# Analisis Kinerja *Deep Learning* untuk Deteksi Penyakit Tanaman Kentang


1st Devi Luthfitaningrum
*Departemen Teknik Elektro dan Teknologi Informasi*
*Universitas Gadjah Mada*
Sleman, Indonesia
deviluthfitaningrum@mail.ugm.ac.id

2nd Indah Soesanti
*Departemen Teknik Elektro dan Teknologi Informasi*
*Universitas Gadjah Mada*
Sleman, Indonesia
indahsoesanti@ugm.ac.id

3rd Dzuhri Radityo Utomo
*Departemen Teknik Elektro dan Teknologi Informasi*
*Universitas Gadjah Mada*
Sleman, Indonesia
dzuhri.r.u@ugm.ac.id



***Abstract*—Kentang adalah salah satu tanaman umbi yang berpotensi menjadi makanan pokok, serta memiliki tinggi manfaat bagi tubuh. Banyaknya peran kentang menyebabkan peningkatan permintaan dan nilai ekonomi kentang. Namun produktivitas kentang di Indonesia pada tahun 2022 masih sangat rendah yaitu 1,93 ton/ha. Selain itu, impor kentang Indonesia pada Januari 2023 sangat tinggi yaitu sebesar 7160 ton. Berdasarkan rendahnya produktivitas dan tingginya impor kentang di Indonesia, dapat diketahui bahwa produksi kentang di Indonesia belum memenuhi permintaan. Faktor utama yang menghambat produksi kentang adalah penyakit tanaman. Penyakit yang umumnya menyerang dan memiliki dampak besar pada tanaman kentang adalah *early blight* (bercak kering) dan *late blight* (hawar daun). Penyakit tersebut ditandai dengan pola pada daun yang dapat diklasifikasikan oleh model *machine learning* maupun *deep learning*. Namun pada penelitian sebelumnya, model belum dapat mendeteksi lokasi daun kentang yang terinfeksi. Pada penelitian ini, model *deep learning* diterapkan untuk mendeteksi kelas dan lokasi penyakit tanaman kentang pada daun. Model *deep learning* yang digunakan adalah model *Convolutional Neural Network* (CNN) dari keluarga *You Only Look Once* (YOLO) yaitu YOLOv5m, YOLOv6m, YOLOv7, dan YOLOv8m. Model dilatih dengan 2100 citra dan divalidasi dengan 600 citra, kemudian diuji dengan 300 citra. Pengujian dilakukan untuk mengetahui kinerja model berdasarkan akurasi deteksi, kompleksitas model, dan waktu komputasi. Hasil pengujian kemudian dievaluasi untuk menemukan model dengan kinerja terbaik. Model dengan kinerja terbaik untuk mendeteksi penyakit tanaman kentang adalah YOLOv5m dengan mAP@0.5 bernilai 0,995, mAP@0.5:0.95 bernilai 0,973, parameter berjumlah 20,9 juta, waktu pelatihan 0,80 jam, dan waktu deteksi 17,50 ms. YOLOv5m dipilih sebagai model terbaik karena memiliki kompleksitas model dan waktu komputasi paling rendah dibanding ketiga model lainnya, tetapi memiliki akurasi yang memuaskan.***



***Index Terms*—*Deep Learning*, *Convolutional Neural Network* (CNN), *You Only Look Once* (YOLO), *Early Blight*, *Late Blight***


## I. PENDAHULUAN

Kentang adalah salah satu tanaman umbi yang berpotensi menjadi makanan pokok, serta memiliki tinggi manfaat bagi tubuh. Banyaknya peran kentang menyebabkan peningkatan permintaan dan nilai ekonomi kentang. Dengan begitu produktivitas kentang di Indonesia perlu dikaji dan ditingkatkan lagi. Berdasarkan data yang dipublikasikan oleh Badan Pusat Statistik [1], panen kentang di Indonesia tahun 2022 berjumlah 1,42 juta ton. Sedangkan luas lahan panen kentang di Indonesia, menurut data Badan Pusat Statistik [2] adalah 73714 ha pada tahun 2022, maka produktivitas kentang di Indonesia tahun 2022 adalah 1,93 ton/ha. Hal ini menunjukkan bahwa produktivitas kentang di Indonesia masih sangat rendah dibanding negara-negara Eropa yang berkisar antara 40-60 ton/ha [3]. Impor kentang Indonesia pada Januari 2023 juga tinggi, yaitu sebanyak 7160 ton [4]. Rendahnya produktivitas dan tingginya impor kentang di Indonesia menunjukkan bahwa produksi kentang di Indonesia belum memenuhi permintaan.

Salah satu penyebab rendahnya produksi kentang adalah adanya penyakit tanaman. Penyakit yang umumnya menyerang dan memiliki dampak besar pada tanaman kentang adalah *early blight* (bercak kering) dan *late blight* (hawar daun). Kedua penyakit tersebut dapat menyebabkan kematian tanaman bahkan memunculkan risiko gagal panen. Dalam penanganan penyakit tanaman kentang, diperlukan proses identifikasi penyakit yang menginfeksi. Untuk mempermudah identifikasi penyakit, beberapa penelitian di bidang visi komputer telah diterapkan. Penyakit *early blight* (bercak kering) dan *late blight* (hawar daun) ditandai munculnya pola pada daun yang dapat dikenali oleh model *machine learning*. Volume data yang bertambah menyebabkan para peneliti beralih ke *deep learning*. Salah satu metode *deep learning* yang banyak diterapkan pada identifikasi citra adalah *Convolutional Neural Network* (CNN). Menurut hasil penelitian Mohamed, S. I. [5], CNN memiliki akurasi lebih tinggi dibanding *machine learning*. Selain itu, model CNN dari keluarga *You Only Look Once* (YOLO) yaitu YOLOv5s pernah diterapkan untuk mengklasifikasi penyakit tanaman kentang dengan hasil akurasi yang tinggi. Dengan demikian, beberapa penelitian sebelumnya menunjukkan bahwa *machine learning* maupun *deep learning* dapat mengidentifikasi kelas penyakit tanaman kentang pada daun, tetapi belum dapat mendeteksi lokasi daun kentang yang terinfeksi.

Pada penelitian ini, model keluarga YOLO akan diterapkan untuk mendeteksi kelas dan lokasi penyakit tanaman kentang pada daun. Namun keluarga YOLO memiliki lebih dari 30 detektor sehingga belum diketahui model YOLO terbaik untuk mendeteksi penyakit tanaman kentang. Dengan demikian, penelitian ini bertujuan untuk menganalisis kinerja model YOLO dalam deteksi penyakit tanaman kentang.

