

**LAPORAN PROJECT AKHIR  
PRAKTIKUM KECERDASAN BUATAN**

**Deteksi Jenis Minuman dalam Botol dan Menampilkan Informasi Harga Otomatis  
secara Real-Time Menggunakan Convolutional Neural Network (CNN)**



Disusun Oleh:

<b>Muhamad Fatan Nayaka</b>	<b>: 23090620076</b>
<b>Muhammad Athaozih</b>	<b>: 230906200</b>
<b>Ahmad Ilman Nadziron</b>	<b>: 230906200</b>

**D4 TEKNIK ELEKTRONIKA  
DEPARTEMEN TEKNIK ELEKTRO DAN ELEKTRONIKA  
FAKULTAS VOKASI  
UNIVERSITAS NEGERI YOGYAKARTA  
2025**

# BAB I

## PENDAHULUAN

### 1.1 Judul Project

Deteksi Jenis Minuman dalam Botol dan Menampilkan Informasi Harga Otomatis secara Real-Time Menggunakan Convolutional Neural Network (CNN)

### 1.2 Latar Belakang

Perkembangan teknologi kecerdasan buatan dan *computer vision* telah membuka peluang baru dalam otomasi sistem pengenalan produk, khususnya pada sektor ritel. Sistem identifikasi produk berbasis kamera kini semakin relevan karena lebih praktis dibandingkan metode tradisional seperti pemindaian barcode yang memerlukan interaksi manual.

Teknologi deep learning, khususnya model deteksi objek YOLOv8, menjadi solusi yang efektif karena mampu melakukan pengenalan objek secara cepat dan akurat dalam waktu nyata (*real-time*), bahkan pada perangkat dengan sumber daya terbatas.

Pada penelitian ini, YOLOv8 digunakan untuk mendeteksi jenis minuman dalam botol melalui input webcam dan secara otomatis menampilkan informasi harga tanpa basis data eksternal. Sistem ini menunjukkan potensi penerapan teknologi otomatisasi di bidang ritel, seperti kasir otomatis, inventori berbasis kamera, dan *cashierless system* pada masa depan.

### 1.3 Dasar Teori

Model yang digunakan dalam proyek ini adalah **YOLOv8 (You Only Look Once Version 8)**, yaitu salah satu arsitektur deep learning modern yang dirancang khusus untuk tugas *object detection* dan *image classification* dengan performa tinggi namun tetap efisien. YOLOv8 dipilih karena mampu bekerja secara real-time dengan latensi rendah, bahkan pada perangkat dengan spesifikasi terbatas seperti laptop dengan webcam bawaan.

YOLOv8 juga mendukung metode *transfer learning*, sehingga proses pelatihan menjadi lebih cepat dan tidak memerlukan dataset yang sangat besar. Selain itu, model ini fleksibel dan dapat dioptimalkan untuk mendeteksi objek sederhana seperti kemasan minuman dengan akurasi yang tinggi.

Langkah-Langkah Pelatihan Model :

#### 1. Persiapan Dataset

Dataset berisi gambar berbagai jenis minuman. Gambar dikumpulkan melalui pengambilan foto secara langsung maupun dari sumber lain seperti internet.

#### 2. Preprocessing Dataset

Seluruh gambar diberi label menggunakan *annotation tool* seperti Roboflow atau LabelImg, kemudian dilakukan preprocessing berupa:

- Resize ke resolusi yang sesuai (misalnya 416×416 piksel).
- Normalisasi piksel.
- *Data augmentation* untuk meningkatkan variasi dataset seperti flip, rotation, blur, atau brightness adjustment.

### **3. Pelatihan Model YOLOv8**

Model YOLOv8 dilatih untuk mengenali ciri khas dari masing-masing produk berdasarkan fitur visual seperti warna, bentuk botol, pola desain label, dan logo.

### **4. Export dan Deployment Model**

Setelah model mencapai performa terbaik (misalnya dengan metrik mAP, precision, dan recall yang optimal), model disimpan dalam format .pt dan digunakan untuk inferensi real-time.

## **1.4 Tujuan**

Tujuan utama dari proyek ini adalah untuk mengembangkan sistem berbasis *machine learning* yang mampu mengenali jenis minuman dalam botol secara otomatis menggunakan kamera secara real-time, serta menampilkan informasi harga dari produk yang terdeteksi secara langsung di layar.

Dalam implementasinya, sistem akan menggunakan webcam untuk menangkap gambar atau video dari objek minuman yang disorot. Kemudian, model *Convolutional Neural Network (CNN)* yang telah dilatih sebelumnya akan melakukan proses klasifikasi untuk menentukan jenis minuman, seperti Aqua, Teh Botol, Coca-Cola, Sprite, Fanta, dll. Setelah jenis minuman berhasil dikenali, program akan menampilkan nama produk dan harga yang sesuai di layar.

## **BAB II ALAT DAN BAHAN**

### **1. Perangkat Keras**

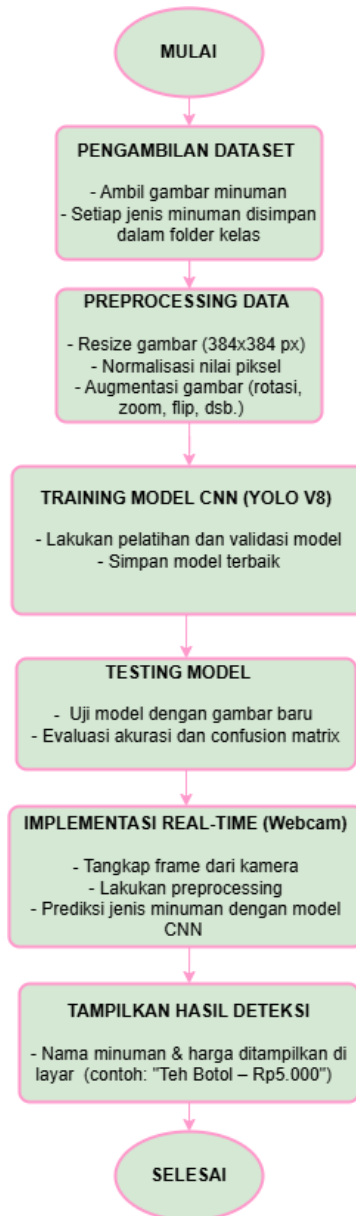
- Laptop
- Webcam (opsional)
- Minuman

### **2. Perangkat Lunak**

- Python IDE (Jupyter/VS Code/Google Colab)
- Library (ultralytics, cv2)

