DETEKSI & KUANTIFIKASI TRIKOMA TIPE GLANDULAR (BULBOSE) PADA CITRA DAUN TANAMAN KENTANG MENGGUNAKAN *DEEP LEARNING*

M. Fauzan Azhari

21917011

Tesis diajukan sebagai syarat untuk meraih gelar Magister Komputer

Konsentrasi Sains Data

Program Studi Informatika Program Magister

Fakultas Teknologi Industri

Universitas Islam Indonesia

2024

Lembar Pengesahan Pembimbing

Deteksi & Kuantifikasi Trikoma Tipe Grandular (Bulbose) pada Citra Daun Tanaman Kentang Menggunakan *Deep Learning*

M. Fauzan Azhari

21917011

Yogyakarta, November 2024

Pembimbing 1	Pembimbing 2
Dr. tech. Rohmatul Fajriyah, S.Si., M.Si.	Izzati Muhimmah, S.T., M.Sc., Ph.D.

Lembar Pengesahan Penguji

Deteksi & Kuantifikasi Trikoma Tipe Grandular (Bulbose) pada Citra Daun Tanaman Kentang Menggunakan *Deep Learning*

M. Fauzan Azhari

21917011

Yogyakarta, November 2024

Tim Penguji,

Dr. tech. Rohmatul Fajriyah, S.Si., M.Si.
Ketua Penguji

Izzati Muhimmah, S.T., M.Sc., Ph.D.
Penguji I

Dhomas Hatta Fudholi, S.T., M.Eng., Ph.D.
Penguji II

Mengetahui,

Ketua Program Studi Informatika Program Magister

Universitas Islam Indonesia

Irving Vitra Paputungan, S.T., M.Sc., Ph.D.

Abstrak

Deteksi & Kuantifikasi Trikoma Tipe Grandular (Bulbose) pada Citra Daun Tanaman Kentang Menggunakan *Deep Learning*

Kentang merupakan salah satu tanaman yang paling banyak diproduksi dan dibudidayakan di Indonesia, karena kentang mempunyai nilai gizi yang sangat tinggi. Supaya budidaya/produksi tanaman kentang memiliki kualitas yang baik, banyak orang dari kalangan petani, peneliti hingga pemulia tanaman berusaha agar mengeksplorasi dan memahami karakteristik sumber resistensi baru, salah satunya yaitu peranan trikoma sebagai mekanisme pertahanan pada tanaman. Trikoma merupakan bagian berupa rambut-rambut halus yang melapisi begian terluar pada daun tanaman yang berfungsi sebagai penghalang fisik dan kimia. Identifikasi dan kuantifikasi pada trikoma biasa dilakukan peneliti dengan perhitungan manual. Pekerjaan seperti ini tentunya akan menghabiskan banyak waktu dan tenaga serta tidak efektif. Maka dari itu kebutuhan akan adanya sistem yang dapat melakukan deteksi dan kuantifikasi trikoma secara otomatis sangat diperlukan agar identifikasi dan kuantifikasi pada trikoma tidak lagi dilakukan secara manual dan dapat dilakukan dalam waktu yang cepat. Penelitian ini menggunakan pendekatan *deep learning* untuk melatih model agar dapat melakukan deteksi dan kuantifikasi objek berupa trikoma. Arsitektur model yang digunakan yaitu YOLOv8. Dari hasil proses *training* yang telah dilakukan diperoleh hasil nilai *mean average precision* (mAP) pada *treshold confidence* = 50 yaitu 0,816 sedangkan nilai mAP pada *treshold confidence* = 90 yaitu 0,38. Dari model ini diharapkan dapat membantu pakar atau peneliti dibidang pertanian dalam melakukan identifikasi trikoma sehingga dapat mengoptimalkan hasil panen tanaman.

Kata kunci

trikoma grandular, kentang, *deep learning*, *convolutional neural network,* YOLOv8

Abstract

Detection and Quantification of Glandular Trichomes (Bulbous) on Potato Plant Leaf Images Using Deep Learning

Potatoes are one of the most widely produced and cultivated crops in Indonesia due to their high nutritional value. To ensure the cultivation and production of high-quality potatoes, farmers, researchers, and plant breeders strive to explore and understand the characteristics of new resistance sources, one of which is the role of trichomes as a defense mechanism in plants. Trichomes are fine hair-like structures that cover the outermost part of the plant leaves, serving as both a physical and chemical barrier. Identification and quantification of trichomes are commonly performed manually by researchers. Such tasks are time-consuming, labor-intensive, and inefficient. Therefore, there is a pressing need for a system capable of automatically detecting and quantifying trichomes to replace manual methods and accelerate the process. This study employs a deep learning approach to train a model for detecting and quantifying trichome objects. The model architecture used is YOLOv8. The training results show a mean average precision (mAP) of 0.816 at a confidence threshold of 50 and a mAP of 0.38 at a confidence threshold of 90. This model is expected to assist experts and researchers in the agricultural field in identifying trichomes, thereby optimizing crop yields.

Keywords

grandular trichomes, potato, *deep learning, convolutional neural network,* YOLOv8

Pernyataan Keaslian Tulisan

Dengan ini saya menyatakan bahwa tesis ini merupakan tulisan asli dari penulis, dan tidak berisi material yang telah diterbitkan sebelumnya atau tulisan dari penulis lain terkecuali referensi atas material tersebut telah disebutkan dalam tesis. Apabila ada kontribusi dari penulis lain dalam tesis ini, maka penulis lain tersebut secara eksplisit telah disebutkan dalam tesis ini.

Dengan ini saya juga menyatakan bahwa segala kontribusi dari pihak lain terhadap tesis ini, termasuk bantuan analisis statistik, desain survei, analisis data, prosedur teknis yang bersifat signifikan, dan segala bentuk aktivitas penelitian yang dipergunakan atau dilaporkan dalam tesis ini telah secara eksplisit disebutkan dalam tesis ini.

Segala bentuk hak cipta yang terdapat dalam material dokumen tesis ini berada dalam kepemilikan pemilik hak cipta masing-masing. Apabila dibutuhkan, penulis juga telah mendapatkan izin dari pemilik hak cipta untuk menggunakan ulang materialnya dalam tesis ini.

Yogyakarta, November 2024

M. Fauzan Azhari

Daftar Publikasi

Publikasi yang menjadi bagian dari tesis

Azhari, M. F., Rohmatul Fajriyah, Izzati Muhimmah, Dan Jeric Arcega Rustia, Smulders, M. J., & Gracianna Devi, M. (2024). Detection and Quantification of Glandular Trichomes (Bulbous) on Potato Plant Leaf Images Using Deep Learning. Enthusiastic : International Journal of Applied Statistics and Data Science, 4(2), 96–108. https://doi.org/10.20885/enthusiastic.vol4.iss2.art2

Sitasi Publikasi 1

Kontributot	Jenis Kontribusi
M. Fauzan Azhari	Mendesain eksperimen (30%) Menulis paper (60%)
Rohmatul Fajriyah	Mendesain eksperimen (30%) Menulis dan mengedit paper (20%)
Izzati Muhimmah	Mendesain eksperimen (5%) Menulis dan mengedit paper (20%)
Dan Jeric Arcega Rustia	Mendesain eksperimen (5%)
Marinus J.M. Smulders	Mendesain eksperimen (5%)
Micha Gracianna Devi	Mendesain eksperimen (25%)

Halaman Kontribusi

Pada penelitian ini terdapat beberapa kontribusi dari pihak lain, yakni :

1. Dr. tech. Rohmatul Fajriyah, S.Si., M.Si. sebagai pembimbing pertama yang telah memberikan bimbingan dan masukan yang sangat berarti selama proses penyusunan tesis ini. Serta membantu penulis memahami konsep-konsep mendasar dalam penelitian ini, memberikan arahan strategis yang tepat, serta memotivasi penulis untuk terus mengembangkan kemampuan dalam bidang bioinformatik.
2. Izzati Muhimmah, S.T., M.Sc., Ph.D. sebagai pembimbing kedua yang telah memberikan pandangan kritis dan konstruktif terhadap penelitian ini. Serta telah banyak membantu penulis dalam memperbaiki kualitas penelitian melalui diskusi yang mendalam, serta memberikan perspektif yang lebih luas terkait implementasi dan dampak penelitian ini.
3. Plant Breeding, Wageningen University Research karena telah mendukung penelitian ini, baik melalui penyediaan data maupun bantuan teknis. Kontribusi yang diberikan sangat membantu penulis dalam mengembangkan model deteksi trikoma dan meningkatkan kualitas penelitian yang saya lakukan.

Halaman Persembahan

Segala puji dan syukur saya panjatkan ke hadirat Allah SWT atas limpahan rahmat, hidayah, dan kasih sayang-Nya sehingga saya dapat menyelesaikan tesis ini. Tiada daya dan upaya selain atas kehendak-Nya, segala proses yang saya lalui menjadi pembelajaran berharga dalam perjalanan akademik saya.

Dengan kerendahan hati, tesis ini saya persembahkan untuk Ayah, Ibu, dan Adik saya tercinta, terima kasih yang sebesar-besarnya atas dukungan moral, doa, dan semangat yang terus menerus diberikan selama proses penyusunan tesis ini. Kehadiran dan motivasi dari keluarga telah menjadi sumber kekuatan utama untuk menyelesaikan penelitian ini.

Semoga ilmu yang saya dapat selama menempuh studi ini dapat memberikan manfaat bagi dunia akademik, khususnya dalam bidang penelitian yang saya tekuni, serta menjadi amal kebaikan bagi semua pihak yang telah terlibat.

Kata Pengantar

Bismillahirrahmanirrahim,

Assalammualaikum Warrahmatullahi Wabarakatuh

Segala puji dan syukur saya panjatkan ke hadirat Allah SWT atas limpahan rahmat, karunia, dan hidayah-Nya, sehingga saya dapat menyelesaikan tesis ini dengan judul **" Deteksi & Kuantifikasi Trikoma Tipe Grandular (Bulbose) pada Citra Daun Tanaman Kentang Menggunakan Deep Learning"**. Penulisan tesis ini merupakan salah satu syarat untuk memperoleh gelar Magister pada Program Studi Informatika Program Magister, Fakultas Teknologi Industri, Universitas Islam Indonesia.

Penyelesaian tesis ini tidak lepas dari dukungan, bimbingan, dan kerja sama dari berbagai pihak. Oleh karena itu, pada kesempatan ini saya ingin menyampaikan ucapan terima kasih yang sebesar-besarnya kepada:

1. Bapak Fathul Wahid, S.T., M.Sc., Ph.D., selaku Rektor Universitas Islam Indonesia yang telah memberikan fasilitas dan dukungan akademik yang sangat baik selama saya menempuh studi di Universitas Islam Indonesia.
2. Bapak Prof. Dr. Ir. Heri Purnomo, MT., selaku Dekan Fakultas Teknologi Industri, Universitas Islam Indonesia yang selalu memastikan kelancaran proses akademik di lingkungan Fakultas Teknologi Industri, Universitas Islam Indonesia.
3. Bapak Irving Vitra Paputungan, S.T., M.Sc., Ph.D., selaku Ketua Program Studi Informatika Program Magister, Fakultas Teknologi Industri, Universitas Islam Indonesia yang memberikan arahan, dukungan dan memberikan motivasi kepada seluruh mahasiswa.
4. Ibu Dr. tech. Rohmatul Fajriyah, S.Si., M.Si. dan Ibu Izzati Muhimmah, S.T., M.Sc., Ph.D. selaku Dosen Pembimbing 1 dan Dosen Pembimbing 2, atas bimbingan, arahan, dan dukungan yang luar biasa selama proses penelitian hingga penyusunan tesis ini.
5. Jajaran Staf Program Studi Magister Informatika, yang selalu siap membantu dan memberikan pelayanan terbaik selama proses studi saya.
6. Rustia Dan, Micha dan Bapak Renne Smulders, rekan-rekan dari Plant Breeding, Wageningen University Research, yang telah bersedia menjadi mitra penelitian dan memberikan akses data dan masukan yang sangat berharga dalam penyusunan tesis ini.

7. Orang Tua dan Keluarga Saya, atas doa, cinta, dan dukungan moral maupun material yang tiada henti selama saya menempuh studi ini. Terima kasih kepada ayah, ibu, dan adik saya tercinta yang selalu menjadi sumber semangat saya.

Saya menyadari bahwa tesis ini masih jauh dari sempurna. Oleh karena itu, saya sangat mengharapkan kritik dan saran yang membangun untuk perbaikan di masa mendatang. Semoga hasil penelitian ini dapat memberikan manfaat bagi dunia akademik, khususnya dalam bidang yang saya tekuni, serta bagi masyarakat secara umum.

Akhir kata, saya mengucapkan terima kasih yang sebesar-besarnya kepada semua pihak yang telah membantu dan mendukung proses penyusunan tesis ini. *Wassalaamu'alaikum warahmatullaahi wabarokaatuh.*

Yogyakarta, November 2024

M. Fauzan Azhari

Daftar Isi

Daftar Tabel

Daftar Gambar

Glosarium

Activation Function	-	Fungsi yang beroperasi pada jaringan syaraf tiruan yang memberikan output pada setiap neuron
Backpropagation	-	Algoritma pada jaringan syaraf tiruan untuk meminimalkan nilai kesalahan pada suatu model dengan cara memperbaharui nilai bobot
Bounding Box	-	Kotak pembatas dalam pendeteksian objek pada arsitektur YOLO
AI	-	Artificial Intelligence
ANN	-	Artificial Neural Network
Akurasi	-	Ukuran ketepatan model dalam memprediksi
Batch Size	-	Jumlah sampe data yang disebarkan ke jaringan syaraf tiruan
CNN	-	Convolutional Neural Network
Convolutional Layer	-	Lapisan pada arsitektur CNN yang mengoperasikan filter atau kernel terhadap feature map
CSP	-	Corss-Stage Partial Network
C2F	-	Cross-Stage Feature Fusion
Data Test	-	Sampel data yang digunkaan untuk menguji hasil penelitian
Data Train	-	Sampel data yang digunakan untuk menjalankan fungsi dari sebuah algoritma
Deep Learning	-	Salah satu cabang ilmu dari machine learning
Epoch	-	Banyak langkah yang dilakukan dalam proses training pada CNN
Feature Map	-	Input yang digunakan pada sebuah layer yang didapatkan dari hasil perhitungan pada layer sebelumnya
Feed Forward	-	Proses komputasi dari input layer hingga didapat output pada jaringan syaraf tiruan
Flatten Layer	-	Layer untuk mengubah data dalam bentuk matriks menjadi vektor
Fully Connected Layer	-	Lapisan yang terhubung penuh satu sama lain sebagai proses klasifikasi/prediksi *neural network*
Hidden Layer	-	Layer pada jaringan syaraf tiruanyang terletak antara input layer dan output layer
Input Layer	-	Layer pertama dalam jaringan syaraf tiruan
Kernel	-	Matriks yang digunakan untuk memfilter matriks gambar input
Learning Rate	-	Parameter dari gradient descent

Loss	-	Nilai kesalahan atau error hasil prediksi
Nilai Confidence	-	Nilai kepercayaan yang menggambarkan seberapa besar probabilitas prediksi terhadap suatu objek
Weight	-	Parameter dalam artificial neural network
XML	-	Extensible Markup Language
YOLO	-	You Only Look Once

BAB 1
Pendahuluan

1.1. Latar Belakang

Indonesia sangat mengandalkan sektor pertanian untuk memegang peranan yang sangat penting dalam kesejahteraan masyarakat, sehingga budidaya tanaman hortikultura sangat banyak dilakukan salah satunya tanaman kentang. Kentang merupakan salah satu tanaman penghasil karbohidrat dan memiliki nilai gizi yang sangat baik, sehingga kentang menjadi salah satu tanaman yang banyak sekali dibudidayakan di Indonesia bahkan negara-negara di Eropa. Berdasarkan data dari Badan Pusat Statistik tahun 2022 kentang berada diurutan nomor 4 sebagai tanaman yang paling banyak diproduksi dan dibudidaya di Indonesia dengan total sebanyak 1,36 juta ton pada tahun 2022.

Gambar 1.1 Sayuran paling banyak diproduksi di Indonesia tahun 2022

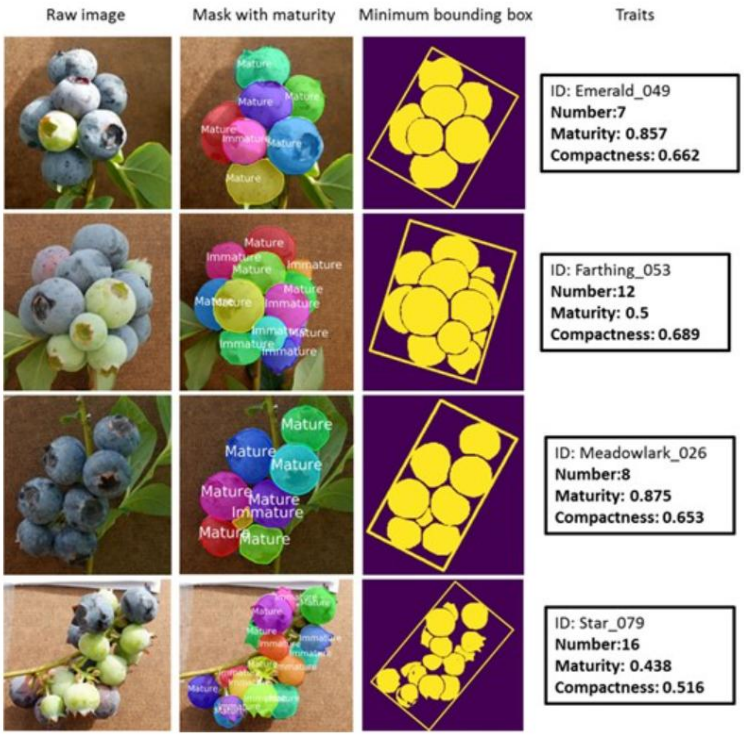
Supaya budidaya dan produksi tanaman kentang memiliki kualitas yang baik, banyak orang dari kalangan petani, peneliti hingga pemulia tanaman berusaha agar mengeksplorasi dan memahami karakteristik sumber resistensi baru, salah satunya yaitu peranan trikoma sebagai mekanisme pertahanan pada tanaman. Trikoma merupakan bagian berupa rambut-rambut halus yang melapisi bagian terluar pada daun tanaman yang berfungsi sebagai penghalang fisik dan kimia (Zhang, et al., 2020). Trikoma berperan penting sebagai mekanisme pertahanan tanaman terhadap cuaca ekstrim ataupun hama serangga, sehingga

peneliti dan pemulia tanaman berusaha untuk melakukan identifikasi kelenjar trikoma untuk dapat menghasilkan kualitas tanaman yang baik (Bergau, Bennewitz, Syrowatka, Hause, & Tissier, 2015). Identifikasi dan kuantifikasi pada trikoma biasa dilakukan peneliti dengan cara manual. Pekerjaan seperti ini tentunya akan menghabiskan banyak waktu dan tenaga serta tidak efektif. Maka dari itu kebutuhan akan adanya sistem cerdas sangat diperlukan agar identifikasi dan kuantifikasi pada trikoma tidak lagi dilakukan secara manual dan dapat dilakukan dalam waktu yang cepat.

Memahami permasalahan tersebut maka penting untuk dapat membangun sebuah sistem atau model berbasi *deep learning* untuk dapat melakukan identifikasi dan kuantifikasi trikoma secara otomatis. Dari model ini diharapkan dapat membantu pakar atau peneliti dibidang pertanian dalam melakukan identifikasi trikoma sehingga dapat mengoptimalkan hasil panen tanaman. Metode yang digunakan untuk membangun model yaitu salah satu algoritma *deep learning* yaitu *convolutional neural network* dengan arsitektur model YOLOv8.

You Only Look Once (YOLO) merupakan salah satu arsitektur varian dalam algoritma CNN. YOLO dapat menjalankan tugas *realtime object detection* dengan menyertakan lokaliasi yang dimana lokalisasi yang dihasilkan oleh YOLO merupakan kumpulan piksel yang memiliki probabilitas kelas yang paling tinggi. Proses lokalisasi dilakukan dengan menerapkan model YOLO ke dalam gambar pada tiap lokasi (piksel) lalu memberika nilai probabilitas pada gambar sebagai hasil untuk pendeteksian. Arsitektur YOLO menggunakan pendekatan yang sedikit berbeda dengan arsitektur varian CNN yang lain, dimana arsitektur YOLO menggunakan jaringan syaraf tunggal pada keseluruhan gambar dan mempartisi gambar menjadi wilayah yang kemudian memprediksi *bounding box* dan nilai probabilitas nya.

Penerapan *deep learning* seperti ini juga pernah diimplementasikan dalam penelitian terdahulu mengenai *digital phenotyping* pada tanaman. Seperti contoh penelitian yang dilakukan oleh (Ni, Li, Jiang, & Takeda, 2020) mengenai *digital phenotyping* pada buah blueberry yang terkait dengan daya dan hasil panen. Penelitian tersebut bertujuan untuk mengembangkan suatu *data processing pipeline* yang dapat menghitung jumlah buah berry, mengukur tingkat kematangan dan mengevaluasi hasil panen secara otomatis dengan menggunakan metode segmentasi gambar dan algoritma *deep learning* untuk mengklasifikasi empat kultivar buah blueberry.



Gambar 1.2 Contoh *deep learning* untuk *digital phenotyping* (Ni, Li, Jiang, & Takeda, 2020)

Berdasarkan latar belakang dan permasalahan tersebut, dengan menggunakan teknologi berbasi *deep learning* dan *computer vision* maka penulis akan mencoba membuat model *deep learning* yang dapat digunakan untuk mengidentifikasi dan juga kuantifikasi terhadap trikoma yang ada pada permukaan daun tanaman dengan menggunakan salah satu arsitektur varian CNN yaitu YOLOv8. Dari penelitian yang telah dilakukan ini harapannya dapat membantu peneliti, pemulia tanaman dan plantologis dalam melakukan identifikasi terhadap trikoma sehingga dapat dilakukan secara otomatis.

1.2. Rumusan Masalah

Berdasarkan latar belakang yang sudah dijelaskan sebelumnya, adapun rumusan masalah yang diangkat pada penelitian ini yaitu:

a. Bagaimana implementasi algoritma *Convolutional Neural Network* dengan mengunakan model arsitektur YOLO v8 dalam mendeteksi trikoma pada daun tanaman kentang.
b. Bagaimana parameter yang tepat untuk menghasilkan model yang baik.
c. Berapa nilai *mean average precision* (mAP) pada *confidence treshold* tertentu yang dihasilkan dari hasil *training* dengan menggunakan arsitektur YOLOv8.

1.3. Tujuan Penelitian

Adapun tujuan dari penelitian ini berdasarkan rumusan masalah yang sudah disebutkan adalah sebagai berikut:

a. Mengetahui implementasi algoritma *Convolutional Neural Network* dengan menggunakan model arsitektur YOLOv8 dalam mendeteksi trikoma pada daun tanaman kentang.
b. Mengetahui bagaimana parameter-parameter yang tepat untuk menghasilkan model yang baik.
c. Mengetahui nilai *mean average precision* (mAP) pada *confidence treshold* tertentu yang dihasilkan dari hasil *training* dengan menggunakan arsitektur YOLOv8.

1.4. Manfaat Penelitian

Adapun manfaat dari penelitian ini yaitu untuk membantu dan memudahkan pemulia tanaman, peneliti dibidang *plantologi* ataupun pihak-pihak terkait untuk dapat melakukan identifikasi dan kuantifikasi trikoma secara otomatis sehingga pekerjaan tersebut dapat dilakukan dalam waktu yang lebih cepat. Hasil penelitian ini juga diharapkan akan menjadi pionir atau tahapan awal penelitian deteksi dan kuantifikasi trikoma berbasis *deep learning*. Sehingga untuk pengembangan penelitian yang akan datang diharapkan dapat melakukan identifikasi terhadap daun tanaman lain termasuk tanaman yang liar, selain itu identifikasi juga dapat dilakukan untuk tipe trikoma yang lebih banyak.

1.5. Batasan Penelitian

Agar penelitian yang dilakukan tidak terlalu melebar dari topik permasalahan, maka penulis memberikan batasan-batasan tertentu sebagai pedoman dalam melakukan penelitian. Adapun batasan-batasan penelitian antara lain sebagai berikut:

a. Data yang digunakan merupakan sampel gambar mikroskopis pada daun tanaman kentang hasil budidaya (bukan tanaman liar). Sampel tanaman budidaya cenderung memiliki kepadatan trikoma yang rendah sehingga lebih mudah untuk dilakukan pelabelan.
b. Klasifikasi yang dilakukan yaitu terhadap trikoma *grandularis bulbose*. Trikoma ini memiliki bentuk bulat atau bulosa yang unik, dengan kepala trikoma yang mengandung sel-sel sekretori yang dapat memproduksi dan menyimpan berbagai

senyawa metabolit sekunder. Pemilihan tipe trikoma tersebut berdasarkan ketersediaan data yang diperoleh oleh peneliti.

c. Model klasifikasi yang dibangun yaitu menggunakan metode *Convolutional Neural Network* dengan arsitektur varian yaitu YOLO v8.
d. Bahasa pemrograman yang digunakan untuk membangun model klasifikasi yaitu menggunakan bahasa pemrograman *Python* dan *lybrary Ultralytics.*

1.6. Sistematika Penulisan

Penulisan tesis ini disusun secara sistematis untuk memberikan gambaran yang terstruktur mengenai proses penelitian yang dilakukan. Sistematika penulisan tesis ini terdiri dari beberapa bab, yaitu sebagai berikut:

BAB 1: Pendahuluan

Bab ini berisi latar belakang penelitian, rumusan masalah, tujuan penelitian, manfaat penelitian, batasan penelitian, serta sistematika penulisan. Pendahuluan memberikan landasan awal mengenai pentingnya penelitian yang dilakukan.

BAB 2: Tinjauan Pustaka

Bab ini mencakup kajian literatur dan teori-teori yang relevan dengan penelitian, serta hasil-hasil penelitian sebelumnya yang mendukung penelitian ini. Bab ini juga memuat landasan teori yang digunakan untuk membangun hipotesis atau model dalam penelitian.

BAB 3: Metodologi Penelitian

Bab ini menjelaskan metode penelitian yang digunakan, termasuk desain penelitian, metode pengumpulan data, teknik analisis data, dan prosedur penelitian. Penjelasan dalam bab ini bertujuan agar penelitian dapat direplikasi oleh pihak lain.

BAB 4: Hasil dan Pembahasan

Bab ini memaparkan hasil penelitian yang diperoleh serta pembahasannya berdasarkan analisis data. Selain itu pada bab ini juga memaparkan tentang implementasi hasil model yang telah dibangun.

BAB 5: Kesimpulan dan Saran

Bab terakhir ini menyajikan kesimpulan dari penelitian yang telah dilakukan sesuai dengan tujuan penelitian. Selain itu, disampaikan pula saran-saran yang relevan berdasarkan hasil penelitian untuk pengembangan lebih lanjut atau aplikasi praktis.

Setiap bab dalam tesis ini disusun secara runtut dan logis sehingga pembaca dapat dengan mudah memahami proses dan hasil penelitian yang dilakukan. Sistematika ini diharapkan mampu memberikan gambaran yang jelas mengenai keseluruhan penelitian, mulai dari latar belakang hingga kesimpulan.