## II. Dasar Teori

### A. Penyakit Tanaman Kentang Early Blight

Fig. 1. Penyakit *Early Blight* [6]

*Early blight* atau penyakit bercak kering disebabkan oleh patogen jamur *Alternaria solani*, patogen ini menyukai suhu di antara 24-29 °C dan kelembaban tinggi [7]. Patogen jamur dapat menginfeksi tanaman kentang melalui bibit atau tanah yang terkontaminasi. Produksi spora jamur *Alternaria solani* meningkat apabila terjadi cuaca basah dan kering secara bergantian. Hal ini biasanya terjadi ketika periode curah hujan tinggi di daerah tropis dan subtropis.

Penyakit *early blight* terjadi pada daun yang lebih tua dengan gejala bercak cincin melingkar berwarna abu-abu hingga coklat kekuningan [8]. Penyakit ini mengakibatkan daun akan gugur sehingga kentang rentan terhadap luka bakar matahari. Bercak ini juga menjalar ke batang dan umbi kentang. Menurut Xue dkk [9], penyakit *early blight* merupakan penyakit daun kentang yang sangat penting bagi perekonomian dunia. Penyakit tersebut bisa menyebabkan kematian hingga gagal panen. Gambar 1 menunjukkan contoh daun dari tanaman kentang yang terinfeksi *early blight*.

### B. Penyakit Tanaman Kentang Late Blight


Fig. 2. Penyakit *Late Blight* [10]

*Late blight* atau penyakit hawar daun disebabkan oleh patogen *oomycete* (jamur air) *Phytopthora infestans*, patogen ini menyukai suhu dingin di antara 12-18 °C [7]. Jamur *Phytopthora infestans* termasuk parasit obligat yang menumpang pada inangnya untuk bertahan hidup melewati musim dingin. Jamur ini menginfeksi inang melalui luka dan robekan di kulit pada kondisi basah yang panjang dengan kelembaban relatif 90%. Penyakit ini umumnya menyerang tanaman kentang berumur 5-6 minggu.

Gejala penyakit *late blight* adalah membusuknya daun dengan munculnya bercak coklat mulai dari tepi daun dan tumbuhnya jamur menyerupai kapas di bagian bawah daun. Gejala ini akan menyebabkan daun layu dan menyebar ke tangkai, batang, serta umbi [11]. Umbi kentang yang terinfeksi akan memiliki kulit berbintik biru dan daging berwarna coklat sehingga tidak bisa dikonsumsi. Oleh karena itu, penyakit ini membahayakan tanaman dan menyebabkan petani gagal panen hingga menimbulkan aroma membusuk pada lahan.

### C. Deteksi Objek


Fig. 3. Perbedaan Klasifikasi Objek dan Deteksi Objek [12]

Deteksi objek adalah pengembangan dari klasifikasi objek. Klasifikasi objek dan deteksi objek merupakan bagian dari bidang ilmu visi komputer. Visi komputer merupakan pengetahuan dan kecerdasan komputer untuk mengenali dan menganalisis citra sehingga menghasilkan suatu interpretasi atau persepsi bagi komputer. Interpretasi yang dihasilkan pada proses klasifikasi objek adalah kelas objek. Sementara interpretasi yang dihasilkan pada proses deteksi objek adalah kelas dan letak suatu objek pada citra. Dengan demikian, proses deteksi objek pada bidang visi komputer adalah proses klasifikasi dan lokalisasi objek. Gambar 3 adalah contoh perbedaan hasil klasifikasi dan deteksi terhadap anjing dan serigala yang sangat mirip apabila dilihat secara sekilas.

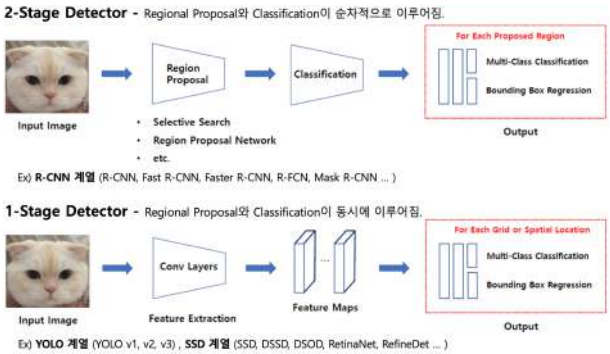


Fig. 4. Perbedaan *two-stage detector* dan *one-stage detector* [13]

Deteksi objek dapat dilakukan dengan 2 metode, yaitu metode tradisional dan *deep learning* [14]. Deteksi objek berbasis *deep learning* dikembangkan sejak tahun 2014 dan

dinyatakan memiliki kinerja lebih baik dibanding metode tradisional [15]. Deteksi objek berbasis *deep learning* tergolong menjadi 2 detektor yaitu *two-stage detector* dan *one-stage detector*. *One-stage detector* lebih efisien dalam komputasi dan lebih cepat dalam inferensi karena model melakukan klasifikasi dan lokalisasi objek dalam satu tahap. Perbedaan proses deteksi menggunakan *one-stage detector* dan *two-stage detector* diilustrasikan pada Gambar 4. Dalam perkembangannya, metode deteksi objek selalu dievaluasi kinerjanya meliputi akurasi deteksi, kompleksitas model, dan waktu komputasi.

### D. Akurasi Deteksi

*1) Confusion matrix: Confusion matrix* dapat ditentukan dengan membandingkan kelas aktual dengan kelas prediksi dari suatu objek. *Confusion matrix* berupa matriks yang berisi nilai kejadian *True Positive, True Negative, False Positive*, dan *False Negative*. Untuk menghitung nilai-nilai kejadian pada *confusion matrix*, dalam deteksi objek dikenal *Jaccard Index* atau disebut juga *Intersection over Union* (IoU). IoU membandingkan area irisan terhadap area gabungan antara kotak label pada objek $(B_{aktual})$ dengan kotak prediksi $(B_{prediksi})$. Perhitungan IoU ditunjukkan pada Persamaan 1 [16].

$$IoU = \{\frac{B_{aktual} \cap B_{prediksi}}{B_{aktual} \cup B_{prediksi}}\} \tag{1}$$

Selanjutnya menggunakan nilai IoU dan ambang batas dapat ditentukan nilai-nilai *confusion matrix* sebagai berikut:

- ***True Positive (TP)***: Model memprediksi objek secara tepat dengan nilai $IoU \geq \alpha$.
- ***True Negative (TN)***: Dalam deteksi objek, hal ini tidak berlaku karena terlalu banyak kemungkinan model dalam memprediksi *background* [17].
- ***False Positive (FP)***: Pada deteksi objek, kejadian ini terjadi apabila kotak prediksi memiliki nilai $IoU < \alpha$ terhadap kotak aktual.
- ***False Negative (FN)***: Model tidak dapat memprediksi objek ke kelas objek yang sebenarnya.

