Jurnal Rekursif, Volume 13 Nomor 2, November 2025, ISSN 2303-0755, e-ISSN 2777-0427
https://ejournal.unib.ac.id/index.php/rekursif/

DETEKSI HAMA *WHITEFLIES* (*ALEYRODIDAE*) PADA TANAMAN CUCURBITACEAE MENGGUNAKAN YOLOV11

Naufal Rizky Ananda[1], Ernawati [2], Endina Putri Purwandari [3]

[1,3] Program Studi Informatika, Fakultas Teknik, Universitas Bengkulu,
Jl. WR. Supratman, Kandang Limun, Bengkulu, 3871A
Telp. (627) 3621170, Faks (627) 3622105

[1]naufalrizky1710@gmail.com
[2]ernawati@unib.ac.id
[3]endinaputri@unib.ac.id

***Abstrak**:* Hama *Whitefly (Aleyrodidae)* merupakan ancaman signifikan bagi produktivitas tanaman *Cucurbitaceae* di Indonesia, menyebabkan kerugian panen dan penyebaran virus. Karena metode deteksi manual lambat dan tidak efisien, diperlukan solusi teknologi untuk identifikasi dini. Penelitian ini membangun dan mengevaluasi model deteksi objek menggunakan arsitektur YOLOv11. Metodologi yang digunakan meliputi empat tahap: persiapan 1.940 citra gambar dari repositori publik, *preprocessing* data melalui anotasi dan *augmentasi (blur, brightness, noise)*, perancangan dan pelatihan model, serta evaluasi kinerja. Model yang telah dilatih kemudian diimplementasikan ke dalam aplikasi *web* untuk deteksi *real-time*. Hasil evaluasi menunjukkan performa model yang sangat baik, dengan mAP@50 sebesar 85,6% dan mAP@50-95 sebesar 81,2%. Selain itu, model ini mencapai nilai *precision* 83,1%, *recall* 89,0%, dan *F1-Score* 86,0%, ini membuktikan model mampu mendeteksi hama secara akurat dan konsisten. Penelitian ini berhasil menghasilkan sebuah sistem deteksi dini yang efektif dan mudah diakses, memberikan kontribusi praktis bagi pengembangan pertanian presisi dan mendukung upaya ketahanan pangan di Indonesia melalui penerapan teknologi deep learning.
Kata Kunci: YOLOv11, Whitefly, Hama, Cucurbitaceae.

Abstract: Whitefly (Aleyrodidae) pests pose a significant threat to the productivity of Cucurbitaceae crops in Indonesia, leading to substantial harvest losses and the transmission of plant viruses. Because traditional manual detection methods are often slow and inefficient, there is a clear need for a technological solution for early identification. This study details the development and evaluation of an object detection model using the YOLOv11 architecture. The methodology involved four primary stages: preparing a dataset of 1,940 images from public repositories, preprocessing the data through annotation and augmentation (including blur, brightness, and noise), training the model, and conducting a thorough performance evaluation. The resulting model was deployed into a web-based application for real-time detection. The evaluation demonstrated the model's excellent performance, achieving a mean Average Precision at a 0.5 IoU threshold (mAP@50) of 85.6% and an mAP@50-95 of 81.2%. Furthermore, it achieved a precision of 83.1%, a recall of 89.0%, and an F1-Score of 86.0%, proving its capacity to consistently and accurately detect these small-sized pests. This research successfully delivers an effective and accessible early detection system, making a practical contribution to precision agriculture and supporting food security in Indonesia through the application of deep learning.
Keywords: YOLOv11,Whitefly,Pest, Cucurbitaceae.

I. PENDAHULUAN

Pemantauan hama secara efektif sangat penting untuk meningkatkan hasil pertanian, terutama di negara dengan biodiversitas tinggi seperti Indonesia. Dengan meningkatnya kebutuhan pangan global, jumlah kerugian yang diakibatkan oleh hama semakin signifikan. Data dari FAO menunjukkan bahwa hama dapat menyebabkan kerugian tanaman hingga 30% di negara

berkembang, dan *whitefly* merupakan salah satu hama utama yang berkontribusi terhadap penurunan hasil pertanian [1], [2].

Hama *whitefly (Aleyrodidae)* adalah salah satu spesies yang paling merugikan dalam pertanian, terutama pada tanaman sayuran dan buah-buahan [3], [4]. Hama ini tidak hanya merusak tanaman secara langsung, tetapi juga bertindak sebagai vektor berbagai virus tanaman yang dapat memperparah kerugian. Penelitian menunjukkan bahwa whitefly dapat menurunkan produksi tanaman secara signifikan di daerah yang terinfeksi di seluruh dunia [5]. Hama *Whitefly*, merupakan masalah serius bagi pertanian di Indonesia. *Whitefly* dapat menyebarkan virus tanaman, mengurangi kualitas serta kuantitas hasil panen, dan menyebabkan kerugian ekonomi yang signifikan [6]. Keterlambatan dalam identifikasi hama ini, terutama ketika hama sudah menyebabkan kerusakan yang signifikan, sering kali disebabkan oleh metode deteksi tradisional yang kurang efisien dan memerlukan banyak waktu, seperti inspeksi manual yang berbasis visual). Oleh karena itu, pemantauan hama secara tepat waktu sangat penting untuk mempertahankan kesehatan tanaman serta hasil panen yang optimal.

Tanaman *family Cucurbitaceae*, seperti mentimun, melon, semangka, dan labu, memegang peran strategis dalam ketahanan pangan dan ekonomi Indonesia.Tanaman *family* ini menjadi komoditas unggulan karena adaptasinya yang luas di berbagai kondisi lahan, termasuk daerah marginal seperti gambut, dengan nilai ekonomi mencapai rasio R/C > 1,5 pada budidaya melon dengan pendapatan panen hingga Rp.900.000 per hektar [7]. Kandungan nutrisinya yang kaya, seperti vitamin A, C, serat, dan senyawa bioaktif (*flavonoid* dan *fenolat*), menjadikannya sumber pangan fungsional untuk pencegahan diabetes, hipertensi, dan imunomodulasi [8]. Selain itu, studi Nursamsu tentang Biodiversitas menegaskan bahwa *Cucurbitaceae* menjadi tulang punggung ketahanan pangan masyarakat pesisir, dengan 10 spesiesnya berkontribusi pada 30% kebutuhan harian serat dan protein [9]. Namun, produktivitasnya terancam oleh serangan hama kutu kebul (*Aleyrodidae*) yang berpotensi menurunkan hasil panen hingga 80%.

Hama kutu kebul (*Aleyrodidae*) merupakan ancaman serius karena perannya sebagai vektor utama penyakit virus seperti *Begomovirus* dan *Cucurbit aphid-borne yellows virus (CABYV).* Meskipun metode pengendalian konvensional seperti insektisida dan *varietas* tahan telah diimplementasikan, resistensi hama dan dampak lingkungan menjadi kendala utama. Oleh karena itu, integrasi pendekatan bioteknologi dan penguatan sistem deteksi dini melalui *deep learning* dalam pemantauan *vector* population menjadi urgensi untuk memastikan keberlanjutan budidaya *Cucurbitaceae* di Indonesia [10].

Terdapat sejumlah penelitian sebelumnya yang telah dieksplorasi dalam konteks deteksi hama. menggambarkan penggunaan teknik berbasis GAN untuk augmentasi deteksi hama, tetapi penelitian ini masih terbatas pada metodologi tertentu dan memerlukan pengembangan lebih lanjut untuk aplikasi lapangan. Penelitian lain oleh [4] menunjukkan potensi penggunaan pembelajaran transfer visual untuk mendeteksi objek pertanian, tetapi aplikasinya terhadap spesies hama individu seperti whitefly belum dieksplorasi secara mendalam.

Meskipun terdapat berbagai metode manual dan semiautomatis untuk mendeteksi hama, keterbatasannya menjadi kendala yang signifikan. Metode manual sering kali mengandalkan pengalaman petani yang terbatas dan dapat menyebabkan keterlambatan dalam mendeteksi

infestasi [11]. Di sisi lain, penggunaan teknologi seperti citra satelit dan penginderaan jauh juga telah menunjukkan potensi, namun efektivitasnya sering kali terhambat oleh kesulitan dalam mendeteksi objek kecil dalam konteks pertanian yang kompleks [12]. Dalam hal ini, penerapan algoritma deteksi objek berbasis deep learning, seperti YOLOv11, menawarkan solusi inovatif yang dapat diandalkan untuk mendeteksi *whitefly* secara akurat dan cepat.

Penelitian ini bertujuan untuk mengisi kesenjangan tersebut dengan mengeksplorasi penerapan YOLOv11 dalam mendeteksi hama *whitefly*. Dengan mengembangkan model yang dapat secara akurat mengidentifikasi *whitefly*, yang diharapkan dapat memberi berkontribusi pada inovasi di bidang pertanian pintar. Secara teoritis, penelitian ini berkontribusi pada pemahaman yang lebih dalam tentang penerapan algoritma *deep learning* dalam konteks pertanian, terutama di negara tropis. Secara praktis, hasil dari penelitian ini diharapkan dapat diintegrasikan ke dalam sistem pemantauan hama yang ada, memberikan alat baru bagi petani untuk membuat keputusan berbasis data [13]. Berdasarkan permasalahan diatas penelitian ini bertujuan untuk mengembangkan aplikasi identifikasi spesies *Whitefly (Aleyrodidae)* menggunakan metode YOLO (*You Only Look Once*). Data penelitian diperoleh dari citra gambar spesies *Whitefly (Aleyrodidae)* yang dikumpulkan dari repositori publik dari website penyedia dataset seperti *Roboflow* dan *Kaggle*, yang menawarkan berbagai koleksi dataset gratis yang dapat diakses dengan mudah. Diharapkan bahwa hasil penelitian ini akan memberikan kontribusi positif bagi peneliti di bidang serangga, mahasiswa pertanian, dan ilmuwan dalam mempermudah identifikasi spesies *Whitefly (Aleyrodidae)*, serta memungkinkan penemuan spesies baru yang sebelumnya belum teridentifikasi.