BAB 2
Tinjauan Pustaka

2.1. Penelitian Terdahulu

Dalam melakukan penelitian, tentunya referensi dibutuhkan untuk memperkuat teori-teori dan membantu pemahaman penulis dalam melakukan penelitian. Beberapa referensi dan penelitian yang berkaitan dengan *plantologi* / trikoma dan juga mengenai *image processing* dan *deep learning* telah penulis rangkum guna membantu penulis dalam melakukan penelitian.

Penelitian pertama yaitu penelitian mengenai perkembangan trikoma tipe VI pada daun tanaman tomat *solanum lycopersicum* dan spesies liar terkait *S. habrochaites* (Bergau, Bennewitz, Syrowatka, Hause, & Tissier, 2015). Penelitian ini menyebutkan bahwa trikoma kelenjar tipe VI merupakan jenis trikoma yang paling banyak terdapat pada daun daun dan batang tanaman tomat dan secara signifikan berkontribusi terhadap resistensi herbivora, terutama pada spesies liar. Penelitian ini bertujuan untuk mempelajari lebih lanjut mengenai perkembangan trikoma tipe VI pada daun dan batang tanaman tomat dengan menggunakan berbagai teknik pencitraan sel. Hasil dari penelitian ini menjelaskan bahwa morfologi adaptif trikoma tipe VI untuk penyimpanan metabolit, lalu menghasilkan dan menyediakan kerangka untuk penelitian yang akan datang untuk penghasil metabolik seluler yang sangat penting. Hal ini diperlukan untuk lebih memanfaatkan potensi trikoma khususnya untuk pemuliaan ketahanan hama pada tomat.

Penelitian kedua berjudul "*Peran Berbagai Jenis Trikoma dalam Tomat Untuk Ketahanan Terhadap Dingin, Kekeringan, Hama Lalat Putih dan Botrytis."* (Zhang, et al., 2020). Cuaca kering/dingin yang ekstrim, efek penyakit dan serangga hama dapat mengurangi produksi tomat dalam skala besar. Penelitian ini ingin membuktikan bahwa trikoma dapat berperan penting dalam melindungi tanaman dari kerusakan melalui struktur fisiknya. Kepadatan dan jenis trikoma yang berbeda dapat berpengaruh terhadap ketahanan stress dari tanaman tomat. Karakteristik kelenjar, jenis dan kepadatan trikoma diamati menggunakan pemindaian *scanning electron microscope* (SEM) dan *transmission electron microscop* (TEM) untuk mengatur struktur subseluler pada daun. Hubungan antara trikoma tomat yang berbeda dan ketahanan stress diamati dengan menggunakan perlakuan terhadap suhu, kekeringan, penyakit dan serangga. Hasil dari penelitian menyebutkan bahwa tomat dengan kepadatan tinggi tipe 1 dan trikoma nonglandular tipe II lebih tahan terhadap stress

akibat cuaca dingin/kering yang ekstrim, trikoma nonglandular tipe III banyak dan memiliki rata-rata yang lebih tinggi dapat menghasilkan resistensi terhadap patogen, kemudian dengan ukuran dan kepadatan trikoma yang lebih tinggi, terutama kelenjar trikoma tipe I dan IV membuat tanaman tomat lebih tahan terhadap hama kutu.

Penelitian selanjutnya yaitu mengenai perbedaan ketahanan terhadap hama serangga antara spesies tomat endemik di Kepulauan Galapagos. Dalam penelitian ini mencoba untuk melihat perbedaan karakteristik dan ketahanan terhadap serangga antara 2 jenis tomat endemik yaitu *solanum galapagense* dan *S. cheesmaniae*. Penelitian dilakukan dengan melakukan karakterisasi pada 12 aksesi *solanum galagapense*, 22 aksesi *S. cheesmaniae* dan satu aksesi *S. lycopersicum* sebagai referensi untuk resistensi terhadap kutu kebul. Resistensi atau ketahanan terhadap kutu kebul hanya ditemukan pada spesies *solanum galapagense* yang memiliki kadar gula yang relatif tinggi dan adanya kelenjar trikoma tipe I dan IV (Lucatti, Heusden, Vos, Visser, & Vosman, 2013).

Kemudian terdapat penelitian mengenai pemetaan QTL terhadap resistensi dan ketahanan serangga dari *solanum galapagense.* Ketahanan tanaman inang sangat penting, namun penggunaan insektisida yang berlebihan juga dilarang karena alasan lingkungan. Pada penelitian ini menjelaskan bahwa tanaman tomat memiliki resistensi yang cukup tinggi terhadap serangga karena keberadaan kelenjar trikoma dan komposisi fitokimia. (Vosman, et al., 2018). Diakhir penelitian diperoleh hasil bahwa QTL utama di akhir kromosom 2 dari *S. Galapagense* terlibat dalam resistensi terhadap dua spesies kutu kebul. QTL ini mengatur produksi trikoma kelejar tipe IV serta produksi metabolit dalam jumlah besar.

Untuk penelitian-penelitian berikutnya merupakan penelitian pada bidang *image processing* dan *deep learning*. Penelitian pertama berjudul "*Deep Learning untuk Klasifikasi Glaukoma dengan menggunakan Arsitektur EfficientNet*". Penelitian ini berusaha untuk membuat sistem yang dapat melakukan klasifikasi glaukoma dalam 5 kelas yaitu *deep, early, moderate, OHT* dan *normal*. Hasil penelitian didapatkan model terbaik dengan menggunakan *hyperparameter optimizer adamax, learning rate* sebesar 0,01, nilai *epoch* sebesar 20 dan nilai *batch size* sebesar 32. Untuk nilai akurasi yang diperoleh yaitu sebesar 1,000 (Perdani, Magdalena, & Pratiwi, 2022).

Penelitian berikutnya berjudul "*Klasifikasi Citra Plasmodium Penyebab Penyakit Malaria dalam Sel Darah Merah Manusia dengan Menggunakan Metode Multi Class Support Vector Machine (SVM)*". Penelitian ini bertujuan untuk membuat model klasifikasi penyakit malaria berdasarkan citra *plasmodium falciparum*. Data citra diekstraksi menggunakan metode *filter gabor* dan *multi class SVM.* Dari penelitian yang telah dilakukan

diperoleh hasil nilai akurasi sebesar 73,33% dengan skema 120 citra untuk *data train* dan 30 citra untuk *data test* (Banyal, Surianti, & Dayat, 2016).

Penelitian selanjutnya yaitu "*Klasifikasi Patologi Makula Retina Melalui Citra OCT menggunakan Convolutional Neural Network dengan Arsitektur MobileNet*". Penelitian ini bertujuan untuk merancang sistem klasifikasi kondisi patologi makula retina dengan menggunakan algoritma *Convolutional Neural Network* dan arsitektur *MobileNet*. Hasil penelitian diperoleh hasil klasifikasi terbaik arsitektur *mobileNet* yaitu dengan menggunakan *optimizer Adam* dengan nilai akurasi sebesar 95,67% untuk data *training*, 92,04% untuk data *testing*, dan nilai *loss* sebesar 0,135 untuk data *training* dan 0,299 untuk data *testing* (Zakiya, Ledya, & Rizal, 2021).

Penelitian berikutnya yaitu "*Deep Learning untuk Klasifikasi Diabetic Retinopathy menggunakan Model EfficientNet*". Penelitian ini berusaha untuk merancang sebuah sistem untuk mendeteksi *diabetic retinopathy* menggunakan algoritma *deep learning* dan *convolutional neural network*. Model yang digunakan yaitu *EfficientNet model* dan diperoleh hasil akurasi sebesar 79,8% dan dapat melakukan klasifikasi terhadap 5 tingkat penyakit *diabetic retinopathy* (Rizal, Ibrahim, Pratiwi, Saidah, & Fu'adah, 2020).

Penelitian selanjutnya yaitu mengenai deteksi sel malaria pada sediaan apus darah berdasarkan fitur morfologi menggunakan jaringan *backpropagation*. Pada penelitian ini bertujuan untuk membuat model deteksi sediaan apus darah sehingga mampu mengenali gejala malaria secara dini. Tahapan yang dilakukan dimulai dari *preprocessing*, segmentasi untuk membagi warna latar pada data gambar, ekstraksi fitur morfologi dan tekstur dan klasifikasi *bakcpropagation*. Akhir penelitian diperoleh hasil nilai akurasi sebesar 85,56% (Hamid, Suratin, & Usman, 2021).

Rangkuman tinjauan pustaka dari beberapa referensi penelitian terdahulu dapat dilihat pada **Error! Reference source not found.** dan **Error! Reference source not found.** be rikut.

Tabel 2.1 Ulasan kritis penelitian trikoma.

Pustaka	Judul	Objek	Metode	Hasil
(Bergau, Bennewitz, Syrowatka, Hause, & Tissier, 2015)	*The development of type VI glandular trichomes in the cultivated tomato Solanum lycopersicum and a related wild*	Trikoma daun dan batang tanaman tomat	Teknik Pencitraan Sel	Hasil dari pengamatan menjelaskan morfologi adaptif untuk penyimpanan metabolit.

	species S. habrochaites			
(Zhang, et al., 2020)	*The Roles of Different Types of Trichomes in Tomato Resistance to Cold, Drought, Whiteflies, and Botrytis*	Trikoma tanaman tomat	Observasi menggunakan *scanning electron microscope*	Hubungan antara trikoma Pada tanaman tomat dapat mempengatuhi ketahanan terhadap suhu, kekeringan, penyakit dan serangga
(Lucatti, Heusden, Vos, Visser, & Vosman, 2013)	*Differences in insect resistance between tomato species endemic to the Galapagos Islands*	Trikoma *S.Galapagense* dan *S.Cheesmaniae*	Eksperimen Karaterisasi	Hasil penelitian menunjukan bahwa *S. Galapagense* dan *S. Cheesmaniae* merupakan morfotipe dari pada dua spesies dan ko-eksistensi merupakan hasil dari tekanan selektif.
(Vosman, et al., 2018)	*Broad spectrum insect resistance and metabolites in close relatives of the cultivated tomato*	*trialerodes vaporariorum, myzus persicae, frankliniella occidentalis* dan *spodoptera exigua*.	Observasi *plantology*	Perbedaan yang signifikan dalam metobolom daun ditemukan antara *S. Galapagense*, mengandung trikoma tipe IV dan yang paling dekat relatif *S. Cheesmaniae* yang tidak memiliki trikoma tipe IV.
(Vosman, et al., 2019)	*QTL mapping of insect resistance components of Solanum galapagense*	Trikoma *solanum galapagense*	Observasi *plantology*	Diperoleh hasil bahwa QTL utama di akhir kromosom 2 dari *S. Galapagense* terlibat dalam resistensi terhadap dua spesies kutu kebul. QTL ini mengatur produksi trikoma kelejar tipe IV serta produksi metabolit dalam jumlah besar.

Tabel 2.2 Ulasan kritis penelitian *deep learning* / *image processing*.

Pustaka	**Judul**	**Objek**	**Metode**	**Hasil**
(Hamid, Suratin, & Usman, 2021)	Deteksi Sel Malaria Pada Sediaan Apus Darah Berdasarkan Fitur Morfologi dan Tekstur Menggunakan Jaringan *Backpropagation*	Sel Malaria	Jaringan *backpropagation*	Hasil akurasi diperoleh sebesar 85,56% dalam pengenalan fitur citra sel malaria
(Rizal, Ibrahim, Pratiwi, Saidah, & Fu'adah, 2020)	*Deep Learning* untuk Klasifikasi *Diabetic Retinopathy* menggunakan Model *EfficientNet*	Citra penyakit *Diabetic Retinopathy*	*Convolutional Neural Network* dengan arsitektur model *EfficientNet*	Dari penelitian yang telah dilakukan diperoleh hasil berupa akurasi 79,8% dalam mengklasifikasi penyakit *diabetic retinopathy*
(Perdani, Magdalena, & Pratiwi, 2022)	*Deep Learning* untuk Klasifikasi *Glaukoma*	Citra Glaukoma	*Convolutional Neural Network* dengan arsitektur	Hasil penelitian diperoleh akurasi, presisi, *recall* dan *F1-score* masing-masing

	dengan menggunakan Arsitektur *EfficientNet*		model *EfficientNet*	mencapai nilai 1,0000 dengan parameter yang digunakan yaitu *optimizer adamax, learning rate* 0,001, *epoch* 20 dan *batch size* 32
(Hanin, Patmasari,, & Fu'adah, 2021)	Sistem Klasifikasi Penyakit Kulit Menggunakan *Convolutional Neural Network* (CNN)	Citra penyakit kulit	*Basic Convolutional Neural Network*	Diperoleh hasil akurasi dan *loss* berturut-turut yaitu sebesar 96% dan 0,2486.
(Ni, Li, Jiang, & Takeda, 2020)	*Deep learning image segmentation and extraction of blueberry fruit traits associated with harvestability and yield*	Citra buah blueberry	R-CNN	Rata-rata nilai presisi untuk dataset validasi dan pengujian masing-masing adalah 78,3% dan 71,6%
(Suyuti, 2023)	Pengembangan Model Klasifikasi Mata Tertutup Dan Terbuka Dalam Identifikasi Kelelahan Menggunakan Arsitektur *Mobile CNN*	Citra mata tertutup dan terbuka	*Mobile CNN*	Dari penelitian yang dilakukan diperoleh hasil nilai presisi, *recall* dan *f1 score* masing-masing sebesar 1,00. Arsitektur *mobileNetV1* menhasilkan model yang lebih baik dibandingkan *MobileNetV3 Large* dan *EfficientNet Lite* 0
(Pailus, 2022)	Pengembangan Model Identifikasi Penyakit Pada Tanaman Padi Berbasiskan *Faster Rcnn*	Penyakit pada tanaman padi	*Faster RCNN*	Dari penelitian yang dilakukan diketahui model dapat mendeteksi penyakit pada tanaman padi dengan hasil pengujian mAP 87,6% dan *recall* 85,7%. Selain itu model juga dapat mendeteksi lebih dari satu penyakit pada satu tanaman padi.
(Miftakhurrokhmat, 2023)	Presensi Kelas Berbasis Pola Wajah Dan Tersenyum Menggunakan *Deep Learning*	Pola wajah tersenyum	*Deep Learning* dengan Arsitektur *FaceNet* dan SVM untuk klasifikasi	Dari penelitian yang telah dilakukan diperoleh nilai akurasi sebesar 92,9% dari model *testing* dan 66,7% dari *live testing*. *live testing* dinilai mengalami *overfitting* sehingga mengakibatkan nilai akurasi nya menurun.
(Khatami, 2022)	Deteksi Kendaraan Menggunakan Algoritma *You Only Look Once* (YOLO) V3	Deteksi Kendaraan	Algoritma YOLO V3	Bobot jaringan telah diuji pada *dataset* rekaman lalu lintas dari website ATCS (*Area Traffic Control System*) Kota Bandung dan rata-rata performa jaringan mencapai lebih dari 70%. Hal ini menunjukkan kemampuan jaringan dapat mendeteksi kendaraan pada dataset penulis dengan baik.

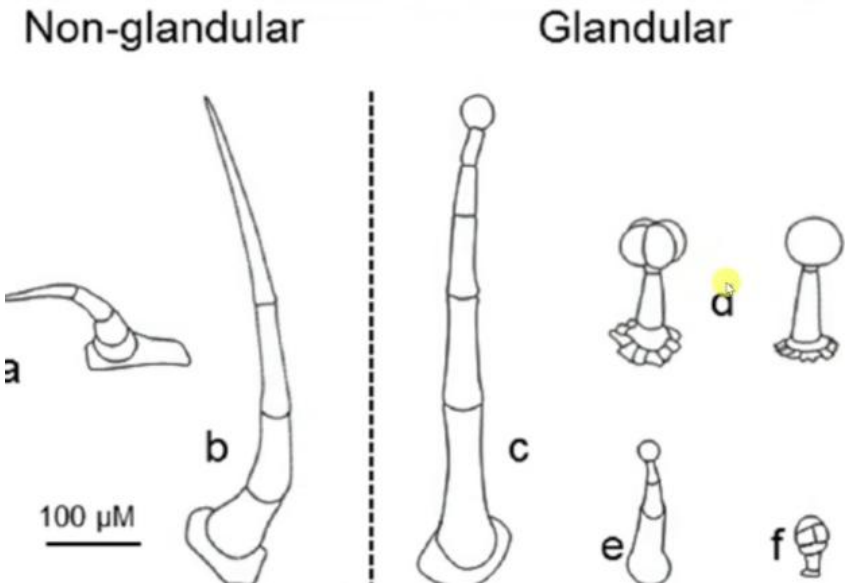
(Amwin, 2021)	Deteksi dan Klasifikasi Kendaraan Berbasis Algoritma *You Only Look Once* (YOLO)	Deteksi Kendaraan	Algoritma YOLO	Algoritma *You Only Look Once* (YOLO) terbukti dapat mengenali objek pada video CCTV yang dipasang di Simpang Air Mancur-Imanuel Kota Medan dengan menggunakan *pre-trained weights* yang telah dilatih sendiri. Hasil penelitian menunjukkan bahwa nilai *mean Average Precision* (mAP) yang dihasilkan mencapai 99,35%.
(Bella, 2021)	Implementasi Algoritma *Deep Learning* Untuk Sistem Deteksi Kantuk Pada Pengemudi Menggunakan YOLO	Deteksi kantuk pada pengemudi	Algoritma CNN dan YOLO	Dalam melakukan konfigurasi parameter model, digunakan *batch size* sebesar 64, *network size* sebesar 416x416, *subdivisions* sebesar 16, *max batch* sebesar 4000, dan filters sebesar 21. Ada empat skenario *training* data dan *testing* data yang digunakan dengan *learning rate* 0.00261 dan 0.001, serta *split dataset* sebesar 90%:10% dan 80%:10%. Dari hasil konfigurasi tersebut, didapatkan nilai IoU terbesar pada konfigurasi dengan split dataset sebesar 80%:20% dengan *learning rate* sebesar 0.00261.

2.2. Landasan Teori

2.2.1 Trikoma

Trikoma adalah struktur yang ditemukan pada tumbuhan dan hewan yang terdiri dari rambut halus yang muncul dari permukaan epidermis. Fungsi trikoma bervariasi tergantung pada jenis dan lokasi tumbuhan atau hewan di mana trikoma tersebut ditemukan. Pada tumbuhan, trikoma dapat melindungi tanaman dari serangan serangga atau hewan herbivora dan membantu mengurangi hilangnya air melalui evaporasi. Selain itu, trikoma juga dapat berfungsi sebagai alat sensor yang sensitif terhadap lingkungan, termasuk suhu dan kelembaban. Beberapa penelitian juga telah menunjukkan bahwa trikoma pada tanaman dapat berperan dalam menghasilkan senyawa-senyawa penting seperti minyak atsiri dan flavonoid. Sementara pada hewan, trikoma dapat berfungsi sebagai alat pertahanan diri dengan menghasilkan racun atau berperan dalam penglihatan seperti pada mata lalat. Studi tentang trikoma sangat penting karena dapat memberikan wawasan tentang adaptasi

organisme terhadap lingkungannya dan potensi penggunaan trikoma sebagai sumber senyawa bioaktif. (Sumber: Martin et al., 2014).



Gambar 2.1 Macam-macam tipe trikoma

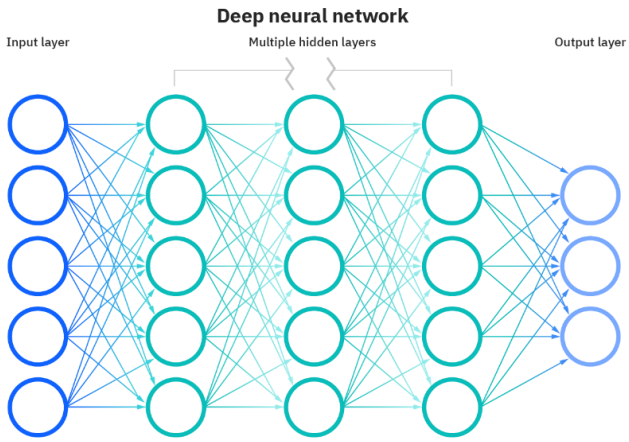
2.2.2 *Deep Learning*

Deep learning merupakan salah satu turunan ilmu dari *machine learning*. Perkembangan *deep learning* di industri modern sudah sangat banyak diimplementasikan seperti *self driving car, image/text generator, chat bot* dan lain-lain. Pada dasarnya *deep learning* berawal dari *neural network* hanya saja pada *deep learning* memiliki lapisan atau *layer* yang sangat banyak.

Deep learning memiliki kemampuan untuk belajar dan beradaptasi pada data yang tidak terstruktur seperti gambar dan audio. Meskipun *deep learning* diteorikan pertama kali pada tahun 1980-an, namun baru saat ini mulai populer, hal ini dikarenakan *deep learning* membutuhkan data yang sudah dilabeli dalam jumlah yang besar, Selain itu untuk membuat model *deep learning* dibutuhkan daya komputasi yang besar dan teknologi GPU dengan performa tinggi baru banyak digunakan di era saat ini (Pailus, 2022).

Model *deep learning* disebut juga sebagai *deep neural network* karena menggunakan arsitektur jaringan saraf. Gambar 3.3 menunjukkan *node*/*cell* yang terdiri dari tiga bagian yaitu input *layer*, *hidden layer*, dan *output layer*. *Hidden layer* memiliki kedalaman yang lebih dalam dibandingkan *input* dan *output layer*. Kedalaman tersebut biasanya mengacu pada jumlah lapisan tersembunyi dalam jaringan saraf. Jaringan saraf tradisional umunya memiliki 2-3 lapisan tersembunyi, namun pada *deep neural network* dapat memiliki hingga

ratusan lapisan tersembunyi. Dalam hal ini, suatu *neural network* dikatakan sebagai *deep learning* apabila memiliki lebih dari satu *hidden layer* (Pailus, 2022).

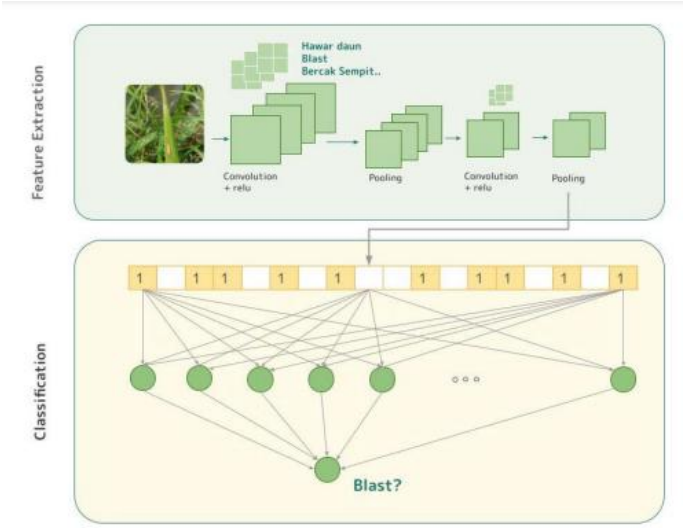


Gambar 2.2 Arsitektur *deep learning*

2.2.3 CNN (*Convolutional Neural Network*)

Convolutional Neural Network (CNN) merupakan salah satu jenis *neural network* yang sering digunakan dalam pengolahan data gambar. Perbedaan antara CNN dengan *neural network* standar terletak pada tahapan *pre-processing* data yang disebut *feature learning/feature extraction*. *Feature learning* pada CNN dilakukan untuk mengubah citra gambar menjadi *feature* dalam bentuk angka dari nilai piksel agar dapat diproses oleh komputer. Dalam *feature learning* terdapat Beberapa proses atau *layer* yaitu *convolution layer*, *pooling layer* dan *fully connected layer*.

Setelah proses *feature learning*, nilai-nilai piksel dikonversi menjadi bentuk vektor untuk digunakan sebagai *input* untuk algoritma *neural network*. Oleh karena itu, algoritma pada CNN dapat dibagi menjadi dua bagian besar, yaitu *feature extraction* (*pre-processing*) dan *neural network* (klasifikasi).



Gambar 2.3 Arsitektur CNN

2.2.4 Ultralytics

Ultralytics merupakan perusahaan teknologi yang berfokus dalam pengembangan solusi *computer vision*, terutama dalam hal deteksi objek dan segmentasi gambar. Ultralytics dikenal berkat kontribusinya dalam mengembangkan algoritma YOLO (*You Only Look Once*), yang memainkan peran krusial dalam kemajuan teknik deteksi objek yang efisien dan presisi. Algoritma YOLO yang dirancang oleh Ultralytics bertujuan untuk mendeteksi objek secara *real-time*, sebuah fitur yang sangat penting untuk berbagai aplikasi seperti kendaraan otonom, pengawasan keamanan, analisis video, dan lain sebagainya.

Ultralytics secara berkelanjutan telah mengembangkan dan memperbarui algoritma YOLO dengan versi-versi terbaru seperti YOLOv8, yang menunjukkan peningkatan dalam akurasi, efisiensi, dan kecepatan pemrosesan. Salah satu keunggulan utama algoritma YOLO adalah kemampuannya memproses gambar dalam satu kali pemrosesan (one-shot), menjadikannya jauh lebih cepat dibandingkan metode deteksi objek tradisional yang memerlukan beberapa tahap pemrosesan. Selain mengembangkan algoritma, ultralytics juga menawarkan berbagai sumber daya, termasuk *source code*, *library*, dokumentasi, dan tutorial, untuk mendukung pengembang dan peneliti dalam menerapkan teknologi deteksi objek ke berbagai macam bidang.

2.2.5 YOLO (*You Only Look Once*)

YOLO atau *You Only Look Once* merupakan salah satu algoritma deteksi objek berbasis *deep learning* yang pertama kali dikembangkan oleh Redmon et al. pada tahun 2015. Algoritma YOLO menggunakan metode Convolutional Neural Network (CNN) untuk

melakukan deteksi objek. Sesuai dengan namanya algoritma YOLO hanya menggunakan satu lapisan jaringan syaraf (*neural network*) pada citra. Jaringan ini membagi citra menjadi beberapa daerah (regions) dan memprediksi *bounding boxes* serta probabilitas setiap daerah secara bersamaan (Zou, Chen, Shi, Guo, & Ye, 2019).

