Peningkatan Kecepatan Inferensi *Mask R-CNN* Menggunakan *MobileNetV3 Small* pada Sistem Deteksi Kardus

1st Vikha Tri Vicika
Universitas Buana Perjuangan Karawang
Karawang, Indonesia
if21.vikhavicika@mhs.ubpkarawang.ac.id

2nd Jamaludin Indra
Universitas Buana Perjuangan Karawang
Karawang, Indonesia
jamaludin.Indra@ubpkarawang.ac.id

3rd Sutan Faisal
Universitas Buana Perjuangan Karawang
Karawang, Indonesia
sutan.faisal@ubpkarawang.ac.id

4th Hanny Hikmayanti
Universitas Buana Perjuangan Karawang
Karawang, Indonesia
hanny.hikmayanti@ubpkarawang.ac.id

Abstract— Penelitian ini bertujuan untuk meningkatkan kecepatan inferensi pada sistem deteksi objek dengan mengoptimalkan arsitektur Mask R-CNN melalui penggantian backbone menjadi MobileNetV3 Small. Studi difokuskan pada deteksi kardus dalam lingkungan gudang menggunakan dataset yang dikumpulkan dari berbagai kondisi pencahayaan dan sudut pengambilan gambar. Proses pelatihan dan validasi dilakukan terhadap data yang telah dianotasi secara manual, dengan evaluasi menggunakan metrik presisi, recall, dan kecepatan pemrosesan dalam satuan frame per second (FPS). Hasil pengujian menunjukkan bahwa model dengan backbone MobileNetV3 Small menghasilkan FPS rata-rata sebesar 8,66, lebih tinggi dibandingkan model default yang hanya mencapai 6,94. Dari segi akurasi, model kustom menunjukkan presisi sebesar 85 persen, sedangkan model default mencapai recall sebesar 85,7 persen. Meskipun segmentasi model default lebih menyeluruh, model kustom lebih ringan dan efisien, sehingga lebih cocok untuk aplikasi waktu nyata pada perangkat dengan sumber daya terbatas. Temuan ini menunjukkan bahwa MobileNetV3 Small dapat meningkatkan efisiensi inferensi tanpa menurunkan akurasi secara signifikan.

Kata kunci — Mask R-CNN, MobileNetV3 Small, Deteksi Kardus, FPS

I. PENDAHULUAN

Manajemen persediaan memiliki peran krusial dalam mendukung efisiensi operasional dan pengembangan sistem logistik di lingkungan pergudangan. Salah satu fungsinya adalah memastikan keakuratan proses verifikasi stok serta pencatatan arus barang, baik saat masuk maupun keluar dari gudang [1]. Dalam praktiknya, kegiatan seperti pemindahan kardus dari kendaraan distribusi ke area penyimpanan menjadi tahap penting yang menentukan keandalan sistem pencatatan. Namun, sering kali ditemukan ketidaksesuaian antara jumlah kardus yang tercatat dan kondisi aktual di lapangan. Ketidaksesuaian ini tidak jarang menimbulkan kerugian finansial dan berdampak pada menurunnya kepercayaan pelanggan. Salah satu penyebab umum dari masalah ini adalah pencatatan manual yang rawan kesalahan, terutama ketika dilakukan oleh tenaga kerja yang mengalami kelelahan [2].

Untuk mengatasi tantangan tersebut, pemanfaatan teknologi modern menjadi solusi yang semakin relevan. Salah satu pendekatan yang menjanjikan adalah penggunaan teknologi deteksi objek berbasis computer vision. Teknologi ini mampu meningkatkan akurasi serta efisiensi proses pencatatan inventaris dengan mengenali objek secara otomatis melalui pengolahan citra digital [3][4]. Dengan demikian, sistem dapat melakukan penghitungan objek seperti kardus secara real-time dan sekaligus mengurangi risiko kesalahan yang biasa terjadi dalam pencatatan manual [5].

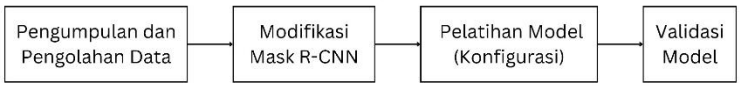
Meskipun demikian, penerapan computer vision dalam konteks industri tidak lepas dari hambatan teknis. Salah satu tantangan yang sering muncul adalah tingginya tingkat kemiripan visual antar objek dalam hal ini kardus yang memiliki ukuran, warna, dan desain kemasan yang hampir seragam [6]. Selain itu, pola tekstur kardus yang berulang juga dapat membingungkan model dalam membedakan antara satu objek dengan objek lain, terutama ketika kardus disusun bertumpuk, sehingga memperbesar kemungkinan kesalahan deteksi [7].

