p-ISSN : 2406-7733
e-ISSN : 2597-9922

# ANALISIS PERBANDINGAN KINERJA SISTEM INVENTORI DENGAN ALGORITMA PENDETEKSI BARANG BERBASIS *YOLOV11* DAN *YOLOV8*

**Muhammad Luqman[1], Kasyfillah Ardaffa Fahrezi[2], Aji Setiawan[3]**
Program Studi Teknologi Informasi, Fakultas Teknik, Universitas Darma Persada
Jl. Taman Malaka Selatan No.8, RT.8/RW.6, Pondok Kelapa, Kec. Duren Sawit, Kota Jakarta Timur, Daerah Khusus Ibukota Jakarta 13450
E-mail: luqmanm296@gmail.com[1], kasyfillahardaffa2004@gmail.com[2], aji_setiawan@ft.unsada.ac.id[3]

**Abstrak** - Peningkatan efisiensi dan akurasi dalam sistem inventori merupakan tantangan utama dalam manajemen logistik modern. Penelitian ini bertujuan untuk menganalisis dan membandingkan kinerja dua algoritma deteksi objek berbasis *deep learning*, yaitu *YOLOv8* dan *YOLOv11*, dalam mendeteksi dan mengidentifikasi barang secara otomatis. Metode penelitian yang digunakan adalah eksperimen dengan melatih kedua model menggunakan *dataset* yang sama, kemudian mengevaluasi performanya berdasarkan metrik *precision*, *recall*, *F1-score*, *mean Average Precision (mAP)*, dan kecepatan inferensi (*FPS*). Hasil penelitian menunjukkan bahwa *YOLOv11* memiliki performa yang lebih baik dengan nilai *mAP* sebesar 0.83 dan *FPS* sebesar 52, dibandingkan *YOLOv8* yang memperoleh *mAP* sebesar 0.81 dan *FPS* sebesar 43. Hal ini menunjukkan bahwa *YOLOv11* mampu memberikan keseimbangan yang lebih baik antara akurasi dan kecepatan. Penelitian ini memberikan kontribusi dalam pemilihan model deteksi objek yang optimal untuk sistem inventori berbasis *computer vision*.

**Kata Kunci:** Analisis kinerja, deteksi objek, *inventory*, manajemen logistik, *YOLOv8*, *YOLOv11*.

## I. PENDAHULUAN

Manajemen inventori merupakan aspek penting dalam operasional logistik modern yang bertujuan untuk menjaga ketersediaan stok secara optimal. Namun, dalam praktiknya, sistem inventori masih menghadapi berbagai permasalahan, seperti kesalahan pencatatan, keterlambatan identifikasi barang, serta rendahnya efisiensi dalam pengolahan data (Villegas-Ch et al., 2024). Permasalahan tersebut dapat mengakibatkan ketidaksesuaian data stok, meningkatnya biaya operasional, dan menurunnya kualitas pelayanan. Oleh karena itu, diperlukan suatu sistem yang mampu meningkatkan akurasi dan efisiensi pengelolaan inventori secara otomatis (Villegas-Ch et al., 2024).

Perkembangan teknologi *computer vision* berbasis *deep learning* telah memberikan solusi terhadap permasalahan tersebut melalui penerapan deteksi objek otomatis. Teknologi ini mengalami perkembangan pesat dan telah bertransformasi dari metode tradisional menjadi pendekatan berbasis *neural network* dengan tingkat akurasi dan kecepatan yang tinggi (L. Liu et al., 2020).

Salah satu algoritma yang paling banyak digunakan dalam deteksi objek adalah YOLO (You Only Look Once), yang dikenal karena kemampuannya dalam melakukan deteksi secara *real-time* dengan performa yang optimal. Perkembangan YOLO hingga versi terbaru menunjukkan peningkatan signifikan dalam keseimbangan antara kecepatan dan akurasi (Plastiras et al., 2018).

Penelitian terbaru menunjukkan bahwa *YOLOv8* memiliki performa deteksi yang sangat baik bahkan pada *dataset* dengan jumlah terbatas serta mampu beradaptasi pada berbagai kondisi lingkungan (Jocher et al., 2023). Selain itu, pengembangan lanjutan dari model YOLO juga terus dilakukan untuk meningkatkan efisiensi dan akurasi dalam berbagai aplikasi industri (Ali & Zhang, 2024).

Meskipun demikian, sebagian besar penelitian masih berfokus pada satu model atau versi tertentu, sehingga belum banyak studi yang secara spesifik membandingkan performa model terbaru seperti *YOLOv8* dan *YOLOv11* dalam konteks sistem inventori.

Berdasarkan hal tersebut, penelitian ini bertujuan untuk menganalisis dan membandingkan kinerja algoritma *YOLOv8* dan *YOLOv11* dalam sistem inventori, dengan fokus pada aspek akurasi, kecepatan, dan efisiensi pemrosesan.