Algoritma YOLO melakukan pembagian citra masukan ke dalam beberapa sel grid berukuran $s \times s$. Jika pusat objek pada citra terletak di dalam salah satu sel grid, maka sel grid tersebut akan bertanggung jawab untuk mendeteksi objek tersebut. Setiap sel grid akan memprediksi *B bounding box* dan *confidence score* untuk setiap *bounding box*. *Confidence score* berisi nilai seberapa yakin (*confident*) model terhadap kemungkinan keberadaan objek dalam *bounding box* dan akurasi dari prediksi *bounding box* tersebut (Khatami, 2022). untuk menghitung nilai *confident* sebuah *bounding box* dapat menggunakan persamaan berikut.

$$confident = P_r(object) \times IOU_{pred}^{truth} \quad (1)$$

Dengan:

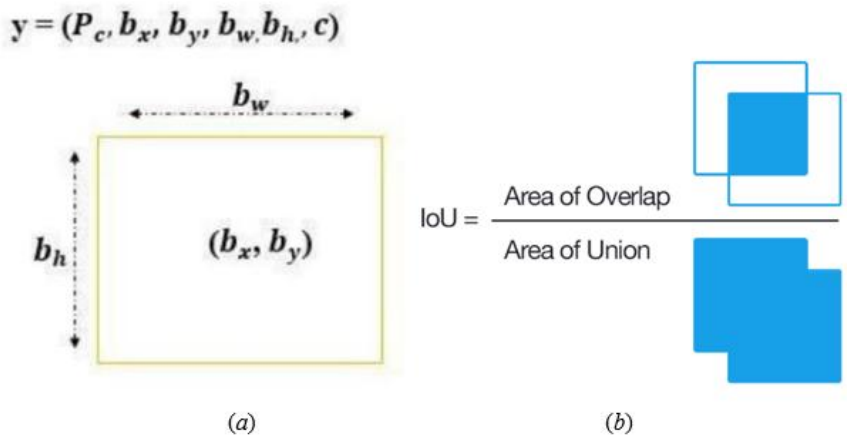
$confidence$: nilai *confidence score.*

$P_r(object)$: probabilitas objek, probabilitas bahwa sebuah objek terdeteksi dalam bounding box tersebut.

IOU_{pred}^{truth} : *Intersection over union,* perbandingan antara luas area yang tumpang tindih antara prediksi bounding box dan ground truth bounding box dengan luas gabungan dari kedua bounding box tersebut.

Confidence score dapat dihitung dengan menggunakan nilai *Intersection Over Union* (IOU) antara *bounding box* yang diprediksi dan *ground truth box*, yang dibuat secara manual saat proses pelabelan dan mengandung informasi kelas dan lokasi objek dengan akurat. Jika tidak ada objek yang terdeteksi di dalam sel grid, maka nilai *confidence* adalah nol. Selain itu, setiap sel grid juga memprediksi probabilitas dari C kelas kondisional (Khatami, 2022).

Setiap *bounding box* memiliki lima nilai prediksi yang terdiri dari x, y, w, h, dan *confidence score*. Nilai x dan y merepresentasikan koordinat titik pusat *bounding box*, sedangkan nilai w dan h masing-masing adalah lebar (*width*) dan tinggi (*height*) *bounding box*. *Confidence score* pada *bounding box* merepresentasikan nilai *Intersection Over Union* (IOU) antara *bounding box* yang diprediksi dengan *ground truth box*. IOU atau *Intersection Over Union* sendiri adalah nilai perbandingan luas area yang tumpang tindih antara area prediksi *bounding box* dan area *ground truth box* dengan luas area gabungan prediksi *bounding box* dan area *ground truth box* (Khatami, 2022).



Gambar 2.4 (a) *bounding box*, (b) *intersection over union*

Dari perbandingan tersebut, akan diperoleh nilai IOU yang berada pada rentang 0 hingga 1. Apabila nilai IOU semakin mendekati 1, maka dapat dikatakan bahwa *bounding box* yang diprediksi memiliki akurasi lokalisasi yang tinggi. Untuk memperoleh probabilitas kemunculan sebuah objek dalam suatu *bounding box* dan seberapa cocok *bounding box* tersebut membatasi objek, YOLO mengalikan nilai probabilitas kelas kondisional dengan nilai prediksi *confidence* sebuah *bounding box*. Persamaan (2) berikut ini menunjukkan cara YOLO menghitung probabilitas tersebut.

$$P_r(Class_i|Object) \times P_r(Object) \times IOU_{pred}^{truth} = P_r(Class_i) \times IOU_{pred}^{truth} \quad (2)$$

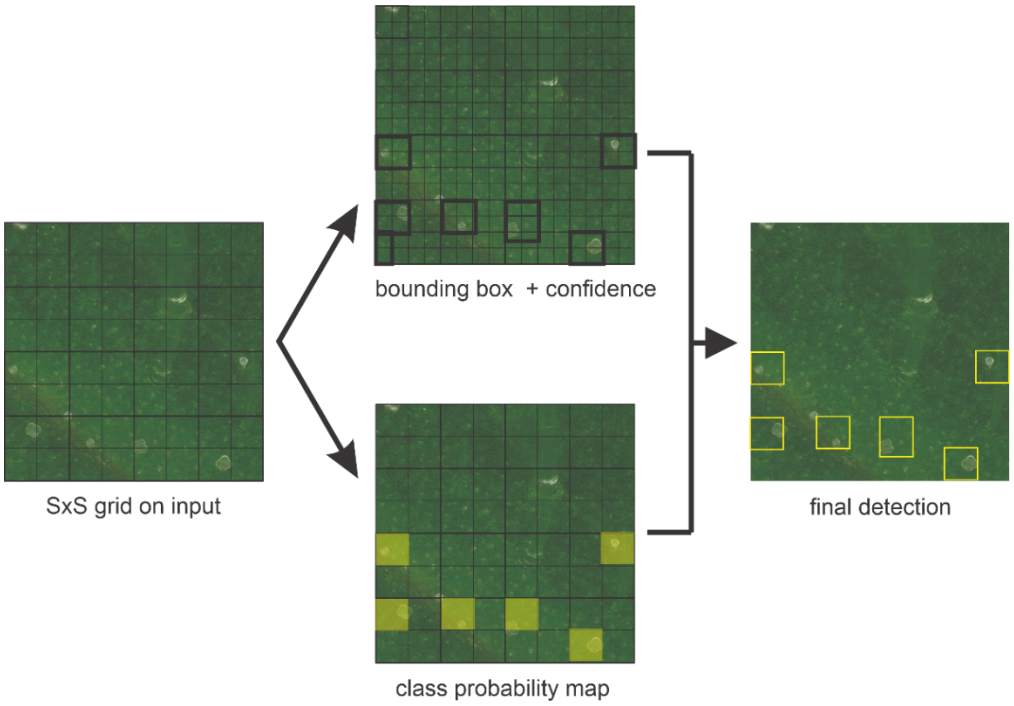
$$Ukuran\ Tensor = S \times S \times (B \times 5 + C) \quad (3)$$

Dengan:

$P_r(Class_i\|Object)$	:	Probabilitas kondisional bahwa objek yang terdeteksi adalah dari kelas objek i, diberikan adanya objek dalam *bounding box*.
$P_r(Object)$	:	Probabilitas bahwa ada objek dalam *bounding box*.
IOU_{pred}^{truth}	:	Nilai *intersection over union* antara prediksi bounding box dan *ground truth bounding box.*
$P_r(Class_i)$	:	Probabilitas bahwa objek yang terdeteksi adalah dari kelas objek i, tanpa mempertimbangkan adanya objek dalam *bounding box*.
S	:	Ukuran spatikal dari grid.
B	:	Jumlah *bounding box* yang akan diprediksi oleh setiap sel grid.
C	:	Jumlah kelas objek yang ingin dideteksi oleh model.

Hasil prediksi deteksi objek algoritma YOLO direpresentasikan sebagai sebuah tensor dengan ukuran yang dinyatakan dalam persamaan (3). Dalam persamaan tersebut, *S* merupakan ukuran sel grid, *B* mewakili jumlah *bounding box* yang diprediksi, dan C adalah jumlah kelas (Khatami, 2022).

Nilai *confidence* digambarkan dengan ketebalan garis pada *bounding box* sehingga apabila garis *bounding box* semakin tebal maka semkain tinggi nilai *confidence* pada grid tersebut dan juga sebaliknya. Apabila tidak terdapat objek atau kelas yang terdeteksi pada suatu grid maka nila *confidence* sama dengan 0.



Gambar 2.5 Cara kerja algoritma YOLO

2.2.6 *Corss-Stage Partial Network* (CSP) & *Cross-Stage Feature Fusion* (C2F)

Pada YOLOv8 memiliki arsitektur dengan *cross-stage partial network* (CSP) dan modul *cross-stage feature fusion* (C2f) yang dirancang untuk meningkatkan ekstraksi fitur, terutama untuk objek yang berukuran kecil. Arsitektur *cross-stage partial network* dan modul *cross-stage feature fusion* telah diterapkan dalam beberapa model YOLO sebelumnya, seperti YOLOv4, YOLOv5 dan dikembangkan lagi pada YOLOv8.

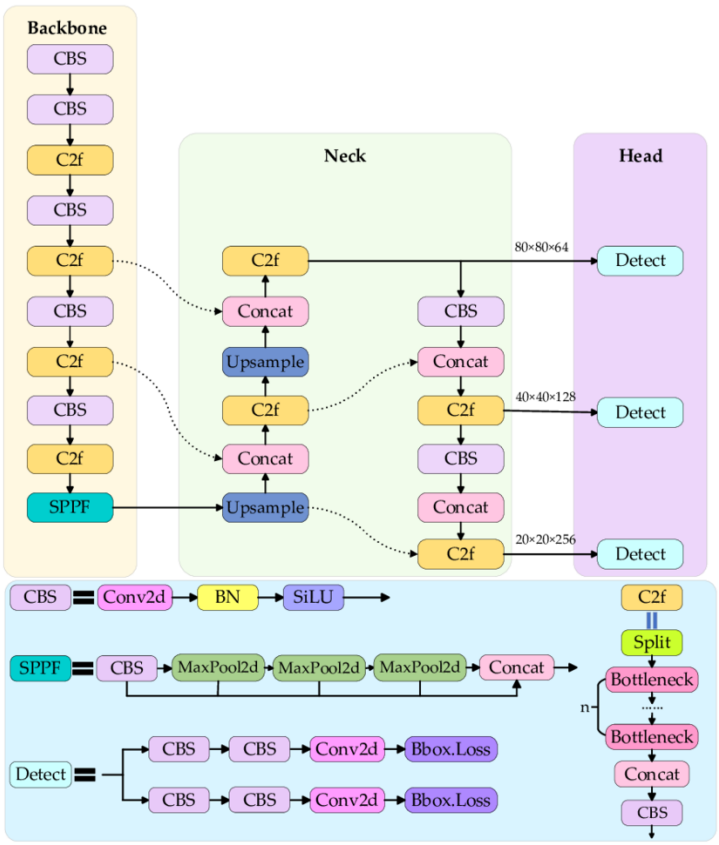
- *Corss-Stage Partial Network* (CSP)

Arsitektur *Cross-Stage Partial Network* (CSP) diperkenalkan sebagai salah satu inovasi dalam jaringan CNN untuk meningkatkan efisiensi ekstraksi fitur sambil mengurangi redundansi komputasi. CSP digunakan untuk memperbaiki representasi fitur dengan

membagi data fitur yang diekstraksi menjadi dua jalur (*paths*), yakni jalur yang diproses lebih lanjut melalui konvolusi dan jalur yang dilewatkan langsung ke tahap penggabungan (*fusion*). Pendekatan ini memungkinkan jaringan untuk meminimalkan redundansi dengan membagi fitur menjadi dua jalur sehingga menghasilkan efisiensi pemrosesan yang lebih baik tanpa mengorbankan akurasi deteksi. Selain itu CSP juga dapat meningkatkan pembelajaran fitur dan membantu menstabilkan aliran gradien (*gradient descent*) selama pelatihan, sehingga mengurangi risiko *vanishing gradient* pada jaringan yang dalam (Ma, et al., 2024). Dalam YOLOv8, CSP digunakan pada bagian *backbone* untuk menangkap fitur multi-skala dengan efisien. Dengan struktur ini, model mampu memproses objek dari berbagai ukuran, termasuk objek kecil seperti trikoma pada daun tanaman.

- *Cross-Stage Feature Fusion* (C2F)

Modul *Cross-Stage Feature Fusion* (C2F) merupakan pengembangan lebih lanjut dari CSP yang bertujuan untuk meningkatkan penggabungan fitur lintas skala (*cross-scale feature fusion*). C2F diimplementasikan dalam YOLOv8 untuk mengoptimalkan deteksi objek dengan lebih baik dengan penggabungan fitur multi-skala, C2F mengintegrasikan fitur yang diekstraksi dari berbagai skala untuk menghasilkan representasi yang lebih kaya. Pendekatan ini memungkinkan model mendeteksi objek kecil maupun besar secara lebih akurat. Selain itu C2F juga meminimalkan jumlah parameter jaringan tanpa mengorbankan performa dengan menggunakan pendekatan berbasis *partial fusion*, sehingga menjadikan YOLOv8 lebih ringan dan cepat. Penggabungan fitur multi-skala juga memungkinkan jaringan memahami konteks spasial yang lebih luas, yang sangat penting untuk mendeteksi objek yang saling tumpang tindih atau yang memiliki ukuran kecil. Dalam YOLOv8, C2F diimplementasikan di bagian *neck*, yaitu lapisan penghubung antara *backbone* dan *head*. Modul ini memastikan bahwa fitur yang ditransfer ke bagian *head* memiliki informasi yang cukup kaya untuk deteksi objek secara presisi (Huang, Mi, & Wang, 2024).



Gambar 2.6 Diagram blok YOLOv8 (Huang, Mi, & Wang, 2024)

Kombinasi CSP dan C2f dalam YOLOv8 memberikan sejumlah manfaat, di antaranya yaitu:

a) **Efisiensi Komputasi**, pengurangan parameter jaringan melalui CSP dan efisiensi pemrosesan multi-skala oleh C2F memungkinkan YOLOv8 bekerja dengan baik pada perangkat keras dengan sumber daya terbatas.

b) **Kemampuan Deteksi Objek Kecil**, Arsitektur ini sangat relevan untuk tugas mendeteksi objek kecil, seperti trikoma, karena penggabungan fitur lintas skala memperkaya informasi detail pada tingkat piksel.

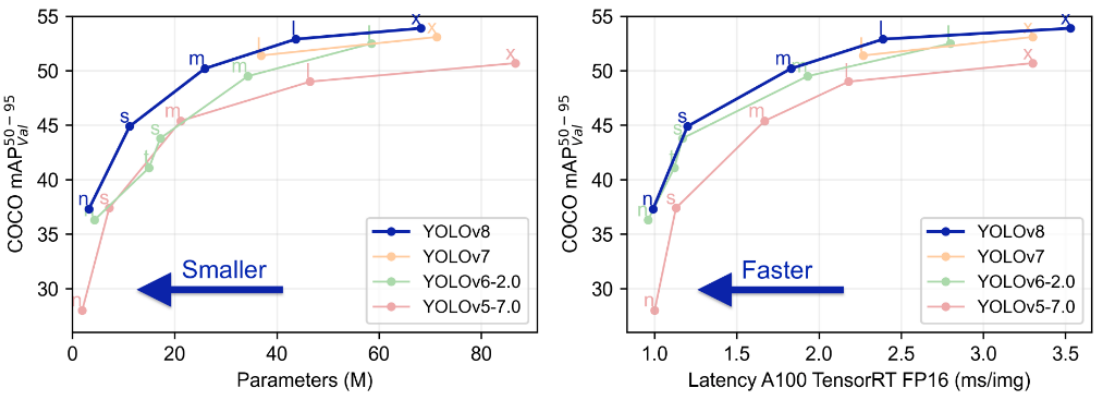
c) **Kecepatan Inferensi**, Struktur modular ini mendukung kecepatan inferensi yang tinggi tanpa mengorbankan akurasi, menjadikan YOLOv8 ideal untuk aplikasi real-time.

BAB 3
Metodologi Penelitian

Penelitian ini akan mencoba untuk melatih sebuah model agar dapat mendeteksi dan mengkuantifikasi citra trikoma pada daun tanaman kentang dengan menggunakan metode *Convolutional Neural Network* dan model arsitektur yang akan digunakan yaitu arsitektur *You Only Look Once v8* (YOLOv8). Arsitektur YOLO dikenal sebagai salah satu arsitektur varian dari CNN yang dapat melakukan pengenalan objek yang tergolong cepat karena arsitektur *frame detection* nya menggunakan model regresi dan tidak dibutuhkan *pipeline* yang kompleks. Selain itu arsitektur YOLO memiliki beberapak kelebihan lain seperti robust terhadap perubahan skala sehingga arsitektur YOLO dapat mengatasi objek dengan skala yang beragam dalam satu gambar dan lebih efektif terhadap variasi ukuran objek.

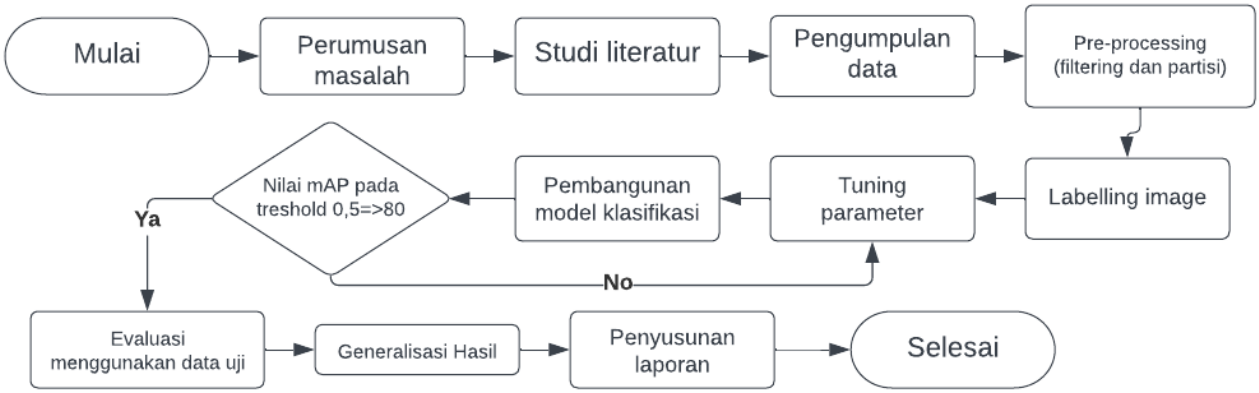
Sampai saat ini arsitektur YOLO mempunyai beberapa versi. YOLOv8 merupakan salah satu model terbaru dan modern yang dikembangkan oleh Ultralytics. YOLOv8 mengembangkan keberhasilan versi-versi YOLO sebelumnya. YOLOv8 memperkenalkan fitur-fitur dan perbaikan baru untuk meningkatkan kinerja serta fleksibilitas. YOLOv8 dirancang untuk memberikan kecepatan, akurasi, dan kemudahan penggunaan, sehingga menjadi pilihan yang sangat baik untuk berbagai tugas seperti klasifikasi, deteksi objek, segmentasi dan lain-lain. Selain itu pada YOLOv8 memiliki arsitektur dengan *cross-stage partial network* (CSP) dan modul *cross-stage feature fusion* (C2f) yang dirancang untuk meningkatkan ekstraksi fitur, terutama untuk objek yang berukuran kecil. Hal ini sangat relevan untuk penelitian yang berfokus pada deteksi trikoma, yaitu objek kecil dan halus pada daun. Arsitektur *cross-stage partial network* dan modul *cross-stage feature fusion* telah diterapkan dalam beberapa model YOLO sebelumnya, terutama dalam YOLOv4 dan YOLOv5. Arsitektur dirancang untuk mengurangi redundansi dalam proses ekstraksi fitur sambil tetap mempertahankan informasi penting, sehingga meningkatkan efisiensi dan akurasi deteksi objek pada model (Ma, et al., 2024).

Keunggulan yang dimiliki YOLOv8 seperti kemampuan deteksi objek kecil ini memungkinkan penulis untuk memanfaatkannya dalam berbagai aplikasi dan domain yang luas termasuk bioinformatik.



Gambar 3.1 Perbandingan kinerja YOLOv8

Secara umum alur penelitian yang akan dilakukan pada penelitian ini digambarkan melalui diagram alir pada gambar 3.2.



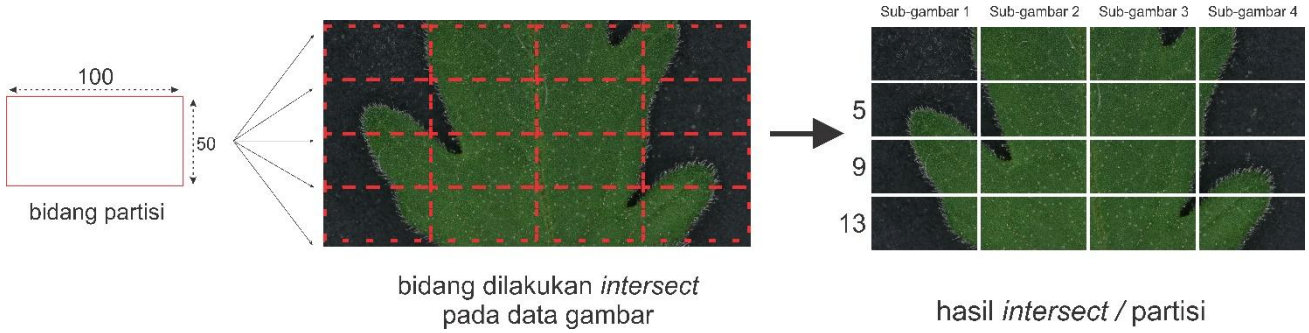
Gambar 3.2 Diagram alur penelitian

Penelitian dimulai dengan membuat rumusan masalah. Rumusan masalah disusun dengan mengidentifikasi topik yang menarik dan relevan terkait bidang bioinformatik. Kemudian penulis melihat bagaimana perkembangan implementasi dari algoritma *deep learning* pada bidang tersebut.

Setelah menentukan rumusan masalah, selanjutnya dilakukan studi literatur dengan mencari jurnal atau referensi terkait sebagai bahan referensi. Referensi yang dicari merupakan referensi dalam bidang lingkup biologi yang membahasa trikoma dan lingkup *image processing / deep learning*.

Tahapan penelitian selanjutnya yaitu melakukan pengumpulan data. Data yang akan digunakan pada penelitian ini merupakan data primer dari citra mikroskopis pada permukaan daun tanaman kentang budidaya. Data yang sudah dikumpulkan kemudian akan melalui

tahapan *preprocessing data* berupa filter dan partisi. Filter data dilakukan untuk memilih data gambar yang akan digunakan dari sekian banyak data yang ada, hal ini dikarenakan data bioinformatik memiliki karakteristik yang kurang konsisten sehingga penting untuk melakukan filter data terlebih dahulu. Kemudian untuk proses partisi bertujuan untuk melakukan perbesaran gambar berdasarkan area atau *region* yang dibagi pada data gambar yang asli. Proses pendeteksian trikoma akan dilakukan melalui area-area (sub-gambar) yang sudah dibagi pada data asli. Hal ini dilakukan agar dapat mempermudah sistem dalam melakukan identifikasi terhadap trikoma dengan ukuran citra yang sudah diperbesar sesuai dengan area masing-masing sub-gambar. Pada penelitian ini melakukan partisi menggunakan metode *intersect* (memotong) melalui media bidang persegi dengan ukuran 100×50 sehingga terdapat 16 bidang persegi yang digunakan untuk memotong seluruh permukaan gambar. proses partisipasi dapat diilustrasikan pada gambar berikut.



Gambar 3.3 Ilustrasi proses partisi data gambar

Proses selanjutnya yaitu proses pemberia label pada data. Proses ini dilakukan untuk memberikan keterangan kelas/kategori pada objek yang akan dideteksi. Pada penelitian ini menggunakan label atau kelas yaitu trikoma tipe glandularis bulbose. Trikoma ini memiliki bentuk bulat atau bulosa yang unik, dengan kepala trikoma yang mengandung sel-sel sekretori yang dapat memproduksi dan menyimpan berbagai senyawa metabolit sekunder. Pemilihan tipe trikoma tersebut berdasarkan ketersediaan data yang ada pada sampel gambar tanaman kentang hasil budidaya yang diperoleh oleh peneliti. Proses pelabelan dilakukan manual oleh penulis dengan memberikan label satu per satu dari setiap trikoma yang ada pada data gambar. Sebelum melakukan proses pelabelan, penulis juga akan berkonsultasi dengan pakar dibidang biologi/*plantology* terkait dengan karakteristik trikoma agar label yang diberikan tidak keliru.

Tahapan selanjutnya kemudian melakukan konfigurasi sistem dan *tuning* parameter untuk proses *neural network*. Proses ini sangat penting karena akan mempengaruhi hasil

akhir dan ketepatan model dalam melakukan deteksi. *Loss function* yang digunakan pada arsitektur YOLOv8 yaitu *box loss* (kotak pelatihan), *classification loss* (klasifikasi pelatihan) dan *Distribution Focal Loss* (distribusi pelatihan). Untuk metrik yang digunakan yaitu *mean Average Precision* (mAP) dan *Intersection Over Union* (IoU). Metrik *mean Average Precision* (mAP) memiliki rentang nilai antara 0 hingga 1. Rentang ini digunakan untuk mengukur kualitas kinerja sistem deteksi objek dalam tugas pengenalan objek. Nilai mAP didapatkan dengan menghitung *Average Precision* (AP) untuk setiap kelas objek yang terdeteksi, dan kemudian mengambil rata-rata dari semua AP tersebut. AP sendiri mengukur akurasi dan kepresisian sistem dalam mendeteksi objek dengan memperhitungkan *recall* (jumlah objek yang berhasil dideteksi) dan *precision* (jumlah deteksi yang benar). Sebuah nilai mAP yang lebih tinggi menunjukkan performa yang lebih baik, dengan 1 sebagai nilai maksimal yang mencerminkan deteksi yang sempurna. Sedangkan nilai mAP yang rendah menunjukkan kinerja yang kurang akurat atau deteksi yang buruk. Pemilihan *threshold* (ambang batas) untuk deteksi objek yang dianggap benar atau salah dapat mempengaruhi nilai mAP. Biasanya, sebuah threshold IoU (*Intersection over Union*) digunakan untuk menentukan apakah sebuah prediksi *bounding box* dianggap benar atau salah. Nilai umum untuk threshold IoU adalah sekitar 0,5 atau 0,5 hingga 0,7, tetapi dapat disesuaikan berdasarkan kebutuhan dan persyaratan tugas deteksi objek yang sedang dijalankan. Dengan memperhatikan rentang nilai 0 hingga 1, mAP memberikan ukuran yang komprehensif tentang performa deteksi objek pada berbagai kelas dan memudahkan perbandingan antara sistem yang berbeda. Pada penelitian ini akan digunakan nilai *threshold* sebesar 0,5 (50%) dan 0,5-0,9 (50%-90%) sebagai acuan keberhasil model yang dilatih.

Setelah proses *learning* model selesai, maka dapat dilakukan evaluasi atau pengujian menggunakan data *testing* / data validasi. Hal ini juga dilakukan untuk melihat bagaimana implementasi dan hasil akhir dari model yang telah dilatih. Berdasarkan dari beberapa referensi/penelitian terdahulu mengenai *image processing/deep learning* yang penulis jadikan sebagai bahan acuan, didapatkan rata-rata hasil akhir mAP sebesar 80,8% dari performa model dalam melakukan klasifikasi atau identifikasi, sehingga angka tersebut akan dijadikan acuan oleh penulis untuk bisa mendapatkan nilai akurasi yang lebih tinggi dan menghasilkan model yang lebih baik.

Untuk melakukan generalisasi hasil yang diperoleh berkaitan dengan deteksi dan kuantisasi trikoma pada pengamatan mikroskopis daun kentang, penulis melakukan beberapa langkah diantaranya yaitu dengan membandingkan jumlah trikoma yang diprediksi oleh model dengan jumlah trikoma sebenarnya pada *data truth*, kemudian penulis

menghitung nilai *precision* dan *recall* dari nilai *true positive, false positive* dan *false negative* yang diperoleh.