Salah satu solusi untuk mengatasi kemiripan visual antar objek adalah dengan menggunakan instance segmentation, yaitu metode yang menggabungkan segmentasi piksel dan deteksi objek secara simultan untuk membedakan setiap objek dalam citra, termasuk yang berasal dari kelas yang sama, di mana Mask R-CNN merupakan model pertama yang dirancang khusus untuk menangani permasalahan ini [8]. Model ini merupakan pengembangan dari Faster R-CNN yang tidak hanya mampu mengklasifikasikan dan mendeteksi objek, tetapi juga menghasilkan segmentasi yang presisi pada masing-masing objek secara instans [9]. Beberapa penelitian terdahulu menunjukkan bahwa Mask R-CNN dengan backbone ResNet-50-FPN mampu memberikan hasil deteksi yang baik, seperti ditunjukkan oleh performa deteksi keypoint dengan nilai AP sebesar 75.76 pada data nyata [10]. Namun, model dengan backbone besar seperti ResNeXt meskipun menghasilkan akurasi deteksi dan segmentasi yang tinggi (mAP 62.62% dan 57.58%), membutuhkan daya komputasi besar yang tidak ideal untuk perangkat terbatas [11]. Kemudian Penelitian oleh [12] membandingkan Mask R-CNN dengan backbone ResNet-50, ResNet-101, dan MobileNet-V1. Hasilnya,

ResNet-101 mencapai mAP tertinggi sebesar 98.3%. Selanjutnya Penelitian oleh [13] menunjukkan bahwa mengganti backbone ResNet101 pada Mask R-CNN dengan MobileNetV2 berhasil mengurangi jumlah parameter sebesar 92,4% (dari 44,5 juta menjadi 3,4 juta), meningkatkan kecepatan pemrosesan lebih dari 50%, dan mempertahankan akurasi deteksi yang tinggi, menjadikannya pilihan ideal untuk aplikasi real-time pada perangkat dengan sumber daya terbatas. Lebih lanjut, pengembangan terbaru pada model deteksi real-time seperti Yolov5 menunjukkan bahwa penggunaan MobileNetV3 sebagai backbone dapat meningkatkan efisiensi deteksi pada objek kecil, sekaligus memangkas parameter hingga 87,4% dibandingkan backbone konvensional [14]. Hal ini menunjukkan potensi besar dalam mengintegrasikan arsitektur ringan ke dalam sistem deteksi berbasis instance segmentation.

Berangkat dari kondisi tersebut, penelitian ini bertujuan untuk mengoptimalkan Mask R-CNN dengan mengganti backbone standar menjadi MobileNetV3 Small, dengan fokus utama pada peningkatan efisiensi kecepatan pemrosesan (frame per second / FPS). Dengan pengurangan jumlah parameter dan peningkatan FPS, model yang diusulkan diharapkan mampu beroperasi secara real-time tanpa mengorbankan akurasi secara signifikan, khususnya saat diterapkan pada sistem deteksi kardus dalam lingkungan dengan keterbatasan sumber daya komputasi.

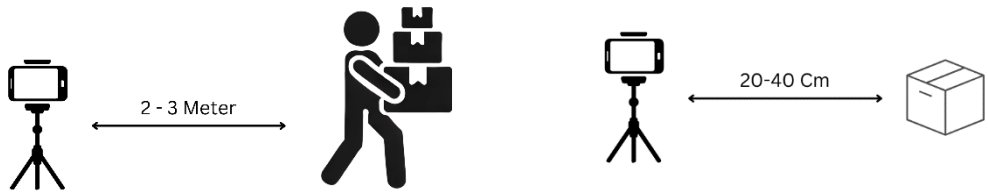
II. METODE PENELITIAN



Gambar 1. Metode Penelitian

A. Pengumpulan Data dan Pengelolaan Data

Data diperoleh melalui proses dokumentasi visual di Gudang Distribusi Minuman Air Toko Anugerah yang berlokasi di Kampung Kedung Lotong, Desa Bantarjaya, Kabupaten Bekasi. Pengambilan gambar dilakukan dari berbagai sisi—depan, samping, dan atas—guna memperoleh representasi menyeluruh terhadap objek kardus. Untuk mendekati kondisi nyata, pencahayaan diatur secara bervariasi. Kamera smartphone dengan resolusi 48 megapiksel digunakan dalam proses ini. Jarak pemotretan disesuaikan: 20–40 cm untuk kardus tunggal (Gambar 2A), dan 2–3 meter untuk kardus yang sedang dibawa oleh pekerja (Gambar 2B). Pencahayaan dalam ruangan menggunakan lampu 15 watt, sementara pengambilan gambar di luar ruangan memanfaatkan cahaya alami sekitar pukul 08.00 pagi. Secara keseluruhan, terkumpul 639 foto dari empat kategori produk yaitu Sky, Fatqua, Levios, dan Elvios.



Gambar 2. Pengambilan Gambar Kardus

Setelah proses pengambilan gambar selesai dilakukan, tahap selanjutnya adalah melakukan anotasi *dataset* untuk memberikan penandaan terhadap objek-objek yang akan dikenali oleh model. Seluruh gambar diunggah ke platform Roboflow [15], dan anotasi dilakukan secara manual menggunakan fitur polygon tool yang tersedia pada situs tersebut. Proses ini menghasilkan segmentasi objek yang sesuai dengan bentuk aktual kardus di setiap gambar. Label yang digunakan dikelompokkan berdasarkan kategori produk, yaitu "Sky", "Fatqua", "Levios", dan "Elvios", guna memastikan bahwa setiap objek teridentifikasi secara tepat sesuai dengan klasifikasi produknya.

