperoleh menunjukkan bahwa sistem memiliki respon kecepatan pemrosesan yang cukup baik. Dengan nilai rata-rata FPS yang konsisten di atas 15, sistem memiliki potensi untuk diterapkan dalam skenario pemantauan *real time*, tergantung pada dukungan spesifikasi perangkat keras dan kebutuhan aplikasi di lapangan.

Bab 5. Kesimpulan dan Saran

5.1. Kesimpulan

Berdasarkan hasil pengujian, analisis, dan pembahasan yang telah dilakukan terhadap sistem deteksi gas LPG 3 kg menggunakan model YOLOv4-*Tiny*, maka diperoleh beberapa Kesimpulan sebagai berikut:

1. Variasi rasio pembagian dataset memberikan dampak signifikan terhadap kinerja model. Rasio 80:10:10 memberikan performa terbaik dengan mAP@0.50 sebesar 91,70%, *precision* 84%, dan F1-*Score 86%.* Hal ini menunjukkan bahwa semakin proporsi data pelatihan, semakin baik model dalam mengenali pola objek.
2. Optimasi *learning rate* ke nilai 0,012 meningkatkan performa model pada seluruh metriks evaluasi, terlihat dari peningkatan mAP@0.50 menjadi 93,62%.
3. Optimasi *batch size* mampu mempercepat dan menstabilkan proses pembelajaran model untuk dataset satu kelas terlihat dari peningkatan *recall* yang meningkat sebesar 26,38% dan mAP sebesar 24,82%.
4. Model memiliki kemampuan inferensi yang cepat dan efisien dengan rata-rata waktu deteksi sebesar 81.07 ms dan memiliki sensitivitas yang baik terhadap kondisi eksternal.
5. Berdasarkan hasil pengujian FPS sistem deteksi mampu di implementasi secara *real time* dengan respon yang cukup cepat namun tetap perlu memperhatikan spesifikasi perangkat (*device*) yang digunakan.

5.2. Saran

1. Pengaturan konfigurasi *max_batches* lebih tinggi dapat dipertimbangkan, karena berdasarkan pengujian menunjukkan adanya peningkatan signifikan pada nilai *precision, recall, F1-Score* sehingga berpotensi meningkatkan performa model secara keseluruhan.
2. Disarankan untuk mengeksplorasi lebih banyak parameter untuk di uji, seperti *momentum, decay, subdivisions* yang berpotensi mempengaruhi proses pembelajaran model.
3. Penelitian ini masih *manual tuning* disarankan pada penelitian selanjutnya menerapkan teknik *hyperparameter tuning* otomatis seperti *Bayesian Optimization, Random Search,* atau *Genetic Algorithm* untuk menemukan kombinasi lebih optimal.

Daftar Pustaka

[1] G. N. Goenawan, A. N. Tjondrowiguno, and Liliana, “Pengenalan Rambu Lalu Lintas di Indonesia Secara Real-time Menggunakan YOLOv4-tiny.,” *Jurnal Infra*, vol. 10, pp. 198–204, 2022.

[2] M. A. Al Haadi, C. Setianingsih, and T. W. Purboyo, “Sistem Pemantauan Aktivitas Keseharian Lansia Berbasis Deteksi Objek Menggunakan Algoritma YOLO,” *E-Proceedings of Engineering*, vol. 10, p. 836, 2023.

[3] P. Hidayatullah, *Buku Sakti Deep Learning Computer Vision Menggunakan YOLO untuk Pemula* , 3rd ed. Cimahi: Stunning Vision AI Academy, 2023.

[4] A. Rifai Arganata, “Analisis Perhitungan Bibit Ikan Gurame Menggunakan Webcam Dengan Metode YOLO (You Only Look Once),” Surabaya, Aug. 2020.

[5] P. Anggia Cahyani, “Sistem Perhitungan Kendaraan Menggunakan Algoritma YOLOv5 dan Deepsort,” Bandar Lampung, Sep. 2023.

[6] M. Elgendy, *Deep Learning For Vision System*, 1st ed. Shelter Island: Manning Publications Co., 2020.