Luaran yang dihasilkan pada penelitian ini berupa deteksi dan kuantifikasi trikoma tipe glandularis bulbose pada sampel citra daun tanaman kentang budidaya. Kuantifikasi dilakukan berdasarkan hasil partisi dari area atau region yang sudah dibagi. Model ini akan menjadi salah satu tahapan awal atau pionir deteksi dan kuantifikasi trikoma berbasis *deep learning*. Untuk penelitian-penelitian berikut nya dapat mengembangkan model yang dapat melakukan deteksi dan kuantifikasi pada tipe trikoma yang lain dan juga pada sampel citra daun tanaman liar.

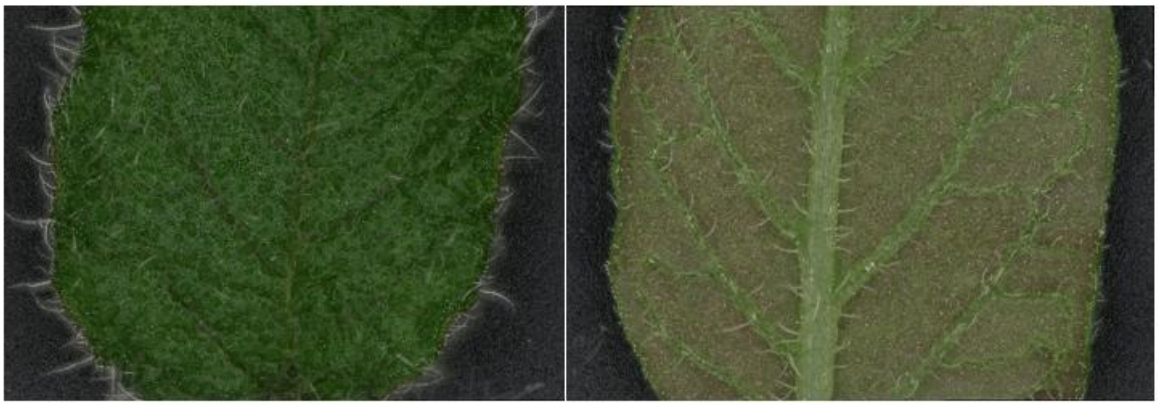
BAB 4
Hasil dan Pembahasan

Pada bagian hasil dan pembahasan, penulis menjelaskan secara detail langkah-langkah untuk memperoleh hasil pembuatan model dari proses pelatihan, evaluasi metrik, dan visualisasi pengujian model.

4.1. Persiapan Dataset

4.1.1 Data yang Digunakan

Penelitian ini menggunakan dataset berupa foto yang diambil menggunakan mikroskopis digital kamera. Data tersebut merupakan hasil observasi trikoma dari peneliti plantologi Wageningen University & Research. Dataset yang telah dikumpulkan sebelumnya telah diverifikasi oleh pakar di bidang plantologi. Pakar melakukan observasi keberadaan trikoma glandural pada beberapa sampel data gambar untuk selanjutnya dijadikan pedoman oleh penulis saat melakukan proses labeling data. Total keseluruhan data berjumlah 159 gambar yang nanti akan dipartisi menjadi beberapa bagian yang lebih kecil (sub-gambar) agar proses pemberian label dapat dilakukan dengan lebih mudah. Dari 159 data gambar, 27 diantaranya merupakan gambar bagian bawah permukaan daun dan 132 gambar diambil dari bagian atas permukaan daun. Gambar berikut merupakan sampel dari gambar bagian atas (a) dan bagian bawah (b) dari permukaan daun kentang yang telah diambil menggunakan mikroskopis digital kamera.



Gambar 4.1 Sampel gambar daun bagian permukaan atas (a) dan permukaan bawah (b)

Mengacu pada gambar 4.1 dapat dilihat bahwa trikoma yang terdapat pada bagian atas permukaan daun terlihat lebih jelas dibanding trikoma yang terdapat pada bagian bawah

daun. Hal ini diakibatkan karena bagian atas permukaan daun cenderung memiliki warna lebih kontras dibanding bagian bawah permukaan daun yang memiliki warna lebih pudar.

4.1.2 Preprocessing Data

Data yang sudah dikumpulkan kemudian masuk kedalam tahap *pre-processing*. Pada tahap ini dibagi menjadi 2 bagian yaitu tahap partisi gambar dan tahap *labeling*/anotasi.

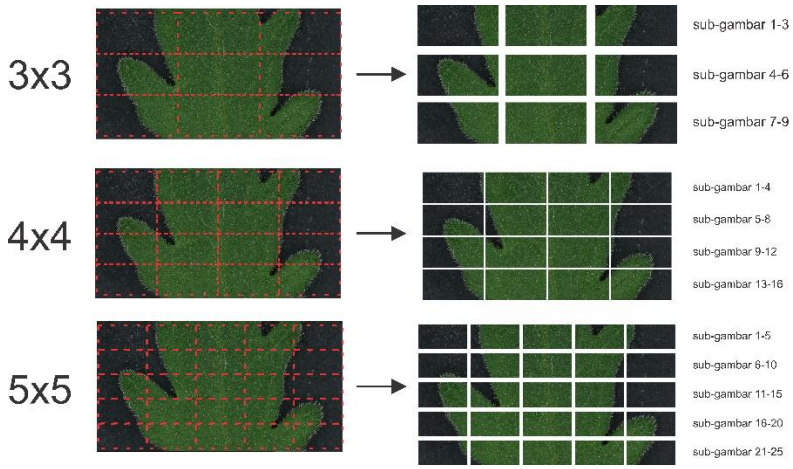
Tahap partisi bertujuan untuk membagi satu gambar menjadi beberapa bagian yang lebih kecil (sub-gambar) agar identifikasi trikoma dapat dilakukan dengan lebih jelas. Pada proses ini penulis melakukan beberapaka kali percobaan *intersect* dengan menggunakan dimensi grid yang berbeda-beda yaitu mulai dari 2×2, 3×3, 4×4, 5×5 dan 6×6. Percobaan ini dilakukan untuk menentukan dimensi grid mana yang paling ideal untuk mendukung proses *training*. Hasil percobaan ditampilkan pada tabel 4.1.

Tabel 4.1 Percobaan dimensi grid

Dimensi	Estimasi Jumlah Sub-Gambar	Visual Sub-Gambar	Catatan
2 × 2	600		Menghasilkan sekitar 600 sub-gambar baru dan visual trikoma masih tampak cukup kecil sehingga proses labeling sulit dilakukan.
3 × 3	1400		Menghasilkan sekitar 1400 sub-gambar baru dan visual trikoma mulai terlihat cukup jelas
4 × 4	2500		Menghasilkan sekitar 2500 sub-gambar baru dan visual trikoma terlihat jelas shingga proses labeling dapat dilakukan dengan lebih mudah.
5 × 5	3900		Walaupun visual trikoma terlihat dengan jelas, namun estimasi sub-gambar yang dihasilkan sangat banyak sehingga waktu yang dibutuhkan dalam proses labeling dan juga semakin lama.

Dimensi	Estimasi Jumlah Sub-Gambar	Visual Sub-Gambar	Catatan
6 × 6	5700		Walaupun visual trikoma terlihat dengan jelas, namun estimasi sub-gambar yang dihasilkan sangat banyak sehingga waktu yang dibutuhkan dalam proses labeling dan juga semakin lama.

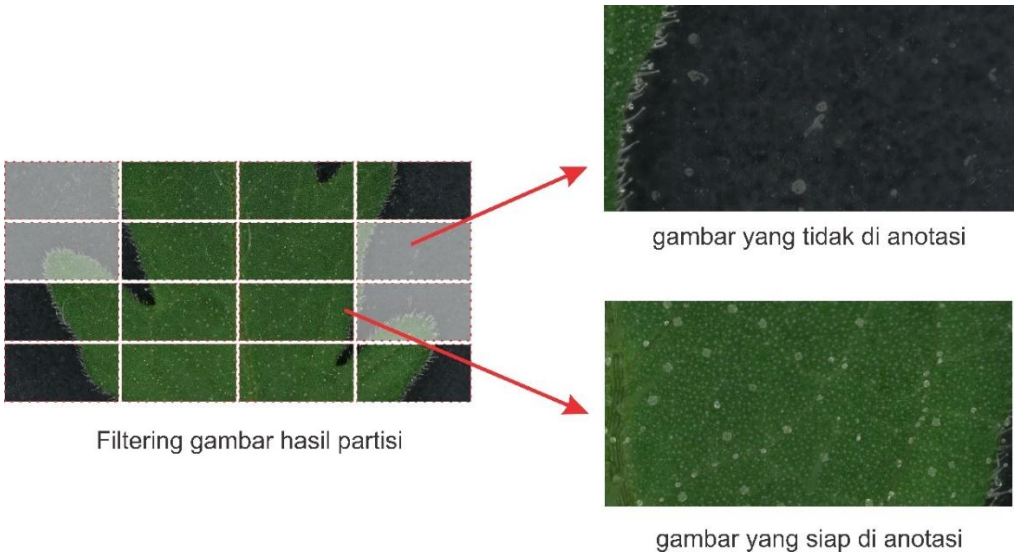
Dari beberapa kali percobaan dengan menggunakan berbagai macam dimensi gird, dapat diketahui bahwa masing-masing dimensi memiliki kelebihan dan kekurangan masing-masing. Apabila dimensi yang digunakan kecil (2×2 atau 3×3) maka sub-gambar yang dihasilkan menampilkan visual trikoma yang masih terlalu kecil sehingga proses labeling dan deteksi masih sulit dilakukan. Kemudian apabila dimensi yang digunakan besar (5×5, 6×6 dan seterusnya) maka sub-gambar menampilkan visual trikoma yang cukup jelas namun jumlah sub-gambar menjadi terlalu banyak sehingga dapat membuat proses labeling membutuhkan waktu yang lebih lama dan semakin membebani resource saat proses training. Oleh karena itu, pada penelitian ini menggunakan dimensi grid 4×4.



Gambar 4.2 Ilustrasi percobaan setiap dimensi grid

Proses partisi membuat sebuah data gambar menjadi beberapa sub-gambar tergantung dengan dimensi grid yang digunakan, namun tidak semua gambar yang dihasilkan merepresentasikan permukaan daun sehingga dari sini perlu dilakukan *filtering* untuk memilah gambar. Gambar yang dipilih berdasarkan komposisi ruas daun atau keberadaan trikoma yang dapat dijadikan data untuk proses labeling/anotasi. Gambar yang

tidak merepresentasikan permukaan daun/tidak terdapat trikoma dapat dibuang dan tidak diikutsertakan kedalam proses *training*.



Gambar 4.3 *Filtering* sampel hasil partisi

4.1.3 *Split Dataset*

dari proses partisi dan *filtering* data, total keseluruhan sug-gambar akan dibagi menjadi 2 yaitu data *train* dan data *test*. Data *train* digunakan sebagai input pada saat proses pelatihan model (*training model*), sedangkan data *test* digunakan untuk memvalidasi atau mengukur tingkat akurasi dari model yang telah dilatih. Pada penelitian ini penulis menggunakan skema 80% data digunakan untuk *training* dan 20% digunakan untuk *testing*. Skema pembagian dataset dapat diilustrasikan pada tabel 4.2.

Tabel 4.2 Pembagian dataset

Data	**Persentase**
Training	80%
Testing	20%
Total	**100%**

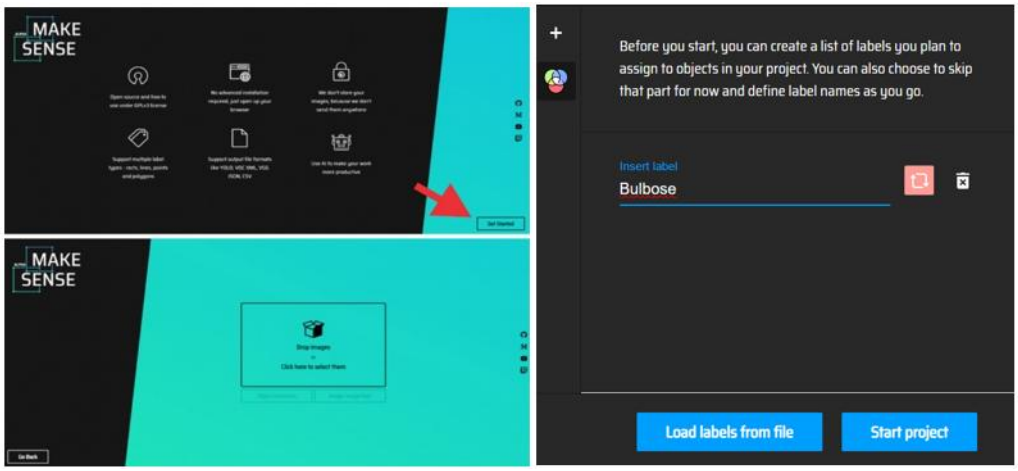
4.1.4 Pemberian Label

Tahap selanjutnya setelah seluruh data siap yaitu proses anotasi atau pemberian label. Pada tugas deteksi objek, tahap anotasi merupakan tahap yang sangat penting. Selain *setup parameter* dan arsitektur, tahap anotasi juga dapat mempengaruhi kualitas akhir dari suatu model. Tahap anotasi sendiri bertujuan untuk memberikan tanda lokasi berupa *bounding box*

dari suatu objek pada data gambar, selain itu anotasi juga dapat memberikan keterangan kelas dari objek yang akan dideteksi.

Pada penelitian ini penulis melakukan pemberian label dengan bantuan makesense.ai. makesense.ai adalah *web apps* yang biasa digunakan untuk memberi label pada foto-foto atau gambar untuk proyek-proyek *deep learning, computer vision* atau *objeck detection* sehingga proses persiapan dataset menjadi lebih mudah dan cepat. Alat ini tidak memerlukan instalasi yang rumit dan dapat digunakan secara gratis melalui *browser*. Label-label yang telah disiapkan dapat diunduh dalam beberapa format yang didukung seperti format YOLO dan XML.

Pengguna dapat menggunakan fitur malalui laman https://www.makesense.ai dan klik menu bagian *get started*. Selanjutnya pengguna dapat langsung melakukan *drag and drop* atau bisa juga dengan mengklik menu yang ada untuk memasukan semua file gambar yang akan diberikan label.

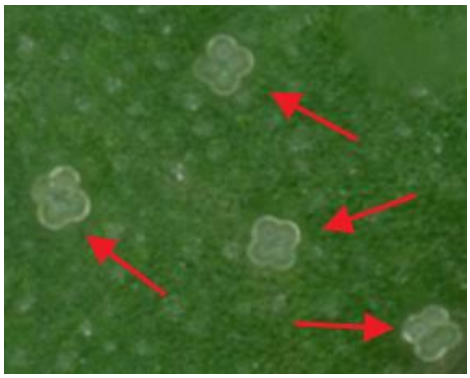


Gambar 4.4 Proses upload data dan label pada *makesense.ai*

Sebelum mulai melakukan pemberian label, terlebih dahulu penulis akan mendefinisikan kelas yang akan digunakan dalam penelitian. Terdapat dua cara yang bisa digunakan untuk mendefinisikan kelas yaitu dengan melakukan *load labels* apabila sudah terdapat file label yang siap digunakan, atau bisa mendefinisikan kelas baru dengan menuliskan nama kelas pada kolom *insert label.* Setelah label kelas siap, dapat dilanjutkan kebagian pemberian label dengan klik menu *start project.*

Trikoma tipe *glandularis* (*bulbose*) memiliki bentuk atau ciri yang khas, yaitu dengan kepala membulat atau bulosa. Kepala trikoma tampak lebih besar atau menonjol daripada bagian batangnya, sehingga memberikan tampilan yang sedikit bengkak atau bulat. Pada bagian kepala trikoma *grandular bulbose* berisi cairan atau zat yang diproduksi oleh

sel-sel sekretori sehingga membuat warna pada bagian kepala menjadi lebih terang dibanding bagian batangnya. Dari ciri-ciri fisik dan bentuk tersebut maka penulis dapat melakukan pemberian label berdasarkan ciri-ciri yang ada. Proses pemberian label juga melibatkan pakar dibidang terkait agar pemberian label data benar-benar valid dan dapat menghasilkan model yang akurat.

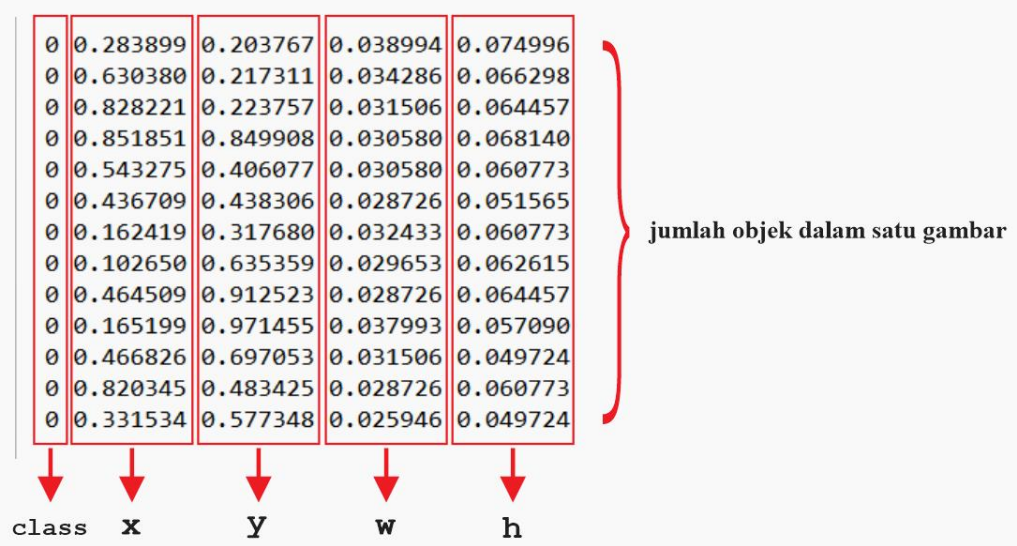


Gambar 4.5 Bentuk atau ciri fisik trikoma

Pemberian label dilakukan dengan cara melakukan *click and drag* pada bagian objek yang terdapat pada setiap data gambar. tahap ini merupakan tahapan yang membutuhkan waktu dan tenaga paling banyak karena pemberian label dilakukan satu persatu pada setiap objek dan setiap data gambar yang ada secara manual. Setelah proses anotasi selesai, semua data hasil label dapat diexport kedalam beberapa jenis format seperti format YOLO (file .txt), format XML dan format CSV. Dalam penelitian ini karena penulis akan melakukan *training* menggunakan arsitektur YOLO maka data hasil label akan diexport kedalam format YOLO (file txt).

Gambar 4.6 Proses pemberian label menggunakan *makesense.ai*

Anotasi citra YOLO akan tersimpan dalam format file .txt dengan nama yang sama dengan file gambar. Setiap file .txt berisi informasi mengenai kode kelas objek, koordinat objek, tinggi (*width*) dan lebar (*height*). Kode kelas objek dimulai dari angka 0 sampai *n* tergantung seberapa banyak kelas yang akan digunakan, pada penelitian ini karena hanya menggunakan 1 kelas maka kode kelas objek akan diisi angka 0. Kemudian koordinat objek (*x,y*) merepresentasikan letak koordinat dari *bounding box* yang dibuat saat proses labeling, lalu *width* dan *height* memberikan informasi mengenai tinggi dan lebar dari *bounding box* yang dibuat. Selain itu perlu diketahui bahwa pada proses anotasi sangat memungkinkan terdapat lebih dari satu objek dengan kelas yang sama ataupun berbeda dalam satu data gambar, sehingga pada satu file .txt dapat berisi informasi dari beberapa hasil label objek. Contoh format anotasi citra YOLO pada file .txt seperti ditunjukkan pada gambar 4.7.



Gambar 4.7 Hasil file anotasi

4.2. *Tuning Hyperparameter*

Langkah awal yang dilakukan sebelum memulai proses *training* model adalah mengatur *hyperparameter*. *Hyperparameter* dalam konteks *machine learning* dan *deep learning* adalah parameter yang nilainya ditentukan sebelum proses pelatihan model dimulai. *Hyperparameter* mengatur aspek-aspek tertentu dari proses pelatihan dan arsitektur model yang tidak bisa dipelajari dari data selama pelatihan. *Hyperparameter* adalah bagian dari desain model dan dapat mempengaruhi hasil akhir model.

4.2.1 Parameter Default

Arsitektur YOLOv8 pada *library ultralytics* menyediakan *hyperparameter default* yang bisa langsung digunakan untuk melakukan *training* model. Berikut beberapa metrik standar (*default*) pada *hyperparameter* YOLOv8 yang ditampilkan pada tabel 4.3.

Tabel 4.3 Metrik *hyperparameter*

Parameter	Definisi	Nilai
`Epoch`	siklus lengkap di mana seluruh dataset dilalui oleh algoritma pembelajaran untuk memperbarui bobot model.	100
`Learning rate`	menentukan seberapa besar perubahan atau penyesuaian yang dilakukan pada bobot model selama proses *training* setiap kali algoritma *backpropagation* memperbarui bobot.	0,01
`Batch size`	jumlah sampel data yang diproses bersama-sama sebelum model melakukan pembaruan bobot selama pelatihan.	16
`Time`	Waktu pelatihan maksimum dalam jam. Jika diatur, akan mengesampingkan parameter *epoch*.	null
`patience`	Jumlah epoch yang harus ditunggu ketika tidak ada peningkatan dalam metrik validasi untuk mencegah *overfitting*.	100
`imgsz`	Ukuran target pada data gambar untuk proses *training*. Semua data gambar diubah ukurannya ke dimensi yang telah ditentukan sebelum masuk ke dalam model.	640×640
`save`	Memungkinkan untuk menyimpan *checkpoint* dan bobot model terakhir agar dapat digunakan untuk penerapan model ataupun menlajutkan proses *training.*	True
`pretrained`	Menentukan apakah akan memulai *training* dari model yang telah dilatih sebelumnya, sehingga dapat meningkatkan efisiensi pelatihan dan kinerja model.	True
`optimizer`	Menentukan algoritma yang digunakan untuk memperbarui bobot model. Meliputi *SGD*, *Adam*,	Auto

Parameter	Definisi	Nilai
	AdamW, *NAdam*, *RAdam*, *RMSProp*, dll., atau dapat juga diset otomatis untuk pemilihan otomatis berdasarkan konfigurasi model.	
`Single_cls`	Membuat semua kelas/label dalam kumpulan data sebagai satu kelas selama proses *training*. Berguna untuk tugas klasifikasi biner atau saat berfokus pada deteksi objek daripada klasifikasi.	True
`amp`	Mengaktifkan *automatic mixed precision* (AMP) untuk mengurangi penggunaan memori dan mempercepat *training* dengan dampak minimal pada akurasi.	True
`momentum`	Digunakan untuk mempercepat *training* dengan mengurangi osilasi.	0,937
`Weight_decay`	Parameter yang mengontrol tingkat penalti pada bobot besar untuk mengurangi *overfitting.*	0,0005
`dropout`	Persentase unit yang secara acak diabaikan selama *training* untuk mencegah *overfitting*.	0,0
`box`	Ukuran dan skala dari *bounding boxes* yang digunakan untuk prediksi objek.	7,5
`nms`	*Non-Maximum Suppression (NMS) Threshold*: Digunakan dalam YOLO untuk menghilangkan prediksi yang tumpang tindih dengan *confidence score* rendah.	False

Pada *deep learning,* terdapat beberapa parameter yang cukup penting dan sering kali digunakan sebagai aspek eksperimen maupun simulasi untuk mendapatkan model yang terbaik. Parameter-parameter tersebut yaitu *epoch, learning rate, batch size* dan *image size*.

Epoch merupakan jumlah siklus pelatihan yang dilakukan pada *dataset*. *Training* model biasanya melibatkan beberapa *epoch* untuk memungkinkan model belajar dan menyempurnakan bobotnya secara bertahap dan meningkatkan akurasi prediksi dalam setiap siklus. Selama satu *epoch*, model memproses setiap contoh dalam *dataset* satu kali dan menyesuaikan bobot berdasarkan kesalahan yang dihitung dari prediksi. Dalam *deep learning,* terlalu sedikit *epoch* dapat menyebabkan *underfitting* atau model tidak punya banyak waktu untuk belajar sehingga performa model cenderung buruk, sementara terlalu

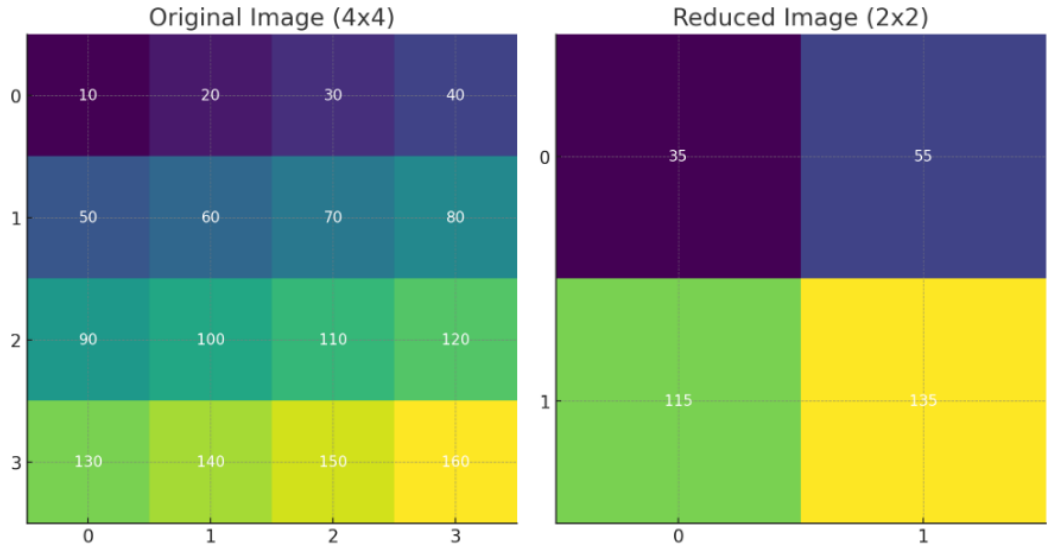
banyak *epoch* bisa menyebabkan *overfitting* yang artinya performa model hanya bagus pada data *training* namun buruk pada data *testing*. Pada penelitian ini penulis akan menggunakan jumlah iterasi sebanyak 100 *epoch* untuk percobaan pertama dan akan mengidentifikasi grafik dari hasil *training* untuk melihat apakah terjadi *overfitting*.

Learning rate pada *deep learning* merupakan sebuah *hyperparameter* yang menentukan seberapa besar perubahan atau penyesuaian yang dilakukan pada bobot model selama proses *training* setiap kali algoritma *backpropagation* memperbarui bobot. *Learning rate* yang terlalu tinggi dapat menyebabkan pelatihan tidak stabil dan membuat model melompati bobot minimum optimal, sedangkan *learning rate* yang terlalu rendah dapat membuat pelatihan sangat lambat dan mungkin terjebak pada solusi suboptimal. Oleh karena itu, pemilihan *learning rate* yang tepat sangat penting untuk kinerja dan efisiensi model *deep learning*. Pada *library ultralytics*, nilai standar *learning rate* yang digunakan yaitu 0,01.