*2) Performance Evaluation Metrics:* Setelah menentukan nilai *true positive, true negative, false positive,* dan *false negative*, kita dapat mengukur akurasi menggunakan metrik evaluasi kinerja. Beberapa metrik yang digunakan untuk mengukur akurasi model deteksi objek adalah *precision* (presisi), *recall* atau *sensitivity* (sensitivitas), dan *mean Average Precision* (mAP).

- ***Precision***: *Precision* merupakan perbandingan antara citra terdeteksi positif benar dengan seluruh citra terdeteksi positif. *Precision* yang dihasilkan model dapat direpresentasikan dengan Persamaan 2 [17].

$$Precision = \{\frac{TP}{TP + FP}\} \tag{2}$$

  *Precision* akan bernilai rendah jika jumlah *false positive* tinggi, hal ini akan terjadi jika ambang batas IoU rendah. Sebaliknya *precision* akan bernilai tinggi jika jumlah *false positive* rendah, hal ini akan terjadi jika ambang batas IoU tinggi. Tetapi ambang batas IoU terlalu tinggi menyebabkan banyak objek tidak terdeteksi sehingga *false negative* menjadi tinggi, sehingga dapat menurunkan akurasi model.
- ***Recall***: *Recall* merupakan perbandingan antara citra terdeteksi positif benar dengan seluruh citra positif. *Recall* yang dihasilkan model dapat direpresentasikan dengan Persamaan 3 [17].

$$Recall = \{\frac{TP}{TP + FN}\} \tag{3}$$

  *Recall* akan bernilai rendah jika jumlah *false negative* tinggi, hal ini akan terjadi jika ambang batas IoU tinggi. Sebaliknya *recall* akan bernilai tinggi jika jumlah *false negative* rendah, hal ini akan terjadi jika ambang batas IoU rendah. Tetapi ambang batas IoU terlalu rendah menyebabkan banyak objek terdeteksi tidak tepat sehingga *false positive* menjadi tinggi, sehingga dapat menurunkan akurasi model.
- ***mean Average Precision***: Adanya *trade-off* antara metrik *precision* dan *recall*, menunjukkan bahwa model yang baik adalah model yang memiliki *precision* dan *recall* tetap tinggi pada berbagai variasi ambang batas $(\alpha)$ IoU. Pada deteksi objek dikenal suatu kurva yang memplotting nilai *precision* terhadap *recall*. Kurva *Precision-Recall* menunjukkan akurasi model dengan menghitung luas area di bawah kurva. Luas *Area Under Curve* (AUC) mempresentasikan rata-rata *precision* di semua nilai *recall*. Luas AUC bernilai antara 0 hingga 1 yang dapat dijadikan metrik untuk mengukur akurasi deteksi objek, metrik ini disebut *Average Precision* (AP). Cara menghitung AP dapat ditunjukkan dengan Persamaan 4 [18].

$$AP@\alpha = \{\int_0^1 p(r)dr\} \tag{4}$$

  Namun metrik ini hanya dapat menghitung akurasi deteksi satu kelas objek. Pada model yang dapat mendeteksi objek *multi-class* atau lebih dari satu kelas, maka perlu dihitung terlebih dahulu nilai AP setiap kelas. Kemudian untuk mengukur akurasi model secara keseluruhan, dicari rata-rata dari nilai AP setiap kelas. Oleh karena itu, terdapat metrik yang disebut *mean Average Precision* (mAP) untuk mengukur akurasi deteksi objek *multi-class*. Cara menghitung mAP ditunjukkan pada Persamaan 5 [18].

$$mAP@\alpha = \{\frac{1}{n}\sum_{i=1}^{n} AP_i\} \tag{5}$$

  Nilai mAP dapat dihitung dengan berbagai ambang batas $(\alpha)$ IoU, metrik yang sering digunakan pada evaluasi deteksi objek adalah mAP@0.5 dan mAP@0.5:0.95. mAP@0.5 artinya menggunakan ambang batas IoU 0,5, sedangkan mAP@0.5:0.95 artinya menghitung metrik dengan variasi nilai ambang batas IoU dari 0,5 hingga 0,95.

### E. Kompleksitas Model

Kompleksitas model dapat diukur dari jumlah parameter pada model [19]. Semakin banyak jumlah parameter model maka kompleksitas modelnya semakin tinggi.

### F. Waktu Komputasi

Pada deteksi objek, waktu yang dibutuhkan dalam proses komputasi terdiri dari waktu pelatihan dan waktu deteksi. Waktu pelatihan adalah waktu yang dibutuhkan dalam melatih model dan memvalidasi model. Waktu deteksi adalah waktu yang dibutuhkan model dalam mendeteksi objek pada suatu citra.

### G. Deep Learning

*Deep learning* merupakan bagian dari *machine learning* yang dikembangkan sehingga memiliki kinerja lebih unggul dibanding *machine learning*. *Deep learning* mampu mempelajari data tanpa diberikan fitur oleh manusia, model akan membuat fitur secara mandiri berdasarkan pola dari data yang diberikan. *Deep learning* memiliki jaringan saraf untuk belajar dan berlatih memahami data yang besar. Jaringan saraf tersebut memiliki komputasi sangat kompleks dan berlapis-lapis menyerupai jaringan saraf manusia.

### H. Convolutional Neural Network (CNN)

*Convolutional Neural Network* (CNN) merupakan teknik *deep learning* yang jaringannya memiliki bobot bersama dan koneksi lokal. Bobot bersama dan koneksi lokal pada CNN menyederhanakan jaringan dan mempercepat proses pelatihan [20]. Fungsi tersebut sangat berguna untuk memproses input 2 dimensi seperti citra. Hal ini menyerupai sistem pengenalan pada *visual cortex* manusia [21]. Arsitektur CNN memiliki beberapa komponen yaitu *convolutional layer, pooling layer, fully connected layer, activation function,* dan *loss function*.