Kemudian seluruh gambar dianotasi, Roboflow secara otomatis membagi dataset ke dalam dua subset: 88% digunakan sebagai data pelatihan dan 12% sebagai data validasi. Untuk meningkatkan keberagaman data dan memperkuat kemampuan generalisasi model, diterapkan teknik augmentasi sebanyak tiga kali lipat terhadap data latih. Teknik augmentasi yang digunakan mencakup rotasi sudut, penyesuaian tingkat kecerahan, serta efek blur. Distribusi jumlah instance pada masing-masing kategori, baik untuk pelatihan maupun validasi, dapat dilihat secara rinci pada Tabel 1.

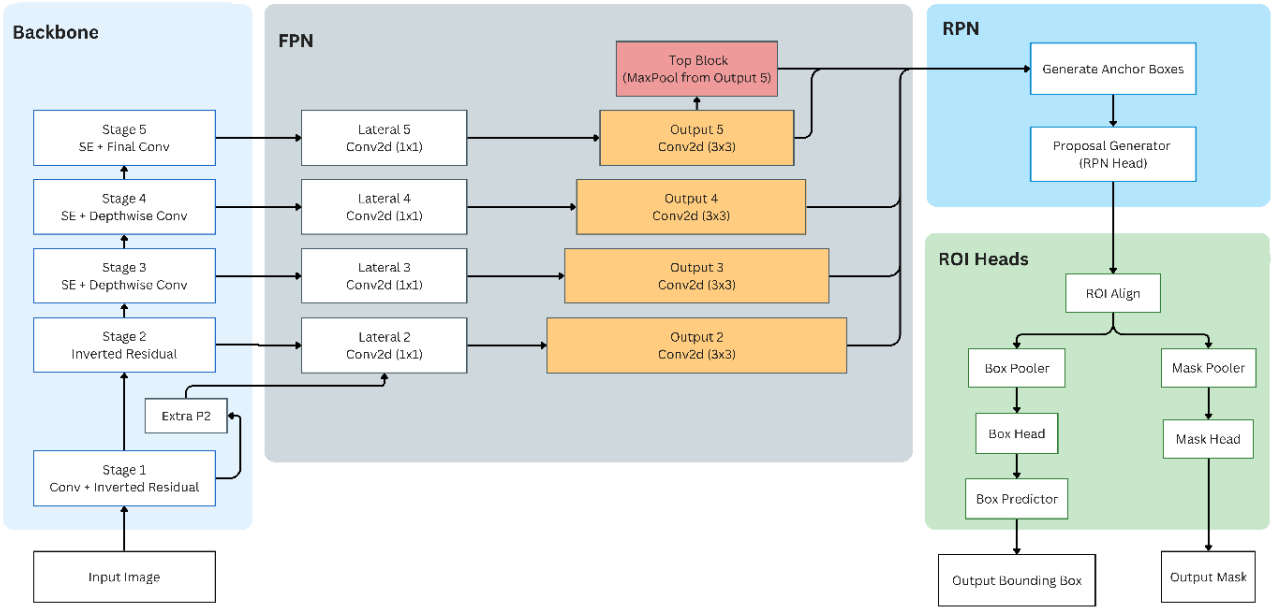
Tabel 1. Jumlah dataset dan pembagian instance

No	Kategori	Jumlah Pelatihan	Jumlah Validasi
1	Elvios	672	59
2	Sky	717	58
3	Fatqua	735	64
4	Levios	668	55
Total		2792	236

B. Modifikasi Mask R-CNN

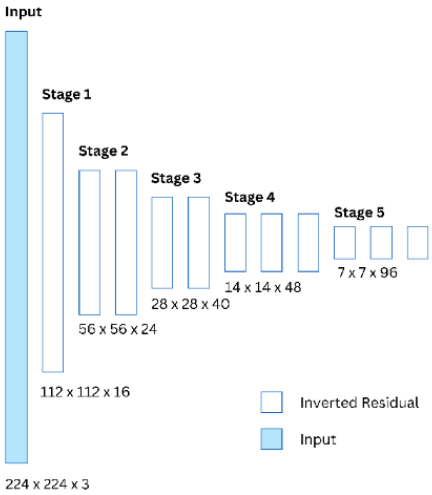
Gambar 3 menggambarkan alur kerja deteksi dan segmentasi objek menggunakan *Mask R-CNN* dengan *backbone MobileNetV3 Small* yang diintegrasikan dengan *Feature Pyramid Network (FPN)*. Proses dimulai dengan ekstraksi fitur dari gambar input melalui *backbone* yang terdiri dari beberapa tahap konvolusi dan *blok residual*, termasuk tambahan *Extra* P2 yang berfungsi untuk meningkatkan representasi fitur pada skala yang lebih kecil, sehingga deteksi objek kecil menjadi lebih akurat. Fitur yang dihasilkan kemudian diteruskan ke *FPN*, yang mengolahnya dalam berbagai skala dengan kombinasi konvolusi 1×1 dan 3×3 untuk meningkatkan representasi spasial. Selanjutnya, fitur dari *FPN* dikirim ke *Region Proposal Network (RPN*), yang menghasilkan