[7] S. R. Dewi, “Deep Learning Object Detection Pada Video Menggunakan Tensorflow Dan Convolutional Neural Network,” Yogyakarta, Jan. 2018.

[8] R. Gelar Guntara, “Pemanfaatan Google Colab Untuk Aplikasi Pendeteksian Masker Wajah Menggunakan Algoritma Deep Learning YOLOv7,” *Jurnal Teknologi Dan Sistem Informasi Bisnis*, vol. 5, no. 1, pp. 55–60, Feb. 2023, doi: 10.47233/jteksis.v5i1.750.

[9] I. Chatterjee, *Machine Learning and Its Application: A Quick Guide for Beginners*, 1st ed. Sharjah, UAE: Bentham Science Publishers, 2021.

[10] Z. Jiang, L. Zhao, S. Li, and Y. Jia, “Real-time object detection method based on improved YOLOv4-tiny,” Nov. 2020, doi: 10.48550/arXiv.2011.04244.

[11] R. Moh Yusup, A. Faris Anugrah, D. Desmonda Muslimah, S. Mentari Widya Ningrum Permana, and S. Yuliani, “Pendeteksian Objek Menggunakan OpenCV dan Metode YOLOv4-Tiny Untuk Membantu Tunanetra,” *Journal of Computer Science and Information Technology (JCSIT)*, vol. 1, Mar. 2023.

[12] R. D. Saputra, “Pengembangan Sistem Deteksi Objek Pada Produk Retail Dengan Arsitektur YOLOv4-Tiny,” Yogyakarta, Jul. 2023.

[13] T. A. A. H. Kusuma, K. Usman, and S. Saidah, “People Counting For Public Transportations Using You Only Look Once Method,” *Jurnal Teknik Informatika (Jutif)*, vol. 2, no. 1, pp. 57–66, Jun. 2021, doi: 10.20884/1.jutif.2021.2.2.77.

[14] M. S. Khatami, “Deteksi Kendaraan Menggunakan Algoritma You Only Look Once (YOLO) V3,” Yogyakarta, May 2022.

[15] S. A. Harjanto, M. Nurhaliza, and J. H. T. Sagala, “Optimalisasi Deteksi Anomali Untuk Pemfilteran Log dan Integrasi Dengan SIEM Menggunakan Machine Learning,” *Madani: Jurnal Ilmiah Multidisiplin*, vol. 2, no. 7, pp. 266–275, Jun. 2024, doi: 10.5281/zenodo.12562358.

[16] F. Fadhlurrahman, “Livestream Pengenalan Penyakit Padi Menggunakan Algoritma YOLO Berbasis Raspberry,” Depok, Oct. 2022.

[17] V. Marcellino, V. C. Mawardi, and N. J. Pradana, “Pendeteksian Jumlah Penumpang Yang Masuk Berdasarkan CCTV Pada Pintu Bus Dengan Metode YOLO,” *Jurnal Ilmu Komputer dan Sistem Informasi*, vol. 10, 2022.

[18] Y. Fadhillah, B. Berlinton, S. Karya, and Samin, “Advancing Vehicle Detection With YOLOv9: Integrating Programmable Gradient Information and Efficient Layer Aggreation,” vol. 5, Aug. 2024.

[19] S. Fitriyati Prisunia, “Pemanfaatan Jetson Nano NVDIA Untuk Mendeteksi Penggunaan Masker Secara Real-Time Menggunakan OPENCV Python,” Semarang, Mar. 2023.

[20] K. S. Nugroho, “Confusion Matrix untuk Evaluasi Model pada Supervised Learning,” Medium. [Online]. Available: https://ksnugroho.medium.com/confusion-matrix-untuk-evaluasi-model-pada-unsupervised-machine-learning-bc4b1ae9ae3f

Biodata

Nama	: Widya Pratiwi
TTL	: Batam, 28 Januari 2002
Agama	: Islam
Alamat	: Taman Marchelia Blok C No. 154
Email	: widyaprxtiwi@gmail.com
Riwayat Pendidikan	SMA/SMK : SMK Negeri 1 Batam
	SMP : SMP Negeri 31 Batam