### I. You Only Look Once (YOLO)

*You Only Look Once* (YOLO) adalah model *one-stage detector* pertama yang mengaplikasikan arsitektur CNN. Prinsip kerja algoritma YOLO dipaparkan pada Gambar 5. YOLO bekerja dengan membagi citra ke beberapa *grid*, kemudian setiap *grid* memprediksi *bounding box* dan probabilitas kelas [22]. *Bounding box* adalah kotak pembatas yang memprediksi 5 nilai terkait objek pada setiap *grid* berukuran s x s. *Bounding box* memprediksi *object confidence score*, x, y, *width*, dan *height* [16]. *Object confidence score* menunjukkan tingkat kepercayaan model terhadap keberadaan objek yang mirip dengan objek pada data pelatihan. Nilai x dan y adalah koordinat titik pusat dari objek yang dideteksi pada *grid*. Nilai *width* dan *height* adalah lebar dan tinggi *bounding box* dari objek. Kemudian *grid* akan memprediksi nilai probabilitas kelas jika berdasarkan *confidence score* diprediksi terdapat objek di dalam *grid* tersebut. YOLOv1 memiliki waktu komputasi yang cepat dan efisien, tetapi memiliki akurasi yang masih rendah dibanding model *two-stage detector*. Kemudian model YOLO selalu diteliti dan dikembangkan hingga adanya model-model versi modern dari YOLO. Model YOLO versi modern ini berhasil meningkatkan akurasinya, seperti YOLOv5, YOLOv6, YOLOv7, dan YOLOv8. Model-model tersebut memiliki arsitektur dan prinsip kerja yang berbeda. Hal ini menyebabkan setiap model memiliki akurasi deteksi, kompleksitas model, dan waktu komputasi yang berbeda pada implementasinya.

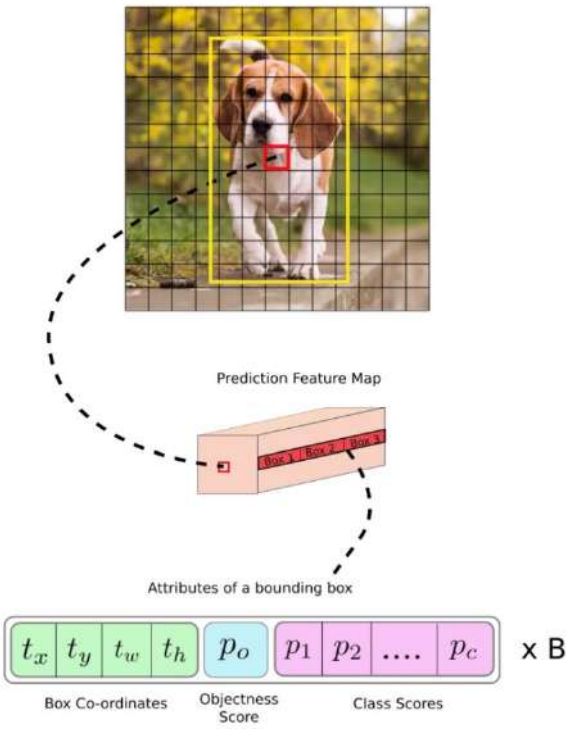


Fig. 5. Prinsip Kerja Algoritma YOLO [16]

## III. Metode Penelitian

### A. Bahan Penelitian

Penelitian ini menggunakan dataset dari 3 sumber yaitu PlantVillage Dataset [23], Potato Leaf Dataset [24], dan Potato Disease Leaf Dataset [25]. Sampel dataset dibagi secara acak dengan persentase 70% data pelatihan, 20% data validasi, dan 10% data pengujian. Jumlah hasil pembagian dataset dapat dirinci pada Tabel I.

TABLE I
Pembagian Dataset

| Kelas | Data Pelatihan | Data Validasi | Data Pengujian |
|---|---|---|---|
| *Healthy* | 700 | 200 | 100 |
| *Early Blight* | 700 | 200 | 100 |
| *Late Blight* | 700 | 200 | 100 |

### B. Metode Penelitian

Pada penelitian ini akan diuji model-model YOLO modern untuk mendeteksi penyakit yang menginfeksi tanaman kentang melalui citra daunnya. Daun kentang bukan merupakan objek yang kecil, sehingga dapat digunakan ukuran citra pelatihan *default* model yaitu ukuran 640 x 640 (dalam piksel).

Perangkat yang digunakan pada penelitian adalah GPU T4, maka analisis kinerja dilakukan terhadap model yang ideal diimplementasikan pada GPU. Model yang ideal diimplementasikan pada GPU adalah model YOLOv5m, YOLOv6m, YOLOv7, dan YOLOv8m. Perbandingan arsitektur dan prinsip kerja keempat model dapat dirangkum pada Tabel II.

TABLE II
ARSITEKTUR YOLOV5M, YOLOV6M, YOLOV7, DAN YOLOV8M

| YOLOv5m | YOLOv6m | YOLOv7 | YOLOv8m |
|---|---|---|---|
| CSPDarknet | EfficientRep | E-ELAN | Darknet |
| SPPF, CSP-PAN | CSPStackRep | PAN | SPPF, PAN |
| YOLOv3 | *decoupled head* | *double head* | *split head* |
| *Anchor-based* | *Anchor-free* | *Anchor-based* | *Anchor-free* |
| *box, obj, cls* | *iou, dfl, cls* | *box, obj, cls* | *box, dfl, cls* |

Penelitian ini termasuk penelitian kuantitatif yang dilakukan dengan eksperimen. Eksperimen diawali dengan melakukan pelatihan terhadap 4 model deteksi objek YOLO. Pelatihan model dikonfigurasi dengan nilai *default* hyperparameter dari model, kecuali nilai epoch. Epoch yang digunakan untuk pelatihan model adalah 50. Eksperimen dilanjutkan dengan melakukan pengujian terhadap kinerja model meliputi akurasi deteksi, kompleksitas model, dan waktu komputasi. Akurasi deteksi dianalisis berdasarkan metrik mAP@0.5 dan mAP@0.5:0.95. Kompleksitas model dibandingkan berdasarkan jumlah parameter. Waktu komputasi dianalisis berdasarkan waktu pelatihan yang diukur dalam satuan jam dan waktu deteksi yang diukur dalam satuan milisekon. Berdasarkan pengujian tersebut dapat dianalisis dan dievaluasi kinerja model sehingga diketahui kelebihan dan kekurangan model. Dengan mengetahui kelebihan dan kekurangan model dapat diketahui model terbaik untuk mendeteksi penyakit tanaman kentang. Dalam melakukan eksperimen, peneliti menggunakan Python sebagai bahasa pemrograman yang dijalankan pada *runtime* Google Colaboratory.

### C. Alur Penelitian

Tugas akhir ini dimulai dengan merumuskan fokus dari tugas akhir. Berdasarkan fokus yang sudah dirumuskan, dilanjutkan perancangan jalannya penelitian pada tugas akhir ini. Perancangan meliputi pemilihan dataset, pemilihan model, dan perancangan evaluasi model. Tahap selanjutnya adalah pelaksanaan rancangan tugas akhir, tahap ini disebut implementasi tugas akhir. Implementasi meliputi anotasi data, pelatihan dan validasi model, pengujian model, dan evaluasi model. Kemudian tugas akhir ini diakhiri dengan tahap penulisan naskah ilmiah. Alur penelitian pada tugas akhir ini dapat dilihat pada Gambar 6.