kandidat wilayah objek melalui *anchor boxes*. Kandidat wilayah ini kemudian diproses oleh *ROI Heads*, di mana dilakukan ekstraksi lebih lanjut menggunakan *ROI Align*. Setelah itu, fitur diproses oleh *Box Head* untuk klasifikasi objek dan *regresi bounding box*, serta oleh *Mask Head* untuk menghasilkan segmentasi objek dalam bentuk *mask*. Dengan adanya alur ini, serta tambahan *Extra* P2 untuk menangkap detail lebih baik pada objek kecil, sistem mampu melakukan deteksi dan segmentasi objek dengan akurasi tinggi serta efisiensi komputasi yang optimal.



Gambar 3. Arsitetur *Mask R-CNN + MobileNetV3 Small*

Pada gambar 4 menunjukkan proses ekstraksi fitur dalam *backbone MobileNetV3 Small*, yang digunakan untuk mendeteksi dan mengenali objek dengan efisiensi tinggi.



Gambar 4. *Backbone MobileNetV3 Small*

Proses dimulai dari gambar *input* berukuran 224 × 224 × 3, yang kemudian diproses melalui serangkaian tahap konvolusi dan *blok Inverted Residual* untuk menghasilkan representasi fitur yang lebih kompresibel dan informatif. *Stage* 1 mengurangi dimensi menjadi 112 × 112 × 16, diikuti oleh *Stage* 2 yang menghasilkan 56 × 56 × 24 menggunakan *Inverted Residual Block* untuk meningkatkan efisiensi pemrosesan. Selanjutnya, *Stage* 3 memperkecil ukuran menjadi 28 × 28 × 40, diikuti oleh *Stage* 4 dengan ukuran 14 × 14 × 48, yang memperkaya informasi fitur sambil mempertahankan detail objek. Akhirnya, *Stage* 5 menghasilkan fitur berukuran 7 × 7 × 96, yang sangat terkompresi tetapi berisi informasi tingkat tinggi yang akan digunakan dalam *FPN* untuk meningkatkan deteksi objek dan segmentasi mask.

C. Pelatihan Model

Pada penelitian ini, proses pelatihan dilakukan dengan menggunakan *optimizer Stochastic Gradient Descent (SGD)* yang memiliki momentum sebesar 0.9 dan *weight decay* sebesar 1e-4 untuk menghindari *overfitting*. Nilai *learning rate* awal yang digunakan adalah 0.002, dengan skema penurunan bertahap (*step decay*) pada iterasi ke 12.000 dan 16.000 dengan faktor penurunan (γ) sebesar 0.1. Untuk meningkatkan stabilitas awal pelatihan, diterapkan skema warm-up selama 1.000 iterasi dengan faktor 1.0/1000. Jumlah gambar yang digunakan per iterasi (batch size) adalah 2, sedangkan jumlah sampel per gambar dalam Region of Interest (ROI) Heads ditetapkan sebanyak 128. Proses pelatihan dilakukan hingga mencapai 20.000 iterasi dengan konfigurasi optimizer yang memastikan konvergensi model secara optimal.

Dalam arsitektur yang digunakan, sistem deteksi objek memanfaatkan FPN untuk menghasilkan fitur dari berbagai skala. Oleh karena itu, sistem anchor generation dikonfigurasi untuk menangani berbagai ukuran objek dengan menerapkan skala berbeda pada setiap level fitur. Pada level p2, *anchor* memiliki ukuran 16 piksel, level p3 sebesar 32 piksel, level p4 sebesar 64 piksel, level p5 sebesar 128 piksel, dan level p6 sebesar 256 piksel.

D. Validasi Model

Validasi dilakukan dengan menggunakan subset dari dataset kardus dan *COCO* untuk mengukur performa model. Metrik evaluasi adalah alat atau metode yang digunakan untuk menilai kinerja suatu model dalam menyelesaikan tugas tertentu, Menurut [16] beberapa validasi yang digunakan untuk menilai kinerja model meliputi:

1. Presisi pada rumus (1) mengacu pada tingkat akurasi hasil yang diperoleh suatu model [17].

$$presisi = \frac{true\ positive}{true\ positive + false\ positive} \quad (1)$$

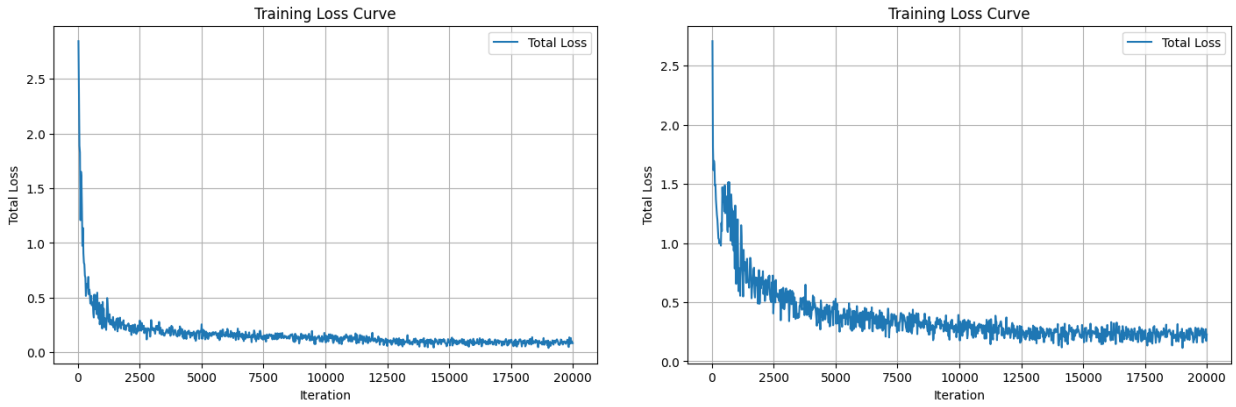
2. *Recall* pada rumus (2) menunjukan seberapa lengkap data relevan yang berhasil diindetifikasi oleh sebuah model [17].

$$recall = \frac{true\ positive}{true\ positive + false\ negative} \quad (2)$$

Selain kedua metrik tersebut, dalam penelitian ini juga ditambahkan pengukuran *frame per second (FPS)* sebagai indikator kecepatan inferensi model. *FPS* menunjukkan jumlah *frame* atau citra yang dapat diproses model dalam satu detik proses inferensi. Penelitian oleh [18] menjelaskan bahwa frame rate merupakan ukuran jumlah bingkai atau citra yang diputar setiap detik dalam suatu karya berbasis waktu, yang relevan pula dalam konteks sistem deteksi waktu nyata. Nilai *FPS* yang tinggi menunjukkan model memiliki kemampuan respon yang cepat dan efisien dalam mengolah visual secara berkelanjutan di lingkungan nyata.

III. HASIL DAN PEMBAHASAN

A. Grafik Training Loss

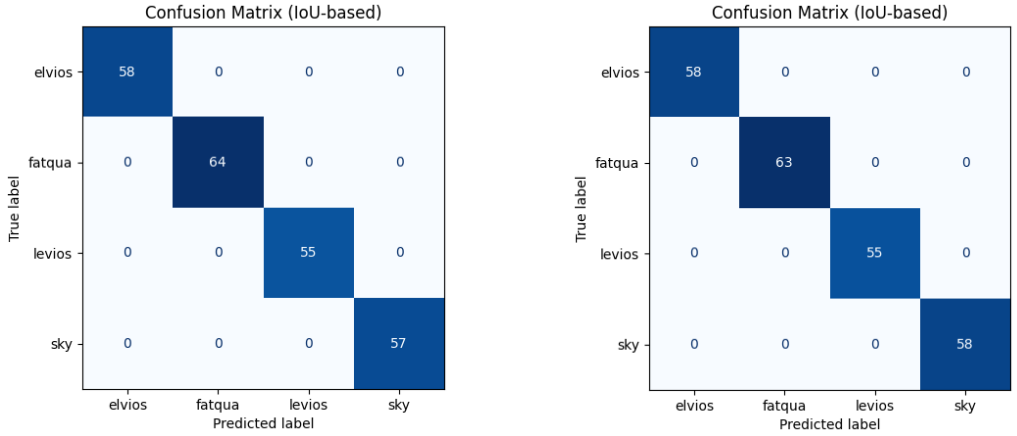


A B

Gambar 5. Hasil Training Model

Gambar 5 menunjukkan grafik training loss dari kedua model, Gambar A adalah grafik model *Mask R-CNN* dan B adalah grafik model *Mask R-CNN + MobileNetV3 Small*. Kedua Grafik menggambarkan perubahan total *loss* selama 20.000 iterasi. Total *loss* awalnya bernilai tinggi sekitar 2.5 dan mengalami penurunan drastis dalam beberapa ribu iterasi pertama. Namun, perbedaan utama terlihat pada pola fluktuasi *loss*. Kurva A menunjukkan penurunan yang lebih stabil dengan sedikit fluktuasi, sementara kurva B memiliki lebih banyak variasi, terutama pada tahap awal pelatihan, yang dapat mengindikasikan bahwa model pada B mengalami lebih banyak kesulitan dalam menemukan jalur konvergensi optimal. Meskipun demikian, keduanya mencapai titik *loss* yang relatif stabil setelah sekitar 10.000 iterasi, menunjukkan bahwa model berhasil belajar dan tidak mengalami *overfitting*.

B. Confusion Matrix



A B

Gambar 6. Hasil Confusion Matrix

Gambar 6 menunjukkan *confusion matrix* berbasis *IoU* untuk dua model yang berbeda, A adalah model *Mask R-CNN* dan B adalah model *Mask R-CNN + MobileNetV3* Small. Kedua model memiliki hasil yang hampir sama. Gambar A dan B sama-sama mengklasifikasikan elvios (58), levios (55) dengan akurasi penuh, sementara hanya ada sedikit perbedaan di kelas fatqua dan sky, di mana model A mencatat 64 prediksi benar untuk fatqua dan 57 untuk sky, sedangkan model B sedikit lebih rendah dengan 63 untuk fatqua dan lebih tinggi dengan 58 untuk sky.