II. TINJAUAN PUSTAKA

A. *Whitefly (Aleyrodidae)*

Kutu putih, yang dalam terminologi ilmiah dikenal sebagai *Aleyrodidae*, merupakan kelompok serangga kecil yang memiliki dampak signifikan terhadap praktik pertanian dan ekosistem global. Mereka berperan utama sebagai vektor virus dan di antara spesies yang ada, Bemisia tabaci menjadi sorotan utama karena keterlibatannya dalam penyebaran virus, terutama *Tomato Yellow Leaf Curl Virus* (TYLCV) [14]. Penelitian menunjukkan bahwa B. tabaci adalah vektor utama dalam epidemi virus ini yang dapat mengakibatkan kerugian besar bagi hasil pertanian, khususnya pada tanaman tomat yang esensial untuk ketahanan pangan di beberapa negara, terutama di Afrika [15].Sejak pertama kali diidentifikasi pada akhir abad ke-19, kutu putih telah menjadi masalah bagi pertanian modern, khususnya pada tanaman komersial seperti tomat dan singkong [16]. Dengan kemampuannya sebagai penghisap getah, kutu putih dapat memengaruhi pertumbuhan dan kesehatan tanaman serta berkontribusi pada penyebaran penyakit [14]. Mereka memiliki siklus hidup yang singkat dengan kemampuan reproduksi yang cepat, yang mendukung pertumbuhan populasi mereka secara eksponensial, sehingga dapat menyebabkan kerusakan besar pada hasil pertanian [17].Para peneliti juga menekankan pentingnya memahami interaksi antara kutu putih, tanaman, dan virus yang mereka bawa. Beberapa studi menunjukkan bahwa perubahan genetik dalam populasi *B. tabaci* serta infeksi virus pada tanaman dapat berakibat pada peningkatan populasi kutu putih yang signifikan, memperparah masalah penyebaran penyakit di ladang [16].

Penelitian lain menunjukkan bahwa interaksi dengan hama lain, seperti kutu daun, dapat memodifikasi perilaku kutu putih dan meningkatkan efisiensi transmisi virus [17].Dampak yang ditimbulkan oleh kutu putih tidak hanya terbatas pada kerusakan langsung yang mereka lakukan terhadap tanaman, tetapi juga berhubungan dengan penyebaran virus yang dapat menghancurkan hasil panen secara keseluruhan [17], [18]. Infeksi virus dapat mengubah preferensi makan kutu putih, yang berpengaruh pada cara dan waktu penyebaran penyakit [17]. Penggunaan pestisida untuk mengendalikan kutu putih sering membawa dampak negatif pada ekosistem, termasuk penurunan populasi predator alami dan perkembangan resistensi pada kutu putih itu sendiri [19]. Oleh karena itu, pendekatan pengendalian terintegrasi (IPM) mulai diusulkan, dengan menekankan penggunaan insektisida yang lebih selektif serta praktik pertanian yang mempertimbangkan kesehatan ekosistem secara keseluruhan [19]. Kesimpulannya, kutu putih (Aleyrodidae) adalah kelompok serangga yang memiliki dampak signifikan terhadap kesehatan tanaman dan produktivitas pertanian. Dengan mengupas sejarah dan interaksi mereka, serta menerapkan pendekatan terpadu dalam pengelolaannya, kita dapat mengurangi dampak negatif yang ditimbulkan oleh kutu putih [15], [16].

B. *Tanaman Cucurbitaceae*

Cucurbitaceae adalah keluarga tanaman yang dikenal sebagai keluarga labu, yang terdiri dari sekitar 800 spesies dan lebih dari 130 genus. Tanaman dalam keluarga ini terkenal akan buahnya yang dapat dimakan, yang mencakup mentimun, labu, semangka, dan melon. Tanaman-tanaman ini tidak hanya penting sebagai sumber pangan, tetapi juga memiliki nilai ekonomi yang tinggi dan peran penting dalam budaya berbagai masyarakat di seluruh dunia. Secara morfologi, tanaman *Cucurbitaceae* umumnya memiliki batang berusuk, daun berbentuk jantung atau lobus, serta bunga yang umumnya bersifat uniseks, yaitu terdapat bunga jantan dan betina terpisah pada satu individu atau pada individu yang berbeda. Klasifikasi Cucurbitaceae sangat kompleks, dengan pembagian yang meliputi sekitar 95-97 genus dan dibagi menjadi 15 suku (*tribe*). Penelitian terkini menggunakan analisis *filogenetik* dan data *molekuler* untuk memperjelas dan memvalidasi hubungan genetik antar genus dalam keluarga ini [20].Genus yang paling terkenal dalam *Cucurbitaceae* termasuk *Cucumis* (mentimun), *Cucurbita* (labu), dan *Citrullus* (semangka), yang menunjukkan keragaman morfologi dan genetik yang tinggi.

C. *Pengolahan Citra Digital*

Visi komputer adalah cabang ilmu yang memiliki kemiripan dengan pemrosesan citra, mencakup aspek seperti akuisisi, pengolahan, dan klasifikasi citra. Tujuan utama dari visi komputer adalah mengizinkan komputer untuk memahami dan memproses gambar atau video secara otomatis, mirip dengan bagaimana manusia melakukan pengolahan visual. Oleh karena itu, pemrosesan citra berfungsi sebagai langkah awal yang penting dalam visi komputer, diikuti dengan pengenalan pola yang lebih kompleks [21].

Aplikasi visi komputer kini semakin beragam, mulai dari industri hingga sektor kesehatan. Misalnya, sistem visi komputer dapat digunakan untuk memberikan analisis yang lebih akurat dalam bidang medis dengan memanfaatkan dataset besar dan algoritma pengenalan pola yang efisien. Penelitian menunjukkan bahwa ukuran

dataset yang besar sangat penting untuk meningkatkan akurasi dalam pengenalan objek dan klasifikasi citra, mendukung perkembangan teknologi modern dalam bidang ini [22] .Selain itu, perkembangan terkini di bidang visi komputer juga banyak berfokus pada penggunaan algoritma pembelajaran mendalam untuk meningkatkan kemampuan deteksi objek dan pengklasifikasian gambar secara lebih efektif [22], [23].

D. YOLO *(You Only Look Once)*

Salah satu pendekatan yang berkembang dalam jaringan saraf konvolusional atau CNN salah satunya adalah metode You Only Look Once (YOLO). Metode ini diperkenalkan oleh Joseph Chet Redmon dan timnya dalam publikasi tahun 2016 berjudul "*You Only Look Once: Unified, Real-Time Object Detection*"[24]. Sejak pengenalan pertamanya, YOLO telah menarik perhatian luas di kalangan peneliti dalam bidang citra komputer, terutama karena kemampuannya dalam melakukan deteksi objek secara real time. Berbeda dengan pendekatan sebelumnya, seperti *Region Convolutional Network* (R-CNN) yang menggunakan *Region Proposal Networks* (RPNs) untuk menghasilkan kotak pembatas sebelum melakukan klasifikasi, YOLO memprediksi posisi dan kategori objek secara bersamaan, menjadikannya jauh lebih efisien dalam pemrosesan gambar [25].

E. YOLOv11

YOLOv11 adalah iterasi terbaru dari seri YOLO yang dikembangkan oleh Ultralytics dengan berbagai peningkatan arsitektur yang signifikan., arsitektur YOLOv11 terdiri dari tiga komponen utama yang telah dioptimalkan yang terdiri dari SPPF, C2PSA, dan C3K2.

1. SPFF (*Spatial Pyramid Feature Fusion*)

Modul ini meningkatkan efisiensi dalam akuisisi fitur dengan memanfaatkan skala berbeda dari objek yang terdeteksi, sehingga mampu mengoptimalkan pengolahan citra pada berbagai ukuran objek. Dengan menggunakan SPFF, YOLOv11 dapat menyatukan informasi dari berbagai resolusi, yang memungkinkan deteksi yang lebih akurat meskipun dalam kondisi lingkungan yang kompleks [26].

2. C2PSA *(Channel and Spatial Attention Module)*

Modul ini dirancang untuk meningkatkan pemfokusan model pada fitur penting di dalam citra dengan mempertimbangkan relasi antara saluran dan spasial. C2PSA memungkinkan model untuk meningkatkan kinerja deteksi objek dengan memperkuat channel yang relevan dan mengurangi *noise* dari *channel* yang kurang penting, sehingga faedahnya sangat nampak dalam peningkatan akurasi deteksi [26].

3. C3K2 *(C3 Kernel Layer)*

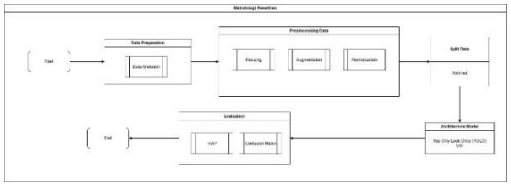
Inovasi ini meningkatkan kecepatan dan efisiensi pemrosesan dengan menggabungkan convolutional layers yang lebih ringan dan lebih cepat. Struktur C3K2 dirancang untuk mempertahankan integritas informasi selama proses deteksi, membantu YOLOv11 mencapai performa yang lebih baik dalam hal kecepatan dan akurasi secara bersamaan [26].