Lampiran

Program Pelatihan Pada *Google Colab*

```
# Verify CUDA
!/usr/local/cuda/bin/nvcc –version

# Check cuDNN
!cat /usr/include/cudnn_version.h | grep CUDNN_MAJOR -A 2

# Clone darknet repository
!git clone https://github.com/AlexeyAB/darknet

# Mount Google Drive
%cd
from google.colab import drive
drive.mount ('/content/gdrive')

# Edit Makefile to enable GPU and OpenCV
%cd /content/darknet
!sed -i 's/OPENCV=0/OPENCV=1/' Makefile
!sed -i 's/GPU=0/GPU=1/' Makefile
!sed -i 's/CUDNN=0/CUDNN=1/' Makefile
!sed -i 's/CUDNN_HALF=0/CUDNN_HALF=1/' Makefile
!sed -i 's/LIBSO=0/LIBSO=1/' Makefile

# Build darknet
!make

# Salin semua file dari yolov4-tiny'folder'
# ke direktori 'darknet' di Colab VM
# hapus semua file lain dari folder data dan cfg kecuali folder label didalam
folder data
%cd data/
!find -maxdepth 1 -type f -exec rm -rf {} \;
%cd ..
%rm -rf cfg/
%mkdir cfg

import glob
```

```
    def generate_list(folder_path, output_file):
        with open(output_file, 'w') as f:
            for img_path in glob.glob(folder_path + '/*.jpg'):
                f.write(img_path + '\n')

    # Ganti ke nama folder kamu di Google Drive
    generate_list('/content/gdrive/MyDrive/UJILR5/train',
'/content/darknet/data/train.txt')
    generate_list('/content/gdrive/MyDrive/UJILR5/valid',
'/content/darknet/data/valid.txt')
    generate_list('/content/gdrive/MyDrive/UJILR5/test',
'/content/darknet/data/test.txt')

    # Menyalin yolov4-tiny-custom.cfg ke folder /content/darknet/cfg/
    !cp /content/gdrive/MyDrive/UJILR5/yolov4-tiny-custom19.cfg
/content/darknet/cfg/

    !ls /content/darknet/cfg/

    # Menyalin obj.names ke folder /content/darknet/data/
    !cp /content/gdrive/MyDrive/UJILR5/obj.names /content/darknet/data/

    # Menyalin obj.data ke folder /content/darknet/data/
    !cp /content/gdrive/MyDrive/UJILR5/obj.data /content/darknet/data/

    # Daftarkan isi folder data untuk memeriksa apakah file train.txt dan test.txt
telah dibuat
    !ls /content/darknet/data

    !cat /content/darknet/data/train.txt
    !cat /content/darknet/data/valid.txt
    !cat /content/darknet/data/test.txt

    #download the newly released yolov4-tiny weights
    %cd /content/darknet
    !wget
https://github.com/AlexeyAB/darknet/releases/download/darknet_yolo_v4_pre
/yolov4-tiny.weights
    !wget
https://github.com/AlexeyAB/darknet/releases/download/darknet_yolo_v4_pre
/yolov4-tiny.conv.29
```

```
    # Run detection
    !./darknet detector train /content/darknet/data/obj.data
/content/darknet/cfg/yolov4-tiny-custom19.cfg /content/darknet/yolov4-
tiny.conv.29 -dont_show -map

    # fungsi untuk menampilkan citra
    def imShow(path):
      import cv2
      import matplotlib.pyplot as plt
      %matplotlib inline

      image = cv2.imread(path)
      height, width = image.shape[:2]
      resized_image = cv2.resize(image,(3*width, 3*height), interpolation =
cv2.INTER_CUBIC)

      fig = plt.gcf()
      fig.set_size_inches(18, 10)
      plt.axis("off")
      #plt.rcParams['figure.figsize'] = [10, 5]
      plt.imshow(cv2.cvtColor(resized_image, cv2.COLOR_BGR2RGB))
      plt.show()

    imShow('chart.png')
```