Penelitian ini tidak hanya membandingkan performa *YOLOv8* dan *YOLOv11*, tetapi juga menganalisis perbedaan performa kedua model berdasarkan kondisi *dataset*, seperti variasi pencahayaan dan kompleksitas latar belakang. Selain membandingkan *YOLOv8* dan *YOLOv11*, penelitian ini juga menggunakan *YOLOv5* sebagai model *baseline*. Pemilihan *YOLOv5* sebagai *baseline* didasarkan pada popularitas dan stabilitasnya dalam berbagai penelitian deteksi objek, serta performanya yang telah teruji dalam memberikan keseimbangan antara akurasi dan kecepatan. Dengan adanya *baseline* ini, perbandingan kinerja model menjadi lebih komprehensif dan memberikan gambaran peningkatan performa dari generasi sebelumnya ke model yang lebih baru.

p-ISSN : 2406-7733
e-ISSN : 2597-9922

## II. TINJAUAN PUSTAKA

### 2.1 Sistem Inventori

Sistem inventori merupakan sistem yang digunakan untuk mengelola data barang, termasuk pencatatan stok, pergerakan barang, serta pengendalian persediaan secara efisien. Sistem ini bertujuan untuk meminimalkan kesalahan dan meningkatkan efisiensi operasional (Tang, 2024).

### 2.2 Deteksi Objek Berbasis *Deep Learning*

Deteksi objek merupakan salah satu bidang dalam *computer vision* yang bertujuan untuk mengidentifikasi dan menentukan lokasi objek dalam suatu citra. Pendekatan berbasis *deep learning*, khususnya *Convolutional Neural Networks* (*CNN*), telah menunjukkan performa yang sangat baik dalam berbagai tugas deteksi objek (He et al., 2016).

### 2.3 Algoritma YOLO

YOLO merupakan algoritma deteksi objek yang menggunakan pendekatan *single-stage detection*, di mana proses klasifikasi dan lokalisasi dilakukan secara bersamaan dalam satu tahap (Plastiras et al., 2018). Versi sebelumnya seperti YOLOv4 telah menunjukkan peningkatan signifikan dalam hal akurasi dan efisiensi dibandingkan generasi awal YOLO (Bochkovskiy et al., 2020).

#### 2.3.1 *YOLOv8*

*YOLOv8* merupakan model deteksi objek terbaru yang dikembangkan untuk meningkatkan efisiensi dan akurasi. Model ini memiliki arsitektur yang lebih ringan dan fleksibel dibandingkan versi sebelumnya (Jocher et al., 2023).

#### 2.3.2 *YOLOv11*

*YOLOv11* merupakan pengembangan lanjutan yang menawarkan peningkatan dalam akurasi dan kemampuan generalisasi terhadap *dataset* kompleks (Khanam & Hussain, 2024).

### 2.4 Evaluasi Kinerja Model

Evaluasi model deteksi objek umumnya menggunakan metrik seperti *precision*, *recall*, dan *mean Average Precision (mAP)*. *Precision* digunakan untuk mengukur tingkat ketepatan model dalam mendeteksi objek, sedangkan *recall* digunakan untuk mengukur kemampuan model dalam menemukan seluruh objek pada *dataset*. Selain itu, *mean Average Precision (mAP)* digunakan untuk mengukur akurasi keseluruhan model berdasarkan hasil deteksi objek yang dilakukan (Tian et al., 2024).

### 2.5 Penelitian Terdahulu

Beberapa penelitian sebelumnya menunjukkan bahwa penggunaan algoritma YOLO dapat meningkatkan akurasi deteksi objek secara signifikan dibandingkan dengan metode konvensional (Ge et al., 2021). Hal ini dikarenakan YOLO mampu melakukan proses deteksi objek secara langsung dalam satu tahap (*single stage detection*), sehingga proses deteksi menjadi lebih cepat dan efisien. Selain itu, YOLO juga banyak digunakan pada sistem *real-time* karena memiliki kecepatan inferensi yang tinggi dengan tingkat akurasi yang baik.

Selain YOLO, terdapat metode lain seperti SSD (*Single Shot MultiBox Detector*) yang juga digunakan dalam deteksi objek. Metode SSD memiliki kemampuan akurasi yang cukup baik dalam mengenali objek dengan berbagai ukuran. Namun, dibandingkan dengan YOLO, SSD umumnya membutuhkan waktu pemrosesan yang lebih lama sehingga kurang optimal untuk implementasi deteksi objek secara *real-time* (W. Liu et al., 2016).

Penelitian lain juga menunjukkan bahwa YOLOv7 mampu mencapai performa yang lebih baik dibandingkan versi sebelumnya dengan peningkatan pada efisiensi pelatihan dan akurasi deteksi (Wang et al., 2023). YOLOv7 dikembangkan dengan arsitektur yang lebih optimal sehingga mampu meningkatkan nilai *precision*, *recall*, dan *mAP*. Selain itu, model ini juga memiliki kemampuan generalisasi yang lebih baik terhadap berbagai kondisi data dan objek.

Berdasarkan beberapa penelitian terdahulu tersebut, dapat disimpulkan bahwa perkembangan algoritma YOLO terus mengalami peningkatan dari segi akurasi, kecepatan, dan efisiensi komputasi. Oleh karena itu, penelitian ini menggunakan model YOLO sebagai metode utama untuk melakukan deteksi objek dan membandingkan performa beberapa versi YOLO guna memperoleh hasil yang paling optimal.