III. METODOLOGI PENELITIAN

A. Metode Pengembangan Sistem

Pada bagian ini, akan dijelaskan rancangan penelitian yang digunakan untuk mengembangkan *object detection* menggunakan YOLOv11 dalam deteksi hama whiteflies (*aleyrodidae*) pada tanaman *cucurbitaceae* . Penelitian ini dilakukan menggunakan metode yang komprehensif serta memiliki tahapan-tahapan penelitian yaitu data

preparation, data *preprocessing*, *arsitecture*, dan *evaluation*.



Gambar 1. Metodologi Penelitian

1. *Data Preparation*

Data *Preparation* adalah langkah awal di mana data yang diperlukan untuk analisis dikumpulkan, dibersihkan, dan disusun. Proses ini melibatkan pengumpulan data dari berbagai sumber, termasuk database, *file*, atau melalui pengamatan langsung. Data juga perlu dibersihkan untuk menghilangkan data yang tidak relevan atau salah. Berbagai teknik seperti normalisasi dan penghapusan duplikat dapat digunakan dalam tahap ini. Persiapan data yang baik sangat penting untuk memastikan akurasi dan efisiensi dalam analisis yang akan dilakukan. Mengidentifikasi dan menangani data hilang adalah aspek yang sangat penting yang akan memengaruhi kualitas dari hasil akhir.

Tahapan ini dilakukan di website *roboflow.com*. Adapun langkah yang dilakukan dalam persiapan data sebelum data masuk tahap *preprocessing*. Pembuatan anotasi adalah proses pelatihan model deteksi objek seperti YOLO. Anotasi mencakup penandaan objek dalam citra dengan *bounding box* serta label yang sesuai, memberikan informasi yang diperlukan bagi model untuk belajar mengenali objek tersebut.

2. *Data Preprocessing*

Data *preprocessing* merupakan langkah krusial dalam pengembangan model deteksi objek menggunakan algoritma YOLO (*You Only Look Once*) dan teknik deep learning lainnya. Tujuan utama dari preprocessing data adalah untuk mempersiapkan data agar dapat digunakan dengan efektif oleh model, sehingga meningkatkan akurasi dan efisiensi dalam proses deteksi. Dalam konteks YOLO, langkah-langkah preprocessing meliputi augmentasi data, normalisasi, dan pembuatan anotasi yang dijelaskan sebagai berikut:

a. *Resizing Data*

Resizing dalam data preprocessing untuk arsitektur YOLO adalah proses penyesuaian dimensi (lebar dan tinggi) semua gambar input ke ukuran yang seragam dan telah ditentukan sebelumnya menjadi 640x640 piksel. Hal ini penting karena arsitektur *object detection* seperti CNN, termasuk YOLO, memerlukan input dengan dimensi yang konsisten untuk inferensi yang efisien dan pelatihan yang stabil, memungkinkan batch *processing* dan memastikan semua gambar diproses dengan cara yang sama oleh arsitektur model.

b. Augmentasi Data

Augmentasi merupakan teknik yang digunakan untuk memperbesar ukuran dataset dengan menciptakan variasi dari gambar asli tanpa harus mengambil gambar baru dengan menggunakan beberapa penambahan atribut ataupun konfigurasi perubahan *seperti blur, noise, brightness* dan sebagainya. Augmentasi data dapat meningkatkan kemampuan model untuk mengenali objek dalam kondisi yang bervariasi dan memperkaya dataset, terutama saat menghadapi masalah ketidakseimbangan kelas dalam data. Tanpa augmentasi yang tepat, model dapat mengalami *overfitting*, mengurangi performa, terutama pada objek kecil atau samar.

c. Normalisasi

Normalisasi adalah langkah penting lain yang bertujuan untuk mengatasi perbedaan skala antara fitur dalam dataset, sehingga model dapat lebih

efisien dalam mempelajari data. Dalam konteks YOLO, normalisasi dapat dilakukan untuk memastikan bahwa semua fitur memiliki skala yang seragam, yang membantu meningkatkan akurasi dan stabilitas model. Normalisasi atribut dapat meningkatkan performa model dengan menghindari bias akibat perbedaan skala fitur dalam data yang akan di pakai.

3. *Architecture*

Tahap arsitektur dalam pengembangan algoritma YOLO (You Only Look Once) dan deep learning berfokus pada desain dan implementasi struktur jaringan saraf yang efisien untuk deteksi objek. Arsitektur ini terdiri dari beberapa komponen penting yang berkolaborasi untuk menghasilkan model deteksi yang cepat dan akurat. Dalam konteks ini, kami akan membahas tiga elemen utama arsitektur dalam YOLO: backbone, neck, dan head:

Gambar 2 Arsitektur Model YOLOv11

a. *Backbone*

Backbone adalah komponen awal dari arsitektur YOLO yang bertugas mengekstraksi fitur dari gambar input. Biasanya, *backbone* terdiri dari beberapa lapisan konvolusi yang mengompresi dimensi gambar sambil mempertahankan informasi penting tentang objek. Dalam variasi terbaru dari YOLO, seperti YOLOv4 dan YOLOv5, arsitektur backbone yang digunakan sering kali adalah *CSPDarknet53* atau lainnya, yang dirancang untuk meningkatkan efisiensi dan keakuratan deteksi. Arsitektur ini memanfaatkan konsep-konsep seperti residual *connections* yang membantu dalam aliran informasi selama proses pelatihan dan mengurangi masalah vanishing gradients.Backbone yang efektif dapat meningkatkan akurasi deteksi secara signifikan sambil mempertahankan kecepatan deteksi *real-time.*

b. *Neck*

Setelah fitur diekstraksi, komponen berikutnya adalah *neck*, yang berfungsi untuk menggabungkan fitur-fitur dari berbagai level untuk memperbaiki konteks dalam deteksi objek. Neck mengimplementasikan teknik seperti *Feature Pyramid Networks (FPN)* atau *PANet (Path Aggregation Network)*, yang memungkinkan model untuk lebih baik dalam mendeteksi objek pada berbagai skala. Menggabungkan fitur dari layer yang berbeda memperkuat kemampuan model dalam menangkap struktur dan konteks objek yang kompleks. *Neck* yang tepat akan memperkaya informasi yang dikumpulkan dari backbone serta memfasilitasi transfer informasi yang lebih baik antara *layer*, mendukung deteksi yang lebih akurat.

c. *Head*

Bagian akhir dari arsitektur adalah head, yang bertugas untuk menghasilkan prediksi akhir berdasarkan fitur yang telah dikombinasikan dari neck. Head mengeluarkan *bounding box* dan probabilitas kelas untuk masing-masing objek yang terdeteksi dalam gambar. Proses ini dilakukan menggunakan teknik regresi yang memungkinkan YOLO untuk memprediksi beberapa bounding box dalam satu langkah. Proporsi dan ukuran dari bounding boxes, yang dihasilkan dari head, sangat penting dalam menentukan kualitas deteksi objek—dengan setiap prediksi memberikan skor kepercayaan yang berkaitan dengan keberadaan objek dalam prediksi tersebut.

4. *Evaluation*

Pada tahapan ini, penulis akan melakukan

evaluasi terhadap model yang telah diperoleh dengan menerapkan metric mAP *(Mean Average Precision*) dan *confusion matrix*. Metrik ini menghitung rata-rata dari nilai *Average Precision* (AP) untuk setiap kelas atau *query*, kemudian dirata-ratakan kembali secara keseluruhan. AP sendiri merupakan ukuran area di bawah kurva *Precision-Recall* (*area under the curve*), yang menggabungkan aspek presisi (ketepatan) dan recall (cakupan) dalam satu nilai metrik.

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \qquad (1)$$

- N adalah jumlah kelas,
- APi adalah nilai average precision untuk kelas ke-i

IV. *HASIL DAN PEMBAHASAN*

A. Analisis Data

Penelitian ini membutuhkan data citra dalam jumlah besar dan berkualitas untuk membangun model deteksi whiteflies yang optimal menggunakan YOLOv11. Data yang digunakan dalam penelitian ini dikumpulkan secara khusus dari tiga sumber utama, yaitu Roboflow, Mendeley dan Kaggle, yang menyediakan dataset publik dan open source yang relevan untuk kebutuhan deteksi hama whiteflies. Berikut adalah tahapan dan rincian penting dalam proses pengumpulan dan pengolahan data:

1. Pengambilan Data dari *Roboflow*, *Kaggle* dan *Mendeley*

Data citra whiteflies diunduh dari repositori *Roboflow*, *Mendeley* dan *Kaggle*, yang dikenal menyediakan dataset citra yang lumayan lengkap.

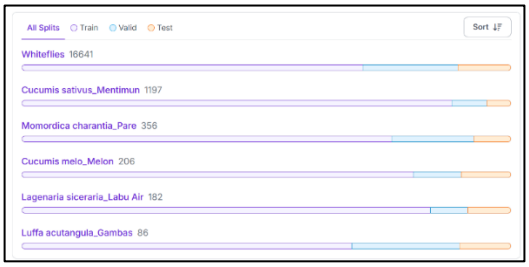
2. Jumlah Data dan Label

Dari proses pengumpulan, diperoleh sebanyak 1.940 data citra *whiteflies* dengan total 18.668 label objek *whiteflies*. Setiap gambar telah dianotasi secara detail dengan *bounding box* pada setiap individu *whiteflies* yang terdeteksi, sehingga satu gambar dapat memiliki lebih dari satu label.