Program Inferensi Deteksi Pada *Google Colab*

```
!./darknet detector test /content/darknet/data/obj.data
/content/darknet/cfg/yolov4-tiny-custom19.cfg
/content/gdrive/MyDrive/UJIPREDIKSI/yolov4-tiny-custom19_best.weights -
dont_show -ext_output
/content/gdrive/MyDrive/UJIPREDIKSI/hasil_resize/Inferensi20_416x416.jpg

from IPython.display import Image, display
display(Image(filename='predictions.jpg'))

!cp predictions.jpg
/content/gdrive/MyDrive/UJIPREDIKSI/Inferensi20_416x416.jpg
```

Program FPS Pada *Google Colab*

```
!./darknet detector demo /content/darknet/data/obj.data
/content/darknet/cfg/yolov4-tiny-custom19.cfg
/content/gdrive/MyDrive/UJILR5/yolov4-tiny-custom19_best.weights
/content/gdrive/MyDrive/UJIFPS/Sore4.mp4 -ext_output -thresh 0.25 -
out_filename /content/gdrive/MyDrive/UJIFPS/output7_Sore4.mp4
```

Hasil Uji Inferensi (Waktu Deteksi)

Inferensi1_416x416

Inferensi2_416x416

Inferensi3_416x416

Inferensi4_416x416

Inferensi5_416x416

Inferensi6_416x416

Inferensi7_416x416

Inferensi8_416x416

Inferensi9_416x416

Inferensi10_416x416

Inferensi11_416x416

Inferensi12_416x416

Inferensi13_416x416

Inferensi14_416x416

Inferensi15_416x416

Inferensi16_416x416

Inferensi17_416x416

Inferensi18_416x416

Inferensi19_416x416

Inferensi20_416x416

Inferensi21_416x416

Inferensi22_416x416

Inferensi23_416x416

Inferensi24_416x416

Inferensi25_416x416

Inferensi26_416x416

Inferensi27_416x416

Inferensi28_416x416

Inferensi29_416x416

Inferensi30_416x416

Hasil Uji FPS

Kondisi 1

```
 FPS:21.3         AVG_FPS:21.7
 OpenCV exception: show_image_mat

  cvWriteFrame
 HObjects:


 FPS:21.4         AVG_FPS:21.7
 OpenCV exception: show_image_mat
 Stream closed.

  cvWriteFrame
 input video stream closed.
  closing... closed!output video writer closed.
```

```
FPS:19.2         AVG_FPS:16.8
OpenCV exception: show_image_mat

 cvWriteFrame
HObjects:


FPS:19.6         AVG_FPS:16.8
Stream closed.
OpenCV exception: show_image_mat

 cvWriteFrame
input video stream closed.
 closing... closed!output video writer closed.
```

```
FPS:21.6          AVG_FPS:20.2
OpenCV exception: show_image_mat

 cvWriteFrame
HObjects:


FPS:22.6          AVG_FPS:20.2
OpenCV exception: show_image_mat
Stream closed.

 cvWriteFrame
input video stream closed.
 closing... closed!output_video_writer closed.
```

```
FPS:18.6          AVG_FPS:20.4
OpenCV exception: show_image_mat

 cvWriteFrame
HObjects:


FPS:18.5          AVG_FPS:20.4
OpenCV exception: show_image_mat
Stream closed.

 cvWriteFrame
input video stream closed.
 closing... closed!output_video_writer closed.
```

```
FPS:20.1          AVG_FPS:20.1
OpenCV exception: show_image_mat

 cvWriteFrame
HObjects:


FPS:20.3          AVG_FPS:20.1
Stream closed.
OpenCV exception: show_image_mat

 cvWriteFrame
input video stream closed.
 closing... closed!output_video_writer closed.
```

```
FPS:20.1         AVG_FPS:18.9
OpenCV exception: show_image_mat

 cvWriteFrame
HObjects:


FPS:20.2         AVG_FPS:18.9
OpenCV exception: show_image_mat
Stream closed.

 cvWriteFrame
input video stream closed.
 closing... closed!output_video_writer closed.
```

```
FPS:22.2         AVG_FPS:12.5
OpenCV exception: show_image_mat

 cvWriteFrame
HObjects:


FPS:22.2         AVG_FPS:12.5
OpenCV exception: show_image_mat
Stream closed.

 cvWriteFrame
input video stream closed.
 closing... closed!output_video_writer closed.
```

```
FPS:17.1         AVG_FPS:22.2
OpenCV exception: show_image_mat

 cvWriteFrame
HObjects:


FPS:16.9         AVG_FPS:22.2
OpenCV exception: show_image_mat
Stream closed.

 cvWriteFrame
input video stream closed.
 closing... closed!output_video_writer closed.
```

```
FPS:23.1          AVG_FPS:14.7
OpenCV exception: show_image_mat

 cvWriteFrame
HObjects:


FPS:22.8          AVG_FPS:14.7
OpenCV exception: show_image_mat
Stream closed.

 cvWriteFrame
input video stream closed.
 closing... closed!output_video_writer closed.
```

```
FPS:13.2          AVG_FPS:15.1
OpenCV exception: show_image_mat

 cvWriteFrame
HObjects:


FPS:13.3          AVG_FPS:15.1
OpenCV exception: show_image_mat
Stream closed.

 cvWriteFrame
input video stream closed.
 closing... closed!output_video_writer closed.
```