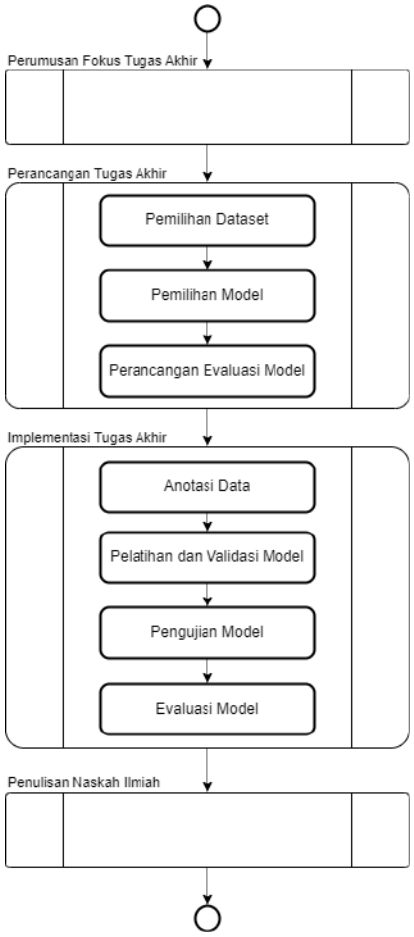


Fig. 6. Alur Tugas Akhir

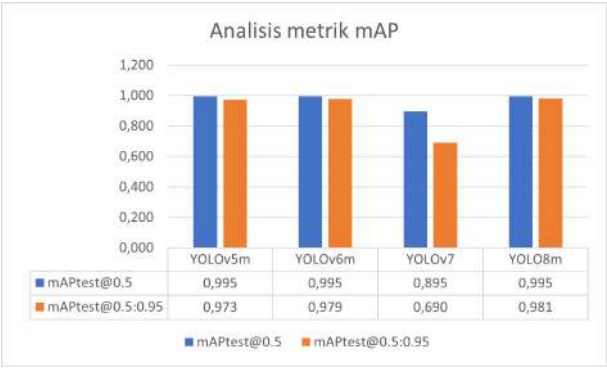


Fig. 7. Hasil Akurasi Deteksi Model

## IV. HASIL DAN PEMBAHASAN

### A. Pengujian Model

*1) Akurasi Deteksi:* Akurasi Deteksi diukur menggunakan metrik mAP@0.5 dan mAP@0.5:0.95. Perhitungan mAP@0.5 dan mAP@0.5:0.95 berdasarkan 300 data pengujian terhadap model yang dilatih selama 50 epoch divisualisasikan pada Gambar 7.

Berdasarkan nilai mAP@0.5 dan mAP@0.5:0.95, YOLOv7 memiliki akurasi deteksi paling rendah secara signifikan di antara keempat model. Sementara ketiga model lainnya memiliki mAP@0.5 yang sama yaitu bernilai 0,995, namun terdapat sedikit perbedaan pada mAP@0.5:0.95 dengan nilai tertinggi dicapai oleh model YOLOv8m.

*2) Kompleksitas Model:* Kompleksitas model dapat diukur berdasarkan jumlah parameter model ketika diimplementasikan pada dataset. Jumlah parameter model YOLOv5m,


Fig. 8. Perbandingan Jumlah Parameter Model

YOLOv6m, YOLOv7, dan YOLOv8m dipaparkan pada Gambar 8

Kompleksitas model paling rendah adalah YOLOv5m dengan jumlah parameter 20,9 juta, kemudian diikuti YOLOv8m dengan jumlah parameter 25,8 juta. Kedua model tersebut memiliki parameter yang jauh lebih sedikit dibanding YOLOv6m yang berjumlah 34,8 juta. Jumlah parameter YOLOv6m tidak memiliki perbedaan signifikan dengan model YOLOv7 yang memiliki jumlah parameter terbanyak. Dengan demikian, YOLOv7 dan YOLOv6m kurang efisien karena kompleksitas modelnya tinggi. Model dengan kompleksitas tinggi membutuhkan memori dan harga komputasi yang tinggi.

*3) Waktu Komputasi:* Waktu komputasi dianalisis berdasarkan waktu pelatihan dan waktu deteksi. Waktu pelatihan adalah waktu yang diperlukan untuk melatih dan memvalidasi model. Pada penelitian ini waktu pelatihan diukur dalam satuan jam. Waktu pelatihan untuk setiap model dapat dilihat pada Gambar 9.

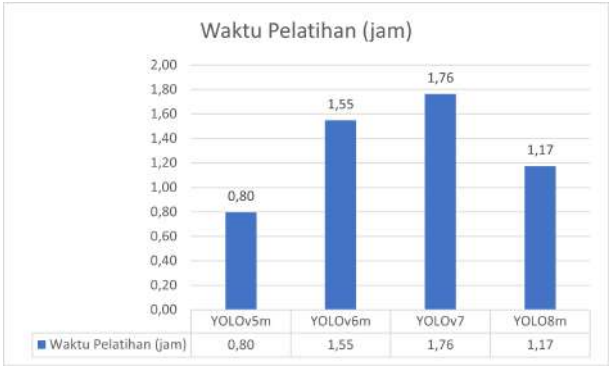


Fig. 9. Waktu Pelatihan Model

YOLOv7 memerlukan waktu pelatihan paling lama dibanding model lainnya dengan waktu 1,76 jam selama pelatihan 50 epoch. Berikutnya adalah model YOLOv6m, model YOLOv8m, kemudian terakhir model YOLOv5m yang memerlukan waktu pelatihan paling cepat. Waktu deteksi adalah waktu yang dibutuhkan model dalam mendeteksi objek pada satu citra.Pada penelitian ini pengujian dilakukan pada 300 citra, sehingga waktu deteksi yang dianalisis adalah rata-rata waktu deteksi dari 300 citra. Waktu deteksi diukur dalam satuan milisekon. Perbandingan waktu deteksi pada pengujian keempat model dapat divisualisasikan pada

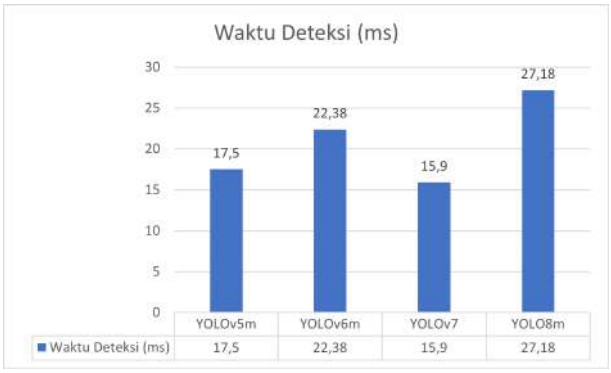


Fig. 10. Waktu Deteksi Model

Gambar 10. YOLOv8m membutuhkan waktu paling lama untuk mendeteksi objek pada citra yaitu selama 27,18 ms. Waktu deteksi paling cepat dilakukan oleh model YOLOv7, kemudian diikuti model YOLOv5m yang tidak berbeda signifikan dengan YOLOv7. Sementara model YOLOv6m memiliki waktu deteksi lebih lama dibanding YOLOv7 dan YOLOv5m.