Batch size pada *deep learning* adalah jumlah sampel data yang diproses bersama-sama sebelum model melakukan pembaruan bobot selama proses *training*. Memilih ukuran *batch* yang sesuai penting karena mempengaruhi efisiensi pelatihan dan penggunaan memori. Ukuran *batch* yang besar bisa memanfaatkan paralelisme lebih baik dan stabil, tetapi membutuhkan lebih banyak memori. Sebaliknya, ukuran *batch* kecil lebih mudah dimanipulasi namun mungkin menyebabkan fluktuasi dalam pelatihan. Pada *library ultralytics*, nilai standar *batch size* yang digunakan yaitu 16.

Selanjutnya pada parameter *imgsz* merupakan parameter yang menentukan ukuran dimensi gambar yang digunakan sebagai input untuk model selama proses *training*, *validation*, dan *inference*. Pada YOLOv8, ukuran gambar input diubah ke dimensi persegi (`*imgsz* x *imgsz*`), agar konsisten dengan ukuran input layer dari arsitektur YOLO. Apabila gambar input (data asli) tidak berdimensi persegi, maka YOLO akan menambahkan padding hitam (*letterboxing*) untuk menjaga rasio aspek gambar. Sebagai contoh, jika gambar input memiliki ukuran 1280×720 piksel, maka sisi terpanjang gambar akan di-*resize* menjadi 640 piksel dan padding hitam akan ditambahkan pada sisi lainnya di sekitar gambar agar hasil akhirnya menjadi 640×640 piksel. Model YOLO mengharuskan gambar untuk memiliki ukuran yang tetap agar dapat diproses dalam *batch* selama pelatihan, sehingga menghindari masalah terkait ukuran yang bervariasi. YOLOv8 pada *ultralytics* menggunakan teknik *image resizing* dengan algoritma *bilinear interpolation*. *Bilinear interpolation* merupakan algoritma untuk memperbesar atau memperkecil ukuran gambar dengan menghitung nilai piksel baru berdasarkan nilai piksel di sekitarnya. Algoritma ini menggunakan interpolasi linear dalam dua arah (horizontal dan vertikal) untuk menentukan warna atau intensitas

piksel baru. Pada *library ultralytics*, nilai standar parameter *image size* yang digunakan yaitu 640×640. Penggunaan ukuran 640×640 piksel dianggap ideal karena seimbang antara akurasi dan kecepatan komputasi. Ilustrasi algoritma *bilinear interpolation* dalam mengubah ukuran gambar ditampilkan pada gambar 4.8.



Gambar 4.8 Ilustrasi *bilinear interpolasi* untuk mereduksi gambar

Pada proses *training* yang pertama, parameter *epoch, learning rate, batch size* dan *hyperparameter* yang lain ditentukan berdasarkan rekomendasi *default* dari *library ultralytics.* Dengan mengatur dan membandingkan nilai *hyperparameter* pada simulasi *training* berikutnya, performa pada model *deep learning* dan YOLO dapat dioptimalkan untuk mencapai hasil yang lebih baik.

4.2.2 Eksperimen Parameter

Pada penelitian ini penulis juga akan melakukan beberapa simulasi atau eksperimen terhadap *hyperparameter* yang sudah disebutkan sebelumnya. Eksperimen ini bertujuan untuk melakukan pembuktian secara ilmiah apakah nilai-nilai yang telah ditentukan pada *hyperparameter* merupakan nilai yang menghasilkan model terbaik. Selain itu penulis juga akan membandingkan dimensi grid yang dilakukan pada proses partisi untuk melihat apakah dimensi 4×4 merupakan dimensi yang paling ideal.

Eksperimen dilakukan dengan metode *sequential search (one-at-a-time)*, yang artinya pengujian dilakukan terhadap satu parameter pada satu waktu, dengan nilai parameter lainnya tetap (*default*). Setelah menemukan nilai terbaik untuk satu parameter, maka nilai tersebut akan digunakan untuk menguji parameter berikutnya. Sebagai contoh, parameter pertama yang akan diuji yaitu *epoch*. Nilai *epoch* diuji dengan nilai 25, 50, 100,

sementara *learning rate*, *batch size* dan *image size* tetap default. Setelah mendapatkan nilai optimal untuk *epoch*, eksperimen dilanjutkan dengan menguji *learning rate* dengan nilai yang berbeda, dan seterusnya. Skema eksperimen yang dilakukan menggunakan *sequential search (one-at-a-time)* disajikan pada tabel 4.4 berikut.

Tabel 4.4 Eksperimen parameter

Parameter	**Nilai**	**Evaluasi**			**Catatan**
		(mAP@50)	**(mAP@50-95)**	**Durasi (jam)**	
Epoch	25	0,761	0,41	0,397	Berdasarkan hasil eksperimen terhadap tiga nilai *epoch* yang berbeda, diperoleh hasil bahwa jumlah *epoch* yang menghasilkan nilai mAP terbaik yaitu 100 *epoch*.
	50	0,792	0,386	0,731	
	100	0,812	0,421	0,941	
Learning rate	0,1	0,758	0,376	1,478	Pada peroses percobaan terhadap tiga nilai *learning rate* yang diuji, diperoleh bahwa nilai default (0,01) dapat menghasilkan mAP lebih tinggi karena *learning rate* yang terlalu kecil membutuhkan lebih banyak iterasi untuk mencapai konvergensi.
	0,01	0,812	0,421	0,941	
	0,0001	0,759	0,363	1,465	
Batch size	8	0,781	0,319	1,714	Berdasarkan eksperimen yang dilakukan terhadap tiga nilai *batch size*, diketahui bahwa nilai *batch size* = 16 menghasilkan mAP yang lebih baik karena dataset yang digunakan mungkin tidak memerlukan terlalu banyak iterasi untuk mencapai generalisasi yang baik. Dalam kasus ini, *batch size* default (16) dapat memberikan keseimbangan yang optimal antara stabilitas dan kecepatan pembaruan parameter dibandingkan *batch size* yang lebih kecil.
	16	0,812	0,421	0,941	
	32	0,786	0,376	1,527	
Image size	320×320	0,78	0,4	0,85	Meskipun secara teori peningkatan ukuran *image size* (dimensi gambar) seharusnya dapat mempertahankan lebih

	640×640	0,812	0,421	0,941	banyak detail dan informasi visual pada data, hasil eksperimen menunjukkan bahwa penggunaan ukuran gambar yang lebih tinggi (1024) tidak menghasilkan *mean Average Precision* (mAP) yang lebih tinggi. Hal ini kemungkinan disebabkan oleh model YOLOv8 mungkin telah mencapai titik saturasi pada *feature extraction*, di mana detail tambahan dari gambar beresolusi tinggi tidak memberikan kontribusi signifikan terhadap peningkatan kinerja model.
	1024×1024	0,76	0,388	2,476	

Berdasarkan tabel 4.4 mengenai hasil eksperimen yang dilakukan terhadap beberapa nilai parameter seperti *epoch*, *learning rate*, *batch size* dan *image size* diperoleh hasil bahwa nilai parameter yang dapat menghasilkan model paling optimal yaitu *epoch* = 100; *learning rate* = 0.01; *batch size* = 16; dan *image size* = 640×640.

4.3. *Training Model*

Proses pelatihan atau *training* model merupakan langkah pemodelan yang menggunakan teknik *machine learning* pada data gambar untuk tujuan klasifikasi, sehingga model yang dihasilkan dapat menjalankan tugas-tugas tertentu. Pada penelitian ini tugas yang dimaksud yaitu mendeteksi keberadaan trikoma pada permukaan daun. Proses pelatihan model pada penelitian ini dilakukan menggunakan *device* atau perangkat dengan spesifikasi berikut:

- Laptop MSI GF63 thin
- Prosesor intel core i5-11400H
- RAM 16GB DDR4
- NVIDIA GeForce GTX 1650
- IDE (*Integrated Development Environment*) yang digunakan yaitu Google Colaboratory dengan bahasa pemrograman *python*
- *Package* yang digunakan yaitu *ultralytics*

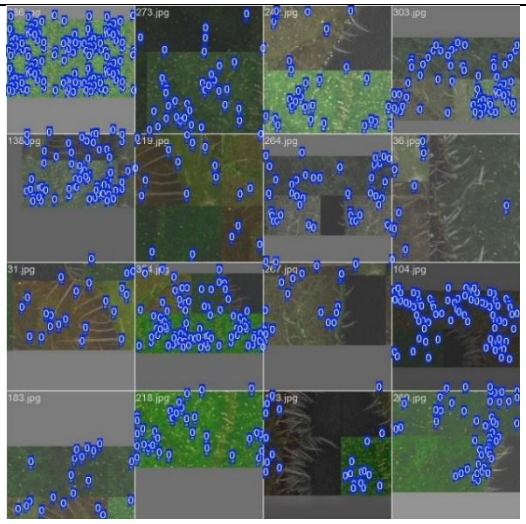
Dalam konteks *training* model YOLO, *library ultralytics* menampilkan file `batch.jpg` yang merupakan sebuah gambar yang dihasilkan selama proses pelatihan model untuk visualisasi. Gambar ini membantu pengguna untuk memeriksa bagaimana model

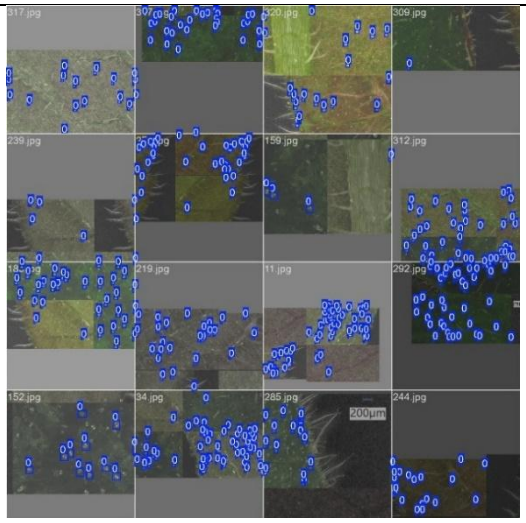
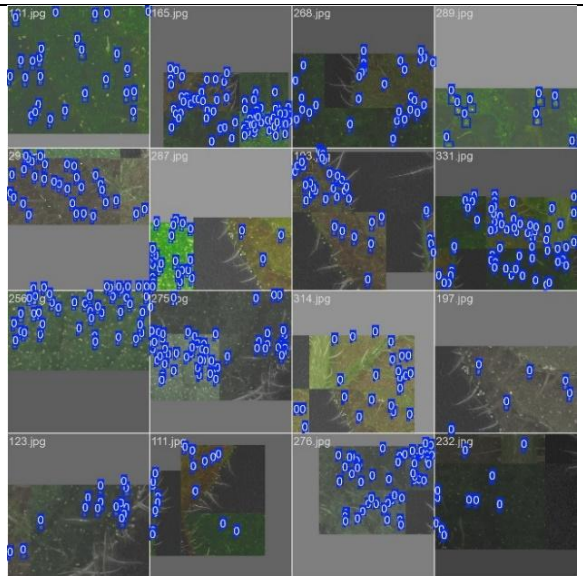
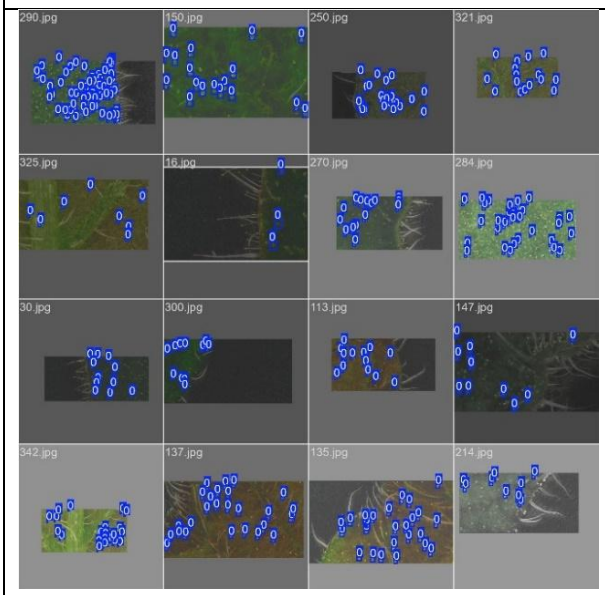
mempelajari dan mendeteksi objek pada gambar pelatihan. *train_batch* menunjukkan beberapa contoh gambar dari *batch* pelatihan saat sedang diproses oleh model. Gambar ini akan mencakup *bounding boxes* yang diprediksi oleh model untuk objek yang terdeteksi dalam gambar tersebut. Ini memungkinkan pengguna untuk melihat apakah model mendeteksi objek dengan benar dan di lokasi yang tepat. *Ultralytics* menampilkan sampel data yang random pada ‘train_batch.jpg` dan val_batch.jpg`, artinya setiap gambar yang ditampilkan merupakan gambar yang berbeda-beda.

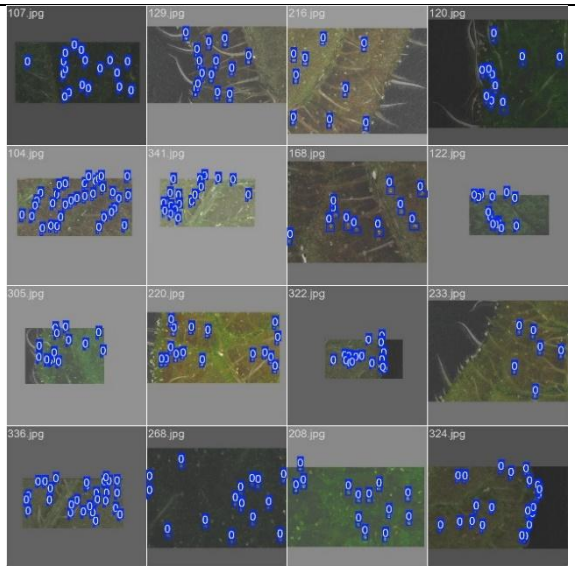
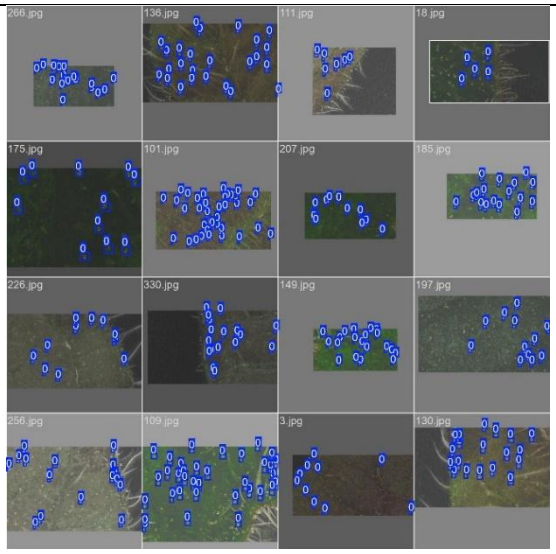
Dengan melihat `train_batch.jpg`, pengguna dapat memantau progres pelatihan model dari waktu ke waktu. `train_batch.jpg` sangat berguna untuk *debugging*. Misalnya, jika model tidak berkinerja seperti yang diharapkan, gambar ini dapat memberikan petunjuk apakah masalahnya terletak pada data, arsitektur model, ataupun proses pelatihan itu sendiri. Jika *bounding boxes* dan label tampak semakin akurat seiring dengan berjalannya *epoch* pelatihan, ini menunjukkan bahwa model sedang belajar dengan baik. Sebaliknya, jika ada masalah dengan deteksi seperti *bounding boxes* yang salah atau label yang tidak sesuai, sehingga pengguna dapat melakukan penyesuaian pada *hyperparameters* atau memperbaiki dataset.

Berikut beberapa visual *train_batch* dari data train dan validasi selama proses *training* yang sudah dilakukan sebanyak 100 *epoch.* Selama proses *training* menampilkan masing-masing 6 *training_batch* dari data train dan data validasi.

Tabel 4.5 Visualisasi *train batch*

Train Batch	***Validation Batch***
	

Train Batch	*Validation Batch*
	
	
	

Train Batch	*Validation Batch*
	
	

Gambar yang ditampilkan pada ‘train_batch.jpg` dan val_batch.jpg` merupakan sampel gambar yang random, artinya setiap gambar yang ditampilkan merupakan gambar yang berbeda-beda sehingga memungkinkan terjadi perubahan posisi dan jumlah *bounding box.*

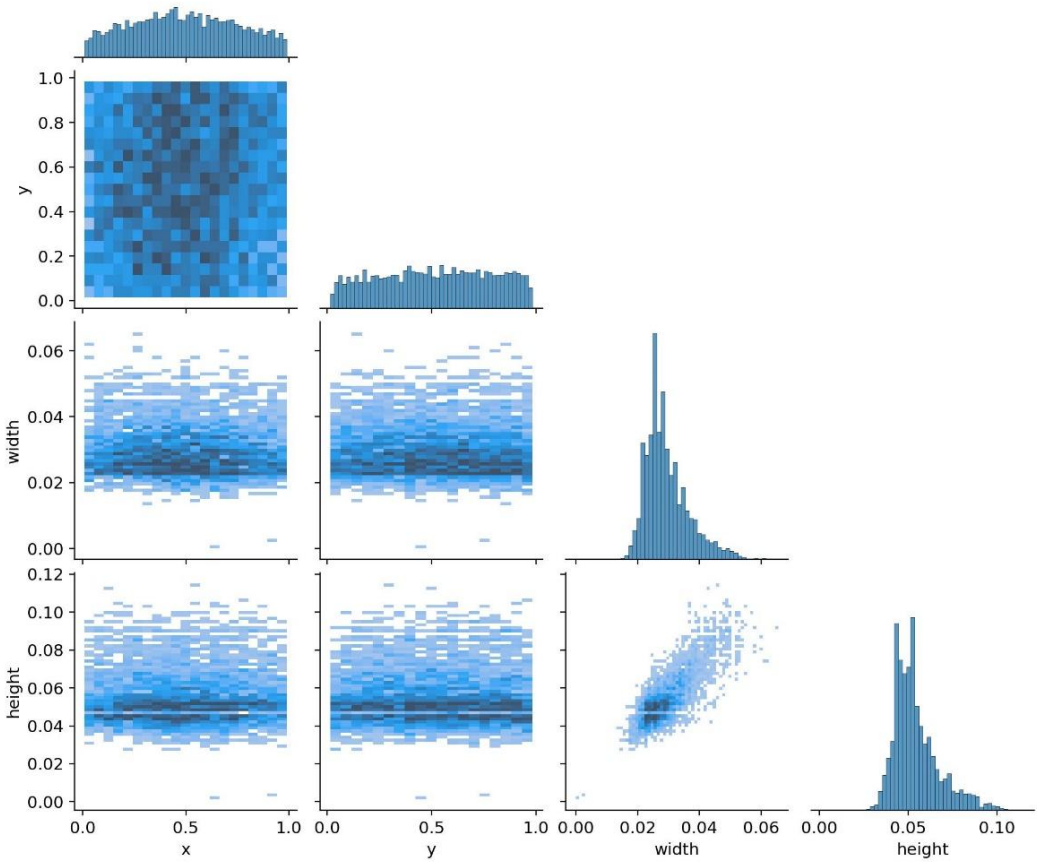
Dari visualisasi *train batch* dan *validation batch* yang ditampilkan, dapat diketahui bahwa deteksi *bounding boxes* tampak sesuai dan semakin akurat seiring dengan berjalannya *epoch* pelatihan, ini menunjukkan bahwa model sedang belajar dengan baik.

4.4. Metrik Evaluasi

4.4.1 *Correlogram*

Pada penelitian ini, saat proses *training* selesai model YOLO menampilkan beberapa visualisasi salah satunya yaitu korelogram. *Correlogram* atau korelogram merupakan

visualisasi yang menunjukkan hubungan korelasi antara beberapa variabel. Dalam konteks *machine learning* dan analisis data, korelogram digunakan untuk menggambarkan tingkat dan arah korelasi antara pasangan variabel, yang dapat membantu dalam mengidentifikasi pola dan hubungan dalam data.



Gambar 4.9 Visualisasi korelogram

Korelogram yang ditampilkan terdiri dari 4 elemen yaitu *x*, *y*, *width* dan *height*. Elemen ini berkaitan dengan ukuran dan lokasi dari *bounding box*. Dalam konteks deteksi objek, model YOLO mengidentifikasi objek dalam gambar dan menentukan *bounding box* yang mencakup objek tersebut. Berikut penejelasan dari masing-masing elemen:

- *x*: Koordinat *X* dari pusat *bounding box*.
- *y*: Koordinat Y dari pusat *bounding box*.
- *width*: Lebar *bounding box*.
- *height*: Tinggi *bounding box*.

Dari visualisasi pada gambar 4.9 dapat dilihat bahwa terdapat korelasi positif yang cukup kuat antara tinggi dan lebar *bounding box* (elemen *width* dan *height*). Hal ini berarti bahwa ketika lebar *bounding box* meningkat, maka tinggi *bounding box* juga akan

meningkat. Hal ini dikarenakan objek trikoma *glandularis bulbose* memiliki bentuk atau proporsi yang konsisten saat ukurannya berubah.

Untuk lokasi koordinat pusat *bounding box* (elemen *x* dan *y*) tidak memiliki korelasi yang kuat yang artinya bahwa posisi horizontal (*x*) dan vertikal (*y*) dari *bounding box* cenderung tidak bergerak bersama-sama atau terdistribusi secara acak. Ini mungkin terjadi karena objek trikoma *glandularis bulbose* dalam gambar tidak bergerak secara diagonal atau tidak mengikuti pola tertentu.

Begitu juga hubungan antara elemen *x*/*y* dengan *widht*/*height* tidak terlihat memiliki korelasi yang cukup kuat secara positif maupun negatif. Hal ini berarti bahwa tidak ada hubungan antara koordinat posisi objek tarkait dengan ukuran *boinding box*-nya.

4.4.2 Plot *Loss* dan *Mean Average Precision* (*mAP*)

Performa model dari proses *training* digambarkan melalui beberapa metrik evaluasi. Metrik evaluasi merupakan ukuran atau standar yang digunakan untuk menilai kinerja suatu model atau algoritma dalam konteks *machine learning* maupun *deep learning*. Metrik evaluasi dapat digunakan untuk mengukur sejauh mana model dapat mencapai tujuan yang diinginkan, seperti klasifikasi yang akurat, prediksi yang tepat, atau deteksi objek yang efektif. Dalam penelitian ini, ditampilkan beberapa metrik seperti plot *box loss*, *classification loss, dfl loss, precision, mAP* dan *recall*.

Pada model YOLO yang digunakan untuk deteksi objek, terdapat beberapa jenis *loss* yang digunakan untuk mengukur kesalahan prediksi selama proses *training*. Berikut penjelasan lengkap mengenai beberapa jenis *loss* tersebut:

a. *Train/Box Loss* (*Loss* Kotak Pelatihan)

Train/Box_loss merupakan metrik yang mengukur kesalahan prediksi dalam menentukan posisi dan ukuran *bounding box* yang mengelilingi objek yang terdeteksi. Metrik ini sangat penting untuk memastikan bahwa model dapat dengan akurat menempatkan *bounding box* di sekitar objek dengan ukuran dan lokasi yang benar. Kesalahan ini biasanya dihitung menggunakan metode seperti *Mean Squared Error* (MSE) antara *bounding box* prediksi dan *bounding box* sebenarnya (*ground truth*).

b. *Train/Cls Loss* (*Loss* Klasifikasi Pelatihan)

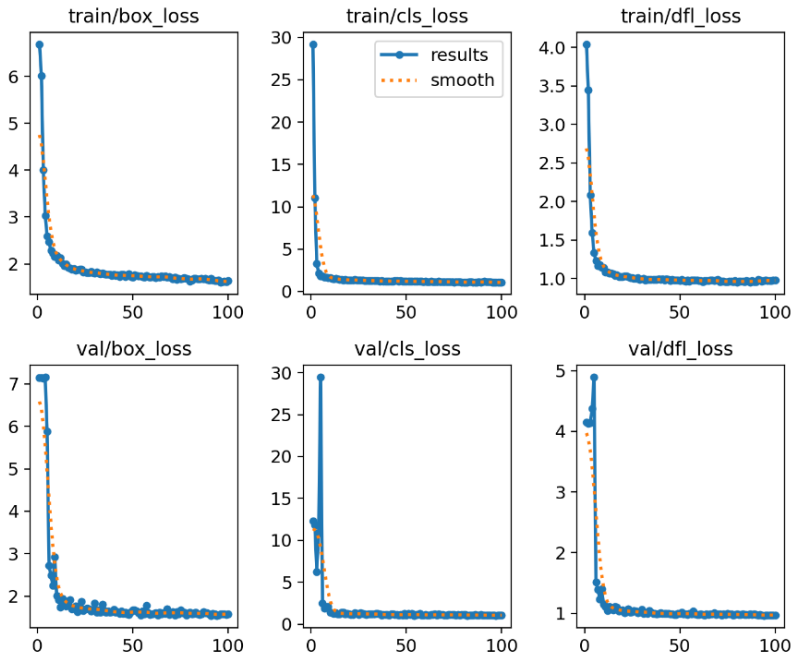
Train/Cls Loss adalah metrik yang mengukur kesalahan dalam prediksi kelas objek dalam *bounding box*. Setiap *bounding box* yang diprediksi tidak hanya memberikan koordinat lokasi dan ukuran, tetapi juga prediksi kelas objek yang ada di dalam kotak tersebut (misalnya, apakah objek tersebut adalah mobil, anjing, manusia, dll.). *Train/Cls*

Loss menghitung kesalahan antara label kelas yang diprediksi oleh model dan label kelas yang sebenarnya (*ground truth*). *Loss* ini membantu model untuk meningkatkan akurasi klasifikasi objek yang ada di dalam *bounding box*. *Loss* ini dihitung berdasarkan perbedaan antara prediksi kelas objek dan kelas sebenarnya, biasanya menggunakan metode seperti *Cross-Entropy Loss*.

c. *Train/DFL Loss* (*Loss* Distribusi untuk Fungsi Logistik Pelatihan)

Train/DFL Loss adalah metrik yang mengukur kesalahan dalam distribusi yang dihasilkan oleh model. DFL (*Distribution Focal Loss*) adalah varian dari *focal loss* yang digunakan untuk menyesuaikan distribusi output dari model. DFL dirancang untuk menangani ketidakseimbangan kelas dan meningkatkan presisi untuk kelas minoritas dengan memfokuskan pelatihan pada contoh-contoh yang sulit. *Train/DFL Loss* menghitung kesalahan distribusi output yang dihasilkan oleh model dan distribusi yang sebenarnya. Ini membantu dalam menyesuaikan distribusi output untuk lebih sesuai dengan distribusi target. *Loss* ini digunakan untuk mengurangi kesalahan dalam distribusi prediksi dan membantu model untuk lebih fokus pada prediksi yang sulit atau kurang terwakili dalam data pelatihan.

Dari visualisasi dari gambar 4.10 dapat dilihat bahwa *train/box loss*, *train/cls loss* dan *train/dfl loss* mengalami penurunan yang sangat signifikan pada 10-20 *epoch* pertama, namun pada *epoch* 30-100 sudah tidak menunjukan penurunan yang signifikan walaupun tetap ada perubahan yg konstan. Hal ini menunjukkan bahwa proses *training* berjalan dengan cukup baik yang ditandai dengan metrik *loss* yang semakin mengecil.