Gambar 3. Jumlah data dan Label

3. Ukuran Dataset

Total ukuran dataset yang digunakan adalah sekitar 1,6 GB. Ukuran ini sudah termasuk seluruh file gambar dan file label yang diperlukan untuk proses pelatihan model YOLOv11. Adapun pembagian split data menggunakan metode hold out dengan persentasi 84%, 10%, 6% untuk setiap batch data.



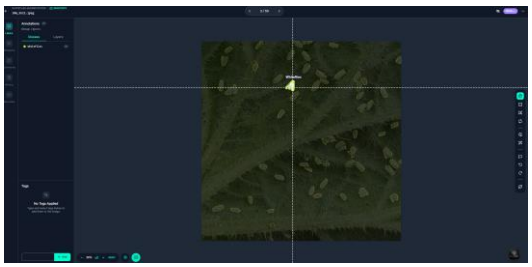
Gambar 4. *Split data* setiap *batch class*

Adapun rincian hasil split data dari tiap class dan gambar yang telah di anotasi

Tabel 1. Jumlah Gambar asli dan Objek Per Kelas

No	Nama Data	Jumlah Gambar Asli	Jumlah Objek
1	Terjangkit *Whiteflies*	1.124	16.641
2	Lagenaria siceraria (Labu Air)	182	182
3	Cucumis sativus (Mentimun)	160	1.197
4	Cucumis melo (Melon)	190	206
5	Momordica charantia (Pare)	252	358
6	Luffa acutangular (Gambas)	79	86
Total		1.940 + (47 null)	18.670

B. Data Preparation



Gambar 5 *Labelling Whiteflies* (*aleyrodidae*) di Web Roboflow pada tanaman *Cucurbitaceae*

Proses *labelling*/anotasi data citra *whiteflies* dilakukan secara terintegrasi di platform Roboflow. Proses dimulai dengan pembuatan akun Roboflow yang terhubung dengan akun Google. Setelah berhasil masuk ke situs Roboflow, *workspace* dapat diakses secara publik, mengingat *workspace* privat memerlukan biaya tambahan. Setelah *workspace* tersedia, data citra diunggah ke platform sesuai dengan fitur yang disediakan. Selanjutnya, barulah melakukan anotasi pada data yang telah berhasil diunggah. Proses anotasi dilakukan dengan menandai setiap objek yang ditemukan menggunakan *bounding box* secara proporsional. Ketelitian dalam memberikan bounding box saat anotasi sangat berpengaruh terhadap kualitas hasil yang diperoleh nantinya. Adapun tahapannya:

1. Pengunggahan Data ke Roboflow

   Seluruh citra hasil pengumpulan dari Roboflow ,Kaggle dan mendeley diunggah ke workspace Roboflow.

2. Anotasi Manual

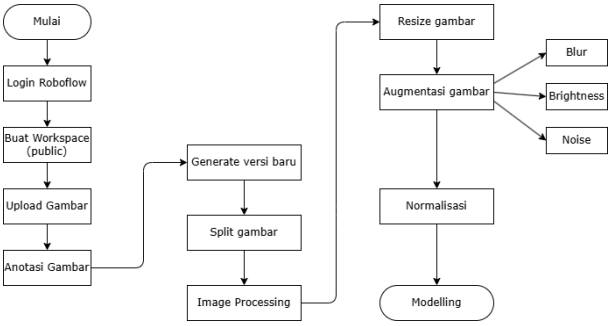
   Setiap gambar dianotasi dengan *bounding box* pada setiap objek *whiteflies* yang terdeteksi, memastikan setiap objek pada gambar dikenali dan diberi label yang sesuai.

3. Ekspor Data

   Hasil labelling dan *preprocessing* diekspor ke format YOLO, dengan struktur folder dataset yang terdiri dari folder *train*, *val*, dan *test*, masing-masing berisi folder images dan labels.

C. Data Preprocessing

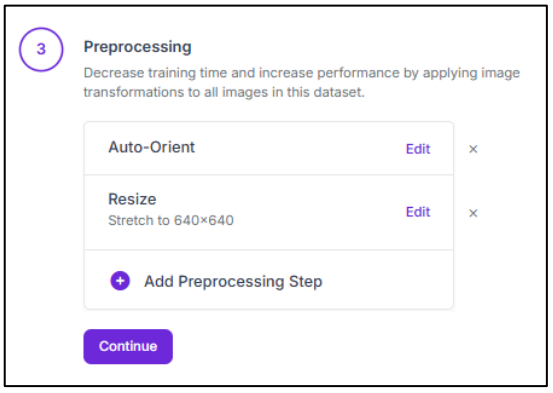
Tahapan data preparation meliputi preprocessing, augmentasi dan split data.



Gambar 6. Data *Preparation* Menggunakan *Roboflow*

1. Preprocessing

Tahapan *preprocessing* data terdiri dari *resize*. Penulis melakukan proses *resize* dengan mengubah ukuran gambar menjadi 640 * 640, karena data yang diperoleh banyak terdiri dari objek yang kecil, sehingga dibutuhkan resolusi yang besar dan seragam. Setelah melakukan proses preprocessing, penulis melakukan proses augmentasi.



Gambar 7. *Preprocessing* Data

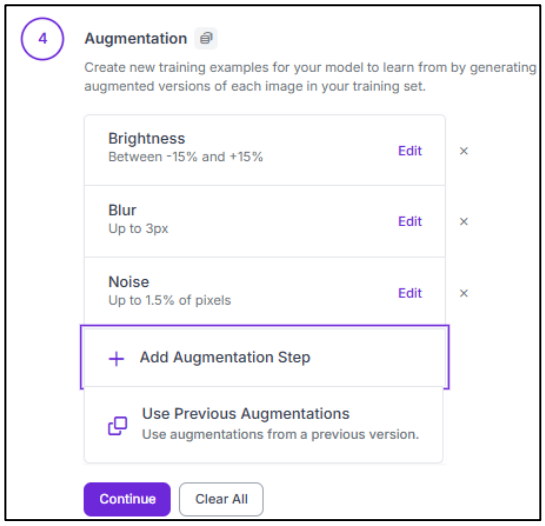
2. Augmentasi

Augmentasi adalah suatu proses yang dapat memperbanyak data. Proses ini akan menghasilkan

gambar dengan variasi yang lebih banyak dan dapat digunakan dalam proses pelatihan model. Berdasarkan tekniknya augmentasi dibagi menjadi 2 yaitu augmentasi spasial dan peningkatan piksel.Augmentasi spesial contohnya yaitu *scaling*, *cropping*, *flipping* dan *rotation*. Peningkatan piksel contohnya berupa *brightness*, *saturation*, dan *hue*. Augmentasi gambar dapat dilakukan menggunakan Roboflow. Penulis melakukan 3 teknik augmentasi pada penelitian ini yaitu *blur*, *brightness*,dan *noise*.

Penulis menggunakan teknik *blur* untuk mengatasi data yang tidak sempurna akibat blur karena resolusi maupun kompresi gambar. Teknik selanjutnya adalah *brightness*. Hal ini bertujuan agar mendapatkan data saat kondisi lapangan gelap dan terang.

Teknik *noise* memberikan kemampuan model untuk mendeteksi gambar dengan lensa yang ada *noise* nya akibat lensa kamera yang kotor terkena debu atau partikel kecil lainnya. Hal ini dilakukan untuk meningkatkan MAP.



Gambar 8. *Augmentation Setting*

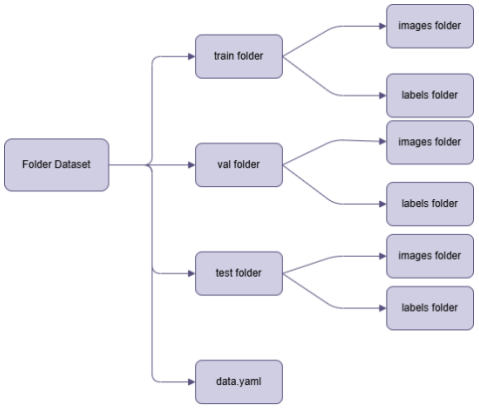
Penulis melakukan augmentasi pada data dan *bounding box*. Tabel 2 akan memperjelas teknik augmentasi yang penulis gunakan.

Tabel 2 Tabel Konfigurasi Augmentasi

No	Teknik	Nilai
1	*Brightness*	-15% terang & 15% gelap
2	*Blur*	3px
3	*Noise*	1.5%

3. *Split data*

Split data merupakan tahapan membagi data yang sudah dilakukan augmentasi menjadi 3 bagian, yaitu data latih, data validasi, dan data uji. Data yang sudah dibagi akan digunakan dalam proses *modeling*.

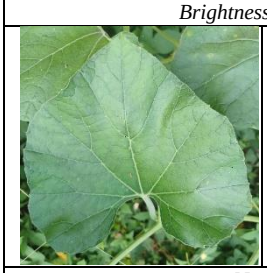


Gambar 9. Struktur *Folder Dataset*

Langkah terakhir adalah mengunduh hasil sesuai dengan format YOLOv11 yang dapat dilihat pada Gambar 9 menunjukan folder *root* dataset yang di dalam nya berisi 3 folder yaitu folder *train*, *val*, dan *test*. Masing-masing folder tersebut berisi 2 folder, yaitu folder *images* yang berisi gambar mentah dan folder *labels* yang berisi hasil *preprocessing* gambar dan augmentasi gambar. Urutan isi dari anotasi tersebut adalah:

1. Kelas
2. Jumlah File Total
3. Jumlah data *train*
4. Jumlah data valid
5. Jumlah Data test

Tabel 3. Perbandingan sebelum dan sesudah augmentasi

Sebelum	Sesudah
1940	3.352
Blur (Objek)	
	
Brightness (Objek)	
	
Noise	
	

D. Modelling

Model yang bagus dapat mempengaruhi fungsionalitas aplikasi dalam mendeteksi objek dengan tepat. Oleh karena itu dibutuhkan perancangan model yang sesuai agar menghasilkan model yang bagus. Tahapan merancang model yang diajukan penulis dapat dilihat pada Gambar 10. merupakan diagram alur perancangan model yang akan dilakukan penulis. Tahapan dimulai dengan memperoleh data berupa hama *whiteflies* yang terdiri dari data latih, data validasi, dan data uji. Setelah data diperoleh, langkah selanjutnya adalah proses pelatihan (*training*). Sebelum melatih model, penulis terlebih dahulu menuliskan *library* yang diperlukan. Kemudian penulis mengatur parameter yang akan digunakan dalam proses pelatihan.