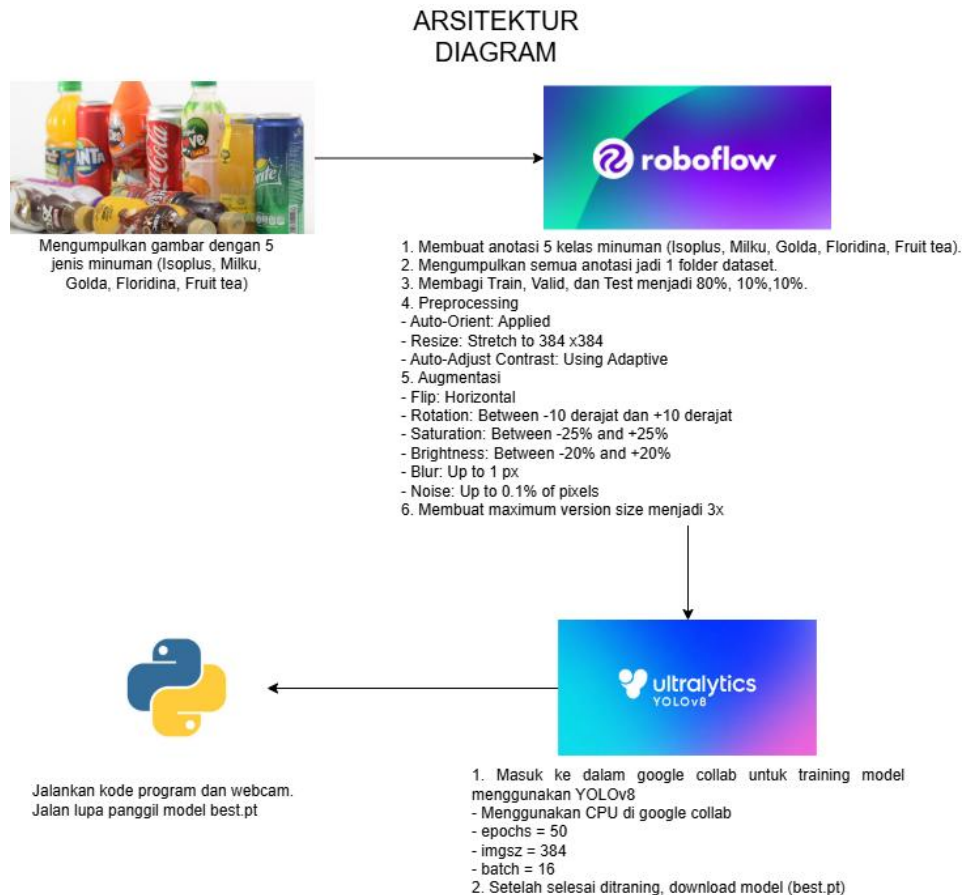
## **BAB III METODOLOGI PRAKTIKUM**

### 3.1 Diagram Alir



Gambar 1. Flowchart

## 3.2 Arsitektur Sistem



Gambar 2. Arsitektur Diagram

## 3.3 Kode Program

### 3.1 Instalasi Library Ultralytics

```
# SEL 1: Install ultralytics (sekali saja)
!pip install ultralytics==8.3.6 -q
```

Tanda seru (!) pada awal perintah menunjukkan bahwa instalasi dijalankan sebagai perintah terminal melalui lingkungan *notebook* seperti Google Colab atau Jupyter Notebook. Penentuan versi (==8.3.6) dilakukan agar penggunaan model tetap konsisten dan menghindari ketidakcocokan fitur apabila terjadi pembaruan library di masa mendatang. Sementara itu, argumen -q (*quiet mode*) digunakan untuk meminimalkan tampilan output selama proses instalasi sehingga hasil eksekusi lebih ringkas.

### 3.2 Uploading ZIP dataset YOLOv8

```
# SEL 2: Upload ZIP dataset YOLOv8 Anda
from google.colab import files
uploaded = files.upload() # Upload ZIP Anda
!unzip -q *.zip -d /content/dataset
```

Bagian `files.upload()` berfungsi untuk membuka dialog unggah sehingga pengguna dapat memilih file dataset dari perangkat lokal. File yang diunggah kemudian disimpan secara sementara di direktori kerja Colab. Perintah selanjutnya `!unzip -q *.zip -d /content/dataset` digunakan untuk mengekstrak seluruh file ZIP yang telah diunggah ke folder tujuan `/content/dataset`. Opsi `-q` (*quiet mode*) digunakan untuk mengurangi tampilan output proses ekstraksi agar lebih ringkas.

### 3.3 Training Menggunakan CPU

```
# =====
# 3. TRAINING DARI AWAL → GRAFIK LENGKAP
# =====
from ultralytics import YOLO
import os
os.environ["WANDB_MODE"] = "disabled" # matikan wandb total

model = YOLO('yolov8n.pt') # mulai dari pretrained

model.train(
    data='/content/dataset/data.yaml',
    epochs=50, # 50
    imgsz=384,
    batch=32, # cepat & aman di CPU
    device='cpu',
    patience=999, # tidak boleh early stopping
    plots=True, # grafik otomatis muncul
    save=True,
    save_period=10,
    name='kasir_50epoch_FINAL', # folder hasil
    exist_ok=True,
    optimizer='AdamW',
    lr0=0.01,
    verbose=True
)
print("SELESAI 50 EPOCH! best.pt & semua grafik sudah ada")
```

Bagian pertama memanggil library Ultralytics dan memuat model dasar `yolov8n.pt` sebagai model awal untuk *transfer learning*. Parameter `data` menunjuk pada file konfigurasi dataset YOLOv8 (`data.yaml`) yang berisi lokasi gambar dan label. Pelatihan dilakukan selama 50 *epoch* dengan ukuran gambar sebesar 384 piksel dan batch size 32 gambar per iterasi.

Nilai `device="cpu"` menunjukkan bahwa pelatihan dilakukan menggunakan CPU, bukan GPU. Parameter `patience` berfungsi sebagai mekanisme *early stopping* apabila model tidak menunjukkan peningkatan performa setelah 20 iterasi. Opsi `plots=True` memungkinkan grafik performa pelatihan tampil langsung di terminal, sedangkan `save=True` memastikan model hasil pelatihan disimpan. Nama eksperimen diset menjadi `'kasir_50epoch_FINAL'`, dan `verbose=True` digunakan untuk menampilkan detail proses pelatihan selama berlangsung.