Kondisi 2

```
 Gas : 47%       (left_x:  757   top_y:    5   width:   62   height:   34)
 Gas : 36%       (left_x:  367   top_y:    8   width:   50   height:   35)
 Gas : 27%       (left_x:  148   top_y:   86   width:   30   height:   46)

 FPS:21.9         AVG_FPS:14.1
 OpenCV exception: show_image_mat
 Stream closed.

  cvWriteFrame
 input video stream closed.
  closing... closed!output_video_writer closed.
```

```
 cvWriteFrame
HObjects:


FPS:21.7         AVG_FPS:18.2
OpenCV exception: show_image_mat

 cvWriteFrame
HObjects:


FPS:21.8         AVG_FPS:18.2
OpenCV exception: show_image_mat
Stream closed.

 cvWriteFrame
input video stream closed.
 closing... closed!output_video_writer closed.
```

```
FPS:18.1         AVG_FPS:12.5
OpenCV exception: show_image_mat

 cvWriteFrame
HObjects:


FPS:18.3         AVG_FPS:12.5
Stream closed.
OpenCV exception: show_image_mat

 cvWriteFrame
input video stream closed.
 closing... closed!output_video_writer closed.
```

```
FPS:17.5         AVG_FPS:14.5
OpenCV exception: show_image_mat

 cvWriteFrame
HObjects:


FPS:19.3         AVG_FPS:14.5
OpenCV exception: show_image_mat
Stream closed.

 cvWriteFrame
input video stream closed.
 closing... closed!output_video_writer closed.
```

```
FPS:20.0         AVG_FPS:16.8
OpenCV exception: show_image_mat

 cvWriteFrame
HObjects:


FPS:20.3         AVG_FPS:16.8
Stream closed.
OpenCV exception: show_image_mat

 cvWriteFrame
input video stream closed.
 closing... closed!output_video_writer closed.
```

```
FPS:20.8         AVG_FPS:13.3
OpenCV exception: show_image_mat

 cvWriteFrame
HObjects:


FPS:21.0         AVG_FPS:13.3
OpenCV exception: show_image_mat
Stream closed.

 cvWriteFrame
input video stream closed.
 closing... closed!output_video_writer closed.
```

```
FPS:23.0         AVG_FPS:19.6
OpenCV exception: show_image_mat

 cvWriteFrame
HObjects:


FPS:22.8         AVG_FPS:19.6
OpenCV exception: show_image_mat
Stream closed.

 cvWriteFrame
input video stream closed.
 closing... closed!output_video_writer closed.
```

```
FPS:18.5         AVG_FPS:18.8
OpenCV exception: show_image_mat

 cvWriteFrame
HObjects:


FPS:18.4         AVG_FPS:18.8
OpenCV exception: show_image_mat
Stream closed.

 cvWriteFrame
input video stream closed.
 closing... closed!output_video_writer closed.
```

```
   FPS:19.8         AVG_FPS:19.2
   OpenCV exception: show_image_mat

    cvWriteFrame
   HObjects:


   FPS:19.9         AVG_FPS:19.2
   OpenCV exception: show_image_mat
   Stream closed.

    cvWriteFrame
   input video stream closed.
    closing... closed!output_video_writer closed.
```

```
FPS:21.6         AVG_FPS:13.4
OpenCV exception: show_image_mat

 cvWriteFrame
HObjects:


FPS:21.8         AVG_FPS:13.4
OpenCV exception: show_image_mat
Stream closed.

 cvWriteFrame
input video stream closed.
 closing... closed!output_video_writer closed.
```