C. Pengujian Data Langsung

Pengujian langsung dilakukan terhadap kedua model menggunakan dataset gambar yang sama. Seluruh proses pengujian dilaksanakan menggunakan platform Google Colab dengan dukungan GPU NVIDIA Tesla T4, guna memastikan efisiensi dan konsistensi dalam inferensi. Pada pengujian ini, digunakan nilai ambang batas (*threshold*) deteksi sebesar 0,8; artinya, objek dengan skor deteksi di bawah nilai tersebut dianggap tidak terdeteksi.

Tabel 2. Tabel hasil pengujian langsung

No	Hasil Pengujian Langsung	Custom	Default
1		TP: 1 FP: 0 FN: 0 S : Tidak Menyeluruh *FPS*: 7,01	TP: 1 FP: 0 FN: 0 S : Menyeluruh *FPS*: 6,94
2		TP: 2 FP: 1 FN: 0 S : Menyeluruh *FPS*: 8,25	TP: 2 FP: 1 FN: 0 S : Menyeluruh *FPS*: 6,24
3		TP: 3 FP: 0 FN: 0 S : Menyeluruh *FPS*: 7,38	TP: 3 FP: 0 FN: 0 S : Menyeluruh *FPS*: 6,22
4		TP: 3 FP: 0 FN: 1 S : Tidak Menyeluruh *FPS*: 6,66	TP: 4 FP: 1 FN: 0 S : Menyeluruh *FPS*: 6,22

5			TP: 3 FP: 1 FN: 1 S : Menyeluruh *FPS*: 7,23	TP: 4 FP: 2 FN: 0 S : Menyeluruh *FPS*: 6,31
6			TP: 0 FP: 0 FN: 4 S : *FPS*: 8,66	TP: 0 FP: 1 FN: 3 S : Menyeluruh *FPS*: 6,12
7			TP: 0 FP: 1 FN: 0 S : Menyeluruh *FPS*: 7,15	TP: 0 FP: 1 FN: 0 S : Tidak Menyeluruh *FPS*: 5,69
8			TP: 0 FP: 0 FN: 0 S : *FPS*: 8,41	TP: 0 FP: 1 FN: 0 S : Menyeluruh *FPS*: 6,78
9			TP: 1 FP: 0 FN: 0 S : Tidak Menyeluruh *FPS*: 8,57	TP: 1 FP: 0 FN: 0 S : Menyeluruh *FPS*: 6,37
10			TP: 1 FP: 0 FN: 0 S : Tidak Menyeluruh *FPS*: 6,92	TP: 1 FP: 0 FN: 0 S : Menyeluruh *FPS*: 6,30

11			TP: 1 FP: 0 FN: 0 S : Tidak Menyeluruh *FPS*: 8,19	TP: 1 FP: 1 FN: 0 S : Menyeluruh *FPS*: 6,78
12			TP: 1 FP: 0 FN: 0 S : Tidak Menyeluruh *FPS*: 7,66	TP: 1 FP: 0 FN: 0 S : Menyeluruh *FPS*: 6,31

D. Evaluasi Presisi dan *Recall*

Nilai presisi dan *recall* dihitung berdasarkan rumus (1) dan (2) pada tahap validasi model. Perhitungan ini didasarkan pada hasil pengujian langsung terhadap 12 gambar, sebagaimana ditampilkan pada Tabel 2.

Presisi untuk model custom dihitung berdasarkan persamaan (1), yaitu:

$$presisi = \frac{17}{17+3} = 85\ \%$$

Recall untuk model custom dihitung berdasarkan persamaan (2), yaitu:

$$recall = \frac{17}{17+6} = 73{,}9\ \%$$

Presisi untuk model default dihitung berdasarkan persamaan (1), yaitu:

$$presisi = \frac{18}{18+8} = 69{,}2\ \%$$

Recall untuk model default dihitung berdasarkan persamaan (2), yaitu:

$$recall = \frac{18}{18+3} = 85{,}7\ \%$$

Berdasarkan hasil tersebut, dapat disimpulkan bahwa model custom memiliki presisi yang lebih tinggi dibandingkan model default, sedangkan model default menunjukkan *recall* yang lebih tinggi dibandingkan model custom. Selain itu, segmentasi yang dihasilkan oleh model default umumnya hampir menyeluruh membentuk area kotak sesuai objek aslinya, sedangkan segmentasi pada model custom cenderung kurang sempurna, dengan cakupan area segmentasi yang hanya sekitar setengah dari objek yang seharusnya terdeteksi.

E. Evaluasi *FPS*

Dalam penelitian ini, kecepatan *inferensi* model diukur menggunakan satuan *frame per second (FPS).* Meskipun istilah *FPS* umumnya digunakan untuk mengukur jumlah frame dalam video, pada konteks ini *FPS* digunakan untuk mengukur berapa banyak gambar (*frame*) yang dapat diproses oleh model dalam satu detik. Setiap gambar diuji satu per satu, dan dihitung berdasarkan kecepatan model dalam memproses sejumlah iterasi prediksi pada gambar tersebut. Semakin tinggi nilai *FPS*, semakin cepat model dalam melakukan prediksi terhadap gambar.