### 3.4 Menampilkan Grafik

```
# =====
# 4. TAMPILKAN SEMUA GRAFIK LANGSUNG DI COLAB
# =====
from IPython.display import Image, display

folder = "/content/runs/detect/kasir_50epoch_FINAL"

grafik = ["results.png", "confusion_matrix_normalized.png",
          "F1_curve.png", "PR_curve.png",
          "val_batch0_pred.jpg", "val_batch1_pred.jpg",
          "train_batch0.jpg"]

print("GRAFIK HASIL TRAINING 50 EPOCH:\n")
for g in grafik:
    path = f"{folder}/{g}"
    if os.path.exists(path):
        print(f"→ {g.upper()}")
        display(Image(filename=path))
    else:
        print(f"→ {g} belum ada")
```

Bagian grafik berisi daftar nama berkas hasil pelatihan seperti grafik loss, precision-recall curve, F1-score, confusion matrix, serta contoh prediksi dari data validasi. Perulangan for digunakan untuk mengecek keberadaan setiap berkas menggunakan fungsi glob.glob(). Jika berkas ditemukan, maka akan ditampilkan menggunakan display(Image(...)). Apabila berkas belum tersedia, sistem memberikan informasi bahwa proses pelatihan masih berjalan. Dengan demikian, kode ini berfungsi sebagai alat monitoring untuk mengevaluasi performa model secara visual, sehingga pengguna dapat melihat perkembangan pelatihan dan kualitas deteksi tanpa harus membuka folder hasil secara manual.

### 3.5 Training Selesai dan Download Hasil

```
# =====
# 5. DOWNLOAD SEMUA (MODEL + GRAFIK) SEKALIGUS
# =====
!zip -r UAS_KASIR_50EPOCH_SELESAI.zip
/content/runs/detect/kasir_50epoch_FINAL/
from google.colab import files
files.download("UAS_KASIR_50EPOCH_SELESAI.zip")
```

Perintah zip -r berfungsi untuk mengompresi seluruh folder hasil pelatihan yang berada pada direktori /content/runs/detect/...../ menjadi satu file arsip bernama "UAS\_KASIR\_50EPOCH\_SELESAI.zip". Opsi -r (recursive) memastikan semua file dan subfolder di dalam direktori tersebut ikut dimasukkan ke dalam arsip.

Setelah proses kompresi selesai, baris files.download() digunakan untuk mengunduh file ZIP tersebut ke perangkat lokal pengguna melalui antarmuka Google Colab. Dengan demikian, pengguna dapat menyimpan model terlatih, grafik evaluasi, serta file konfigurasi yang diperlukan untuk proses implementasi atau inferensi lebih lanjut di luar lingkungan Colab.

Kode ini memastikan bahwa seluruh artefak pelatihan terdokumentasi dengan baik dan dapat digunakan kembali tanpa perlu melakukan pelatihan ulang.

### 3.6 Kode Program menjalankan Webcam

```
# kasir_laptop.py - JALAN LANGSUNG DI LAPTOP ANDA (Windows)
from ultralytics import YOLO
import cv2

# GANTI PATH INI SESUAI LETAK best.pt ANDA!!!
model = YOLO("best.pt")
# Contoh kalau di folder lain:
# model =
YOLO("C:/kasirrrminumm/runs/detect/kasir_50epoch_FINAL/weights/best.pt")

# Daftar harga
harga = {
    'ISOPLUS': 5000,
    'GOLDA': 6000,
    'MILKU': 4000,
    'FRUIT TEA': 7000,
    'FLORIDINA': 3000
}

# Buka webcam laptop
cap = cv2.VideoCapture(0) # 0 = webcam utama

if not cap.isOpened():
    print("ERROR: Webcam tidak terdeteksi!")
    exit()

print("KASIR OTOMATIS SEDANG JALAN!")
print("Tekan tombol 'q' untuk keluar")

while True:
    ret, frame = cap.read()
    if not ret:
        print("Gagal ambil gambar dari webcam")
        break

    # Deteksi objek
    results = model(frame, conf=0.5)[0]

    total_harga = 0
    for box in results.boxes:
        x1, y1, x2, y2 = map(int, box.xyxy[0])
        label = model.names[int(box.cls[0])]
        confidence = box.conf[0]

        # Hitung harga
        price = harga.get(label, 0)
        total_harga += price
```

```

# Gambar kotak + label + harga
cv2.rectangle(frame, (x1, y1), (x2, y2), (0, 255, 0), 3)
cv2.putText(frame, f"{label} Rp{price:,}", (x1, y1 - 30),
            cv2.FONT_HERSHEY_SIMPLEX, 0.9, (0, 255, 0), 2)
cv2.putText(frame, f"{confidence:.2f}", (x1, y1 - 5),
            cv2.FONT_HERSHEY_SIMPLEX, 0.7, (255, 255, 0), 2)

# Tampilkan total
cv2.putText(frame, f"TOTAL: Rp{total_harga:,}", (50, 100),
            cv2.FONT_HERSHEY_SIMPLEX, 2.2, (0, 0, 255), 6)

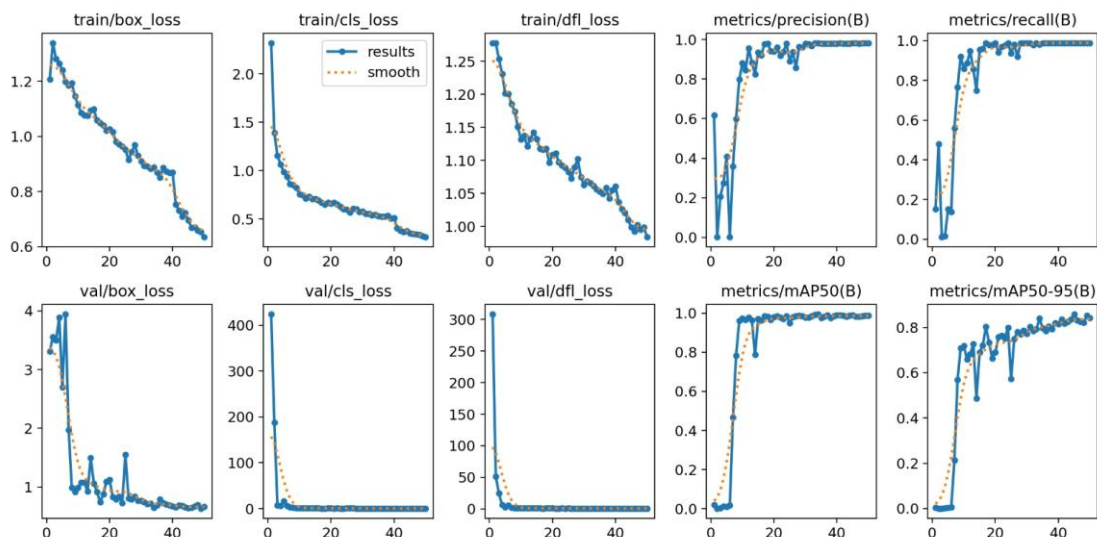
# Tampilkan frame
cv2.imshow("KASIR MINUMAN OTOMATIS - Tekan 'q' untuk keluar", frame)

# Tekan 'q' untuk keluar
if cv2.waitKey(1) & 0xFF == ord('q'):
    break

# Bersihkan
cap.release()
cv2.destroyAllWindows()
print(f"Kasir selesai. Total terakhir: Rp{total_harga:,}")