## III. METODE PENELITIAN

### 3.1 Metode Penelitian

Penelitian ini menggunakan metode eksperimen untuk membandingkan kinerja dua algoritma deteksi objek, yaitu *YOLOv8* dan *YOLOv11* (Plastiras et al., 2018)(Khanam & Hussain, 2024). Pendekatan eksperimen dilakukan dengan melatih kedua model menggunakan *dataset* yang sama, kemudian mengevaluasi performanya berdasarkan metrik tertentu seperti akurasi dan kecepatan.

Metode eksperimen dipilih karena memungkinkan peneliti untuk menguji dan membandingkan performa masing-masing model secara objektif dalam kondisi yang sama. Seluruh proses pelatihan dilakukan menggunakan *dataset*, parameter, dan lingkungan pengujian yang identik sehingga hasil yang diperoleh dapat

p-ISSN : 2406-7733
e-ISSN : 2597-9922

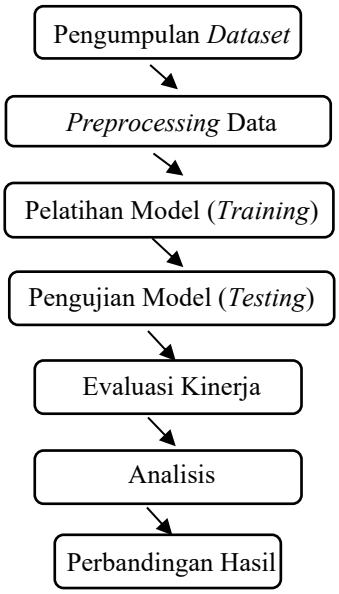
merepresentasikan kemampuan sebenarnya dari masing-masing algoritma.

Tahapan penelitian dimulai dari pengumpulan *dataset*, *preprocessing* data, pelatihan model, pengujian model, hingga evaluasi hasil. *Dataset* yang digunakan dibagi menjadi data pelatihan dan data pengujian untuk memastikan model dapat dievaluasi secara adil terhadap data yang belum pernah dilihat sebelumnya.

Hasil pengujian kemudian dianalisis menggunakan beberapa metrik evaluasi, yaitu *Precision*, *Recall*, *F1-Score*, *mean Average Precision (mAP)*, dan *Frames Per Second (FPS)*. Metrik tersebut digunakan untuk mengukur tingkat akurasi deteksi objek serta kecepatan pemrosesan model. Berdasarkan hasil evaluasi tersebut, dilakukan analisis perbandingan untuk menentukan model yang memiliki performa terbaik pada *dataset Product Detection*.

### 3.2 Alur Penelitian

Alur penelitian yang dilakukan terdiri dari beberapa tahapan, yaitu:




### 3.3 *Dataset*

*Dataset* yang digunakan dalam penelitian ini diperoleh dari platform Roboflow Universe, yaitu *dataset Product Detection* yang tersedia secara publik melalui tautan berikut https://universe.roboflow.com/project-frdpl/product-ebb1e/dataset/6. *Dataset* ini terdiri dari ±4155 citra dengan 3 kelas objek yang telah dilengkapi dengan anotasi berupa *bounding box* pada setiap objek.
*Dataset* dibagi menjadi data pelatihan sebesar 80% dan data pengujian sebesar 20% untuk memastikan evaluasi model yang objektif. Data yang digunakan mencakup variasi kondisi pencahayaan, posisi objek, serta kompleksitas latar belakang.

Selain itu, Roboflow menyediakan *dataset* dalam berbagai format seperti YOLO, COCO, dan Pascal VOC, sehingga memudahkan integrasi dengan berbagai *framework deep learning* https://public.roboflow.com/.

### 3.4 *Preprocessing* Data

Tahap *preprocessing* data dilakukan untuk mempersiapkan *dataset* sebelum digunakan dalam pelatihan model *YOLOv8* dan *YOLOv11*. Proses ini bertujuan untuk menyeragamkan format data serta meningkatkan kualitas masukan agar model dapat bekerja secara optimal.

- Resize gambar (640×640)
  Seluruh gambar diubah ukurannya menjadi 640 × 640 piksel untuk memastikan keseragaman dimensi data. Ukuran ini dipilih karena sesuai dengan standar input model YOLO dan mampu memberikan keseimbangan antara akurasi deteksi serta kecepatan pemrosesan.
- Normalisasi
  Normalisasi dilakukan dengan mengubah nilai piksel dari rentang 0–255 menjadi 0–1 melalui pembagian setiap nilai piksel dengan 255. Proses ini membantu model melakukan pelatihan secara lebih stabil, mempercepat konvergensi, dan meningkatkan kemampuan dalam mengenali pola pada gambar.

### 3.5 Arsitektur Model

#### 3.5.1 *YOLOv5*

*YOLOv5* digunakan sebagai model *baseline* dalam penelitian ini untuk memberikan pembanding tambahan terhadap performa *YOLOv8* dan *YOLOv11*. *YOLOv5* merupakan salah satu model deteksi objek yang digunakan karena memiliki keseimbangan antara kecepatan dan akurasi yang baik, sehingga sering dijadikan sebagai *baseline* dalam berbagai penelitian (Saidani, 2023)(Maideen & Mohanarathinam, 2023)(Jocher, 2020).