Pengukuran *FPS* dilakukan untuk setiap gambar uji yang telah digunakan pada tahap validasi model. Meskipun pengujian berbasis gambar statis, perhitungan *FPS* tetap relevan untuk mengevaluasi kecepatan inferensi model secara individu, dengan asumsi bahwa setiap gambar diproses seperti sebuah frame dalam aliran data berkelanjutan. Hal ini penting, terutama untuk penerapan model pada sistem real-time, seperti monitoring gudang atau aplikasi berbasis kamera.

Proses pengukuran *FPS* dilakukan menggunakan fungsi measure_*FPS*, yang diawali dengan tahap untuk menstabilkan eksekusi GPU sebelum pengambilan waktu dilakukan. Fungsi ini diubah ke bentuk pseudocode berikut:

```
def measure_FPS(predictor, image, num_iter=50, warmup=10):
    for _ in range(warmup):
        predictor(image)
        if CUDA tersedia:
            sinkronisasi CUDA
    mulai waktu
    for _ in range(num_iter):
```

```
    predictor(image)
    if CUDA tersedia:
        sinkronisasi CUDA
akhir waktu
FPS = num_iter dibagi selisih waktu akhir dan waktu mulai
return FPS
```

Nilai *FPS* yang diperoleh dari pengujian model custom dan default untuk 12 gambar uji dapat dilihat pada Tabel 3 berikut.

Tabel 3. Perbandingan *FPS*

Model	**Gambar**											
	1	2	3	4	5	6	7	8	9	10	11	12
Custom	7,01	8,25	7,38	6,66	7,23	8,66	7,15	8,41	8,57	6,92	8,19	7,66
Default	6,94	6,22	6,22	6,22	6,31	6,12	5,69	6,78	6,37	6,3	6,78	6,31

Berdasarkan hasil pengujian pada Tabel 2, model custom konsisten menunjukkan nilai *FPS* yang lebih tinggi dibandingkan model default. Model custom mencapai *FPS* tertinggi sebesar 8,66 dan *FPS* terendah sebesar 6,66. Sementara itu, model default memiliki *FPS* tertinggi sebesar 6,94 dan *FPS* terendah sebesar 5,69. Hal ini menunjukkan bahwa model custom lebih efisien dalam proses inferensi dan lebih cocok digunakan untuk aplikasi *real-time* yang membutuhkan respons cepat.

IV. KESIMPULAN DAN SARAN

Penggantian *backbone Mask R-CNN* dari *ResNet-50* ke *MobileNetV3 Small* terbukti meningkatkan kecepatan *inferensi* secara signifikan dengan rata-rata *FPS* yang lebih tinggi pada model kustom, yaitu mencapai 8,66 dibandingkan 6,94 pada model default, menjadikannya lebih efisien untuk implementasi *real-time*, meskipun akurasi deteksi dan segmentasi pada model default secara umum masih unggul dalam beberapa konfigurasi, model kustom menunjukkan presisi lebih tinggi pada pengujian langsung (85% dibandingkan 69,2%), sehingga tetap kompetitif terutama dalam konteks sistem yang membutuhkan kecepatan dan efisiensi komputasi.

Penelitian selanjutnya disarankan untuk menggunakan data dengan kondisi lebih kompleks seperti objek kecil dan bertumpuk, serta melakukan anotasi yang lebih akurat. Selain itu, disarankan mengeksplorasi backbone ringan lain seperti *EfficientNet* dan menguji model pada perangkat *edge* seperti Jetson Nano atau Raspberry Pi untuk menilai performa nyata dan tantangan implementasi.

1. PENGAKUAN

Makalah ini merupakan bagian dari penelitian Tugas Akhir penulis yang disusun sebagai kontribusi akademik dalam bidang visi komputer, dengan judul "Peningkatan Kecepatan *Inferensi Mask R-CNN* Menggunakan *MobileNetV3 Small* pada Sistem Deteksi Kardus". Penelitian ini dilakukan secara mandiri tanpa dukungan sponsor dari pihak mana pun, dan disusun sebagai bagian dari pemenuhan syarat akademik di Universitas Buana Perjuangan Karawang.

2. DAFTAR PUSTAKA

[1] Yohanes Mbiri, Kristina Sara, and Anastasia Mude, "Rancang Bangun Sistem Informasi Adiministrasi Kependudukan Berbasis Website Menggunakan Metode Agile Di Desa Nginamanu Barat Kecamatan Wolomeze Kabupaten Ngada," *Simtek J. Sist. Inf. dan Tek. Komput.*, vol. 8, no. 1, pp. 148–153, Apr. 2023, doi: 10.51876/simtek.v8i1.236.