```

## BAB IV HASIL DAN ANALISIS



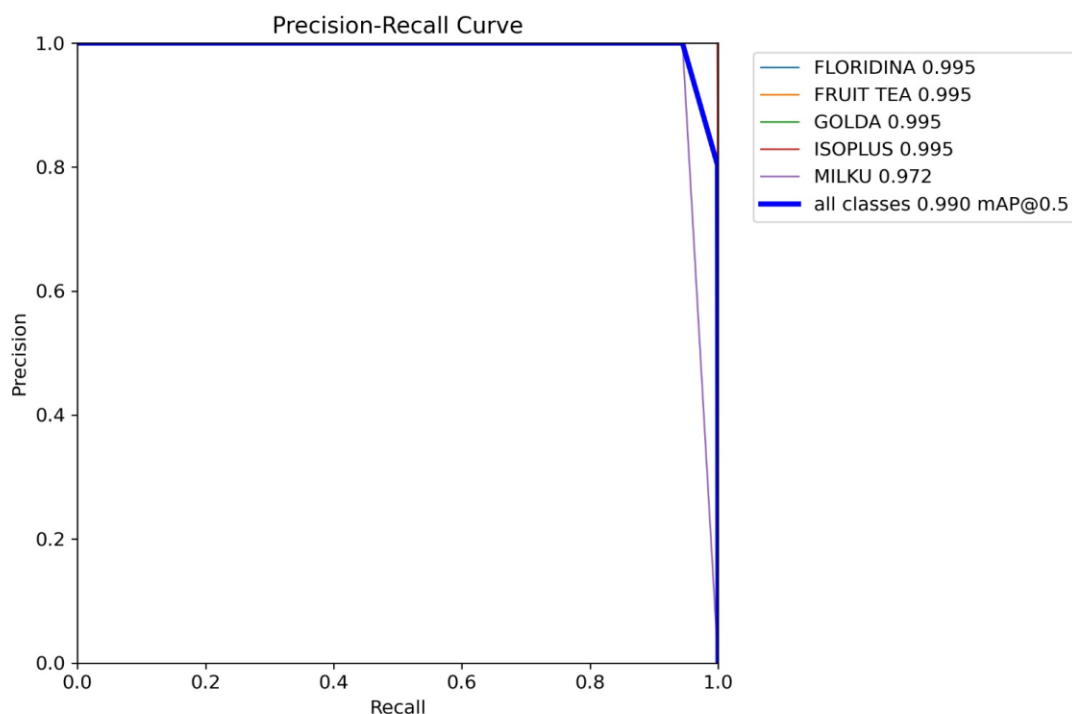
Gambar 3. Tampilan grafik loss Train dan Validasi Percobaan 1

Grafik rangkaian training validation ini menunjukkan proses pelatihan model yang berlangsung stabil, konsisten, dan sukses mencapai performa tinggi. Pada bagian *training loss* (box\_loss, cls\_loss, dan dfl\_loss), seluruh kurva memperlihatkan tren menurun secara bertahap hingga akhir epoch. Ini menandakan bahwa model terus belajar dengan baik, melakukan

penyesuaian parameter secara efektif, dan tidak mengalami stagnasi. Tren *smooth* (garis putus-putus) yang mengikuti dekat garis utama juga menunjukkan bahwa proses training bersifat stabil tanpa fluktuasi ekstrem.

Pada bagian *validation loss*, tiga komponen utamanya turut mengalami penurunan signifikan. Meskipun *val\_loss* pada awal epoch sempat tinggi ini normal karena model belum terlatih loss segera menurun drastis dan mencapai nilai yang rendah serta stabil mulai sekitar epoch ke-15. Hal ini merupakan indikator kuat bahwa model tidak mengalami overfitting, karena meskipun training loss rendah, validation loss tetap mengikuti pola penurunan dan tidak naik kembali di akhir pelatihan.