Gambar 4.10 Plot *loss function*

Untuk detail nilai *loss* pada setiap metrik disajikan pada tabel 4.6, pada tabel ini penulis menampilkan nilai *loss* pada setiap 10 iterasi (*epoch*). Dari nilai-nilai tersebut dapat dipastikan bahwa model tidak mengalami *overfitting* yang ditandai dengan nilai *loss* yang terus mengecil meskipun tidak terlalu signifikan.

Tabel 4.6 Detail nilai *box loss, cls loss* dan *dfl loss*

Epoch	Box Loss	Cls Loss	DFL Loss
1	6,6873	29,202	4,0418
10	2.1736	1.5998	1.1446
20	1.8595	1.3711	1.0218
30	1.8048	1.2864	0.99852
40	1.7759	1.2387	0.99015
50	1.7188	1.2043	0.97118
60	1.7315	1.1836	0.97211
70	1.7087	1.1468	0.98791
80	1.6381	1.1008	0.95157
90	1.659	1.1173	0.95701
100	1.6322	1.0823	0.94855

Selanjutnya terdapat metrik *precision, recall* dan *mean average precision* (mAP). Berikut adalah penjelasan lengkap mengenai masing-masing metrik tersebut yang muncul setelah proses training model deteksi objek menggunakan YOLO:

a. *metrics/precision(B)*

Precision adalah metrik yang mengukur seberapa banyak prediksi yang benar dari keseluruhan prediksi yang dibuat oleh model. Dalam konteks deteksi objek, *precision* adalah proporsi dari prediksi *bounding box* yang benar-benar berisi objek dari semua prediksi *bounding box* yang dibuat. *Precision* penting karena menunjukkan akurasi prediksi model ketika mendeteksi objek. Tinggi rendahnya *precision* memberi tahu kita seberapa baik model dalam menghindari *false positives* (prediksi salah yang mengira ada objek padahal tidak ada). *Precision* tinggi berarti model sangat baik dalam memastikan bahwa apa yang dideteksi memang objek yang sebenarnya. *Precision* rendah menunjukkan bahwa model sering salah mendeteksi objek (banyak *false positives*).

b. *metrics/recall(B)*

Recall adalah metrik yang mengukur seberapa banyak objek yang benar-benar terdeteksi dari semua objek yang ada di gambar. Dalam deteksi objek, *recall* adalah proporsi dari objek yang benar-benar terdeteksi dari semua objek yang ada dalam gambar. *Recall* penting untuk memahami seberapa baik model dalam mendeteksi semua objek yang ada dalam gambar. Ini mengukur kemampuan model dalam menghindari *false negatives* (objek yang ada tapi tidak terdeteksi). *Recall* tinggi berarti model sangat baik dalam mendeteksi hampir semua objek yang ada dalam gambar. *Recall* rendah menunjukkan bahwa model sering melewatkan objek yang sebenarnya ada dalam gambar (banyak *false negatives*).

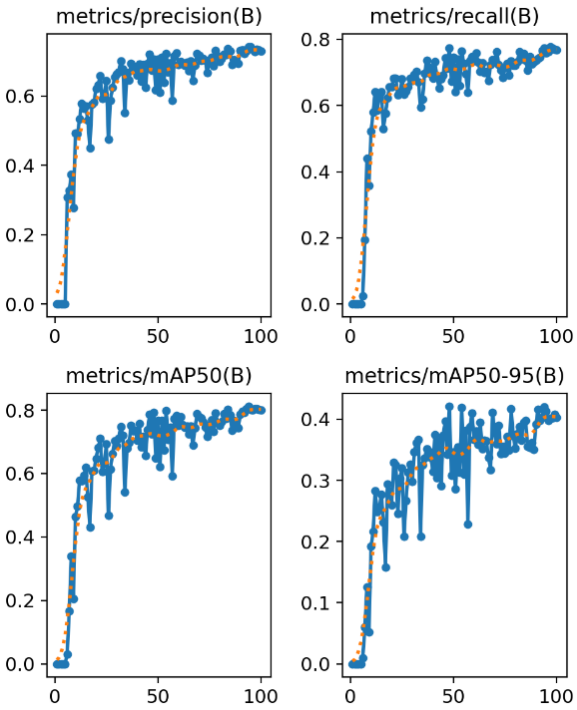
c. *metrics/mAP50(B)*

Mean Average Precision at IoU 0.50 (mAP50) adalah rata-rata dari *precision* di berbagai *threshold* deteksi pada IoU (*Intersection over Union*) *threshold* 0.50. IoU 0.50 berarti bahwa prediksi *bounding box* harus memiliki setidaknya 50% *overlap* dengan *ground truth bounding box* untuk dianggap sebagai deteksi yang benar. *mAP50* adalah metrik yang sering digunakan untuk mengevaluasi kinerja keseluruhan dari model deteksi objek pada IoU *threshold* yang ditentukan. *mAP50* tinggi menunjukkan bahwa model sangat baik dalam mendeteksi objek dengan tingkat *overlap* yang cukup antara prediksi dan *ground truth*. *mAP50* rendah berarti model memiliki banyak prediksi dengan *overlap* kurang dari 50% dengan *ground truth*, sehingga kinerja deteksi secara keseluruhan kurang baik.

d. metrics/mAP50-95(B)

Mean Average Precision at IoU 0.50 to 0.95 (mAP50-95) adalah rata-rata dari *precision* di berbagai *threshold* deteksi di rentang IoU dari 0.50 hingga 0.95. Ini memberikan gambaran yang lebih komprehensif tentang kinerja model pada berbagai tingkat ketelitian deteksi. mAP50-95 adalah metrik yang lebih ketat dan komprehensif dibandingkan mAP50, karena mengevaluasi model pada berbagai tingkat *overlap* dari 50% hingga 95%. mAP50-95 tinggi menunjukkan bahwa model memiliki kinerja deteksi yang sangat baik di berbagai tingkat ketelitian *overlap* antara prediksi dan *ground truth*. mAP50-95 rendah menunjukkan bahwa model mungkin hanya baik pada beberapa tingkat *overlap* tapi tidak secara konsisten baik di berbagai tingkat *overlap* yang lebih tinggi.

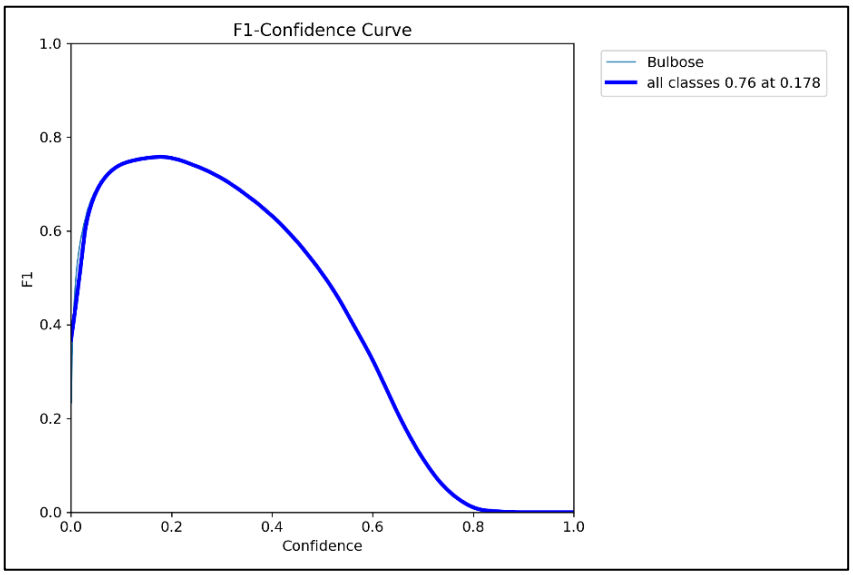
Dari visualisasi dari gambar 4.11 dapat dilihat bahwa metrik *precision, recall,* mAP50 dan mAP50-90 mengalami peningkatan yang sangat signifikan pada 10-20 epoch pertama, namun pada epoch 30-100 sudah tidak menunjukan kenaikan yang signifikan walaupun tetap ada perubahan yg cukup konstan. Hal ini menunjukkan bahwa proses *training* berjalan dengan cukup baik yang ditandai dengan metrik yang semakin meningkat.



Gambar 4.11 Plot *precision*, *recall* dan *mean average precision* (mAP)

4.4.3 *F1Curve*

F1-Confidence Curve menunjukkan hubungan antara nilai *confidence threshold* dan F1 *score*. F1 *score* adalah metrik yang menggabungkan *precision* dan *recall*, memberikan gambaran tunggal tentang performa model. Pada *F1-Confidence Curve* sumbu *x* menunjukkan nilai *confidence threshold*, sumbu *y* menunjukkan *F1 score*. Titik puncak pada kurva menunjukkan nilai *confidence threshold* di mana model mencapai keseimbangan terbaik antara *precision* dan *recall*.



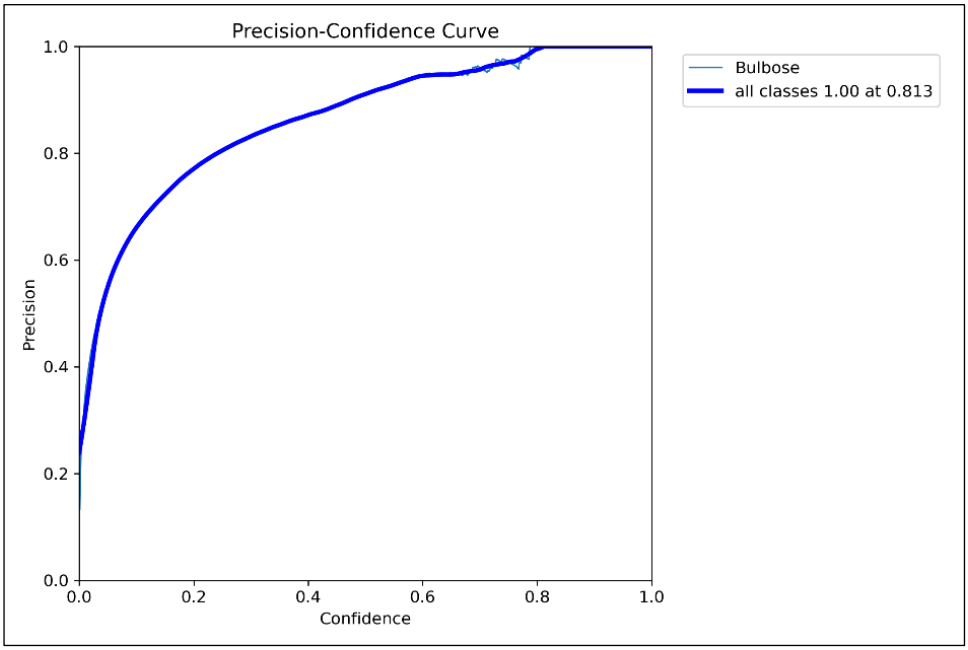
Gambar 4.12 Plot F1-*Confidence Curve*

Pada gambar 4.12 dapat dilihat bahwa nilai F1 *score* tertinggi yang dicapai oleh model adalah 0.76. F1 *score* adalah metrik yang menggabungkan *precision* dan *recall*, sehingga nilai 0.76 menunjukkan performa model dalam mendeteksi objek dengan mempertimbangkan keduanya.

Kemudian nilai *confidence threshold* yang pada titik tersebut model mencapai F1 score 0.76 adalah 0.178. *Confidence threshold* adalah ambang batas untuk menentukan apakah suatu prediksi dianggap sebagai deteksi yang valid atau tidak. Pada nilai *confidence threshold* 0.178, model mencapai keseimbangan terbaik antara *precision* dan *recall*, menghasilkan F1 score 0.76.

4.4.4 *Precision-Confidence Curve*

Precision-Confidence Curve menunjukkan bagaimana *precision* berubah saat nilai *confidence threshold* meningkat. Sumbu *x* menunjukkan nilai *confidence threshold* dan sumbu *y* menunjukkan *precision*. Kurva ini menunjukkan seberapa akurat model saat nilai *confidence threshold* meningkat. Semakin tinggi nilai *precision* pada nilai *confidence threshold* tertentu, semakin sedikit *false positives* yang dihasilkan.



Gambar 4.13 Plot *Percision-Confidence Curve*

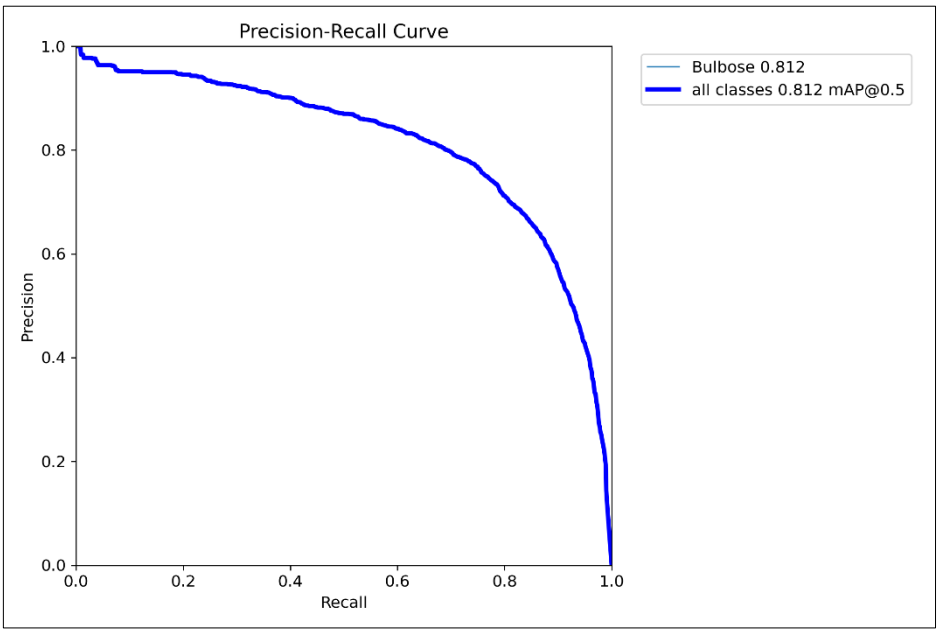
Pada gamabr 4.13 dapat dilihat bahwa nilai *precision* tertinggi yang dicapai oleh model yaitu 1,00. *Precision* adalah metrik yang mengukur proporsi dari prediksi positif yang benar-benar benar (*true positives*) dibandingkan dengan semua prediksi positif (*true*

positives + *false positives*). Nilai 1.00 menunjukkan bahwa pada *threshold* tersebut, semua prediksi positif model adalah benar (tidak ada *false positives*).

Nilai *confidence threshold* yang pada titik tersebut model mencapai precision 1.00 yaitu 0.813. Pada nilai *confidence threshold* 0.813, model menganggap prediksi dengan *confidence score* di atas 0.813 sebagai deteksi yang valid, sehingga mencapai *precision* 1.00. Pada *confidence threshold* 0.813, model sangat selektif dalam membuat prediksi, hanya menganggap prediksi dengan *confidence score* yang sangat tinggi sebagai valid. Ini mengurangi kemungkinan *false positives*, sehingga *precision* mencapai 1.00.

4.4.5 *Precision-Recall Curve*

Precision-Recall Curve menunjukkan hubungan antara *precision* dan *recall* pada berbagai nilai *threshold*. Sumbu *x* menunjukkan *recall* dan sumbu *y* menunjukkan *precision*. *Recall* adalah proporsi dari total objek positif yang terdeteksi (TP / (TP + FN)). Kurva ini membantu dalam memahami *trade-off* antara *precision* dan *recall*. Titik pada kurva yang dekat dengan sudut kanan atas (tinggi *precision* dan tinggi *recall*) menunjukkan performa model yang baik.



Gambar 4.14 Plot *Percision-Recall Curve*

Pada gambar 4.14 dapat dilihat bahwa nilai *mean Average Precision* (mAP) yang dihitung untuk semua kelas objek pada nilai *threshold* = 0,5 adalah 0.812, yang artinya model mampu mencapai nilai mAP sebesar 0,812 pada *confidence treshold* = 0,5. Kurva yang melengkung mendekati sudut kanan atas dari plot (*precision* dan *recall* tinggi) menunjukkan bahwa model memiliki kinerja yang baik. Sebaliknya, kurva yang lurus

mendekati garis diagonal (*precision* dan *recall* rendah) menunjukkan kinerja yang lebih buruk.

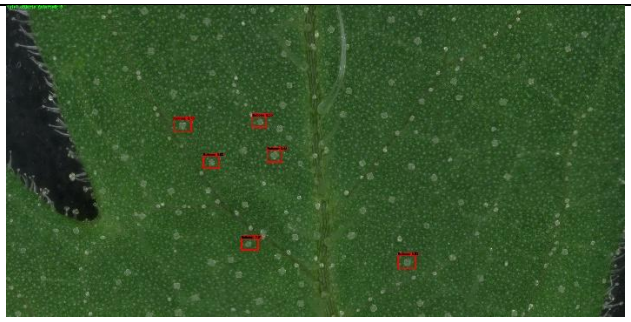
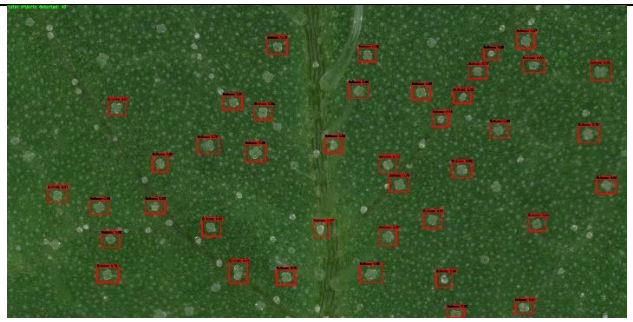
4.5. Implementasi Model

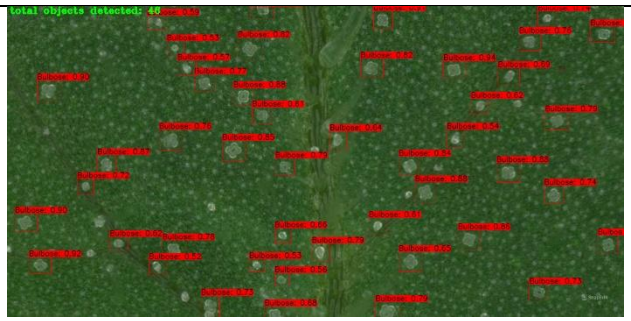
4.5.1 Persiapan Data Uji

Setelah proses *training* model YOLOv8 selesai, langkah selanjutnya adalah melakukan pengujian model dengan menggunakan data uji yang terpisah. Data uji yang digunakan dalam penelitian ini terdiri dari gambar trikoma yang tidak termasuk dalam dataset pelatihan. Setiap gambar telah melalui proses anotasi manual untuk memastikan akurasi pengujian.

Implementasi model juga akan dilakukan terhadap beberapa dimensi grid untuk melihat dimensi grid mana yang menghasilkan visual trikoma paling ideal. Perbandingan implementasi dilakukan pada sampel gambar dan area daun yang sama, namun dengan ukuran visual trikoma yang berbeda yang dihasilkan dari setiap dimensi grid yang berbeda. Perbandingan implementasi model pada berbagai dimensi grid ditampilkan pada tabel 4.7 berikut.

Tabel 4.7 Hasil implementasi model pada berbagai dimensi grid

Dimensi	Sampel Sub-Gambar	Trikoma Terdeteksi	Trikoma Data Truth	Akurasi
2 × 2		6	62	9,6%
3 × 3		40	57	70,2%

4 × 4		46	49	93,8%
5 × 5		36	40	90%
6 × 6		28	31	90,3%

Berdasarkan hasil percobaan pada tabel 4.7 dapat dilihat bahwa ukuran dimensi grid yang kecil (2 × 2 dan 3 × 3) tidak efektif untuk digunakan karena ukuran visual trikoma yang dihasilkan setiap sub-gambar masih terlalu kecil. Hal ini menyebabkan masih banyak *miss* deteksi dan banyak trikoma yang tidak dapat dideteksi oleh model yaitu hanya 9% trikoma yang terdeteksi untuk dimensi 2 × 2 dan 70% untuk dimensi 3 × 3 dari keseluruhan keberadaan trikoma pada data *truth* pada setiap hasil sampel sub-gambar.

Sedangkan untuk dimensi grid yang lebih besar (4 × 4, 5 × 5 dan 6 × 6) dapat menghasilkan akurasi yang lebih baik dengan dimensi grid 4 × 4 yang menghasilkan akurasi yang paling tinggi yaitu sbeesar 93,8% trikoma yang dapat dideteksi. Selain itu dimensi grid 4 × 4 juga menghasilkan sekitar 2500 sub-gambar, jumlah ini jauh lebih sedikit jika dibandingkan dengan dimensi 5 × 5 dan 6 × 6 yang menghasilkan 3900 dan 5700 sub-gambar. Hal ini dapat membuat proses labeling dan *training* dapat berjalan dengan lebih efisien. Dimensi 4 × 4 dapat membuat keseimbangan antara kesesuaian visual trikoma,

akurasi deteksi dan efisiensi labeling & *training* sehingga dimensi 4 × 4 merupakan dimensi grid untuk partisi yang paling optimal untuk penelitian ini.

4.5.2 Deteksi Trikoma

Setelah data gambar hasil partisi diperoleh, langkah selanjutnya yaitu melakukan deteksi dan implementasi model berdasarkan masing-masing sub-gambar. Karena proses partisi menghasilkan 16 sub-gambar baru, maka pengujian juga akan dilakukan sebanyak 16 kali dan hasil kuantifikasi akan diakumulasikan dari setiap pengujian. Sehingga total akumulasi jumlah trikoma dari 1 ruas gambar daun (gambar asli) dapat dihitung dengan menggunakan rumus berikut.

$$N = \sum_{i=1}^{n} T_i \quad (4)$$

Dengan:

N : Jumlah akumulasi trikoma

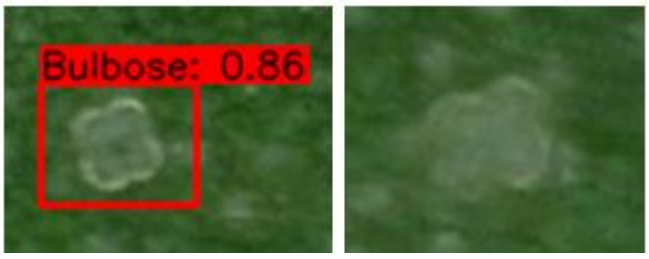
n : Jumlah total sub-gambar yang dihasilkan dari pemecah gambar asli

T_i : Jumlah trikoma yang terdeteksi pada sub-gambar ke-*i*

Visualisasi dari hasil deteksi pada salah satu sampel sub-gambar dapat dilihat melalui gambar 4.15 berikut.

Gambar 4.15 Implementasi deteksi dan kuantifikasi trikoma pada sampe gambar permukaan daun bagian atas (gambar atas) dan bawah (gambar bawah)

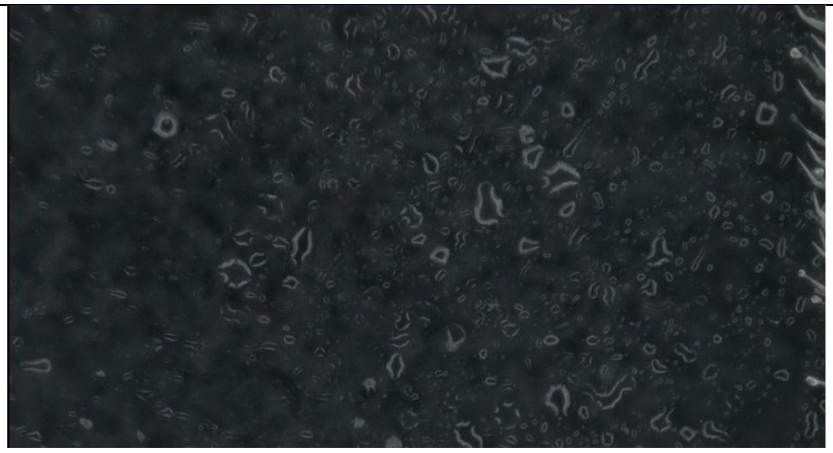
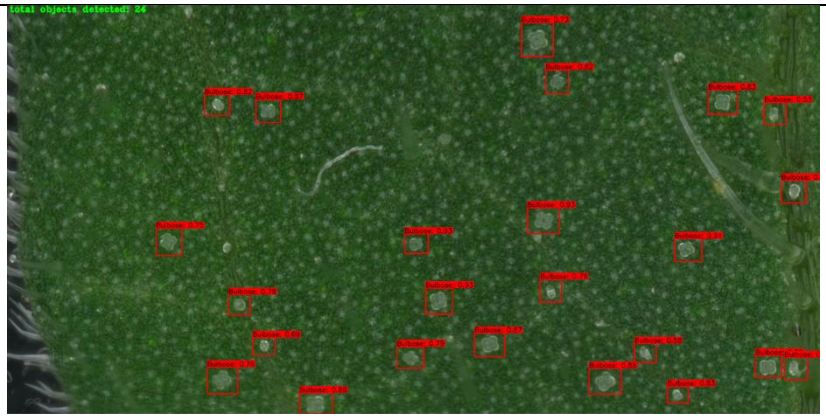
Pada gambar 4.15 dapat dilihat bahwa model yang sudah dilatih dapat melakukan deteksi trikoma tipe *glandularis bulbose* pada gambar daun permukaan atas dan bawah dengan cukup baik dan penempatan *bounding box* cukup presisi. Namun, terlihat beberapa trikoma yang belum berhasil diprediksi atau dideteksi oleh model (*false negative*). Secara bentuk, trikoma yang tidak terdeteksi oleh model tidak memiliki ciri-ciri yang berbeda dengan trikoma yang berhasil dideteksi yang dimana trikoma tersebut memiliki bentuk bengkak dan membulat. Namun trikoma yang tidak terdeteksi cenderung memiliki warna yang lebih pudar dan tidak memiliki garis pinggir yang jelas.