Beberapa penelitian menunjukkan bahwa *YOLOv5* juga dapat ditingkatkan performanya melalui modifikasi arsitektur untuk meningkatkan akurasi dan kecepatan deteksi (Ji et al., 2024).

#### 3.5.2 *YOLOv8*

*YOLOv8* merupakan model deteksi objek berbasis *deep learning* yang dikembangkan oleh Ultralytics dan dirancang untuk meningkatkan efisiensi serta akurasi deteksi objek secara *real-time* (Jocher et al., 2023).

#### 3.5.3 *YOLOv11*

*YOLOv11* merupakan pengembangan lanjutan dari algoritma YOLO yang menawarkan peningkatan pada arsitektur dan performa deteksi objek dalam berbagai kondisi lingkungan (Khanam & Hussain, 2024).

p-ISSN : 2406-7733
e-ISSN : 2597-9922

### 3.6 Proses Pelatihan Model

Proses pelatihan model deteksi objek ini dilakukan dengan mengonfigurasi beberapa parameter utama. Pengaturan *hyperparameter* yang digunakan selama proses *training* dapat dilihat pada Tabel 1.

Tabel 1. Parameter Konfigurasi Pelatihan Model

| No | Parameter | Nilai | Keterangan |
|---|---|---|---|
| 1 | Epoch | 50 | Jumlah iterasi keseluruhan proses pelatihan *dataset* |
| 2 | *Batch Size* | 16 | Jumlah sampel data yang diproses dalam satu iterasi |
| 3 | *Image Size* | 640 × 640 | Resolusi dimensi citra masukan ke dalam model |
| 4 | *Learning Rate* | 0.01 | Nilai laju pembelajaran awal |

Berdasarkan Tabel 3.1, proses pelatihan model dilakukan menggunakan parameter yang sama pada setiap model untuk memastikan perbandingan hasil yang adil dan konsisten. Parameter *epoch* digunakan untuk menentukan jumlah iterasi pelatihan, *batch size* menentukan jumlah data yang diproses dalam satu kali pelatihan, sedangkan *image size* digunakan untuk menentukan ukuran citra yang diproses oleh model. Selain itu, *learning rate* sebesar 0.01 digunakan untuk mengatur kecepatan model dalam mempelajari pola dari *dataset* selama proses *training* berlangsung.

### 3.7 Evaluasi Kinerja

Evaluasi dilakukan menggunakan metrik berikut:

- *Precision*
- *Recall*
- *F1-Score*
- *mAP* (*mean Average Precision*) (Tian et al., 2024)
- *FPS* (*Frames Per Second*)

*Precision* digunakan untuk mengukur ketepatan model dalam mendeteksi objek. *Recall* digunakan untuk mengetahui kemampuan model dalam menemukan seluruh objek pada *dataset*. *F1-Score* digunakan untuk melihat keseimbangan antara *precision* dan *recall*. Selanjutnya, *mAP* (*mean Average Precision*) digunakan untuk mengukur akurasi keseluruhan model dalam deteksi objek. Selain itu, *FPS* (*Frames Per Second*) digunakan untuk mengukur kecepatan model dalam memproses gambar atau video secara *real-time.*

### 3.8 Analisis Perbandingan

Analisis dilakukan berdasarkan:

- Akurasi (*mAP*)
- Kecepatan (*FPS*)

## IV. HASIL DAN PEMBAHASAN

### 4.1 Hasil Pengujian Model

Berdasarkan hasil pada Tabel 2, *YOLOv5* sebagai model *baseline* menunjukkan performa yang cukup baik dengan nilai *mAP* sebesar 0.81 dan *FPS* sebesar 63. Meskipun memiliki kecepatan inferensi yang lebih tinggi dibandingkan model lainnya, akurasi *YOLOv5* masih berada di bawah *YOLOv11*. Hal ini menunjukkan bahwa model yang lebih baru cenderung mengalami peningkatan dalam hal akurasi, meskipun dalam beberapa kasus terjadi *trade-off* terhadap kecepatan. Dengan demikian, penggunaan *YOLOv5* sebagai *baseline* memberikan gambaran yang jelas mengenai perkembangan performa model YOLO dari generasi sebelumnya ke generasi yang lebih baru.

### 4.2 Hasil Evaluasi Kinerja

Setelah proses pelatihan dan pengujian selesai dilakukan, tahap berikutnya adalah evaluasi kinerja model menggunakan parameter *precision*, *recall*, *F1-score*, *mAP@0.5*, dan *FPS*. Hasil perbandingan kinerja *YOLOv5*, *YOLOv8*, dan *YOLOv11* ditunjukkan pada Tabel 2.