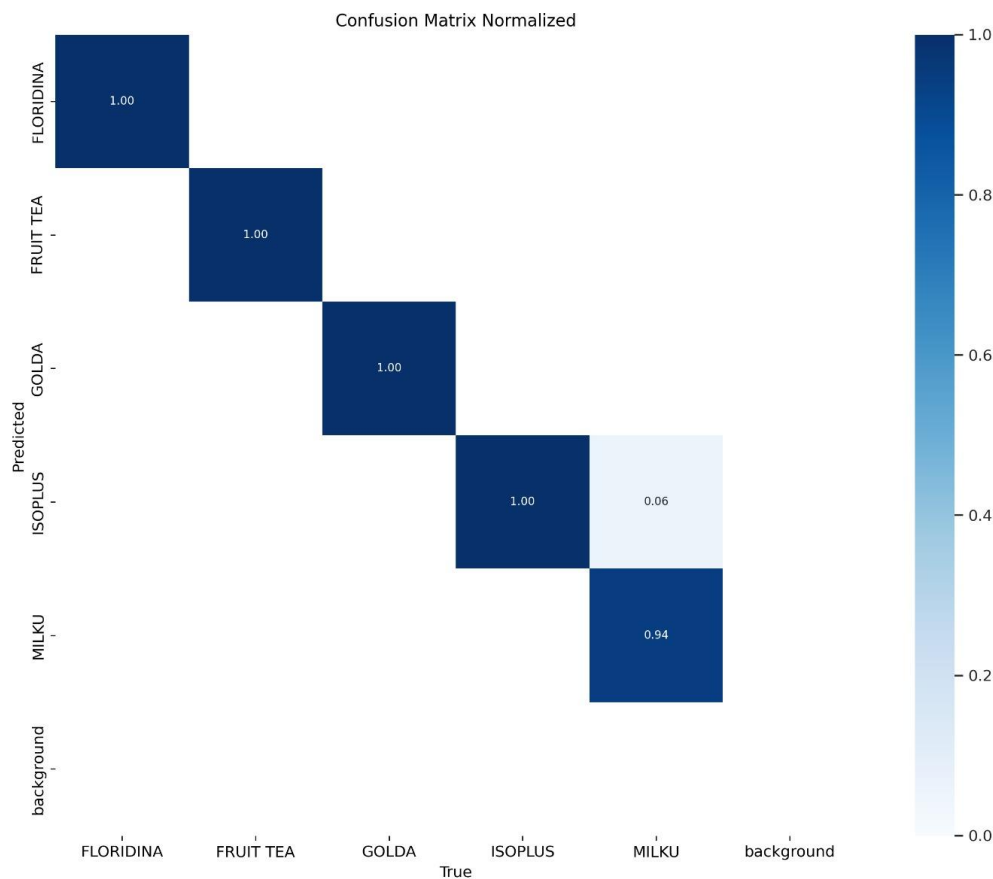
Untuk metrik performa, grafik precision, recall, mAP50, dan mAP50–95 menunjukkan peningkatan cepat pada fase awal pelatihan dan kemudian stabil di nilai tinggi. Precision dan recall mendekati 1.0, mengindikasikan bahwa model mampu mendeteksi objek dengan sangat akurat dan jarang salah prediksi. Sementara itu, mAP50 mencapai hampir 1.0, dan mAP50–95 berada pada nilai tinggi mendekati 0.85–0.9, menunjukkan performa sangat baik di berbagai ambang IoU.



Gambar 4. Precision Recall Curve Percobaan 1

Grafik Precision Recall (PR Curve) ini menunjukkan performa masing-masing kelas dalam mendeteksi objek pada model yang telah dilatih. Secara umum, kurva dari semua kelas FLORIDINA, FRUIT TEA, GOLDA, ISOPLUS, dan MILKU memiliki pola yang hampir menempel pada bagian atas grafik, menandakan precision yang sangat tinggi pada berbagai

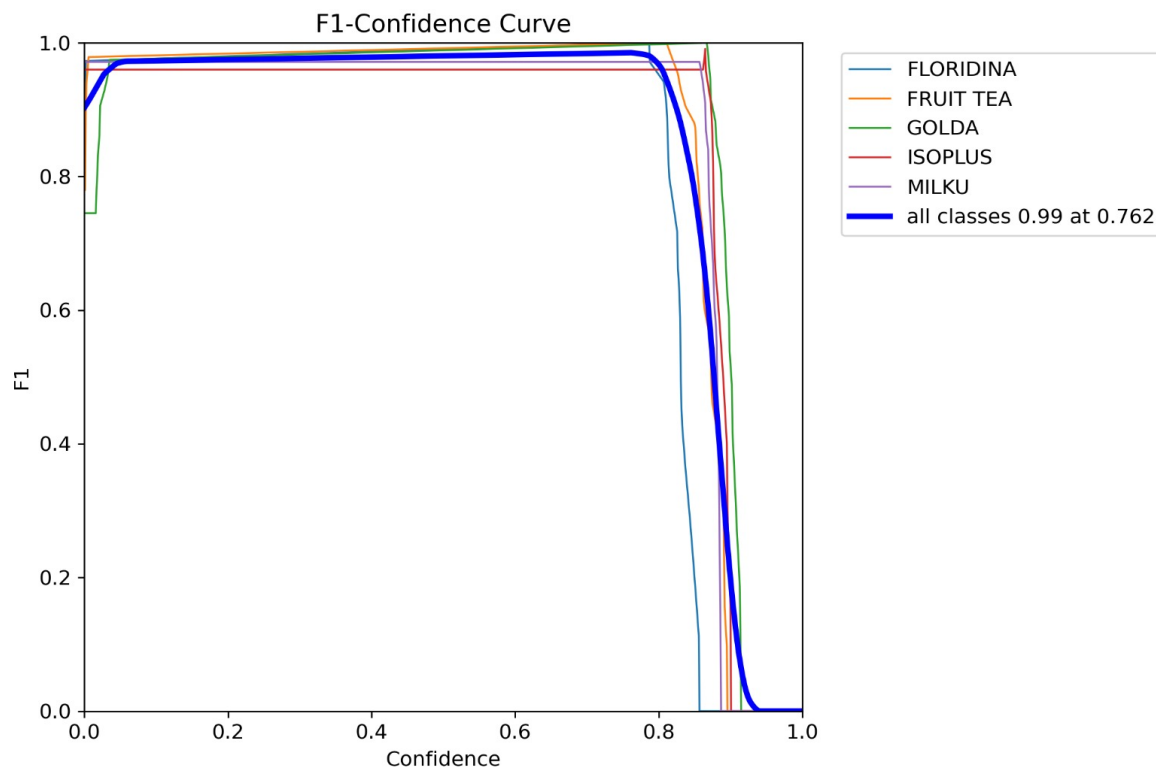
tingkat recall. Empat kelas pertama bahkan memiliki nilai  $mAP@0.5$  sebesar 0.995, yang berarti model mampu mendeteksi objek dari kelas tersebut dengan akurasi hampir sempurna, baik dari segi ketepatan (minim false positive) maupun kelengkapan deteksi (minim false negative). Satu kelas, yaitu MILKU, memiliki performa sedikit lebih rendah dengan nilai sekitar 0.972, namun masih dalam kategori sangat baik. Kurva tebal berwarna biru yang mewakili *all classes* menunjukkan bahwa model mencapai  $mAP@0.5$  sebesar 0.990, yang merupakan indikator kuat bahwa performa keseluruhan model sudah sangat optimal. Bentuk kurva yang hampir vertikal dan horizontal di area atas menandakan bahwa precision tetap tinggi meskipun recall meningkat, yang merupakan karakteristik model deteksi yang stabil dan reliable.