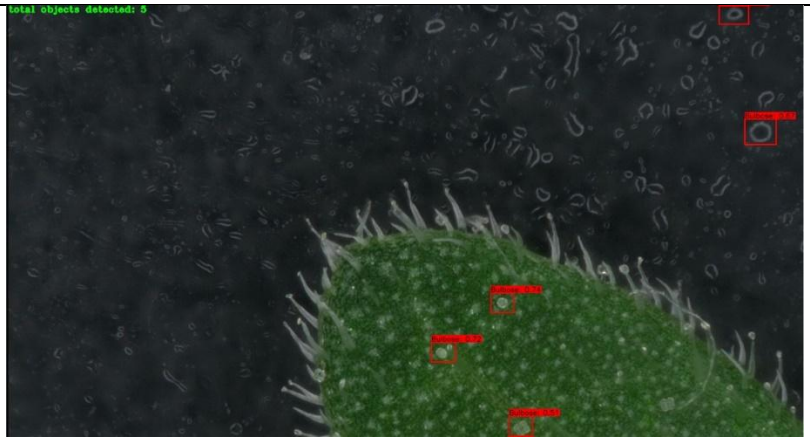


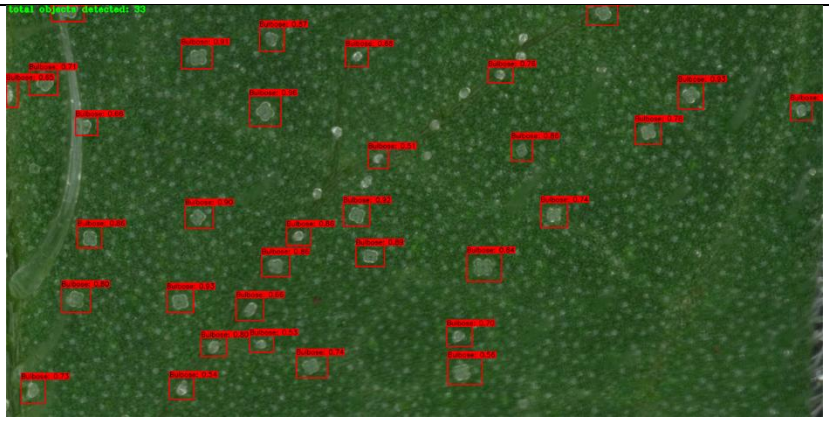
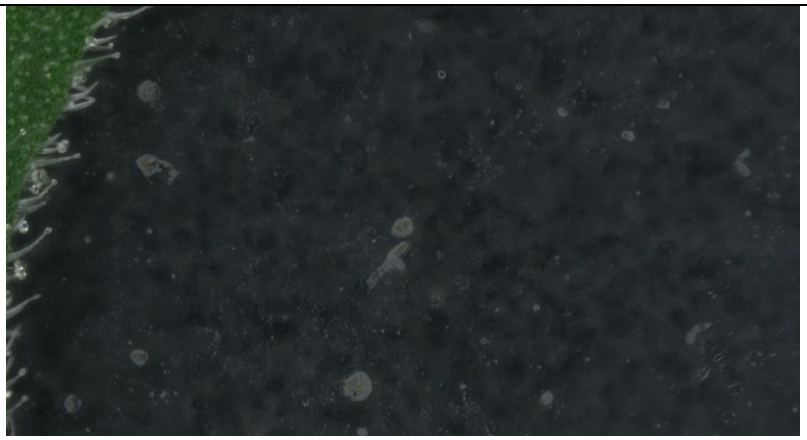
Gambar 4.16 Bentuk dan ciri trikoma yang terdeteksi oleh model (kiri) dan yang tidak terdeteksi (kanan)

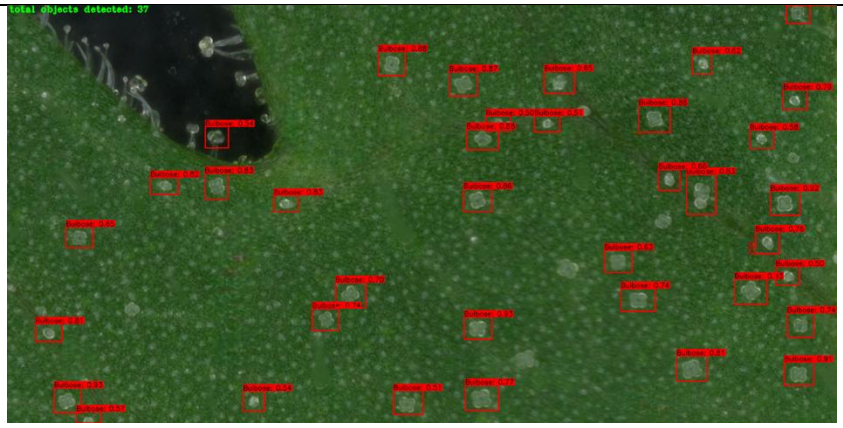
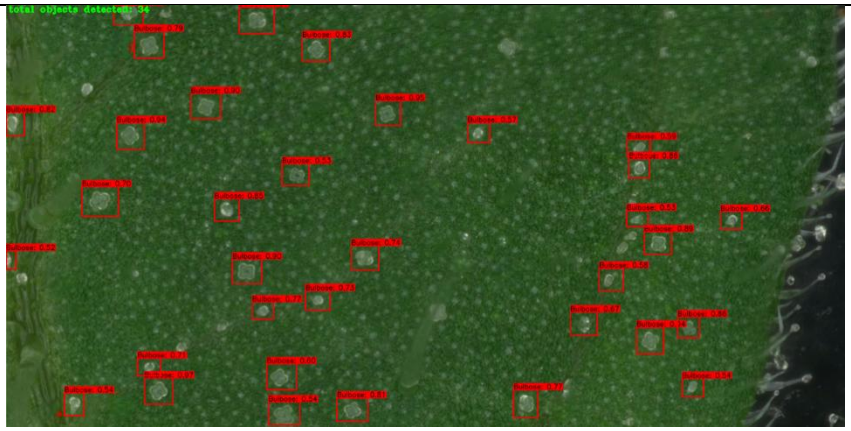
Dalam proses deteksi ini, penulis menambahkan fungsi untuk melakukan kuantifikasi terhadap jumlah objek yang terdeteksi dan menampilkan hasilnya pada bagian sudut kiri atas gambar. Pada hasil kuantifikasi gambar 4.15 dapat dilihat bahwa model mendeteksi sebanyak 24 trikoma pada sampel daun bagian atas dan 12 trikoma pada sampel daun bagian bawah. Untuk pengujian terhadap seluruh sub-gambar pada 1 sampel gambar keseluruhan permukaan daun ditampilkan melalui tabel 4.8.

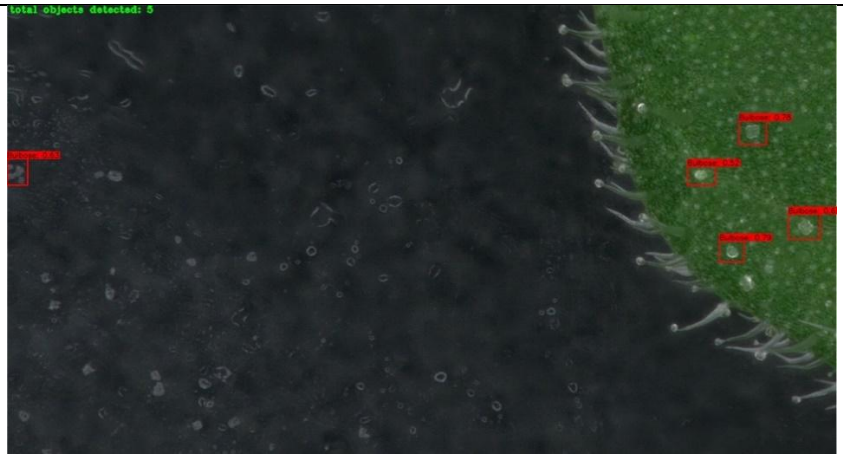
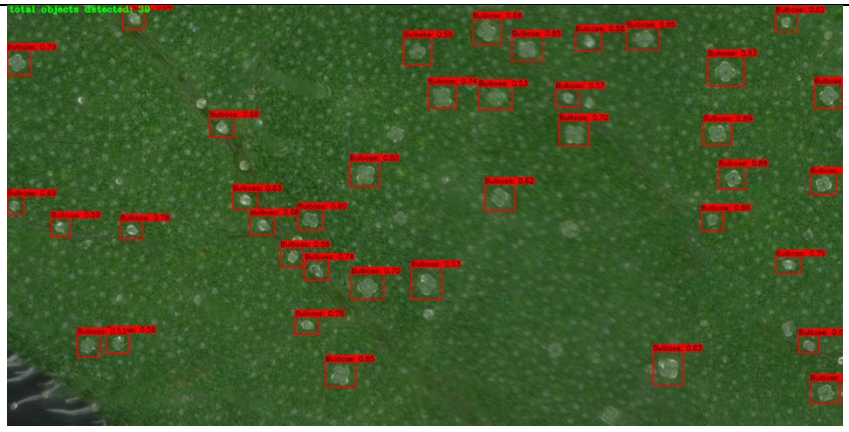
Tabel 4.8 Visualisasi deteksi pada setiap sub-gambar

Sub-Gambar	Visualisasi Deteksi	Trikoma terdeteksi	Trikoma yang tidak terdeteksi	Objek lain yang salah diprediksi sebagai trikoma
Sub-Gambar 1		0	0	0
Sub-Gambar 2		24	2	0
Sub-Gambar 3		20	3	0

Sub-Gambar	Visualisasi Deteksi	Trikoma terdeteksi	Trikoma yang tidak terdeteksi	Objek lain yang salah diprediksi sebagai trikoma
Sub-Gambar 4		0	0	0
Sub-Gambar 5		3	1	2
Sub-Gambar 6		30	3	0

Sub-Gambar	Visualisasi Deteksi	Trikoma terdeteksi	Trikoma yang tidak terdeteksi	Objek lain yang salah diprediksi sebagai trikoma
Sub-Gambar 7		33	8	0
Sub-Gambar 8		0	0	0
Sub-Gambar 9		7	4	0

Sub-Gambar	Visualisasi Deteksi	Trikoma terdeteksi	Trikoma yang tidak terdeteksi	Objek lain yang salah diprediksi sebagai trikoma
Sub-Gambar 10		37	4	0
Sub-Gambar 11		34	4	0
Sub-Gambar 12		0	1	0

Sub-Gambar	Visualisasi Deteksi	Trikoma terdeteksi	Trikoma yang tidak terdeteksi	Objek lain yang salah diprediksi sebagai trikoma
Sub-Gambar 13		4	2	1
Sub-Gambar 14		39	8	0
Sub-Gambar 15		37	8	0

Sub-Gambar	Visualisasi Deteksi	Trikoma terdeteksi	Trikoma yang tidak terdeteksi	Objek lain yang salah diprediksi sebagai trikoma
Sub-Gambar 16		10	5	0
TOTAL		**278**	**53**	**3**

Visualisasi pada tabel 4.8 dapat dilihat bahwa model dapat mendeteksi dengan cukup baik pada seluruh sub-gambar, namun terdapat beberapa *false negatives* yang artinya model melewatkan objek yang sebenarnya adalah trikoma namun tidak dapat terdeteksi. Selain itu pada sub-gambar 5 dan sub-gambar 13 juga terdapat *false positives* yang terdeteksi oleh model, yang artinya terdapat objek yang diprediksi sebagai trikoma namun sebenarnya bukan trikoma.

Dari hasil pengujian yang telah diperoleh, selanjutnya diterapkan kedalam persamaan (5) untuk menghitung jumlah akumulasi trikoma pada seluruh sub-gambar (gamdar asli).

$$N = \sum_{i=1}^{n} T_i$$

$$N = \sum_{i=1}^{16} T_i$$

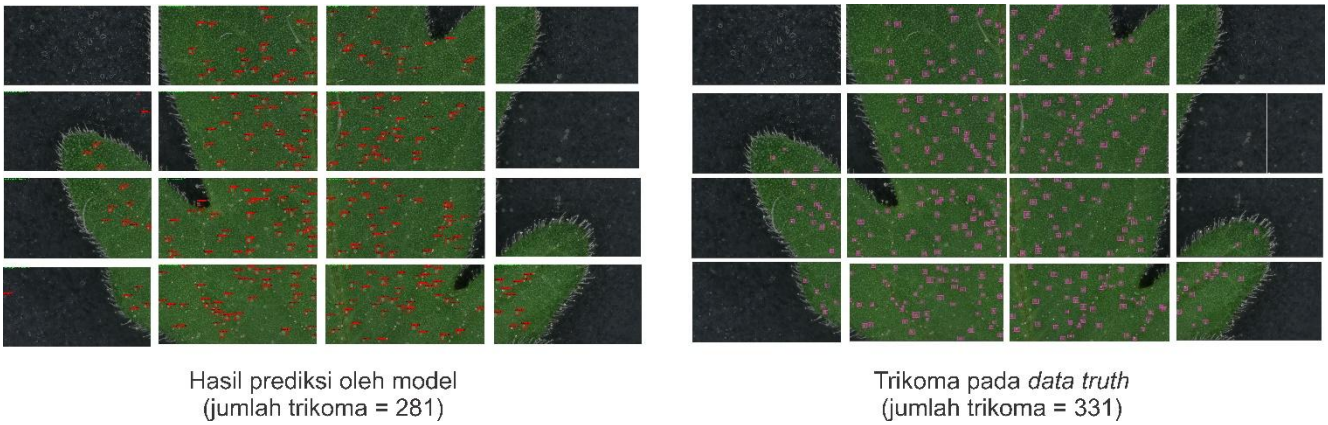
$$N = 0 + 24 + 20 + 0 + 5 + 30 + 33 + 0 + 7 + 37 + 34 + 0 + 5 + 39 + 37 + 10$$

$$N = 281$$

Dari persamaan diatas maka diperoleh hasil dari akumulasi trikoma yang diprediksi oleh model terhadap seluruh sub-gambar yaitu sebanyak 281 trikoma.

4.5.3 Generalisasi Hasil

Setelah melakukan pengujian dan implementasi model, penulis kemudian melakukan generalisasi hasil untuk melihat seberapa baik model dalam mendeteksi trikoma dengan membandingkan data hasil prediksi dengan data sebenarnya (*data truth*).



Gambar 4.17 Perbandingan jumlah trikoma antara hasil prediksi dengan *data truth*

Pada gambar 4.17 dapat dilihat bahwa antara jumlah trikoma yang diprediksi oleh model dengan jumlah trikoma pada *data truth* terdapat perbedaan untuk jumlah prediksi yang benar ataupun salah lebih detail akan disajikan pada tabel 4.9.

Tabel 4.9 Prediksi jumlah

Prediksi	**Jumlah**
True Positive (TP)	278
False Positive (FP)	3
False Negative (FN)	53

Dari tabel 4.9 mengenai detail dari jumlah prediksi pada data *true positive, false negative* dan *false negative* maka dapat dihitung nilai *precision* dan *recall* dengan menggunakan persamaan berikut.

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \times 100\%$$

$$Precision = \frac{278}{278 + 3} \times 100\%$$

$$Precision = \frac{278}{281} \times 100\%$$

$$Precision = 0{,}98 \times 100\%$$

$$\boldsymbol{Precision = 98\%}$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \times 100\%$$

$$Recall = \frac{278}{278 + 53} \times 100\%$$

$$Recall = \frac{278}{331} \times 100\%$$

$$Recall = 0{,}839 \times 100\%$$

$$\boldsymbol{Recall = 83{,}9\%}$$

Dari nilai *precission* yang telah dihitung berarti, dari seluruh gambar yang telah diprediksi ada trikoma terdapat 98% yang benar-benar ada trikoma sedangkan 2% sisanya bukan trikoma. Sedangkan dari nilai *recall* berarti dari semua gambar yang benar-benar ada trikoma, model berhasil mendeteksi sebanyak 83,9% dari gambar tersebut.

Jumlah trikoma pada tanaman memang dapat memengaruhi kualitas perlindungan dan ketahanan tanaman terhadap berbagai faktor lingkungan dan serangan hama. Namun, tidak selalu berarti bahwa semakin banyak trikoma, maka semakin baik juga untuk tanamannya. Faktor-faktor seperti jenis tanaman, lingkungan di mana tanaman tumbuh, dan jenis trikoma itu sendiri memainkan peran penting dalam menentukan manfaat dari trikoma. Misalnya, beberapa tanaman mungkin hanya membutuhkan jumlah trikoma yang moderat untuk melindungi diri dari predator, sementara terlalu banyak trikoma dapat memengaruhi proses fotosintesis atau mengurangi efisiensi tanaman dalam menangkap cahaya matahari.

Dengan adanya sistem ini harapannya dapat menjadi pioner atau langkah awal dalam penelitian identifikasi trikoma berbasis *deep learning*. Model ini dapat melakukan identifikasi trikoma dengan tingkat akurasi hingga 80-90% dan membantu para peneliti sebelum melakukan tindakan lanjutan dengan menyesuaikan terhadap faktor-faktor lain seperti jenis tanaman dan faktor lingkungan.

BAB 5
Kesimpulan dan Saran

5.1. Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan mengenai implementasi model YOOv8 terhadap trikoma *glandularis bulbose* diperoleh beberapa kesimpulan antara lain sebagai berikut:

1. Implementasi model YOLOv8 dalam melakukan deteksi dan kuantifikasi trikoma tipe *gandularis bulbose* pada gambar daun tanaman kentang budidaya dapat dilakukan dengan cukup baik. Dari hasil menggunakan data uji terhadap 1 gambar yang telah dipartisi menjadi beberapa bagian sub-gambar, diperoleh nilai *precision* = 98% dan nilai *recall* = 83,9%.
2. Dari ke 16 sub-gambar yang diuji, terdapat *false positive* = 3 pada sub-gambar 5 dan 13. Yang artinya pada sub-gambar tersebut terdapat 3 objek yang diprediksi sebagai trikoma oleh model padahal sebenarnya bukan trikoma. Selain itu terdapat *false negative* sebanyak 53 yang artinya terdapat 53 objek yang sebenarnya trikoma namun tidak dapat diprediksi oleh model.
3. *Tuning parameter* yang digunakan pada penelitian ini yaitu *epoch* 100, *learning rate* 0,01, *batch size* 16 dan *img size 640*. *Tuning parameter* sangat penting dilakukan agar dapat melakukan *trial and error* hingga menemukan model yang terbaik.
4. Dari hasil proses *training* selama 100 iterasi (*epoch*) diperoleh nilai *mean average precision* (mAP) pada *confidence treshold* = 50 yaitu 0,816. Sedangkan nilai mAP pada *confidence treshold* = 50-90 yaitu 0,38. Untuk nilai *loss* diperoleh *box_loss* = 1,408; *cls_loss* = 0,8149 dan *dfl_loss* = 0,9129.

5.2. Saran

Adapun saran yang diberikan oleh penulis untuk pengembangan dalam penelitian selanjutnya antara lain adalah sebagai berikut:

1. Untuk penelitian selanjutnya dapat melakukan proses *training* dengan kuantitas data gambar yang lebih banyak lagi. Selain itu penelitian juga dapat dikembangkan menjadi beberapa kelas atau tipe trikoma yang lain seperti trikoma *non glandularis*.
2. Kedepan dapat dikembangkan sistem berupa *web apps* ataupun IoT agar sistem deteksi dapat diterapkan dengan lebih luas dan lebih mudah, sehingga dapat membantu para peneliti dan pemulia tanaman dalam melakukan identifikasi trikoma.

3. Keterbatasan sumber daya komputasi dalam melakukan *training* dan evaluasi data membatasi jumlah eksperimen yang dapat dilakukan oleh peneliti. Untuk pengembangan lebih lanjut, dapat melakukan proses *training* dengan *resource* yang lebih tinggi. Penelitian selanjutnya juga dapat menggunakan arsitektur model yang lain untuk dibandingkan atau menggunakan arsitektur yang lebih baru.

Daftar Pustaka

Abdullah, A. H. (2018). *APLIKASI BAHASA PEMROGRAMAN R UNTUK ANALISIS QUANTITATIVE TRAIT LOCI (QTL) TERKAIT KETAHANAN PENYAKIT LAYU PHYTOPHTHORA CAPSICI PADA TANAMAN CABAI (Capsicum annuum).* Diambil kembali dari dspace uii: https://dspace.uii.ac.id/bitstream/handle/123456789/9807/02%20preliminari.pdf?sequence=2&isAllowed=y

Amwin, A. (2021). *DETEKSI DAN KLASIFIKASI KENDARAAN BERBASIS.* Diambil kembali dari DSpace UII: https://dspace.uii.ac.id/bitstream/handle/123456789/34154/17523176%20Aldhiyatika%20Amwin.pdf?sequence=1&isAllowed=y

Banyal, N. A., Surianti, & Dayat, A. R. (2016). KLASIFIKASI CITRA PLASMODIUM PENYEBAB PENYAKIT MALARIA DALAM SEL DARAH MERAH MANUSIA DENGAN MENGGUNAKAN METODE MULTI CLASS SUPPORT VECTOR MACHINE (SVM. *Jurnal Ilmiah ILKOM Volume 8 Nomor 2*.

Bella, M. A. (2021). *IMPLEMENTASI ALGORITMA DEEP LEARNING UNTUK SISTEM DETEKSI KANTUK PADA PENGEMUDI MENGGUNAKAN YOLO.* Diambil kembali dari DSpace UII: https://dspace.uii.ac.id/bitstream/handle/123456789/38792/17522216.pdf?sequence=1&isAllowed=y

Bergau, N., Bennewitz, S., Syrowatka, F., Hause, G., & Tissier, A. (2015). The development of type VI glandular trichomes in the cultivated tomato Solanum lycopersicum and a related wild species S. habrochaites. *BMC Plant Biology*.

Bochkovskiy, A., Arsenyev, A., Arsenyev, A., & Redmon, J. (2020). YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv preprint arXiv*.

Denata, I., Rismawan, T., & Ruslianto, I. (2021). Implementation of Deep Learning for Classification Type of Orange Using The Method Convolutional Neural Network. *Telematika: Jurnal Informatika dan Teknologi Informasi*.

Fronzetti, N. (2019). Predictive Neural Network Applications for Insurance Processes. *Bachelor's Degree internship report*.

Glas, J. J., Schimmel, B. C., Alba, J. M., Bravo, R. E., Schuurink, R. C., & Kant, M. R. (2012). Plant Glandular Trichomes as Targets for Breeding or Engineering of Resistance to Herbivores. *International Journal of Molecular Sciences*.

Hamid, M., Suratin, M., & Usman, A. A. (2021). DETEKSI SEL MALARIA PADA SEDIAAN APUS DARAH BERDASARKAN FITUR MORFOLOGI DAN TEKSTUR MENGGUNAKAN JARINGAN BACKPROPAGATION. *Indonesian Journal on Information System*.

Hanin, M. A., Patmasari,, R., & Fu'adah, R. N. (2021). SISTEM KLASIFIKASI PENYAKIT KULIT MENGGUNAKAN CONVOLUTIONAL NEURAL NETWORK (CNN). *e-Proceeding of Engineering.*

Huang, M., Mi, W., & Wang, Y. (2024). EDGS-YOLOv8: An Improved YOLOv8 Lightweight UAVDetection Model. *Drones*, 4.

Indolia, S., Gosmawi, A. K., Mishra, S. P., & Asopa, P. (2018). Conceptual Understanding of Convolutional Neural Network-A Deep Learning Approach. *International Conference on Computational Intelligence and Data Science (ICCIDS 2018)*.

Irsyam, M., Khirstianto, W., & Diniyah, N. (2023). Pemetaan Jenis Tanaman Menggunakan Pendekatan Machine-Learning dan Citra Sentinel-2: Studi Kasus di Lumajang, Jawa Timur, Indonesia. *INTESI (Seminar Nasional Teknik Sipil)*.

Khatami, M. S. (2022, 05 25). *DETEKSI KENDARAAN MENGGUNAKAN ALGORITMA.* Diambil kembali dari DSpace UII: https://dspace.uii.ac.id/bitstream/handle/123456789/38956/17523142.pdf?sequence=1&isAllowed=y

Lucatti, A. F., Heusden, A. W., Vos, R. C., Visser, R. G., & Vosman, B. (2013). Differences in insect resistance between tomato species endemic to the Galapagos Islands. *BMC Evolutionary Biology*.

Ma, B., Hua, Z., Wen, Y., Deng, H., Zhao, Y., Pu, L., & Song, H. (2024). Using an improved lightweight YOLOv8 model for real-time detection of multi-stage apple fruit in complex orchard environments. *Artificial Intelligence in Agriculture*.

Miftakhurrokhmat. (2023, 01). *Presensi Kelas Berbasis Pola Wajah Dan Tersenyum Menggunakan Deep Learning.* Diambil kembali dari DSpace UII: https://dspace.uii.ac.id/handle/123456789/42509

Ni, X., Li, C., Jiang, H., & Takeda, F. (2020). Deep learning image segmentation and extraction of blueberry fruit traits associated with harvestability and yield . *Horticulture Research*.

Pailus, M. (2022, 10). *Pengembangan Model Identifikasi Penyakit Pada Tanaman Padi Berbasiskan Faster Rcnn.* Diambil kembali dari DSpace UII: https://dspace.uii.ac.id/handle/123456789/41487

Perdani, W. R., Magdalena, R., & Pratiwi, N. K. (2022). Deep Learning untuk Klasifikasi Glaukoma dengan menggunakan Arsitektur EfficientNet. *ELKOMIKA: Jurnal Teknik Energi Elektrik, Teknik Telekomunikasi, & Teknik Elektronika*.

Permatasari, D. I. (2024). IMPLEMENTASI METODE CONVOLUTIONAL NEURAL NETWORK (CNN) UNTUK KLASIFIKASI TANAMAN HERBAL BERDASARKAN CITRA DAUN. *KOHESI (Jurnal Multidisiplin Saintek)*.

Primartha, R. (2018). *Belajar Machine Learning Teori dan Praktik.* Bandung: Informatika.

Purwanto, P., & Sumardi, S. (2022). Perancangan Klasifikasi Tanaman Herbal Menggunakan Transfer Learning Pada Algoritma Convolutional Neural Network (CNN). *Jurnal Ilmiah INFOKAM (Informasi Komputer Akuntansi dan Manajemen)*.

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016, 05 09). *You Only Look Once: Unified, Real-Time Object Detection.* Diambil kembali dari arXiv Cornell University: https://arxiv.org/abs/1506.02640

Rizal, S., Ibrahim, N., Pratiwi, N. K., Saidah, S., & Fu'adah, R. Y. (2020). Deep Learning untuk Klasifikasi Diabetic Retinopathy menggunakan Model EfficientNet. *Elkomika: Jurnal Teknik Energi Elektrik, Teknik Telekomunikasi & Teknik Elektronika*.

Rosalina, R., & Wijaya, A. (2020). Pendeteksian Penyakit pada Daun Cabai dengan Menggunakan Metode Deep Learning. *JuTISI (Jurnal Teknik Informatika dan Sistem Informasi)*.

Satria, A., Badri, R. M., & Safitri, I. (2023). Prediksi Hasil Panen Tanaman Pangan Sumatera dengan Metode Machine Learning. *Digital Transformation Technology (Digitech)*.

Sibarani, J. S., Damanik, S. T., Nurkhalizah, R., Mulyana, S., & Nasution, B. (2023). Klasifikasi Tanaman Hias Menggunakan Algoritma Convolutional Neural Network. *Journal of Information Technology Ampera (Journal ITA)*.

Suyuti, M. (2023, 1 26). *Pengembangan Model Klasifikasi Mata Tertutup Dan Terbuka Dalam Identifikasi Kelelahan Menggunakan Arsitektur Mobile CNN.* Diambil kembali dari Dspace UII: https://dspace.uii.ac.id/handle/123456789/42508

TiaraSari, A., & Haryatmi, E. (2021). Penerapan Convolutional Neural Network Deep Learning dalam Pendeteksian Citra Biji Jagung Kering. *JURNAL RESTI (Rekayasa Sistem dan Teknologi Informasi)*.

Vosman, B., Kashaninia, A., Westende, W. v., Dekens, F. M., Eekelen, H. v., Visser, R. G., . . . Voorrips, R. E. (2019). QTL mapping of insect resistance components of Solanum galapagense. *Theoritical and Applied Genetics*.

Vosman, B., Westende, W. P., Henken, B., Eekelen, H. D., Vos, R. C., & Voorrips, R. E. (2018). Broad spectrum insect resistance and metabolites in close relatives of the cultivated tomato. *Euphytica*.