Tabel 2. Hasil perbandingan kinerja *YOLOv5*, *YOLOv8*, dan *YOLOv11*

| Model | *Precision* | *Recall* | *F1-Score* | *mAP@0.5* | *FPS* |
|---|---|---|---|---|---|
| *YOLOv5* | 0.79 | 0.77 | 0.78 | 0.81 | 63 |
| *YOLOv8* | 0.79 | 0.76 | 0.77 | 0.81 | 43 |
| *YOLOv11* | 0.81 | 0.77 | 0.79 | 0.83 | 52 |

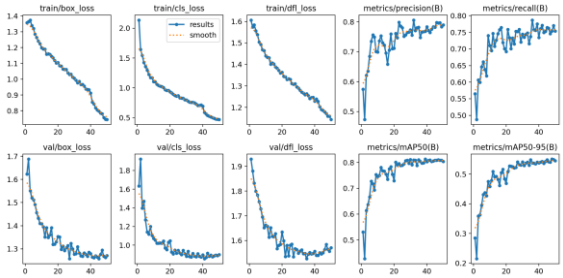
Berdasarkan Tabel 4.1, *YOLOv5* memiliki performa yang cukup baik dengan nilai *mAP@0.5* sebesar 0.81 dan kecepatan inferensi tertinggi yaitu 63 *FPS*. *YOLOv8* memperoleh hasil yang hampir sama dengan *YOLOv5*, namun memiliki *FPS* yang lebih rendah yaitu 43 *FPS*.

Sementara itu, *YOLOv11* menghasilkan performa terbaik dengan nilai *precision* 0.81, *F1-score* 0.79, dan *mAP@0.5* sebesar 0.83. Hasil tersebut menunjukkan bahwa *YOLOv11* memiliki akurasi deteksi objek yang lebih baik dibandingkan model lainnya dengan performa yang tetap stabil.

### 4.3 Visualisasi Hasil Pelatihan

Hasil pelatihan model *YOLOv8* dan *YOLOv11* ditunjukkan pada Gambar 1 dan Gambar 2. Grafik tersebut memperlihatkan perubahan nilai loss dan metrik evaluasi selama proses pelatihan.

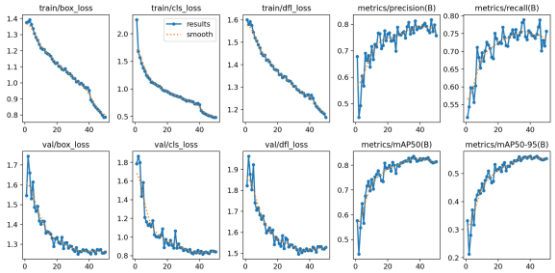
p-ISSN : 2406-7733
e-ISSN : 2597-9922




Gambar 1. Grafik hasil pelatihan *YOLOv8*

Pada Gambar 1 terlihat bahwa nilai *train loss* dan *validation loss* pada model *YOLOv8* mengalami penurunan secara bertahap selama proses pelatihan. Hal ini menunjukkan bahwa model semakin baik dalam mengenali objek dan mempelajari pola data.

Nilai *precision* dan *recall* juga meningkat secara stabil, yang menandakan kemampuan model dalam mendeteksi objek semakin akurat. Selain itu, nilai *mAP50* dan *mAP50-95* mengalami peningkatan hingga mencapai performa yang cukup baik pada akhir pelatihan. Dengan demikian, model *YOLOv8* dapat dikatakan berhasil melakukan proses deteksi objek dengan baik.




Gambar 2. Grafik hasil pelatihan *YOLOv11*

Pada Gambar 2 terlihat bahwa model *YOLOv11* mengalami penurunan nilai *loss* yang lebih cepat dan stabil dibandingkan *YOLOv8*. Hal ini menunjukkan bahwa model memiliki kemampuan pembelajaran yang lebih optimal.

Nilai *precision*, *recall*, *mAP50*, dan *F-95* juga meningkat dengan lebih baik selama proses pelatihan. Hasil tersebut menunjukkan bahwa *YOLOv11* memiliki tingkat akurasi deteksi objek yang lebih tinggi dan performa yang lebih stabil dibandingkan *YOLOv8*.

Berdasarkan grafik, terlihat bahwa nilai loss mengalami penurunan yang konsisten, sementara nilai *precision*, *recall*, dan *mAP* meningkat seiring bertambahnya *epoch*.

### 4.4 Analisis Hasil

Berdasarkan hasil pengujian pada Tabel 4.1, dapat dilihat bahwa model *YOLOv11* memiliki performa yang lebih baik dibandingkan dengan *YOLOv8* dalam hal akurasi. Hal ini ditunjukkan oleh nilai *precision*, *recall*, *F1-score*, dan *mAP* yang lebih tinggi.

Nilai *mAP@*0.5 pada *YOLOv11* mencapai 0.83, lebih tinggi dibandingkan dengan *YOLOv8* sebesar 0.81. Hal ini menunjukkan bahwa *YOLOv11* memiliki kemampuan yang lebih baik dalam mendeteksi dan mengklasifikasikan objek dengan tingkat akurasi yang lebih tinggi.

Dari aspek kecepatan, *YOLOv11* memiliki performa yang lebih baik dengan nilai *FPS* sebesar 52 dibandingkan *YOLOv8* yang memiliki *FPS* sebesar 43. Hal ini menunjukkan bahwa *YOLOv11* menunjukkan performa yang lebih baik dalam hal kecepatan pemrosesan dan lebih cocok digunakan pada sistem yang membutuhkan *real-time* processing dengan latensi rendah.

### 4.5 Pembahasan

Hasil penelitian menunjukkan bahwa model *YOLOv11* memiliki performa yang lebih baik dibandingkan dengan *YOLOv8*, baik dari aspek akurasi maupun kecepatan inferensi. Hal ini terlihat dari nilai evaluasi seperti *precision*, *recall*, *F1-score*, dan *mAP* yang lebih tinggi pada *YOLOv11*, serta nilai *FPS* yang juga lebih besar.