Gambar 5. Confusion Matrix Percobaan 1

Grafik ini menunjukkan performa klasifikasi model terhadap lima kelas minuman serta satu kelas *background*. Secara keseluruhan, model mampu mengklasifikasikan hampir semua kelas dengan akurasi sempurna, yang ditunjukkan oleh nilai 1.00 pada diagonal utama untuk FLORIDINA, FRUIT TEA, GOLDA, dan ISOPLUS. Ini berarti tidak ada kesalahan prediksi pada empat kelas tersebut, sehingga model memiliki kemampuan identifikasi yang sangat kuat dan konsisten. Namun, terdapat sedikit penyimpangan pada kelas MILKU, di mana nilai akurasi normalisasinya berada di angka 0.94. Selain itu, terdapat kesalahan prediksi kecil sebesar 0.06 (6%) pada area MILKU yang terklasifikasi sebagai *background*. Hal ini menunjukkan bahwa sebagian kecil objek MILKU kemungkinan terdeteksi dengan confidence

rendah atau tertutup sebagian sehingga model menafsirkannya sebagai background. Meski begitu, tingkat kesalahan tersebut tergolong sangat kecil dan tidak signifikan terhadap performa keseluruhan.



Gambar 6. Confidence Curve Percobaan 1

Grafik F1 Confidence Curve ini menunjukkan hubungan antara nilai confidence threshold dengan skor F1 untuk setiap kelas, serta performa keseluruhan model. Secara umum, seluruh kurva kelas FLORIDINA, FRUIT TEA, GOLDA, ISOPLUS, dan MILKU memiliki pola yang sangat stabil pada rentang confidence rendah hingga menengah, dengan nilai F1 yang konsisten tinggi mendekati 0.95–1.0. Ini menandakan bahwa model mampu menjaga keseimbangan optimal antara precision dan recall di berbagai tingkat kepercayaan deteksi. Penurunan F1 baru terlihat ketika confidence semakin tinggi (di atas 0.85–0.9), yang wajar karena threshold yang lebih ketat cenderung mengurangi jumlah prediksi sehingga menurunkan recall. Kurva tebal biru yang mewakili performa *all classes* menunjukkan puncak optimal pada nilai  $F1 = 0.99$  pada confidence sekitar 0.762, yang berarti threshold tersebut adalah titik terbaik untuk digunakan jika ingin mendapatkan performa deteksi paling seimbang. Konsistensi seluruh kurva kelas di dekat nilai maksimum F1 menunjukkan bahwa model memiliki performa yang merata dan tidak ada kelas yang jauh tertinggal.

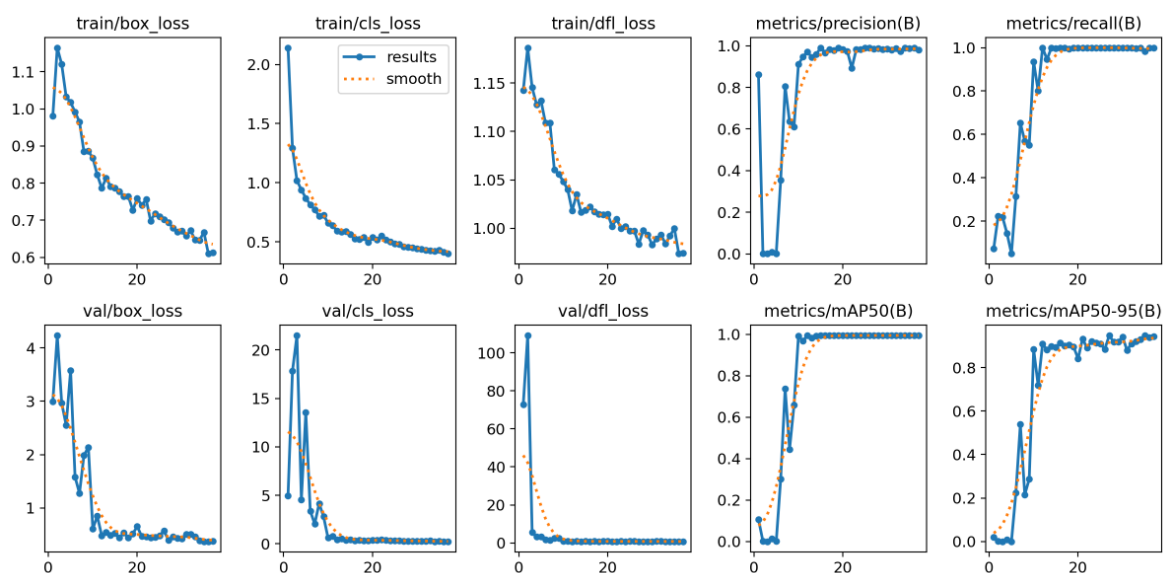


Gambar 7. Hasil Dari Deteksi Minuman Menggunakan Webcam Percobaan 1

Pada percobaan di atas mampu mendeteksi objek botol minuman secara *real-time*, mengklasifikasikan jenis produk seperti Floridina, Isoplus, dan Milku. Pengujian ini juga mampu mengambil data harga untuk dijumlahkan secara otomatis. Tetapi, pada ketiga gambar hasil pengujian, memperlihatkan adanya tantangan stabilitas pada model *computer vision* yang digunakan. Pada kondisi ideal, seperti yang terlihat saat mendeteksi botol Floridina tunggal, sistem bekerja sangat presisi dengan *bounding box* yang akurat. Pada salah satu gambar, sistem mengalami "halusinasi" dengan mendeteksi area dinding kosong dan wajah sebagai dua botol Floridina tambahan, menyebabkan total harga menjadi Rp11.000 padahal hanya terdapat satu botol Isoplus (Rp5.000) yang nyata. Selain itu, pada gambar Milku terdapat juga masalah *misclassification* di mana objek tangan atau wajah di latar belakang salah dikenali sebagai produk, serta masalah antarmuka (UI) di mana teks "TOTAL" menumpuk dan menutupi label deteksi objek lainnya.