Wu, J. W., Cai, W., Yu, S. M., & Xu, Z. L. (2020). Optimized visual recognition algorithmin service robots. *International Journal of Advanced Robotic Systems*, 4.

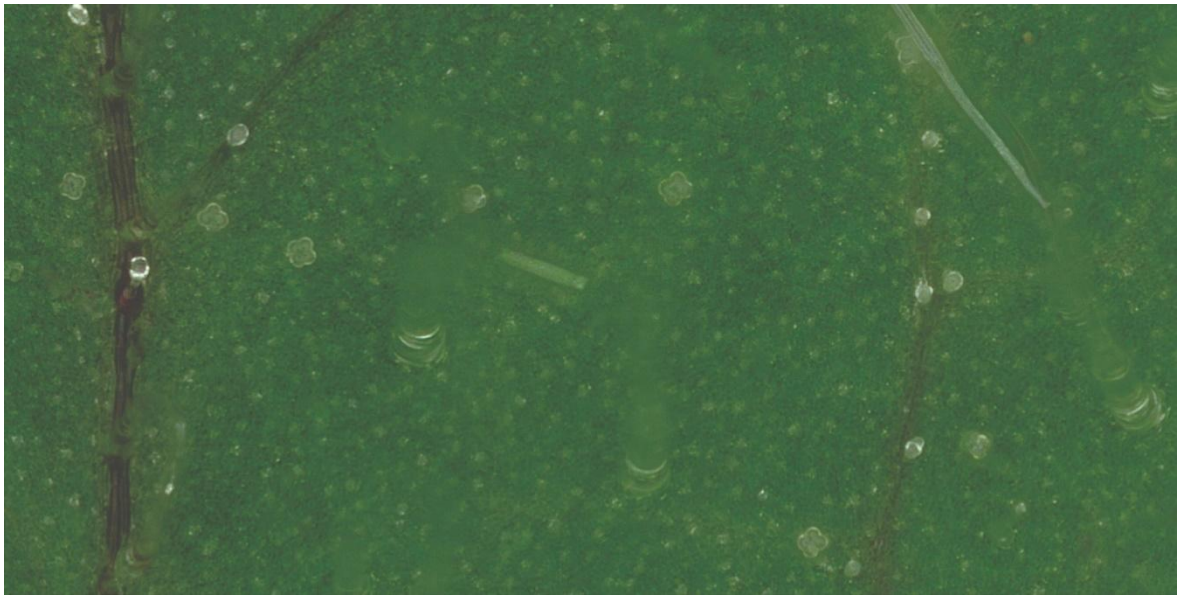
Yanto, B., Fimawahib, L., Supriyanto, A., Hayadi, B. H., & Pratama, R. R. (2021). Klasifikasi Tekstur Kematangan Buah Jeruk Manis Berdasarkan Tingkat Kecerahan Warna Dengan Metode Deep Learning Convolutional Neural Network. *Jurnal INOVTEK Polbeng*.

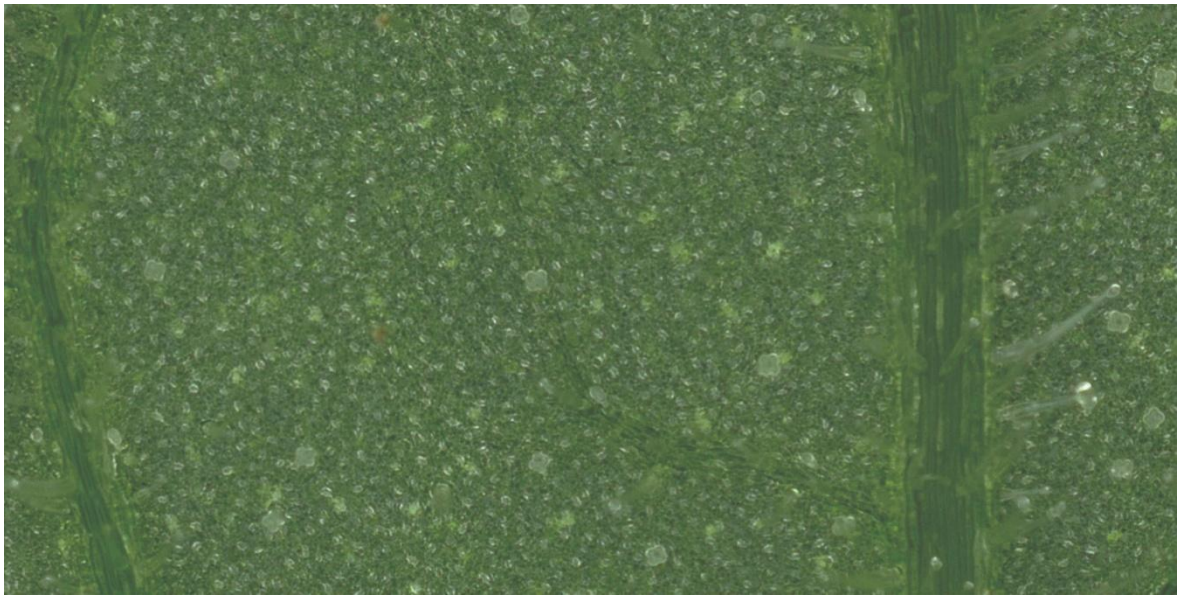
Zakiya, P. N., L. N., & Rizal, S. (2021). KLASIFIKASI PATOLOGI MAKULA RETINA MELALUI CITRA OCT MENGGUNAKAN CONVOLUTIONAL NEURAL NETWORK DENGAN ARSITEKTUR MOBILENET. *e-Proceeding of Engineering : Vol.8, No.5*.

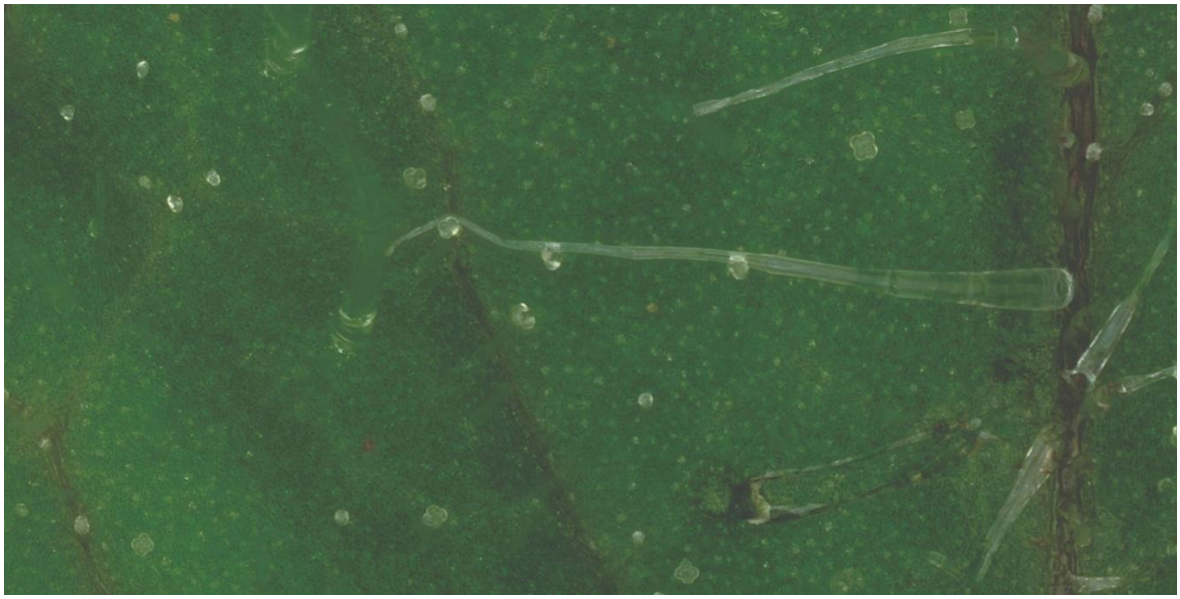
Zhang, Y., Song, H., Wang, X., Zhou, X., Zhang, K., Chen, X., . . . Wang, A. (2020). The Roles of Different Types of Trichomes in Tomato Resistance to Cold, Drought, Whiteflies, and Botrytis. *Journal Agronomy*.

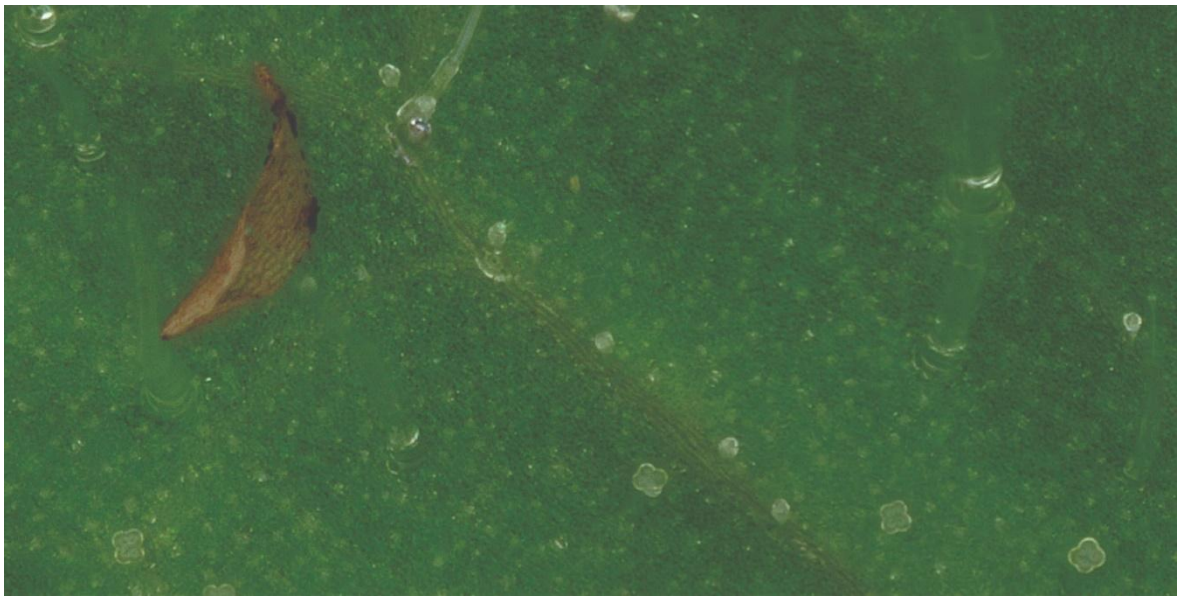
Zou, Z., Chen, K., Shi, Z., Guo, Y., & Ye, J. (2019). *Object Detection in 20 Years: A Survey.* Diambil kembali dari https://arxiv.org/abs/1905.05055

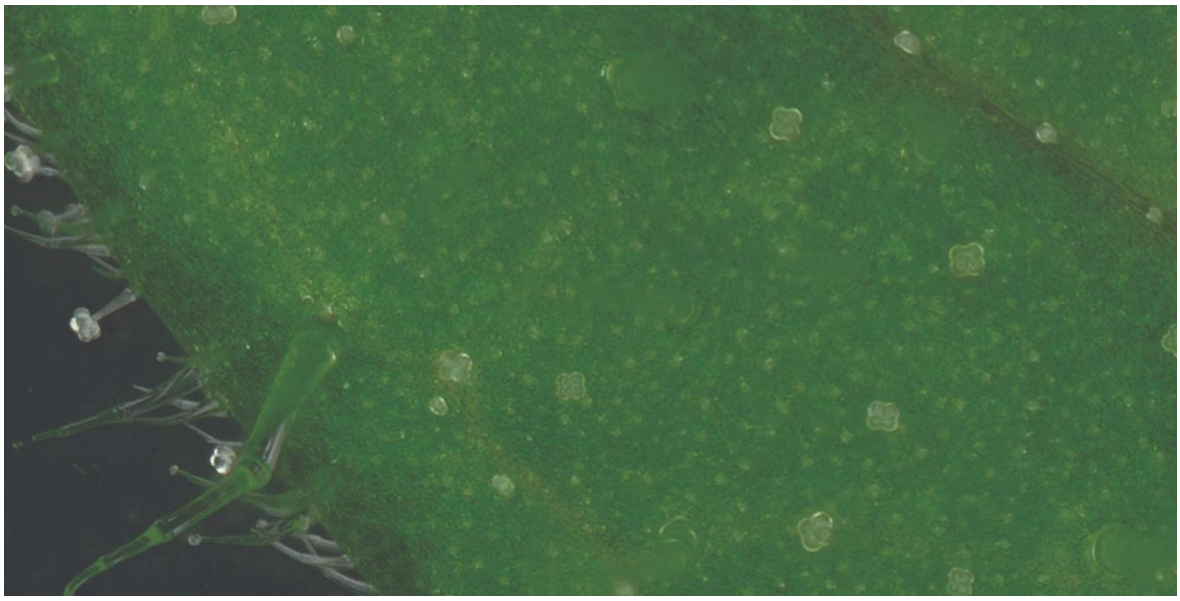
LAMPIRAN 1 (sampel data gambar)

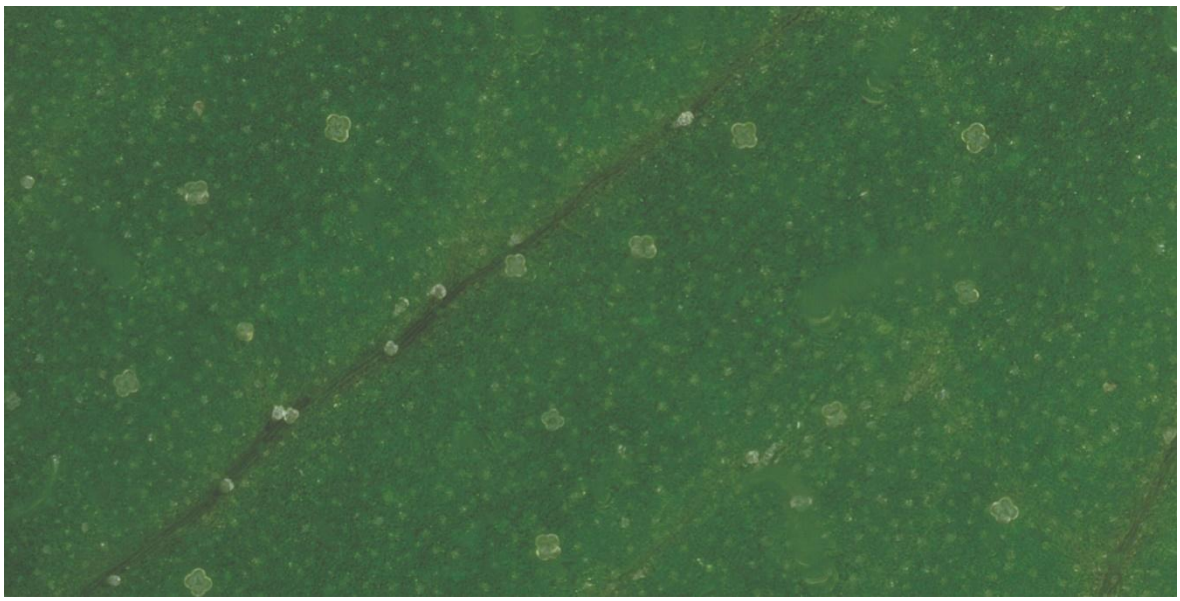




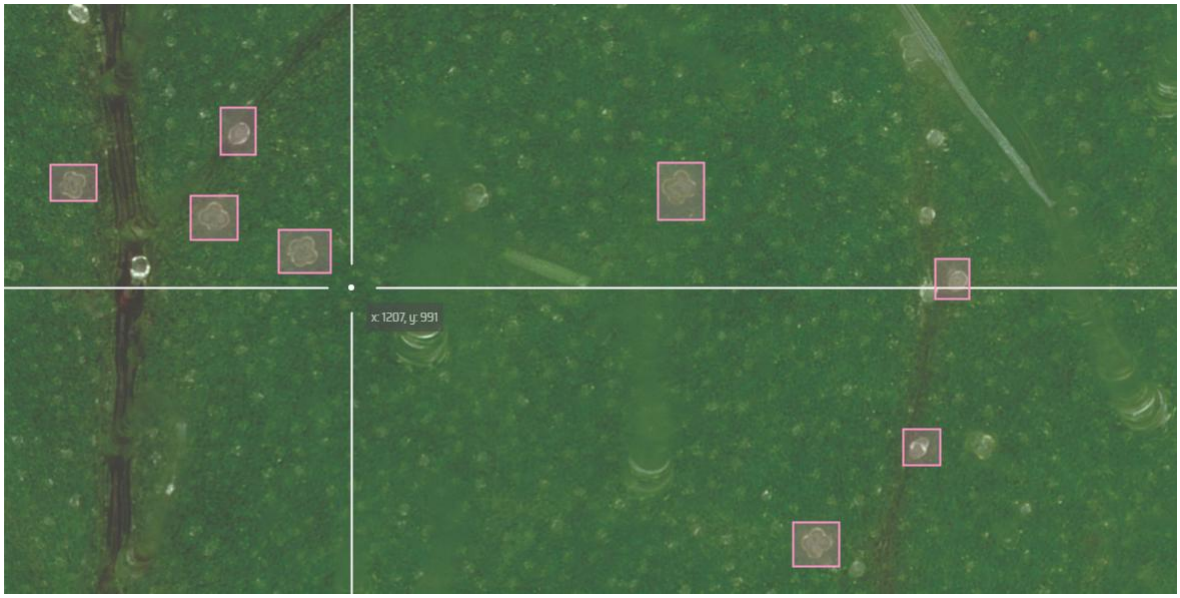


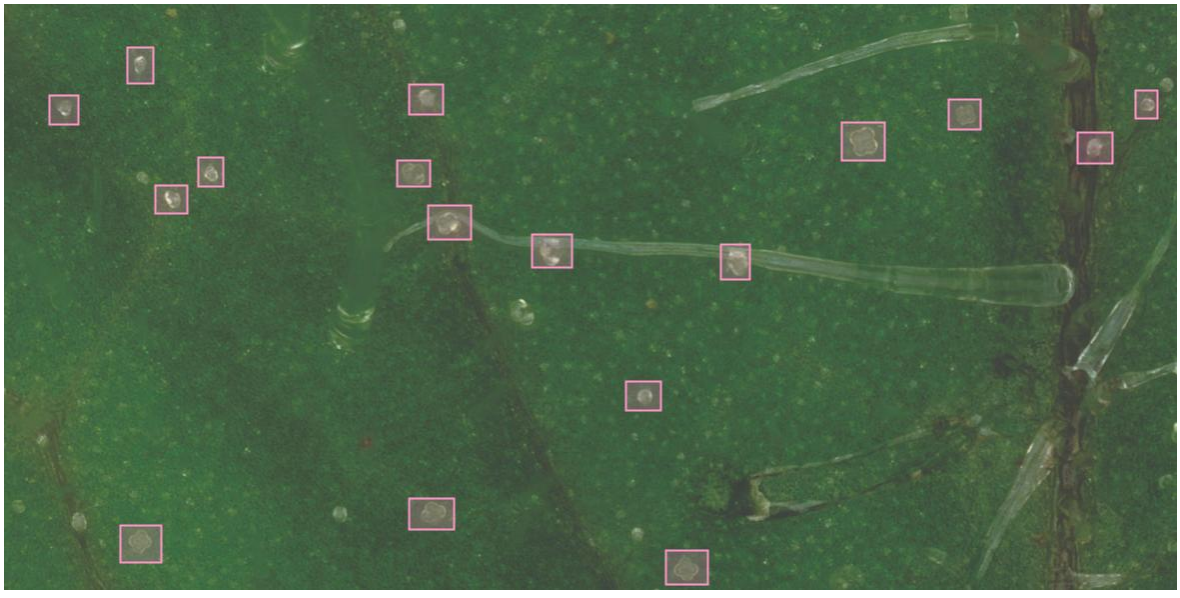


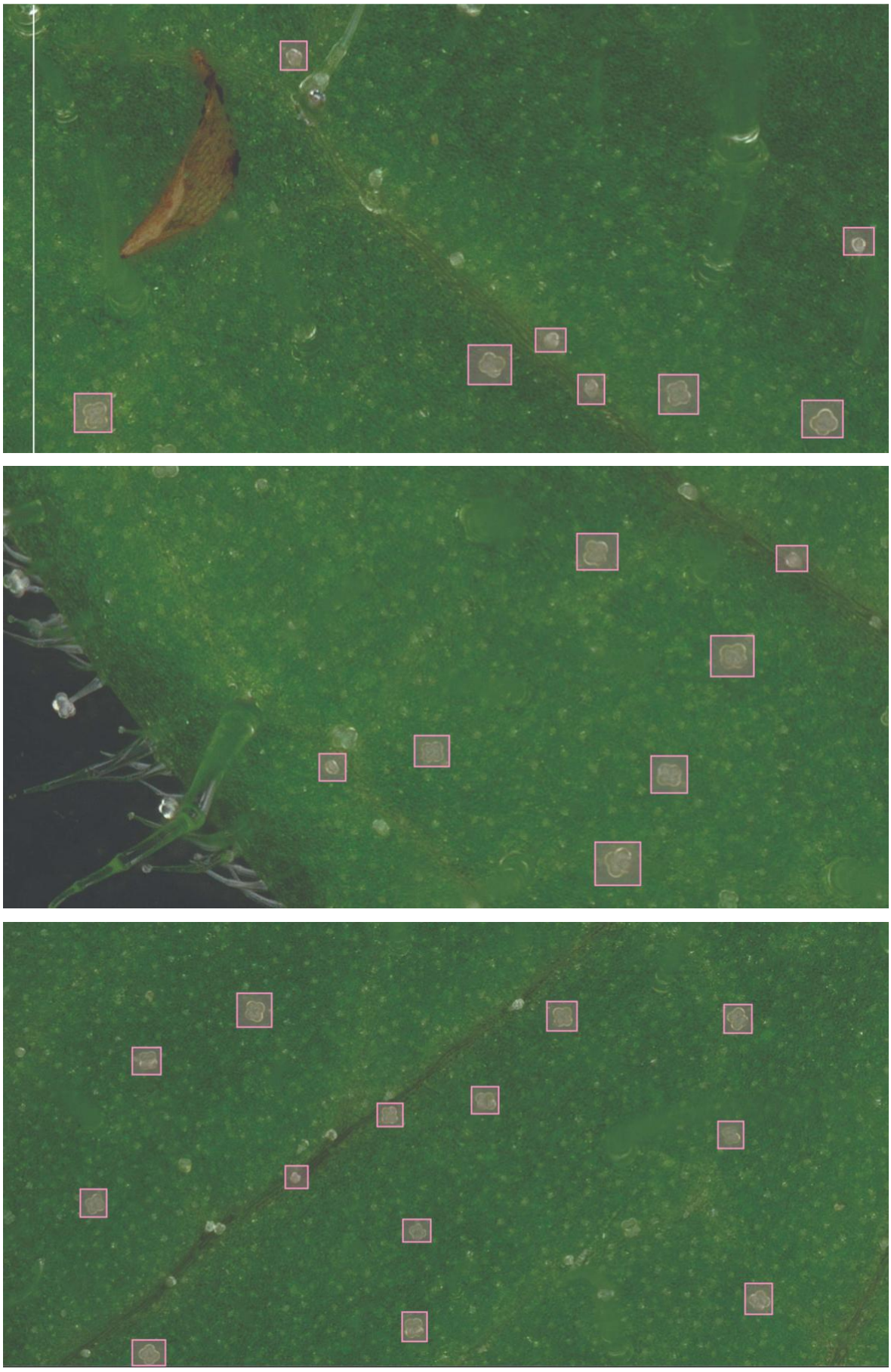




LAMPIRAN 2 (Proses pemberian label)







LAMPIRAN 3 (Hasil anotasi format .txt)

```
0 0.059097 0.304788 0.038920 0.060773
0 0.199022 0.218232 0.029653 0.079190
0 0.178173 0.362799 0.039846 0.073665
0 0.255085 0.419890 0.043553 0.073665
0 0.574318 0.318600 0.038920 0.095764
0 0.689223 0.913444 0.038920 0.073665
0 0.804592 0.466851 0.028726 0.068140
0 0.778182 0.750460 0.031506 0.060773
```

```
0 0.125635 0.447514 0.042647 0.066298
0 0.451976 0.462247 0.031522 0.077348
0 0.654085 0.615101 0.031522 0.066298
0 0.557666 0.930939 0.033376 0.060773
0 0.257747 0.769797 0.039866 0.066298
0 0.438069 0.022099 0.033376 0.044198
0 0.979035 0.698895 0.026886 0.057090
0 0.949368 0.538674 0.032449 0.060773
0 0.985114 0.132597 0.029772 0.077348
```

```
0 0.818955 0.194291 0.027800 0.049724
0 0.732776 0.240331 0.037066 0.064457
0 0.623894 0.443831 0.025020 0.058932
0 0.468216 0.425414 0.034286 0.055249
0 0.381110 0.376611 0.036140 0.057090
0 0.350530 0.294659 0.028726 0.044199
0 0.582658 0.958564 0.035213 0.057090
0 0.366284 0.868324 0.037993 0.053407
0 0.118867 0.918969 0.034286 0.062615
0 0.929227 0.251381 0.029653 0.053407
0 0.973243 0.177716 0.019460 0.046041
0 0.361650 0.169429 0.028726 0.051565
0 0.545591 0.669429 0.029653 0.049724
0 0.178173 0.291897 0.021313 0.049724
0 0.144813 0.337937 0.026873 0.046041
0 0.118403 0.111418 0.022240 0.060773
0 0.053537 0.186924 0.024093 0.049724
```

```
0 0.328671 0.115101 0.029667 0.064457
0 0.103848 0.910681 0.041720 0.086556
0 0.548395 0.803867 0.048209 0.086556
0 0.616537 0.748619 0.034303 0.053407
0 0.662429 0.858195 0.029667 0.066298
0 0.760238 0.869245 0.043574 0.088398
0 0.921555 0.923573 0.045428 0.082873
0 0.962347 0.529466 0.034303 0.060773
```

```
0 0.691540 0.893186 0.050966 0.095764
0 0.749456 0.695212 0.040773 0.082873
0 0.819881 0.430939 0.048186 0.092081
0 0.668373 0.198895 0.045406 0.081031
0 0.887064 0.212707 0.034286 0.057090
0 0.482579 0.642726 0.038920 0.069982
0 0.370917 0.678637 0.030580 0.060773
```

```
0 0.283899 0.203767 0.038994 0.074996
0 0.630380 0.217311 0.034286 0.066298
0 0.828221 0.223757 0.031506 0.064457
0 0.851851 0.849908 0.030580 0.068140
0 0.543275 0.406077 0.030580 0.060773
0 0.436709 0.438306 0.028726 0.051565
0 0.162419 0.317680 0.032433 0.060773
0 0.102650 0.635359 0.029653 0.062615
0 0.464509 0.912523 0.028726 0.064457
0 0.165199 0.971455 0.037993 0.057090
0 0.466826 0.697053 0.031506 0.049724
0 0.820345 0.483425 0.028726 0.060773
0 0.331534 0.577348 0.025946 0.049724
```

LAMPIRAN 4 (file konfigurasi)

```
1
2  path: '/content/drive/MyDrive/yolov8' # dataset root dir
3  train: images/train  # train images (relative to 'path')
4  val: images/val  # val images (relative to 'path')
5
6  # Classes
7  names:
8    0: Bulbose
```