[2] S. Teja, M. I. Jalil, S. Nurakmalia, F. A. Rizaldi, and A. Saifudin, "Analisis dan Perancangan Sistem Informasi Inventory pada PT Insan Data Permata," *JURIHUM J. Inov. dan Hum.*, vol. 1, pp. 231–239, Apr. 2023, doi: 10.30998/jrami.v1i02.231.

[3] E. Fontana, W. Zarotti, and D. Lodi Rizzini, "A Comparative Assessment of Parcel Box Detection Algorithms for Industrial Applications," in *2021 European Conference on Mobile Robots (ECMR)*, IEEE, Aug. 2021, pp. 1–6. doi: 10.1109/ECMR50962.2021.9568825.

[4] M. A. Masril and D. P. Caniago, "Optimasi Teknologi Computer Vision pada Robot Industri Sebagai Pemindah Objek Berdasarkan Warna," *ELKOMIKA J. Tek. Energi Elektr. Tek. Telekomun. Tek. Elektron.*, vol. 11, no. 1, p. 46, Jan. 2023, doi: 10.26760/elkomika.v11i1.46.

[5] T. Anjali Dompeipen, S. R. U. . Sompie, and M. E. . Najoan, "Computer Vision Implementation for Detection and Counting the Number of Humans," *J. Tek. Inform. vol. 16 no. 1*, vol. 16, no. 1, pp. 65–76, 2021, doi: 10.35793.

[6] A. Y. Firmandicky and Y. A. Susetyo, "Klasifikasi Kardus Barang di PT XYZ Menggunakan Convolutional Neural Network dengan Pendekatan Fine Grained Image Classification," *J. JTIK (Jurnal Teknol. Inf. dan Komunikasi)*, vol. 8, no. 4, pp. 954–964, Oct. 2024, doi: 10.35870/jtik.v8i4.2337.

[7] J. Yang *et al.*, "SCD: A Stacked Carton Dataset for Detection and Segmentation," *Sensors*, vol. 22, no. 10, p. 3617, May 2022, doi: 10.3390/s22103617.

[8] S. Goel and D. Koundal, "A MaskFormer EfficientNet instance segmentation approach for crowd counting," *Sci. Rep.*, vol. 15, no. 1, pp. 1–13, 2025, doi: 10.1038/s41598-025-95174-9.

[9] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 386–397, 2020, doi: 10.1109/TPAMI.2018.2844175.

[10] A. Naumann, F. Hertlein, L. Dörr, and K. Furmans, "TAMPAR: Visual Tampering Detection for Parcel Logistics in Postal Supply Chains," in *2024 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, IEEE, Jan. 2024, pp. 8061–8071. doi: 10.1109/WACV57701.2024.00789.

[11] S. Fang, B. Zhang, and J. Hu, "Improved Mask R-CNN Multi-Target Detection and Segmentation for Autonomous Driving in Complex Scenes," *Sensors*, vol. 23, no. 8, 2023, doi: 10.3390/s23083853.

[12] R. Rubin, C. Jacob, S. M. Anzar, and A. Panthakkan, "Mask R-CNN with Multi-Backbones - A Comparative Analysis," *2022 5th Int. Conf. Signal Process. Inf. Secur. ICSPIS 2022*, no. December, pp. 55–60, 2022, doi: 10.1109/ICSPIS57063.2022.10002546.

[13] C. Huang, Y. Zhou, and X. Xie, "Intelligent Diagnosis of Concrete Defects Based on Improved Mask R-CNN," *Appl. Sci.*, vol. 14, no. 10, 2024, doi: 10.3390/app14104148.

[14] L. Cao, P. Song, Y. Wang, Y. Yang, and B. Peng, "An Improved Lightweight Real-Time Detection Algorithm Based on the Edge Computing Platform for UAV Images," *Electron.*, vol. 12, no. 10, 2023, doi: 10.3390/electronics12102274.

[15] D. D. Karyanto, D. D. Karyanto, J. Indra, A. R. Pratama, and T. Rohana, "Detection of the Size of Plastic Mineral Water Bottle Waste Using the Yolov5 Method," *JIKO (Jurnal Inform. dan Komputer)*, vol. 7, no. 2, pp. 123–130, 2024, doi: 10.33387/jiko.v7i2.8535.

[16] A. Ardiansyah, J. Triloka, and Indera, "Evaluasi Kinerja Model YOLOv8 dalam Deteksi Kesegaran Buah," *JUPITER*, vol. 16, no. 2, pp. 357–368, 2024, doi: 10.5281/zenodo.11296226.

[17] A. P. Nardilasari, A. L. Hananto, S. S. Hilabi, T. Tukino, and B. Priyatna, "Analisis Sentimen Calon Presiden 2024 Menggunakan Algoritma SVM Pada Media Sosial Twitter," *JOINTECS (Journal Inf. Technol. Comput. Sci.*, vol. 8, no. 1, p. 11, 2023, doi: 10.31328/jointecs.v8i1.4265.

[18] S. Teh, V. Perumal, and H. Abdul Hamid, "Investigating How Frame Rates in Different Styles of Animation Affect the Psychology of the Audience," *Int. J. Creat. Multimed.*, vol. 4, no. 2, pp. 10–31, 2023, doi: 10.33093/ijcm.2023.4.2.2.