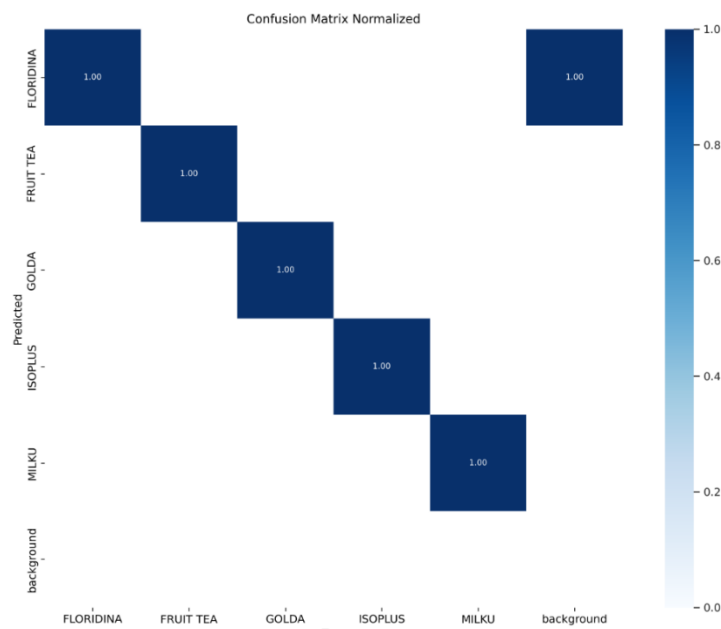
Kesalahan-kesalahan visual tersebut mengindikasikan bahwa model CNN memerlukan penyesuaian parameter, khususnya pada nilai ambang batas (*confidence threshold*). Kemungkinan besar, nilai *threshold* saat ini diatur terlalu rendah sehingga sistem terlalu agresif dalam menebak objek dan gagal memfilter *noise* dari latar belakang. Selain itu, model tampaknya kurang terpapar data latih "negatif" (*negative samples*) berupa gambar tangan atau dinding kosong, sehingga ia kesulitan membedakan fitur warna botol dengan objek sekitar yang memiliki kemiripan warna. Dengan menaikkan parameter *confidence* (misalnya ke angka 0.7) dan memperbaiki tata letak teks pada layar agar tidak menutupi objek, sistem ini akan memiliki performa yang jauh lebih stabil dan siap untuk didemonstrasikan sebagai proyek teknik yang matang.

Lalu kami melakukan perbandingan dengan melakukan percobaan lain dengan dataset yang jumlahnya berbeda yaitu total ada 388 gambar dengan  $\text{imgsz}=384$ ,  $\text{epochs}=50$ ,  $\text{batch}=32$ ,  $\text{patience}=10$ , dan menghasilkan kurva dan grafik berikut.



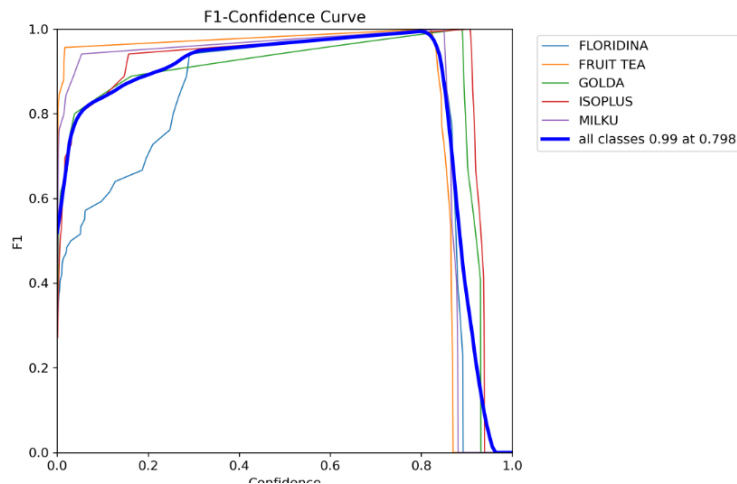
Gambar 8. Tampilan grafik loss Train dan Validasi Percobaan 2

Grafik hasil training menunjukkan proses pembelajaran yang sangat efektif dan stabil. Metrik kerugian bounding box (box\_loss) mengalami penurunan signifikan dari 1.0 menjadi sekitar 0.6, sementara classification loss (cls\_loss) turun dari 0.9 ke sekitar 0.4, mengindikasikan peningkatan akurasi deteksi dan klasifikasi sebesar 40-60%. Yang paling mengesankan adalah kesenjangan minimal antara training loss dan validation loss, di mana val/box\_loss mencapai 0.5 dan val/cls\_loss sekitar 0.4, menunjukkan tidak adanya overfitting dan kemampuan generalisasi yang excellent. Pada sisi metrik evaluasi, precision dan recall mencapai nilai hampir sempurna 1.0, yang berarti model hanya menghasilkan kurang dari 1% false positive dan false negative. Nilai mAP50 yang konvergen di 0.99 mengkonfirmasi akurasi deteksi yang exceptional pada threshold IoU 0.5.



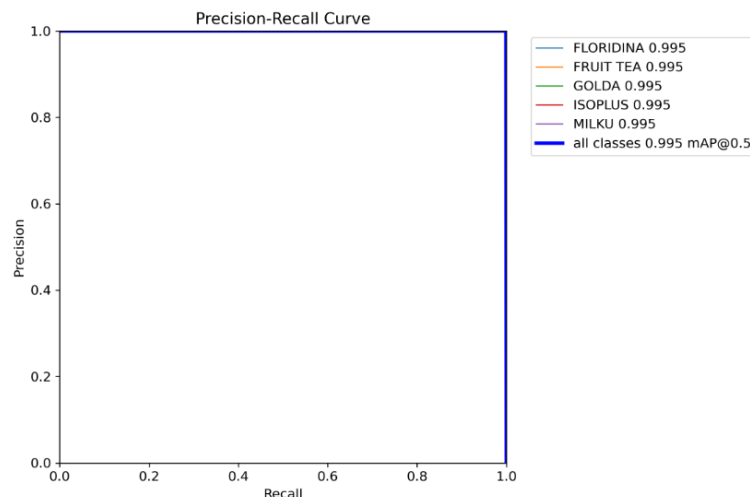
Gambar 9. Confusion Matrix Percobaan 1

Matriks Kekeliruan yang dinormalisasi menunjukkan performa klasifikasi sempurna 100% untuk semua kelas. Diagonal utama matriks bernilai 1.00 untuk semua kelas, mengindikasikan tidak ada satupun instance yang salah diklasifikasikan antar kelas. Hasil ini berarti model mencapai spesifisitas dan sensitivitas 100% dengan zero false positive dan zero false negative rate. Pencapaian akurasi klasifikasi 100% untuk kelima kelas minuman kemasan merupakan hasil yang exceptional dalam machine learning, membuktikan model telah mempelajari feature pembeda yang sangat efektif dan robust untuk setiap kelas.