Kondisi 3

```
FPS:18.9         AVG_FPS:16.2
OpenCV exception: show_image_mat

 cvWriteFrame
HObjects:


FPS:19.2         AVG_FPS:16.2
Stream closed.
OpenCV exception: show_image_mat

 cvWriteFrame
input video stream closed.
 closing... closed!output_video_writer closed.
```

```
FPS:20.2         AVG_FPS:19.2
OpenCV exception: show_image_mat

 cvWriteFrame
HObjects:


FPS:20.3         AVG_FPS:19.2
Stream closed.
OpenCV exception: show_image_mat

 cvWriteFrame
input video stream closed.
 closing... closed!output_video_writer closed.
```

```
 FPS:20.7         AVG_FPS:18.0
 OpenCV exception: show_image_mat

  cvWriteFrame
 HObjects:


 FPS:21.0         AVG_FPS:18.0
 OpenCV exception: show_image_mat
 Stream closed.

  cvWriteFrame
 input video stream closed.
  closing... closed!output_video_writer closed.
```

```
FPS:22.3          AVG_FPS:20.2
OpenCV exception: show_image_mat

 cvWriteFrame
HObjects:


FPS:23.8          AVG_FPS:20.2
OpenCV exception: show_image_mat
Stream closed.

 cvWriteFrame
input video stream closed.
 closing... closed!output_video_writer closed.
```

```
FPS:14.4          AVG_FPS:14.5
OpenCV exception: show_image_mat

 cvWriteFrame
HObjects:


FPS:15.0          AVG_FPS:14.5
OpenCV exception: show_image_mat
Stream closed.

 cvWriteFrame
input video stream closed.
 closing... closed!output_video_writer closed.
```

```
FPS:23.4          AVG_FPS:19.9
OpenCV exception: show_image_mat

 cvWriteFrame
HObjects:


FPS:25.2          AVG_FPS:19.9
OpenCV exception: show_image_mat
Stream closed.

 cvWriteFrame
input video stream closed.
 closing... closed!output_video_writer closed.
```

```
FPS:15.4         AVG_FPS:17.4
OpenCV exception: show_image_mat

 cvWriteFrame
HObjects:


FPS:15.2         AVG_FPS:17.4
OpenCV exception: show_image_mat
Stream closed.

 cvWriteFrame
input video stream closed.
 closing... closed!output_video_writer closed.
```

```
FPS:14.2         AVG_FPS:18.9
OpenCV exception: show_image_mat

 cvWriteFrame
HObjects:


FPS:14.1         AVG_FPS:18.9
OpenCV exception: show_image_mat
Stream closed.

 cvWriteFrame
input video stream closed.
 closing... closed!output_video_writer closed.
```

```
FPS:23.6         AVG_FPS:20.8
OpenCV exception: show_image_mat

 cvWriteFrame
HObjects:


FPS:23.7         AVG_FPS:20.8
OpenCV exception: show_image_mat
Stream closed.

 cvWriteFrame
input video stream closed.
 closing... closed!output_video_writer closed.
```

```
FPS:20.7         AVG_FPS:20.0
OpenCV exception: show_image_mat

 cvWriteFrame
HObjects:


FPS:20.9         AVG_FPS:20.0
OpenCV exception: show_image_mat
Stream closed.

 cvWriteFrame
input video stream closed.
 closing... closed!output_video_writer closed.
```