Gambar 10. Diagram Alur Tahapan *Modelling*

1. Implementasi Model Yolov11

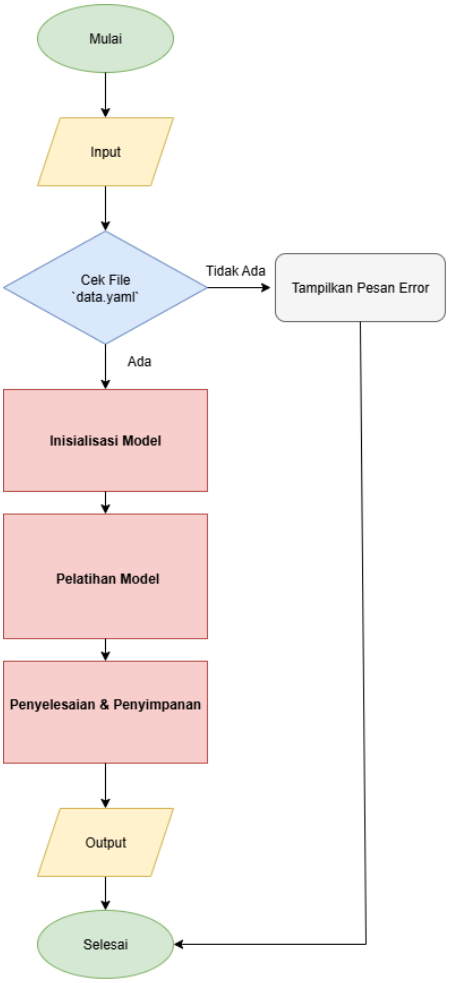
Pada tahap ini gambar yang telah melalui preprocessing serta augmentasi akan di implementasikan menggunakan model YOLOv11 dengan melakukan proses yang dapat dilihat pada *source code* berikut:

```
# ==========================================================================
# LANGKAH 3: PELATIHAN MODEL YOLO
# ==========================================================================
print("\n" + "="*80)
print("LANGKAH 3: PELATIHAN MODEL YOLO (MENGGUNAKAN ANOTASI ASLI)")
print("="*80)

from ultralytics import YOLO

# @markdown ### 🔁 Lanjutkan Pelatihan (Opsional)
RESUME_TRAINING = False  #@param {type:"boolean"}
s
if os.path.exists(YAML_PATH):
    model = YOLO("yolo11n.pt")

    results = model.train(
        data=YAML_PATH, # Menggunakan path absolut ke YAML
        epochs=150,
        patience=50,
        imgsz=640,
        optimizer='AdamW',
        project=DRIVE_PROJECT_PATH,
        name=f'{PROJECT_NAME}_YOLOv11n_original_annot_results',
        exist_ok=True,
        resume=RESUME_TRAINING
    )

    print(f"\n🎉 Pelatihan selesai. Hasil disimpan di: {results.save_dir}")
else:
    print("❌ Gagal memulai pelatihan. Pastikan Langkah 2 berhasil dijalankan.")
```

Gambar 10. *Script* Pelatihan Model

Kode ini menjalankan proses pelatihan untuk model deteksi objek YOLOv11. Prosesnya diawali dengan memeriksa apakah file konfigurasi dataset (YAML_PATH) tersedia. Jika ya, kode akan memuat model YOLOv11 dengan bobot *pre-train* (yolo11n.pt), kemudian

memulai proses pelatihan selama 150 epoch menggunakan parameter spesifik seperti ukuran gambar 640x640 dan optimizer AdamW. Setelah selesai, seluruh hasil pelatihan, termasuk model yang telah jadi, akan disimpan ke direktori yang telah ditentukan.



Gambar 11. Diagram Alur *Train Model*

1. Pengecekan Keberadaan File

a. Proses: Sistem pertama-tama melakukan pengecekan untuk memastikan keberadaan file data.yaml.

b. Tujuan: File data.yaml adalah "*map*" bagi model. Di dalamnya terdapat informasi vital seperti: Lokasi direktori untuk data latih (*train*), data validasi (*validation*), dan data uji (*test*). Jumlah kelas objek yang akan dideteksi. Nama dari setiap kelas (contoh: 'whitefly', 'mentimun', dll.).

c. Hasil:

Jika Gagal (Tidak Ada): Apabila file ini tidak ditemukan, proses tidak dapat dilanjutkan karena model tidak tahu data apa yang harus dipelajari. Sistem akan menampilkan pesan error "Gagal memulai pelatihan" dan proses berhenti. Jika Berhasil (Ada): Sistem melanjutkan ke tahap inisialisasi model.

2. Instalasi Model

a. Proses: Kode akan memuat arsitektur dasar YOLOv11 dan mengisinya dengan bobot (*weights*) awal dari file yolo11n.pt.

b. Tujuan: Penerapan dari konsep *Transfer Learning*. Model tidak belajar dari nol, melainkan memulai dengan "pengetahuan dasar" dari model yolo11n.pt yang sudah dilatih pada dataset besar (seperti COCO). Model ini sudah bisa mengenali bentuk, tekstur, dan fitur umum. Dengan fondasi ini, proses pelatihan untuk mendeteksi objek spesifik (seperti hama *Whitefly*) menjadi jauh lebih cepat dan efektif..

3. Pelatihan Model

a. Proses: Sistem menjalankan fungsi model.train() dengan serangkaian parameter (disebut *hyperparameters*) yang telah ditentukan.

b. Tujuan Parameter:

Epochs: 150: Model akan "melihat" dan belajar dari keseluruhan dataset sebanyak 150 kali putaran. Setiap epoch membantu model mempertajam kemampuannya dalam mengenali pola. Patience: 50: Ini adalah mekanisme *early stopping*. Jika performa model pada data validasi tidak menunjukkan peningkatan selama 50 epoch berturut-turut, pelatihan akan berhenti secara otomatis. Tujuannya adalah untuk

efisiensi dan mencegah *overfitting* (model terlalu hafal data latih tapi buruk pada data baru).Image Size: 640x640: Semua gambar dari dataset akan diubah ukurannya menjadi 640x640 piksel sebelum diproses. Ukuran yang lebih besar ini membantu model mendeteksi objek-objek kecil dengan lebih baik.

4. Penyelesaian dan Penyimpanan

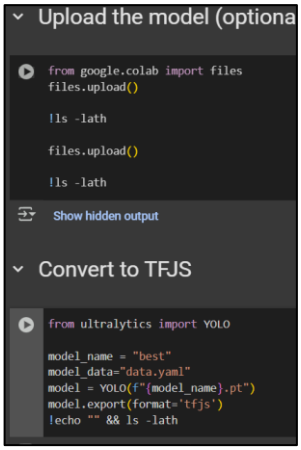
a. Proses: Sistem menyimpan semua *instance* yang dihasilkan selama pelatihan.

b. Tujuan: Output terpenting dari tahap ini adalah file bobot best.pt. File ini berisi "otak" dari model yang sudah terlatih khusus untuk mendeteksi objek Anda. Selain itu, disimpan juga hasil lain seperti grafik kurva *loss* dan *precision/recall*.

c. Hasil Akhir: Model best.pt kini siap untuk digunakan (di-*deploy*) dalam aplikasi untuk melakukan deteksi objek pada gambar baru atau video secara *real-time*.

2. Konversi Model Yolov11 ke TFJS

Konversi model ini miliki tujuan utama yaitu untuk mengubah file model YOLO yang sudah terlatih (best.pt) menjadi format TensorFlow.js (TFJS). Format ini diperlukan agar model deep learning Anda dapat berjalan secara langsung di browser web.



Gambar 12. Script ubah Bobot ke TFJS

Gambar 13. Skema Convert TFJS

1. Persiapan Model

Bagian ini digunakan jika Anda menjalankan kode di environment seperti *Google Colab* dan file *best.pt* Anda berada di komputer lokal.from *google.colab import files & files.upload():* Kode ini memanggil fungsi dari *Google Colab* untuk membuka dialog file *upload. !ls -lath*: Ini adalah perintah terminal untuk menampilkan daftar file di direktori saat ini, beserta detailnya. Perintah ini dijalankan untuk memverifikasi bahwa file yang sudah berhasil diunggah.

2. Convert ke TFJS

Ini adalah bagian inti dari proses konversi. Mengambil model *PyTorch* (.pt) dan mengekspornya ke format yang dapat dibaca dan dieksekusi oleh library *TensorFlow*.js di sisi klien (*browser*). Dari ultralytics import yolo: Mengimpor library yolo yang diperlukan untuk memanipulasi model, model_name = "*best" & model_data="data.yaml*": Menentukan nama file model utama (yaitu *best*.pt) dan file konfigurasinya (*data.yaml*), model = *yolo(f"{model_name}.pt*"): Membuat objek model dengan memuat file bobot best.pt yang telah dilatih sebelumnya, model.export (*format='tfjs'*): Ini adalah perintah utama.