Dengan demikian, tidak ditemukan *trade-off* yang signifikan antara akurasi dan kecepatan pada kedua model, karena *YOLOv11* mampu mengungguli *YOLOv8* pada kedua aspek tersebut. Hal ini menunjukkan adanya peningkatan performa pada arsitektur *YOLOv11* dibandingkan dengan generasi sebelumnya.

Hasil ini sejalan dengan penelitian sebelumnya yang menunjukkan bahwa perkembangan arsitektur YOLO dari versi sebelumnya ke versi terbaru cenderung menghasilkan peningkatan performa baik dari aspek akurasi maupun kecepatan (Ali & Zhang, 2024), (Ge et al., 2021). Hal ini menunjukkan bahwa evolusi model YOLO terus mengarah pada optimasi keseimbangan antara efisiensi komputasi dan ketepatan deteksi objek.

Dalam konteks sistem inventori, model *YOLOv11* lebih direkomendasikan karena mampu memberikan hasil deteksi yang lebih akurat sekaligus mempertahankan kecepatan pemrosesan yang tinggi. Hal ini sangat penting dalam sistem inventori modern yang membutuhkan ketepatan identifikasi barang serta respons sistem yang cepat.

Secara teknis, peningkatan performa *YOLOv11* diduga disebabkan oleh perbaikan pada arsitektur backbone dan neck yang memungkinkan ekstraksi fitur yang lebih representatif. Dengan kemampuan tersebut, model dapat mengenali pola objek dengan lebih baik, terutama pada kondisi data yang kompleks atau bervariasi.

Selain itu, performa kedua model juga dipengaruhi oleh kondisi *dataset*, seperti variasi pencahayaan, sudut pengambilan gambar, dan kompleksitas latar belakang. Berdasarkan hasil pengujian, *YOLOv11* menunjukkan stabilitas yang

p-ISSN : 2406-7733
e-ISSN : 2597-9922

lebih baik dalam menghadapi kondisi lingkungan yang kompleks dibandingkan dengan *YOLOv8*.

Temuan ini juga menunjukkan bahwa kualitas *dataset* memiliki peran penting dalam menentukan performa model. *Dataset* dengan variasi yang tinggi akan membantu model dalam meningkatkan kemampuan generalisasi, sehingga dapat bekerja lebih optimal pada kondisi nyata.

### 4.6 Visualisasi Hasil Deteksi

Pengujian model dilakukan menggunakan *dataset* grid multi-kelas untuk melihat sejauh mana arsitektur model mampu mengenali objek dalam berbagai kondisi. Visualisasi dari hasil pengujian *inference* tersebut dapat dilihat pada Gambar 3 dan 4.



Gambar 3. Hasil deteksi objek menggunakan *YOLOv8*

Hasil deteksi objek menggunakan arsitektur *YOLOv8* menunjukkan kemampuan model dalam mengidentifikasi multi-kelas objek seperti jeruk (*orange*), minuman kaleng (*coca-cola*), dan mi instan (*jin-ramen*) dengan nilai kepercayaan (*confidence score*) tertinggi mencapai 0.7 pada objek tunggal, meskipun pada area objek yang padat masih menyisakan beberapa *bounding box* yang saling tumpang tindih (*overlapping*).



Gambar 4. Hasil deteksi objek menggunakan *YOLOv11*

Hasil pengujian pada arsitektur *YOLOv11* berhasil melokalisasi seluruh target objek secara presisi dengan tingkat kepastian (*confidence score*) yang kuat mencapai nilai 0.8 pada objek jeruk tunggal, serta menunjukkan isolasi *bounding box* yang sangat rapi dan mandiri pada pengelompokan objek yang padat sekaligus mampu menekan munculnya label redundan di area latar belakang.

### 4.7 Keterbatasan Penelitian

Penelitian ini memiliki beberapa keterbatasan yang perlu diperhatikan. Pertama, jumlah *dataset* yang digunakan masih terbatas sehingga dapat mempengaruhi kemampuan generalisasi model terhadap kondisi nyata. Kedua, pengujian model hanya dilakukan pada *dataset* uji dan belum diterapkan secara langsung pada lingkungan operasional sistem inventori di dunia nyata. Selain itu, penelitian ini hanya membandingkan tiga model YOLO tanpa melibatkan metode deteksi objek lainnya sebagai pembanding. Oleh karena itu, hasil penelitian ini masih memiliki ruang untuk dikembangkan lebih lanjut.

### 4.8 Analisis Kesalahan

Berdasarkan hasil pengujian, ditemukan beberapa kesalahan deteksi yang terjadi pada model *YOLOv5*, *YOLOv8*, dan *YOLOv11*. Kesalahan tersebut umumnya terjadi pada kondisi pencahayaan rendah, objek dengan ukuran kecil, serta objek yang saling bertumpuk (*overlapping*).