### *B. Evaluasi Model*

Evaluasi model dilakukan dengan menganalisis kelebihan dan kekurangan model dari berbagai sudut pandang. Analisis kelebihan dan kekurangan model dipertimbangkan terhadap hasil pengujian model yaitu akurasi deteksi, kompleksitas model, dan waktu komputasi.


Fig. 11. Sampel Hasil Deteksi Citra pada Data Pengujian

Berdasarkan hasil pengujian model, YOLOv5m, YOLOv6m, dan YOLOv8m memiliki nilai mAP@0.5 yang sama. Sementara nilai mAP@0.5:0.95 dari ketiga model terdapat perbedaan, dengan urutan dari nilai tertinggi diperoleh model YOLOv8m, YOLOv6m, kemudian YOLOv5m. Perbedaan nilai mAP@0.5:0.95 ketiga model tidak berbeda secara signifikan. Perbedaan signifikan terlihat pada nilai mAP model YOLOv7 dibanding ketiga model lainnya, baik nilai mAP@0.5 maupun mAP@0.5:0.95. Gambar 11 menunjukkan

sampel hasil deteksi pada data pengujian memiliki nilai *confidence* yang tinggi, kecuali hasil deteksi model YOLOv7. Hal ini membuktikan bahwa model YOLOv7 memang memiliki akurasi terendah.


Fig. 12. Sampel Hasil Deteksi Citra *Cluttered-background*

Deteksi objek pada data pengujian tergolong mudah karena variasi citranya serupa dengan data pelatihan dan data validasi. Namun pada penerapan di dunia nyata, citra daun yang dideteksi dapat lebih bervariasi. Oleh karena itu, pada penelitian ini dilakukan uji coba model untuk mendeteksi citra dengan variasi yang lebih sulit dideteksi. Model dicoba untuk mendeteksi citra *cluttered-background* dan citra *mosaic*.

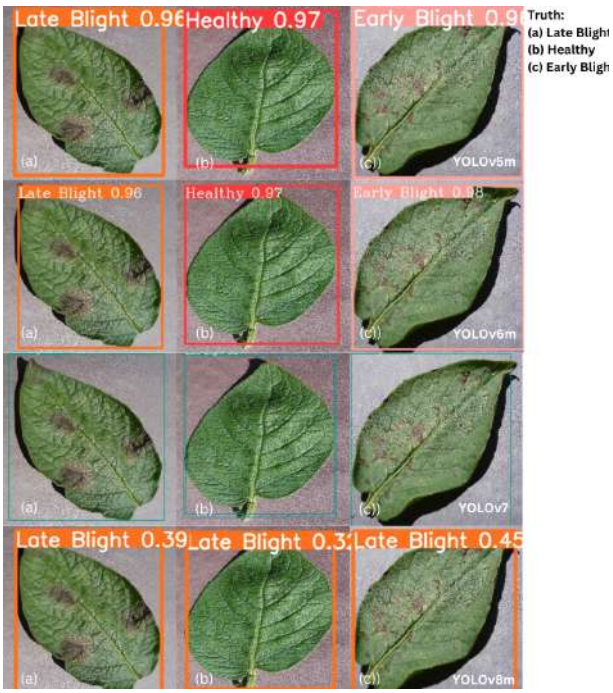


Fig. 13. Sampel Hasil Deteksi Citra *Mosaic*

Hasil deteksi keempat model pada citra *cluttered-background* ditunjukkan pada Gambar 12. Berdasarkan percobaan pada citra *cluttered-background* dapat diketahui bahwa hasil deteksi YOLOv7 dan YOLOv8m kurang akurat. Hal ini ditunjukkan dengan nilai *confidence box* yang rendah. Sementara hasil deteksi terbaik dilakukan oleh model YOLOv6m, diikuti model YOLOv5m dengan perbedaan yang tidak signifikan.

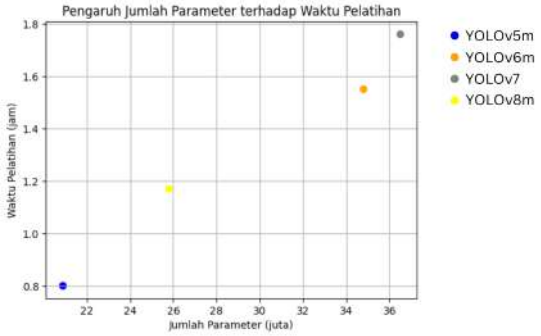


Fig. 14. Pengaruh Jumlah Parameter terhadap Waktu Pelatihan

Kemudian hasil deteksi keempat model pada citra *mosaic* ditunjukkan pada Gambar 13. Pada percobaan citra *mosaic* dapat diketahui bahwa model YOLOv7 dan YOLOv8m tidak berhasil mendeteksi penyakit tanaman dengan tepat. Sementara deteksi penyakit tanaman pada citra *mosaic* berhasil dideteksi dengan benar oleh model YOLOv5m dan YOLOv6m. Maka meskipun YOLOv8m memiliki nilai mAP@0.5:0.95 tertinggi, YOLOv8m memiliki kelemahan dalam mendeteksi objek pada variasi citra *cluttered-background* dan *mosaic*. Dengan demikian, model yang unggul dalam akurasi deteksi adalah YOLOv5m dan YOLOv6m.

Akurasi model dalam mendeteksi objek dapat ditingkatkan dengan penambahan epoch dalam proses pelatihan. Namun hal tersebut kurang efisien apabila diterapkan pada model yang kompleks. Hal ini dipengaruhi oleh jumlah parameter yang dapat meningkatkan waktu pelatihan seperti yang dapat dilihat pada Gambar 14.