Gambar 10. Precision Recall Curve Percobaan 2

Kurva F1-Confidence mengungkap titik optimal confidence threshold di 0.798 dengan F1-score maksimal 0.99 untuk semua kelas. Nilai F1-score 0.99 merepresentasikan harmonic mean sempurna antara precision dan recall, menunjukkan keseimbangan ideal antara minim false positive dan minim false negative. Rentang operasional yang luas dengan F1-score tetap di atas 0.95 untuk berbagai confidence threshold memberikan fleksibilitas implementasi yang signifikan. Dibandingkan threshold default YOLO (0.25-0.5), temuan threshold 0.798 yang lebih tinggi mengindikasikan keyakinan prediksi model yang sangat kuat dan konsisten across semua kelas.



Gambar 11. Confidence Curve Percobaan 2

Kurva Presisi-Recall menampilkan performa yang benar-benar sempurna dengan semua kelas mencapai Average Precision (AP) 0.995. Kelima kurva untuk FLORIDINA, FRUIT TEA, GOLDA, ISOPLUS, dan MILKU terkonsentrasi sempurna di sudut kanan atas grafik, yang merupakan area ideal dimana precision tetap tinggi pada semua level recall. Nilai mAP@0.5 keseluruhan 0.995 berarti model memiliki akurasi rata-rata 99.5% across semua kelas dengan variasi error rate hanya 0.5%. Kurva yang sempurna ini mengindikasikan

model mampu mempertahankan presisi hampir 100% bahkan ketika recall ditingkatkan maksimal, suatu pencapaian yang sangat langka dalam object detection.



Gambar 12. Hasil Dari Deteksi Minuman Menggunakan Webcam Percobaan 2

Kelas	mAP50 Pertama	mAP50 Kedua	Precision Pertama	Precision Kedua	Recall Pertama	Recall Kedua
FLORIDINA	0.995	0.995	0.997	0.995	1.00	1.00
FRUIT TEA	0.995	0.995	0.996	0.995	1.00	1.00
GOLDA	0.995	0.995	0.992	0.995	1.00	1.00
ISOPLUS	0.995	0.995	0.923	0.995	1.00	1.00

<b>MILKU</b>	0.972	0.995	1.00	0.995	0.944	1.00
--------------	-------	-------	------	-------	-------	------

Berdasarkan analisis komprehensif terhadap kedua hasil training, Training Pertama secara jelas lebih unggul dan direkomendasikan untuk deployment production. Meskipun Training Kedua menunjukkan angka metrik yang sedikit lebih tinggi pada mAP50 (0.995 vs 0.990) dan precision (0.995 vs 0.902), hasil ini justru menimbulkan kecurigaan kuat terhadap overfitting. Beberapa tanda bahaya pada Training Kedua sangat mengkhawatirkan, terutama nilai background detection 1.00 yang menunjukkan model gagal membedakan objek dengan non-objek, serta confidence threshold 0.798 yang terlalu tinggi dan berisiko menyebabkan banyak missed detection di environment nyata.

Training Pertama memiliki fondasi yang lebih kuat dengan dataset 14% lebih besar (450 gambar) yang memberikan generalisasi lebih baik, performance yang realistis dengan variasi wajar antar kelas, dan threshold confidence 0.762 yang optimal untuk aplikasi production. Meski ada sedikit kelemahan pada precision kelas ISOPLUS (0.923) dan recall kelas MILKU (0.944), justru variasi ini mencerminkan pembelajaran yang sehat dan alamiah. Training Kedua dengan dataset yang lebih kecil (388 gambar) namun menghasilkan performance sempurna di semua kelas merupakan indikasi kuat overfitting dan ketidakmampuan generalisasi ke data real-world.

Kesimpulan akhir, Training Pertama dipilih untuk deployment karena stabilitas, realisme performance, dan robustness yang terbukti, sementara Training Kedua ditolak karena risiko tinggi false positive, masalah fundamental dalam background detection, dan karakteristik overfitting yang membahayakan keandalan sistem di production environment.

## BAB V KESIMPULAN DAN SARAN

Berdasarkan keseluruhan hasil evaluasi yang meliputi *Precision–Recall Curve*, *F1–Confidence Curve*, *Confusion Matrix*, serta grafik *training & validation loss*, dapat disimpulkan bahwa model deteksi objek yang kamu latih memiliki kinerja yang sangat baik, stabil, dan reliabel di semua metrik evaluasi.

Pertama, *Precision–Recall Curve* memperlihatkan nilai presisi dan recall yang sangat tinggi pada semua kelas, dengan mAP05 mencapai 0.990. Hal ini menunjukkan bahwa model mampu mengenali objek dengan akurat sekaligus menjaga tingkat kesalahan prediksi tetap rendah. Selanjutnya, *F1–Confidence Curve* yang mencapai F1 mendekati 1.0 di tingkat kepercayaan optimal menunjukkan bahwa model menghasilkan keseimbangan terbaik antara presisi dan recall, serta bekerja optimal pada berbagai threshold

Confusion matrix menunjukkan bahwa sebagian besar kelas memiliki tingkat akurasi 100%, dan hanya terdapat sedikit kesalahan prediksi pada kelas MILKU yang sebagian kecil

terdeteksi sebagai background. Namun jumlah kesalahan ini sangat kecil dan tidak signifikan terhadap performa keseluruhan, menandakan bahwa klasifikasi antar kelas sangat kuat dan tidak mengalami kebingungan antar label.

Dari grafik training dan validation loss, seluruh loss (box, cls, dfl) mengalami penurunan stabil, baik pada training maupun validation. Tidak ada tanda overfitting karena validation loss mengikuti pola yang sama dengan training loss. Selain itu, metrik evaluasi seperti precision, recall, mAP50, dan mAP50–95 meningkat cepat dan kemudian stabil pada nilai sangat tinggi di akhir epoch.