Pada kondisi pencahayaan rendah, model cenderung mengalami penurunan performa yang ditunjukkan dengan meningkatnya *false negative*, di mana objek tidak terdeteksi. Selain itu, pada objek kecil, *bounding box* yang dihasilkan kurang presisi sehingga mempengaruhi nilai *mAP*.

p-ISSN : 2406-7733
e-ISSN : 2597-9922

Dibandingkan dengan model lainnya, *YOLOv11* menunjukkan kemampuan yang lebih baik dalam mengurangi kesalahan deteksi, terutama pada kondisi lingkungan yang kompleks. Hal ini menunjukkan bahwa arsitektur *YOLOv11* lebih *robust* dalam menghadapi variasi data dibandingkan model sebelumnya.

Permasalahan pada deteksi objek berukuran kecil juga telah dilaporkan dalam penelitian sebelumnya, di mana model *YOLOv5* mengalami penurunan performa pada objek kecil (Shi et al., 2024).

## V. KESIMPULAN DAN SARAN

### 5.1 Kesimpulan

Penggunaan *YOLOv5* sebagai *baseline* dalam penelitian ini berhasil memberikan acuan yang jelas dalam mengevaluasi peningkatan performa model *YOLOv8* dan *YOLOv11*. Hasil perbandingan menunjukkan adanya peningkatan akurasi pada model yang lebih baru, khususnya pada *YOLOv11*.

**Ringkasan Hasil Pengujian**

Berdasarkan hasil pengujian yang telah dilakukan, diperoleh bahwa model *YOLOv11* memiliki performa yang lebih baik dibandingkan *YOLOv8*. Hal ini ditunjukkan dari nilai *precision* (0.81), *recall* (0.77), *F1-score* (0.79), dan *mAP*@0.5 (0.83) yang lebih tinggi dibandingkan *YOLOv8* dengan nilai *precision* (0.79), *recall* (0.76), *F1-score* (0.77), dan *mAP*@0.5 (0.81). Selain itu, dari segi kecepatan, *YOLOv11* juga menunjukkan performa yang lebih baik dengan *FPS* sebesar 52, sedangkan *YOLOv8* sebesar 43.

**Analisis Performa Model**

Dari hasil analisis, model *YOLOv11* menjadi model dengan akurasi terbaik karena memiliki nilai *mAP* dan metrik evaluasi lainnya yang lebih tinggi. Selain itu, *YOLOv11* juga menjadi model dengan kecepatan terbaik berdasarkan nilai *FPS* yang lebih tinggi. Dengan demikian, *YOLOv11* mampu memberikan keseimbangan yang optimal antara akurasi dan kecepatan, sehingga menunjukkan performa yang lebih baik dibandingkan *YOLOv8*.

**Kelebihan dan Kekurangan Model**

*YOLOv8* memiliki kelebihan pada arsitektur yang lebih ringan dan fleksibel, sehingga cukup baik untuk digunakan pada perangkat dengan spesifikasi terbatas. Namun, kekurangannya terletak pada tingkat akurasi dan kecepatan yang masih berada di bawah *YOLOv11*.

Sementara itu, *YOLOv11* memiliki kelebihan dalam hal akurasi yang lebih tinggi, kecepatan pemrosesan yang lebih cepat, serta kemampuan yang lebih baik dalam menghadapi kondisi lingkungan yang kompleks. Kekurangan *YOLOv11* adalah kemungkinan membutuhkan sumber daya komputasi yang lebih besar dibandingkan *YOLOv8*.

**Implikasi pada Sistem Inventori**

Penggunaan model deteksi objek berbasis YOLO dalam sistem inventori memberikan dampak positif terhadap efisiensi pengelolaan barang. Dengan akurasi yang lebih tinggi, kesalahan dalam identifikasi barang dapat diminimalkan. Selain itu, kecepatan pemrosesan yang tinggi memungkinkan sistem bekerja secara *real-time*. Dalam hal ini, *YOLOv11* lebih sesuai untuk diimplementasikan pada sistem inventori modern karena mampu memberikan hasil yang cepat dan akurat.

**Jawaban Tujuan Penelitian**

Tujuan penelitian untuk menganalisis dan membandingkan kinerja *YOLOv8* dan *YOLOv11* telah tercapai. Berdasarkan hasil yang diperoleh, dapat disimpulkan bahwa *YOLOv11* merupakan model yang lebih direkomendasikan untuk digunakan dalam sistem inventori karena memiliki performa yang lebih baik dari segi akurasi dan kecepatan.

Penggunaan *YOLOv5* sebagai *baseline* dalam penelitian ini berhasil memberikan acuan yang jelas dalam mengevaluasi peningkatan performa model *YOLOv8* dan *YOLOv11*. Hasil perbandingan menunjukkan adanya peningkatan akurasi pada model yang lebih baru, khususnya pada *YOLOv11*.

### 5.2 Saran

**Saran Pengembangan Model**

Penelitian selanjutnya disarankan untuk menggunakan *dataset* yang lebih besar dan lebih beragam agar model dapat memiliki kemampuan generalisasi yang lebih baik. Selain itu, perlu dilakukan optimasi parameter seperti jumlah *epoch*, *batch size*, dan *learning rate* untuk meningkatkan performa model.