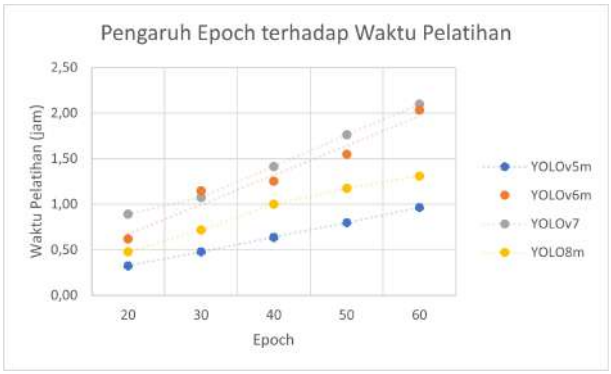


Fig. 15. Perbandingan Jumlah Epoch dengan Waktu Pelatihan Model

Dengan demikian, penambahan epoch untuk meningkatkan akurasi deteksi pada pelatihan model yang kompleks kurang efisien karena akan meningkatkan waktu pelatihan secara signifikan seperti yang dapat dilihat pada Gambar 15.

Selain waktu pelatihan, waktu deteksi adalah hal yang perlu dipertimbangkan dalam evaluasi model. Model yang lama dalam mendeteksi objek, kurang tepat jika diimplementasikan

pada sistem berbasis *real-time*. Penelitian ini bertujuan menemukan model yang berbasis *real-time* karena deteksi penyakit tanaman memerlukan waktu yang cepat dalam penanganannya. Berdasarkan tinjauan pustaka, dinyatakan bahwa akurasi yang tinggi cenderung berhubungan dengan lamanya waktu deteksi. Perbandingan nilai mAP@0.5:0.95 terhadap waktu deteksi model dapat dilihat pada Gambar 16 yang menunjukkan bahwa semakin tinggi nilai mAP@0.5:0.95 menyebabkan waktu yang diperlukan model untuk mendeteksi objek semakin lama.

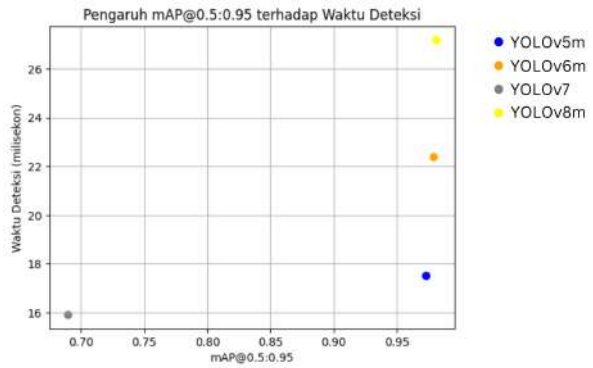


Fig. 16. Perbandingan mAP@0.5:0.95 terhadap Waktu Deteksi

Berdasarkan hasil pengujian dan evaluasi model, nilai mAP@0.5:0.95 tertinggi dicapai oleh YOLOv8m, tetapi YOLOv8m memiliki kelemahan dalam mendeteksi objek pada citra *cluttered-background* dan *mosaic*. Penambahan epoch untuk meningkatkan kemampuan deteksi YOLOv8m kurang efisien karena jumlah parameternya cukup banyak dan waktu pelatihannya cukup lama. Selain itu, YOLOv8m memiliki waktu deteksi paling lama dibanding model lainnya.

YOLOv7 memiliki akurasi deteksi yang sangat rendah, jumlah parameter yang sangat banyak, dan waktu pelatihan yang sangat lama. YOLOv7 memiliki waktu deteksi yang cepat tetapi hasil deteksinya memiliki akurasi yang sangat rendah dibanding model lainnya. Penambahan epoch untuk meningkatkan akurasi YOLOv7 tidak efisien karena jumlah parameter dan waktu pelatihannya tidak sebanding dengan kerendahan akurasinya.

Hasil terbaik dalam mendeteksi objek pada data pengujian maupun pada citra *cluttered-background* dan *mosaic* dicapai oleh model YOLOv6m. Namun YOLOv6m memiliki kompleksitas model dan waktu komputasi yang tinggi sehingga kurang efisien.

Di antara model YOLOv5m, YOLOv6m, dan YOLOv8m, model YOLOv5m memiliki nilai mAP@0.5:0.95 paling rendah tetapi tidak signifikan. YOLOv5m juga masih unggul dalam percobaan deteksi objek pada variasi citra *cluttered-background* dan *mosaic* sehingga akurasi deteksi masih dapat ditingkatkan dengan penambahan epoch. Hal ini dapat dilihat pada Gambar 15 dimana waktu pelatihan YOLOv5m dengan 60 epoch masih lebih cepat dibanding waktu pelatihan model lainnya dengan 50 epoch, maka akurasi deteksi YOLOv5m dapat ditingkatkan secara efisien dan efektif dengan penambahan epoch. Dengan demikian, arsitektur dan prinsip kerja model YOLOv5m memiliki kemampuan optimal dalam mendeteksi penyakit tanaman menggunakan dataset daun kentang.

## V. Kesimpulan dan Saran

### A. Kesimpulan

Penelitian tugas akhir ini menganalisis kinerja 4 model deteksi objek berbasis *deep learning* yaitu model YOLOv5m, YOLOv6m, YOLOv7, dan YOLOv8m. Analisis kinerja dilakukan agar diketahui kinerja model dan ditemukan model terbaik untuk mendeteksi penyakit tanaman kentang. Analisis kinerja menghasilkan poin-poin berikut:

1) YOLOv5m dapat mencapai mAP yang lebih tinggi dibanding YOLOv7 serta hampir sama dibanding YOLOv6m dan YOLOv8m. YOLOv5m memiliki kompleksitas model paling rendah dan waktu komputasi paling cepat dibanding model lainnya.
2) YOLOv6m memiliki akurasi deteksi sedikit lebih tinggi dibanding YOLOv5m, tetapi memiliki kompleksitas model yang tinggi dan waktu komputasi yang lama sehingga kurang efisien.
3) YOLOv7 memiliki hasil paling buruk karena dengan kompleksitas model yang tinggi dan waktu komputasi yang lama, akurasi deteksi yang dicapai sangat rendah.
4) YOLOv8m mencapai mAP tertinggi, tetapi tidak jauh berbeda dibanding YOLOv6m dan YOLOv5m. YOLOv8m memiliki kelemahan dibanding YOLOv5m dan YOLOv6m ketika mendeteksi citra *cluttered-background* dan *mosaic*. Kompleksitas dan waktu komputasi YOLOv8m lebih tinggi dibanding YOLOv5m.

Berdasarkan hasil analisis kinerja, YOLOv5m memiliki banyak keunggulan diantaranya memiliki kompleksitas model dan waktu komputasi paling rendah tetapi dapat mencapai akurasi deteksi yang tinggi. Dengan demikian, model YOLO terbaik yang dapat digunakan untuk mendeteksi penyakit tanaman kentang adalah model YOLOv5m.