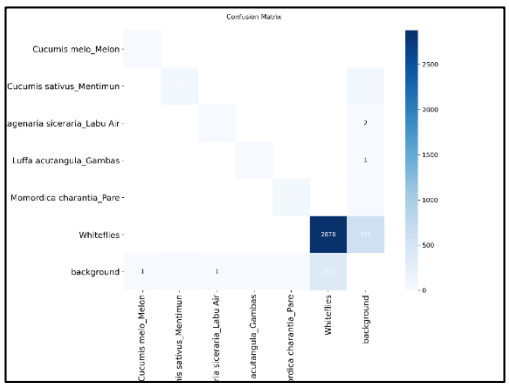
Fungsi.*export*() dijalankan pada model dengan argumen format='tfjs', yang menginstruksikan library untuk melakukan proses konversi. Hasilnya adalah sebuah folder baru (biasanya bernama *best_web_model*) yang berisi file-file yang diperlukan oleh *TensorFlow*.js, dan !echo "" && ls -lath: Sama seperti sebelumnya, perintah ini digunakan untuk menampilkan daftar file dan

memverifikasi bahwa folder hasil konversi (*best_web_model*) telah berhasil dibuat.

E. *Evaluation*

Pada tahapan ini, penulis akan melakukan proses validasi model menggunakan dataset validasi terhadap weight yang didapat pada proses pelatihan model. Berdasarkan hasil validasi akan diperoleh MAP dan confusion matrix.

1. Confusion Matrix



Gambar 14. Confusion Matrix Hasil Pelatihan Model

Confusion Matrix ini adalah sebuah tabel yang merangkum kinerja model klasifikasi. Sumbu vertikal (*Predicted*) menunjukkan kelas apa yang diprediksi oleh model, sedangkan sumbu horizontal (*True*) menunjukkan kelas yang sebenarnya dari data tersebut.

Tabel 4. 1 *Confusion matrix summary*

Parameter	**Prediksi: *Whiteflies***	**Prediksi: Bukan *Whiteflies***
Class: *Whiteflies*	TP: 2.878	FN: 585
Class: Bukan *Whiteflies*	FP: 355	TN: -

Dari tabel *confusion matrix* diatas dapat dijabarkan untuk mencari nilai *precision, recall* dan *F1-Score*:

1. *Precision* (Presisi): Seberapa akurat prediksi positif model.

a. Rumus: TP / (TP + FP)

b. Perhitungan: 2.878 / (2.878 + 585) = 2.878 / 3.463= 0.831 atau 83.1%

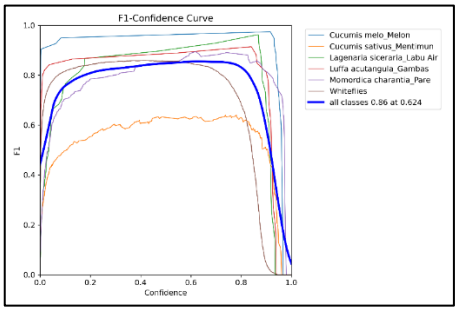
c. Kesimpulan: Dari semua objek yang dideteksi oleh model sebagai *'Whiteflies'*, 83.1% di antaranya adalah benar-benar hama Whiteflies pada daun tanaman *cucurbitae* yang terjangkit oleh jenis hama ini dan memang terbukti ada di lapangan.

2. *Recall* (Sensitivitas): Seberapa baik model menemukan semua hama yang ada.

a. Rumus: TP / (TP + FN)

b. Perhitungan: 2.878 / (2.878 + 365) = 2.878 / 3.243= 0.890 atau 89.0% Kesimpulan: Model berhasil menemukan dan mengidentifikasi 89% dari total hama *Whiteflies* yang ada di seluruh dataset uji. Dalam konteks pertanian, kondisi ini lebih menguntungkan daripada gagal mendeteksi hama yang ada (*False Negative*), namun kurang efektif dalam segi biaya penangan. Deteksi yang terlewat maupun tak terdeteksi dapat menyebabkan penyebaran wabah yang tidak terkendali.

1. F1-Score : Keseimbangan antara Presisi dan Recall.

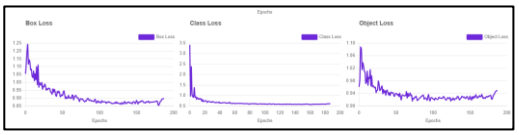


Gambar 15. Grafik F1 Score

1. Rumus: 2 * (*Precision* * *Recall*) / (*Precision* + *Recall*)
2. Perhitungan: 2 * (0.831 * 0.890) / (0.831 + 0.890) = 1.480/1.721
   0.86 atau 86%
3. Kesimpulan: Model mencapai F1-Score sebesar 86%, yang menunjukkan adanya

keseimbangan yang sangat baik antara kemampuan untuk membuat deteksi yang akurat (presisi) dan kemampuan untuk menemukan semua target yang relevan (*recall*). bahwa F1-score tertinggi untuk semua kelas dicapai pada tingkat kepercayaan (*confidence*) 0.697, dengan nilai F1-score rata-rata 0.86.

2. F1-Score : Keseimbangan antara Presisi dan Recall.



Gambar 15. Grafik evaluasi kurva *loss*

1. Box Loss

Box Loss bertujuan untuk mengukur tingkat kesalahan dalam prediksi koordinat (x, y), lebar, dan tinggi dari bounding box yang dihasilkan oleh model. Analisa hasil dari gambar 4.18 menunjukkan penurunan yang konsisten dan signifikan seiring bertambahnya jumlah *epoch*. Pada awal pelatihan, Box Loss berada pada nilai sekitar 1.15, yang menandakan adanya ketidakakuratan yang relatif tinggi dalam prediksi koordinat dan dimensi bounding box. Namun, seiring dengan proses pelatihan, nilai loss ini secara signifikan menurun dan mencapai stabilitas di bawah 0.90. Hal ini menandakan bahwa model secara efektif belajar untuk mengenali hama whiteflies dengan akurat yang semakin tinggi pada data train.

2. Class Loss

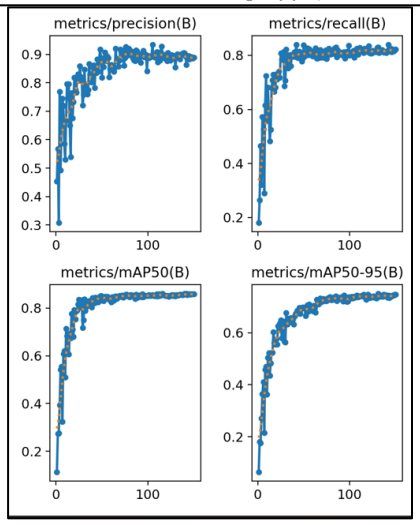
Class Loss bertujuan mengukur kesalahan dalam memberikan label kelas yang benar untuk objek yang terdeteksi, misalnya membedakan antara 'Whitefly' dan kelas lainnya. Analisa hasil dari gambar 4.18 memperlihatkan penurunan yang paling tajam di antara ketiga jenis loss. Nilai Class Loss awal yang berada di atas 3.0 dengan cepat merosot dan stabil pada level yang sangat rendah, sekitar 0.5. Penurunan drastis ini menandakan *class loss* memiliki efisiensi yang luar biasa dalam mempelajari fitur-fitur antar jenis loss, memungkinkannya untuk mengidentifikasi hama *whiteflies* dari objek non-hama dengan cepat dan akurat.

3. Object Loss

Object Loss bertujuan mengukur kesalahan dalam menentukan tingkat keyakinan (*confidence*) apakah sebuah objek benar-benar ada di dalam bounding box yang diprediksi atau tidak. Dari gambar 4.18 juga menunjukkan penurunan yang positif. Meskipun tidak setajam *Class Loss*, penurunan *Object Loss* secara bertahap dari nilai awal yang lebih tinggi menuju stabilitas pada level yang lebih rendah menunjukkan bahwa model semakin akurat dalam mengidentifikasi apakah suatu objek benar-benar ada di dalam *bounding box* yang diprediksi. Secara keseluruhan, Analisa dari ketiga kurva *loss* ini mengkonfirmasi efektivitas model YOLOv11 dalam mendeteksi hama *whiteflies*. Konsistensi penurunan *loss* pada data pelatihan dan validasi, serta konvergensi pada nilai yang rendah, membuktikan bahwa model telah dilatih dengan baik, mampu melokalisasi, mengklasifikasikan, dan mengidentifikasi hama *whiteflies* dengan akurasi dan stabilitas yang memadai.

c. mAP

Mean Average Precision (mAP) adalah metrik wajib untuk mengevaluasi akurasi model deteksi objek seperti YOLO, metrik ini menentukan apakah model mampu memberikan label kelas yang benar untuk objek yang dideteksi dan seberapa presisi kotak pembatas (*bounding box*) yang digambar oleh model.



Gambar 16. Grafik mAP

1. mAP@50 (Akurasi Deteksi Standar)

Kurva ini merepresentasikan mAP yang dihitung pada ambang batas *Intersection over Union* (IoU) sebe sar 0.5. Sebuah deteksi dianggap benar jika kotak prediksi tumpang tindih minimal 50% dengan kotak asli. Grafik menunjukkan fase pembelajaran yang sangat cepat pada ~20 epoch pertama, di mana mAP naik dari ~0.3 menjadi ~0.7. Setelah itu, kurva terus menunjukkan peningkatan bertahap hingga mencapai titik konvergensi (stabilitas) di nilai puncak sekitar 0.856 (85.6%). Nilai mAP@50 yang mencapai 85.6% menandakan bahwa model memiliki efektivitas yang bagus dalam mendeteksi keberadaan hama *whiteflies* dengan benar.

