IDENTIFIKASI JENIS PENYAKIT PADA CITRA DAUN SELADA MENGGUNAKAN SUPPORT VECTOR MACHINE DENGAN OPTIMASI PARTICLE SWARM OPTIMIZATION

SKRIPSI

Diajukan sebagai Salah Satu Syarat Untuk Mendapat Gelar Sarjana Komputer (S.Kom) Program Studi Informatika

MUH. ULIL AMRI

105841106221

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS MUHAMMADIYAH MAKASSAR

2025

IDENTIFIKASI JENIS PENYAKIT PADA CITRA DAUN SELADA MENGGUNAKAN SUPPORT VECTOR MACHINE DENGAN OPTIMASI PARTICLE SWARM OPTIMIZATION

Diajukan sebagai Salah Satu Syarat Untuk Mendapat Gelar Sarjana Komputer (S.Kom) Program Studi Informatika

Di Susun dan Diajukan Oleh :

MUH. ULIL AMRI

105841106221

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS MUHAMMADIYAH MAKASSAR

2025

PENGESAHAN

Skripsi atas nama MUH. ULIL AMRI dengan nomor induk Mahasiswa 105841106221, dinyatakan diterima dan disahkan oleh Panitia Ujian Tugas Akhir/Skripsi sesuai dengan Surat Keputusan Dekan Fakultas Teknik Universitas Muhammadiyah Makassar Nomor : 0004/SK-Y/55202/091004/2025, sebagai salah satu syarat guna memperoleh gelar Sarjana Komputer pada Program Studi Informatika Fakultas Teknik Universitas Muhammadiyah Makassar pada hari Sabtu, 30 Agustus 2025.

Panitia Ujian :

Makassar, 15 Rabi'ul Awal 1447 H / 8 September 2025 M

1. Pengawas Umum
   a. Rektor Universitas Muhammadiyah Makassar
      Dr. Ir. H. Abd. Rakhim Nanda, ST.,MT., IPU : .....
   b. Dekan Fakultas Teknik Universitas Hasanuddin
      Prof. Dr. Eng. Muhammad Isran Ramli, S.T., M.T., ASEAN. Eng : .....
2. Penguji
   a. Ketua : Dr. Ir. ZAHIR ZAINUDDIN, M.Sc .....
   b. Sekertaris : Rizki Yusliana Bakti, S.T., M.T .....
3. Anggota
   1. Ir. Muhammad Faisal, S.SI., M.T., Ph.D., IPM : .....
   2. DESI ANGGREANI, S.Kom., MT : .....
   3. LUKMAN, S.Kom., MT : .....

Mengetahui :

Pembimbing I	Pembimbing II
Chyquitha Danuputri, S.Kom., M.Kom	Rizki Yusliana Bakti, S.T., M.T

Dekan

Ir. Muh. Syafaat S. Kuba, S.T., M.T.
NBM : 795 288

HALAMAN PENGESAHAN

Tugas Akhir ini diajukan untuk memenuhi syarat ujian guna memperoleh gelar Sarjana Komputer (S.Kom) Program Studi Informatika Fakultas Teknik Universitas Muhammadiyah Makassar.

Judul Skripsi : **IDENTIFIKASI JENIS PENYAKIT PADA CITRA DAUN SELADA MENGGUNAKAN SUPPORT VECTOR MACHINE DENGAN OPTIMASI PARTICLE SWARM OPTIMIZATION**

Nama : MUH. ULIL AMRI
Stambuk : 105 841106221

Makassar, 15 Rabi'ul Awal 1447 H 2025

Telah Diperiksa dan Disetujui
Oleh Dosen Pembimbing;

Pembimbing I	Pembimbing II
Chyquitha Danuputri, S.Kom., M.Kom	Rizki Yusliana Bakti, S.T., M.T

Mengetahui,
Ketua Prodi Informatika

Rizki Yusliana Bakti, S.T., M.T.
NBM : 1307 28400

MOTTO DAN PERSEMBAHAN

MOTTO

"Hidup dijalani dengan kerja keras, mimpi diraih dengan keteguhan.
Menjalani dua peran bukan kelemahan, tetapi bukti kekuatan.
Lelah menjadi teman, tekad menjadi alasan untuk terus melangkah."