### B. Saran

Penelitian lanjutan dapat dilakukan untuk meningkatkan manfaat *deep learning* di bidang pertanian, khususnya dalam mendeteksi penyakit tanaman. Dalam implementasinya, model deteksi objek dapat ditingkatkan kinerjanya. Peningkatan dapat dilakukan berdasarkan perspektif pertanian maupun teknologi. Dalam perspektif pertanian dapat dilakukan penambahan objek yang dideteksi, seperti jenis tanaman, penyakit, gulma, dan hama. Berdasarkan perspektif teknologi dapat dilakukan peningkatan hyperparameter, jumlah data, augmentasi, dan optimisasi dalam pelatihan model. Selain itu dapat diuji model deteksi objek lainnya, seperti model YOLO-NAS yang baru dirilis tanggal 3 Mei 2023.

## References

[1] BPS, "Produksi tanaman sayuran menurut provinsi dan jenis tanaman," 2022. [Online]. Available: https://www.bps.go.id
[2] ——, "Luas panen tanaman sayuran menurut provinsi dan jenis tanaman," 2022. [Online]. Available: https://www.bps.go.id

[3] hortiindonesia, "Peluang kentang industri terbuka, kenali permasalahannya," Desember 2021. [Online]. Available: https://hortiindonesia.com/berita/peluang-kentang-industri-terbuka-kenali-permasalahannya

[4] A. Rachman, "Ri borong kentang jerman as, impor naik gila-gilaan," February 2023. [Online]. Available: https://www.cnbcindonesia.com/news/20230216072813-4-414181/ri-borong-kentang-jerman-as-impor-naik-gila-gilaan

[5] S. I. Mohamed, "Potato leaf disease diagnosis and detection system based on convolution neural network," *International Journal of Recent Technology and Engineering (IJRTE)*, vol. 9, no. 4, pp. 254–259, 2020.

[6] A. Irawan, "Ciri ciri fisik daun tanaman yang terkena penyakit bercak kering (alternaria)," Agrokompleks Kita, 2019. [Online]. Available: https://agrokomplekskita.com/penyakit-bercak-kering-alternaria-pada-tanaman-kentang/ciri-ciri-fisik-daun-tanaman-yang-terkena-penyakit-bercak-kering-alternaria/

[7] "Difference between early blight and late blight of potato," BYJU'S, 2023. [Online]. Available: https://byjus.com/biology/difference-between-early-blight-and-late-blight-of-potato/

[8] L. Amatullah, I. Ein, and M. M. Santoni, "Identifikasi penyakit daun kentang berdasarkan fitur tekstur dan warna dengan menggunakan metode k-nearest neighbor," in *Seminar Nasional Mahasiswa Ilmu Komputer dan Aplikasinya (SENAMIKA)*, 2021.

[9] W. Xue, K. G. Haynes, and X. Qu, "Characterization of early blight resistance in potato cultivars," *Plant Disease*, vol. 103, 2019. [Online]. Available: https://doi.org/10.1094/PDIS-05-18-0794-RE

[10] "Perlu di waspadai, penyakit busuk daun yang menyerang tanaman kentang," PT. Hextar Fertilizer Indonesia. [Online]. Available: https://www.hextarfertilizerindonesia.com/perlu-di-waspadai-penyakit-busuk-daun-yang-menyerang-tanaman-kentang/

[11] R. J. Harahap, "Tim dosen usu dampingi petani tingkatkan produksi kentang," Universitas Sumatera Utara, 2022. [Online]. Available: https://www.usu.ac.id/id/pengabdian-masyarakat/tim-dosen-usu-dampingi-petani-tingkatkan-produksi-kentang

[12] Aaradhya, "11 dogs similar to wolf," The Arena Group, March 2023. [Online]. Available: https://pethelpful.com/dogs/11-dogs-similar-to-wolf

[13] G. M. Su, "Deep learning bible," May 2023. [Online]. Available: https://wikidocs.net/163640

[14] M. S. Khatami, "Deteksi kendaraan menggunakan algoritma you only look once (yolo) v3," Universitas Islam Indonesia, 2022.

[15] D. S. Trigueros, L. Meng, and M. Hartnett, "Face recognition: From traditional to deep learning methods," *arXiv:1811.00116v1 [cs.CV] 31 Oct 2018*, 2018. [Online]. Available: https://arxiv.org/abs/1811.00116

[16] S. Oni, "Understanding anchors(backbone of object detection) using yolo," Becoming Human: Artificial Intelligence Magazine, April 2020. [Online]. Available: https://becominghuman.ai/understanding-anchors-backbone-of-object-detection-using-yolo-54962f00fbbb

[17] N. Wulandari, "Perbandingan implementasi metode deep learning pada deteksi objek di bawah air," Universitas Gadjah Mada, 2022.

[18] K. E. Koech, "Object detection metrics with worked example," Towards Data Science, August 2020. [Online]. Available: https://towardsdatascience.com/should-i-look-at-precision-recall-or-specificity-sensitivity-3946158aace1

[19] F. Dang, D. Chen, Y. Lu, and Z. Li, "Yoloweeds: A novel benchmark of yolo object detectors for multi-class weed detection in cotton production systems," *Computers and Electronics in Agriculture*, vol. 205, 2023.

[20] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, S. J, M. A. Fadhel, M. Al-Amidie, and L. Farhan, "Review of deep learning: concepts, cnn architectures, challenges, applications, future directions," *Journal of Big Data*, vol. 8, no. 53, pp. 1–74, 2021.

[21] M. Ahmad, J. Joe, and D. Han, "Cortexnet: Convolutional neural network with visual cortex in human brain," in *2018 IEEE International Conference on Consumer Electronics - Asia (ICCE-Asia)*. IEEE, 2018.

[22] M. Y. A. Thoriq, K. E. Permana, and I. A. Siradjuddin, "Deteksi wajah manusia berbasis one stage detector menggunakan metode you only look once (yolo)," *JURNAL TEKNOINFO*, vol. 17, no. 1, pp. 66–73, 2023.

[23] T. O. Emmanuel, Kaggle, Google LLC, 2018. [Online]. Available: https://www.kaggle.com/datasets/emmarex/plantdisease

[24] A. Ataiemontazer, Kaggle, Google LLC, 2020. [Online]. Available: https://www.kaggle.com/datasets/abbasataiemontazer/potato-leaf

[25] R. Saeed, Kaggle, Google LLC, 2021. [Online]. Available: https://www.kaggle.com/datasets/rizwan123456789/potato-disease-leaf-datasetpld