2. mAP@50-95 (Akurasi Deteksi Ketat)

Kurva ini adalah metrik yang lebih menantang dan menjadi standar emas dalam evaluasi. Nilainya adalah rata-rata dari mAP yang dihitung pada sepuluh ambang batas IoU berbeda (dari 0.50 hingga 0.95).

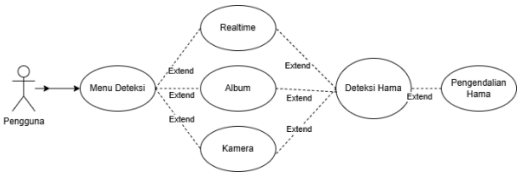
Trennya serupa dengan mAP@50, namun peningkatannya lebih landai, yang merupakan hal wajar karena tugas yang lebih sulit. Kurva ini mencapai stabilitas pada nilai akhir sekitar 0.8 (81.2%).

F. *Deployment*

Penulis melakukan proses *deployment* dengan merancang aplikasi yang dimulai dengan membuat UML (*Unified Modeling Language*), *wireframe*, *interface*, kemudian mengembangkan aplikasi berdasarkan rancangan yang telah dibuat.

1. Perancangan UML (*Unified Modeling Language*)

a. *Use Case* Diagram



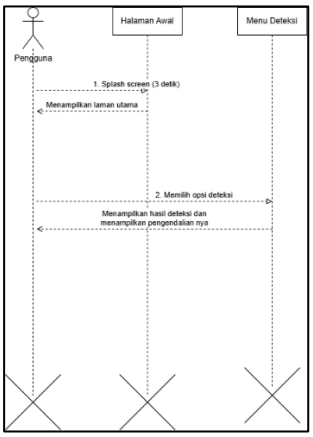
Gambar 17. *Use Case Diagram*

Aplikasi ini akan terdiri dari 2 menu, yaitu menu deteksi dan menu informasi. Pada menu deteksi, pengguna dapat memilih mendeteksi objek terkait secara *realtime*, melalui album atau dari kamera. Jika pengguna memilih secara *realtime* maka aplikasi akan membuka kamera secara otomatis dan objek terkait akan terdeteksi, jika pengguna ingin melakukan deteksi yang objek nya didapat dari album, maka dapat memilih opsi album, dan jika ingin mendeteksi objek dari kamera, maka kamera akan muncul secara otomatis, pengguna menekan tombol jepret dan tekan “Ok”, maka gambar yang didapat akan berhasil dideteksi.

Menu yang kedua yaitu menu informasi. Menu ini akan menampilkan pestisida apa saja yang digunakan dalam pengendalian hama tanaman. Pengguna juga dapat melihat detail dari pestisida terkait, sehingga akan menjadi wawasan bagi pengguna dalam melakukan pengendalian hama tanaman.

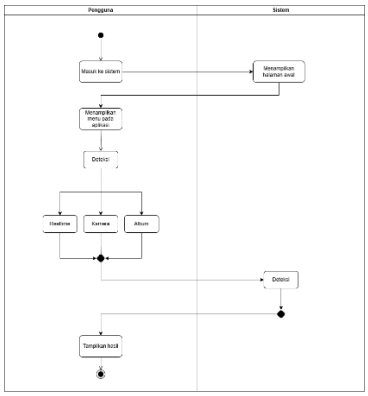
b. *Sequence* Diagram

Menunjukan urutan dari proses penggunaan sistem. Aplikasi dimulai dengan menampilkan halaman awal hingga masuk ke halaman utama. Setelah berada di halaman utama, pengguna dihadapkan dengan 3 menu, yaitu menu beranda, dan menu deteksi. Pada menu deteksi, pengguna melakukan deteksi sampai aplikasi menampilkan hasil deteksinya.



Gambar 18. *Sequence Diagram*

c. *Activity* Diagram



Gambar 19. *Activity Diagram*

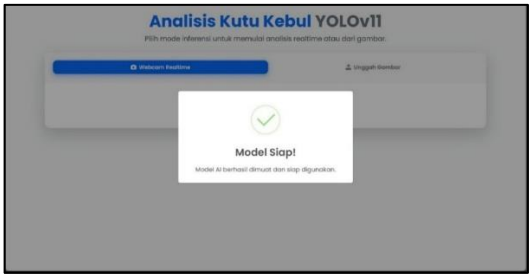
Aktivitas dimulai oleh pengguna yang membuka aplikasi, sehingga aplikasi menampilkan halaman awal. Setelah menampilkan halaman awal, pengguna diarahkan dengan menampilkan menu pada aplikasi. Pada aplikasi ini, pengguna dapat melakukan Deteksi dengan tiga opsi utama: pertama, secara Realtime, di mana pengguna cukup mengarahkan kamera yang ditampilkan pada aplikasi ke objek, dan objek akan terdeteksi secara langsung; kedua, melalui

Kamera, di mana kamera akan tampil pada aplikasi dan pengguna memfoto objek sehingga objek akan tampil pada layer deteksi, kemudian objek akan terdeteksi; ketiga, melalui Album, di mana pengguna memilih objek dari album, objek yang dipilih akan tampil di layer deteksi, kemudian objek akan terdeteksi.

Setelah objek berhasil dideteksi oleh sistem, aplikasi akan menampilkan hasil deteksi tersebut kepada pengguna.

2. Perancangan Infterface

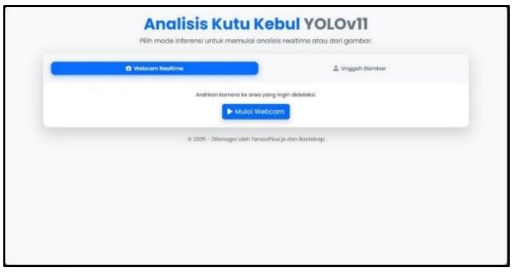
a. Halaman Awal



Gambar 20. Model Berhasil di *Load*

Halaman awal yang mengindikasikan bahwa model bobot yang telah dikonversi menjadi tfjs telah berhasil di-load. Apabila *hover* box nya sudah menghilang maka artinya model berhasil di load dan siap untuk digunakan.

b. Menu Deteksi *Realtime*

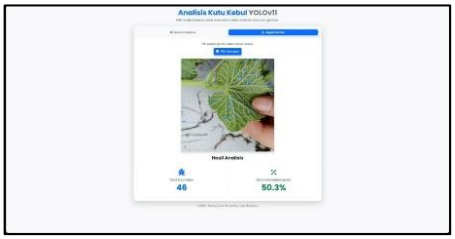


Gambar 21. Menu Deteksi *Realtime*

Setelah pengguna memilih mode "Webcam Realtime", antarmuka ini muncul sekaligus meminta *action* dari user.

Dengan menekan tombol "► Mulai Webcam", pengguna memberikan izin kepada browser untuk mengakses kamera, yang setelahnya akan langsung menampilkan *feed* video untuk dianalisis oleh model YOLOv11.

c. Menu Deteksi Album



Gambar 22. Deteksi Hama dari Album

Jendela ini merupakan jendela untuk deteksi yang dapat dilakukan menggunakan gambar atau foto yang sudah ada,di halaman ini memuat fungsi yang sama seperti pada halaman sebelumnya namun dengan dilengkapi dengan keterangan informasi seberapa tingkat *confidence* aplikasi atau model dalam menentukan hama *whitefly* serta jumlah total yang berhasil di deteksi.

Aplikasi dirancang menggunakan bahasa pemrograman php dan JS tanpa menggunakan database. Model yang telah dipanggil dalam file JS nya yang sebelumnya dalam bentuk TFJS akan diload diawal aplikasi web dibuka. Langkah selanjutnya dalam model yang berhasil di load dapat melakukan tugasnya dalam mendeteksi hama *whiteflies* dalam hal ini dapat berupa *live detector* maupun kita dapat mengambilnya langsung dari local machine yang ada di PC kita.

V. KESIMPULAN DAN SARAN

A. Kesimpulan

Pengembangan model deteksi hama *whiteflies* (*Aleyrodidae*) pada tanaman *Cucurbitaceae* berbasis YOLOv11 berhasil dilakukan dengan tahapan yang sistematis, mulai dari pengumpulan dan anotasi dataset citra daun tanaman *Cucurbitaceae* yang terinfestasi *whiteflies*, *preprocessing* data, pelatihan model, hingga *deployment* aplikasi berbasis web yang mudah diakses oleh pengguna. Model YOLOv11 menunjukkan performa yang baik dalam mendeteksi whiteflies pada daun tanaman *Cucurbitaceae*. Hasil evaluasi menunjukkan nilai *precision* sebesar 83,1%, recall 89,0%, dan F1-score 86,0%. Selain itu, nilai mAP@50 mencapai 85,6% dan akruasi *overall* menggunakan mAP@50-95 sebesar 81,2%, yang menandakan model mampu melakukan deteksi objek kecil (*whiteflies*) secara akurat dan konsisten pada berbagai kondisi gambar.

Implementasi aplikasi deteksi secara real-time maupun dari gambar album memberikan kemudahan bagi pengguna, terutama petani dan peneliti, untuk melakukan identifikasi infestasi *whiteflies* secara cepat, tepat, dan responsif di lapangan. Hal ini mendukung upaya deteksi dini dan pengendalian hama secara presisi demi menjaga produktivitas dan keberlanjutan budidaya tanaman *Cucurbitaceae*. Pemanfaatan dataset publik dan teknik augmentasi data terbukti efektif dalam meningkatkan variasi dan kualitas data pelatihan, sehingga model lebih *robust* terhadap kondisi pencahayaan, *noise*, dan variasi visual pada daun tanaman. Penelitian ini memberikan kontribusi nyata dalam pengembangan teknologi pertanian cerdas (*smart farm*ing) berbasis deep learning di Indonesia, khususnya untuk pengendalian hama penting seperti *whiteflies* yang berdampak besar pada produktivitas dan ekonomi pertanian nasional.