**Saran Implementasi Sistem**

Model yang dihasilkan sebaiknya diuji pada lingkungan nyata untuk mengetahui performa secara langsung dalam kondisi operasional. Selain itu, sistem dapat dikembangkan dengan mengintegrasikan model deteksi objek ke dalam aplikasi berbasis *web* atau *mobile* untuk mempermudah pengelolaan inventori.

**Saran Penelitian Selanjutnya**

Penelitian selanjutnya dapat melakukan perbandingan dengan algoritma lain selain YOLO untuk mendapatkan hasil evaluasi yang lebih luas. Selain itu, dapat dikembangkan dengan memanfaatkan teknologi tambahan seperti *Internet of Things (IoT)*, *cloud computing*, atau *edge computing* untuk meningkatkan efisiensi sistem.

p-ISSN : 2406-7733
e-ISSN : 2597-9922

**Saran Teknis**

Disarankan untuk menggunakan perangkat keras (*hardware*) yang lebih optimal, seperti *GPU* dengan spesifikasi tinggi, agar proses pelatihan dan pengujian model dapat berjalan lebih cepat. Selain itu, pengujian pada kondisi lingkungan yang lebih kompleks juga perlu dilakukan untuk menguji ketahanan model dalam berbagai situasi.

## DAFTAR PUSTAKA

Ali, M. L., & Zhang, Z. (2024). The YOLO Framework: A Comprehensive Review of Evolution, Applications, and Benchmarks in Object Detection. *Computers*, *13*(12). https://doi.org/10.3390/computers13120336

Bochkovskiy, A., Wang, C.-Y., & Liao, H.-Y. M. (2020). *YOLOv4: Optimal Speed and Accuracy of Object Detection*. http://arxiv.org/abs/2004.10934

Ge, Z., Liu, S., Wang, F., Li, Z., & Sun, J. (2021). *YOLOX: Exceeding YOLO Series in 2021*. 1–7. http://arxiv.org/abs/2107.08430

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2016-Decem*, 770–778. https://doi.org/10.1109/CVPR.2016.90

Ji, C. L., Yu, T., Gao, P., Wang, F., & Yuan, R. Y. (2024). Yolo-tla: An Efficient and Lightweight Small Object Detection Model based on YOLOv5. *Journal of Real-Time Image Processing*, *21*(4), 1–19. https://doi.org/10.1007/s11554-024-01519-4

Jocher, G. (2020). *YOLOv5 by Ultralytics*. Ultralytics. https://doi.org/10.5281/zenodo.3908559

Jocher, G., Chaurasia, A., & Qiu, J. (2023). *Ultralytics YOLOv8*. Ultralytics. https://github.com/ultralytics/ultralytics

Khanam, R., & Hussain, M. (2024). *YOLOv11: An Overview of the Key Architectural Enhancements*. *2024*, 1–9. http://arxiv.org/abs/2410.17725

Liu, L., Ouyang, W., Wang, X., Fieguth, P., Chen, J., Liu, X., & Pietikäinen, M. (2020). Deep Learning for Generic Object Detection: A Survey. *International Journal of Computer Vision*, *128*(2), 261–318. https://doi.org/10.1007/s11263-019-01247-4

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016). SSD: Single shot multibox detector. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *9905 LNCS*, 21–37. https://doi.org/10.1007/978-3-319-46448-0_2

Maideen, A., & Mohanarathinam, A. (2023). Computer Vision-Assisted Object Detection and Handling Framework for Robotic Arm Design Using YOLOV5. *Advances in Distributed Computing and Artificial Intelligence Journal*, *12*, 1–16. https://doi.org/10.14201/ADCAIJ.31586

Plastiras, G., Kyrkou, C., & Theocharides, T. (2018). Efficient convnet-based object detection for unmanned aerial vehicles by selective tile processing. *ACM International Conference Proceeding Series*. https://doi.org/10.1145/3243394.3243692

Saidani, T. (2023). Deep Learning Approach: YOLOv5-based Custom Object Detection. *Engineering, Technology and Applied Science Research*, *13*(6), 12158–12163. https://doi.org/10.48084/etasr.6397

Shi, Y., Jia, Y., & Zhang, X. (2024). FocusDet: an efficient object detector for small object. *Scientific Reports*, *14*(1), 1–16. https://doi.org/10.1038/s41598-024-61136-w

Tang, W. (2024). Improvement of Inventory Management and Demand Forecasting by Big Data Analytics in Supply Chain. *Applied Mathematics and Nonlinear Sciences*, *9*(1), 1–15. https://doi.org/10.2478/amns-2024-2213

Tian, J., Jin, Q., Wang, Y., Yang, J., Zhang, S., & Sun, D. (2024). Performance analysis of deep learning-based object detection algorithms on COCO benchmark: a comparative study. *Journal of Engineering and Applied Science*, *71*(1), 1–18. https://doi.org/10.1186/s44147-024-00411-z

Villegas-Ch, W., Navarro, A. M., & Sanchez-Viteri, S. (2024). Optimization of inventory management through computer vision and machine learning technologies. *Intelligent Systems with Applications*, *24*(June), 200438. https://doi.org/10.1016/j.iswa.2024.200438

Wang, C. Y., Bochkovskiy, A., & Liao, H. Y. M. (2023). YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2023-June*, 7464–7475. https://doi.org/10.1109/CVPR52729.2023.00721