B. Saran

Berdasarkan hasil penelitian dan keterbatasan yang ditemui, berikut beberapa saran untuk pengembangan lebih lanjut:

1. Perluasan dan diversifikasi dataset: Penelitian selanjutnya disarankan untuk memperluas jumlah dan jenis citra daun tanaman Cucurbitaceae, termasuk variasi kondisi lingkungan, fase pertumbuhan tanaman, serta tingkat keparahan infestasi whiteflies, agar model semakin general dan adaptif terhadap kondisi nyata di lapangan.

2. Pengujian pada perangkat mobile dan integrasi IoT: Mengingat kebutuhan mobilitas tinggi di sektor pertanian, pengembangan aplikasi deteksi berbasis YOLOv11 dapat dioptimalkan untuk perangkat mobile (Android/iOS) atau diintegrasikan dengan perangkat IoT (kamera lapangan), sehingga deteksi hama dapat dilakukan secara otomatis dan real-time di berbagai lokasi pertanian.

3. Perbandingan dengan algoritma lain: Untuk memperoleh hasil yang lebih komprehensif, penelitian berikutnya dapat membandingkan performa YOLOv11 dengan algoritma deteksi objek lain seperti Faster R-CNN, SSD, atau versi YOLO terbaru, khususnya dalam mendeteksi objek kecil dan tersembunyi pada citra daun.

4. Integrasi sistem rekomendasi pengendalian: Aplikasi dapat dikembangkan lebih lanjut dengan menambahkan fitur rekomendasi tindakan pengendalian hama berbasis data deteksi, seperti saran penggunaan pestisida ramah lingkungan, rotasi tanaman, atau tindakan agronomis lainnya yang sesuai dengan tingkat infestasi yang terdeteksi.

DAFTAR PUSTAKA

[1] A. J. Bharadwaj, C. Thakuria, J. M. Rabha, G. K. Das, and K. Das, “INDIAN JOURNAL OF SCIENCE AND TECHNOLOGY Smart Agriculture via Object Detection,” 2023, doi: 10.17485/IJST/v16iSP2.2438.

[2] H. Ezzy, M. Charter, A. Bonfante, and A. Brook, “How the small object detection via machine learning and uas-based remote-sensing imagery can support the achievement of sdg2: A case study of vole burrows,” *Remote Sensing*, vol. 13, no. 16, Aug. 2021, doi: 10.3390/rs13163191.

[3] M. Aoun *et al.*, “OPEN ACCESS EDITED BY GAN-based semi-automated augmentation online tool for agricultural pest detection: A case study on whiteflies.” [Online]. Available: https://github.com/ChristopheKar/cpb-gen,

[4] W. Zhao, W. Yamada, T. Li, M. Digman, and T. Runge, “remote sensing Augmenting Crop Detection for Precision Agriculture with Deep Visual Transfer Learning-A Case Study of Bale Detection,” 2020, doi: 10.3390/rs1301.

[5] A. Ramcharan *et al.*, “A mobile-based deep learning model for cassava disease diagnosis,” *Frontiers in Plant Science*, vol. 10, Mar. 2019, doi: 10.3389/fpls.2019.00272.

[6] P. Hidayat, D. Bintoro, L. Nurulalia, and M. Basri, “SPECIES, HOST RANGE, AND IDENTIFICATION KEY OF WHITEFLIES OF BOGOR AND SURROUNDING AREA,” *Journal of Tropical Plant Pests and Diseases*, vol. 18, no. 2, pp. 127–150, Sep. 2018, doi: 10.23960/j.hptt.218127-150.

[7] U. Tanjungpura, “ANALYSIS OF THE ECONOMIC POTENTIAL AND EFFICIENCY OF CUCURBITACEAE AGRICULTURAL BUSINESSES IN THE PEATLANDS,” 2024.

[8] Z. E. D. Zuraida, “HUBUNGAN KEKERABATAN TUMBUHAN FAMILI CUCURBITACEAE BERDASARKAN KARAKTER MORFOLOGI DI KABUPATEN PIDIE SEBAGAI SUMBER BELAJAR BOTANI TUMBUHAN TINGGI,” *Jurnal Agroristek*, vol. 2, no. 1, Art. no. 1, May 2019, doi: 10.47647/jar.v2i1.88.

[9] N. Nursamsu, E. Susantini, Y. Yuliani, and N. Nurhafidhah, “Diversity and utilization of vegetables and spices by coastal community in East Aceh, Indonesia,” *Biodiversitas*, vol. 26, no. 2, Feb. 2025, doi: 10.13057/biodiv/d260232.

[10] I. G. R. M. Temaja *et al.*, “Begomovirus diversity and distribution on melon plants in Bali, Indonesia,” *Biodiversitas Journal of Biological Diversity*, vol. 26, no. 2, Feb. 2025, doi: 10.13057/biodiv/d260206.

[11] D. Lu and Y. Wang, “MAR-YOLOv9: A multi-dataset object detection method for agricultural fields based on

YOLOv9," *PLoS ONE*, vol. 19, no. 10 October, Oct. 2024, doi: 10.1371/journal.pone.0307643.

[12] G. Schrader, M. Camilleri, R. M. Ciubotaru, M. Diakaki, and S. Vos, "Pest survey card on Aleurocanthus spiniferus and Aleurocanthus woglumi," *EFSA Supporting Publications*, vol. 16, no. 2, Feb. 2019, doi: 10.2903/sp.efsa.2019.EN-1565.

[13] E. Mujkic, O. Ravn, and M. P. Christiansen, "Framework for environment perception: Ensemble method for vision-based scene understanding algorithms in agriculture," *Frontiers in Robotics and AI*, vol. 9, Jan. 2023, doi: 10.3389/frobt.2022.982581.

[14] K. Li *et al.*, "Satellite-based Crop Identification and Risk Profiling for Area Wide Management of Whitefly and Tomato Yellow Leaf Curl Virus in Southwest Florida," *Plant Disease*, Apr. 2025, doi: 10.1094/PDIS-12-24-2634-RE.

[15] W. Yan *et al.*, "Managing Super Pests: Interplay between Pathogens and Symbionts Informs Biocontrol of Whiteflies," *Microorganisms*, vol. 12, no. 5, Art. no. 5, May 2024, doi: 10.3390/microorganisms12050887.

[16] O. Z. Aregbesola *et al.*, "Production Characteristics and Strategies for Adapting to the Impact of Climate Change on Cassava Whiteflies and Viruses in Tanzania," *Vietnam Journal of Agricultural Sciences*, vol. 4, no. 1, Art. no. 1, Jun. 2021, doi: 10.31817/vjas.2021.4.1.03.

[17] A. Kumar *et al.*, "Validation of IPM strategy in Bt cotton in whitefly (Bemisia tabaci) hot spot of North-West India," *Indian J Agri Sci*, vol. 91, no. 7, Sep. 2021, doi: 10.56093/ijas.v91i7.115138.

[18] E. Rodríguez, M. M. Téllez, and D. Janssen, "Whitefly Control Strategies against Tomato Leaf Curl New Delhi Virus in Greenhouse Zucchini," *IJERPH*, vol. 16, no. 15, p. 2673, Jul. 2019, doi: 10.3390/ijerph16152673.

[19] C. Rapisarda and G. E. M. Cocuzza, Eds., *Integrated pest management in tropical regions*. Boston, MA: CABI, 2018.

[20] Juwita Lestari and Firman Ali Rahman, "Okulasi Tanaman Mangga dan Langsat untuk Membuktikan Adanya Komunikasi Sel Tanaman Beda Jenis," *OEJBPB*, vol. 1, no. 1, pp. 16–22, Aug. 2023, doi: 10.62588/n74r0888.

[21] S. Wang, "Research on Computer Vision Image Classification," *AJCIS*, vol. 6, no. 6, 2023, doi: 10.25236/AJCIS.2023.060604.

[22] C. Nakkach, A. Zrelli, and T. Ezzeddine, "Deep Learning Algorithms Enabling Event Detection: A Review:," presented at the 2nd International Conference on Industry 4.0 and Artificial Intelligence (ICIAI 2021), Sousse, Tunisia, 2022. doi: 10.2991/aisr.k.220201.030.

[23] B. Ma, F. Li, and Y. Ren, "Review of Research Progress in Computer Vision," *AETR*, vol. 11, no. 1, p. 558, Jul. 2024, doi: 10.56028/aetr.11.1.558.2024.

[24] J. Terven, D.-M. Córdova-Esparza, and J.-A. Romero-González, "A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS," *MAKE*, vol. 5, no. 4, pp. 1680–1716, Nov. 2023, doi: 10.3390/make5040083.

[25] G. Lavanya and S. D. Pande, "Enhancing Real-time Object Detection with YOLO Algorithm," *EAI Endorsed Trans IoT*, vol. 10, Dec. 2023, doi: 10.4108/eetiot.4541.

[26] A. Bhagwat, S. Dutta, D. Saha, and M. J. B. Reddy, "An online 11 kv distribution system insulator defect detection approach with modified YOLOv11 and mobileNetV3," *Sci Rep*, vol. 15, no. 1, p. 15691, May 2025, doi: 10.1038/s41598-025-99756-5.