PERSEMBAHAN

Dengan penuh cinta dan hormat, skripsi ini penulis persembahkan teristimewa untuk kedua orang tua tercinta, Ayahanda Ismail Saleh dan Ibunda Nursyam, yang dengan doa, kasih sayang, pengorbanan, dan perjuangan tanpa henti telah membesarkan serta mendidik penulis hingga mampu berada pada titik ini. Setiap tetes keringat, setiap doa yang terucap dalam keheningan, dan setiap pengorbanan yang tak terhitung jumlahnya menjadi sumber kekuatan yang tidak pernah habis. Skripsi ini juga dipersembahkan kepada kakanda tercinta, Albar Arief, S.E., yang senantiasa hadir memberi dukungan, semangat, dan motivasi dalam berbagai situasi; serta kepada adik tersayang, Atifa Dwi Aulia, yang dengan kebersamaan, semangat, dan keceriaannya selalu menjadi pengingat bahwa keluarga adalah sumber kebahagiaan sejati. Selain itu, dengan penuh ketulusan, karya ini penulis persembahkan kepada Reski Anugrah Sari, yang senantiasa memberikan doa, dukungan, serta semangat tanpa henti; dan kepada sahabat-sahabat terbaik Syahril Akbar, Muhammad Adil Saputra, Sony Achmad Djalil, Gempar Perkasa Tahir, Risal, dan Nur Alam yang selalu setia menemani, memberi dukungan, dan menjadi bagian penting dalam perjalanan hidup serta perkuliahan. Akhirnya, karya sederhana ini penulis jadikan persembahan sebagai wujud terima kasih yang tak terhingga kepada semua pihak yang selalu memberikan doa, restu, dan semangat. Semoga persembahan ini dapat menjadi kebanggaan bagi keluarga serta menjadi langkah awal bagi penulis dalam menatap masa depan yang lebih baik.

ABSTRAK

MUH. ULIL AMRI, Identifikasi Jenis Penyakit Pada Citra Daun Selada Menggunakan Support Vector Machine Dengan Optimasi Particle Swarm Optimization (Di Bimbing Oleh Chyquitha Danuputri, S.Kom., M.Kom, dan Rizki Yusliana Bakti, S.T., M.T,)

Identifikasi penyakit tanaman secara manual memiliki keterbatasan subjektivitas dan akurasi, sehingga menghambat tindakan pengendalian yang efektif. Penelitian ini mengembangkan dan mengevaluasi sebuah sistem identifikasi penyakit otomatis pada citra daun selada. Sistem ini berbasis model klasifikasi *Support Vector Machine* (SVM) yang dilatih menggunakan fitur hibrida, yaitu kombinasi fitur warna dari ruang *Hue, Saturation, Value* (HSV) dan fitur tekstur dari *Gray-Level Co-occurrence Matrix* (GLCM). Untuk mengatasi keterbatasan data dan memaksimalkan kinerja, diterapkan teknik augmentasi data serta optimasi *hyperp arameter* SVM menggunakan algoritma *Particle Swarm Optimization* (PSO). Hasil pengujian komparatif menunjukkan peningkatan kinerja yang signifikan, di mana akurasi model meningkat dari 60,87% pada model dasar, menjadi 91,60% setelah augmentasi data, dan mencapai puncaknya pada 96,00% pada model final yang dioptimalkan dengan PSO. Model final juga menunjukkan F1-Score yang seimbang (di atas 0,91) untuk seluruh kelas (Sehat, Bercak Cercospora, Tip Burn, dan Etiolasi). Penelitian ini membuktikan bahwa integrasi rekayasa fitur, augmentasi data, dan optimasi PSO merupakan strategi yang efektif untuk membangun model SVM yang akurat dan robust untuk deteksi penyakit tanaman otomatis.

Kata Kunci: *Support Vector Machine*, Pengolahan Citra, Daun Selada, Optimasi PSO, Klasifikasi Penyakit, Augmentasi Data.

ABSTRACT

***MUH.ULIL AMRI**, Disease Type Identification in Lettuce Leaf Images Using Support Vector Machine with Particle Swarm Optimization (supervised by Chyquitha Danuputri, S.Kom., M.Kom, and Rizki Yusliana Bakti, S.T., M.T,)*

Manual identification of plant diseases is limited by subjectivity and inaccuracy, hindering effective control measures. This research develops and evaluates an automated disease identification system for lettuce leaf images. The system is based on a Support Vector Machine (SVM) classification model trained using hybrid features, combining color features from the Hue, Saturation, Value (HSV) space and texture features from the Gray-Level Co-occurrence Matrix (GLCM). To address data limitations and maximize performance, data augmentation techniques and SVM hyperparameter tuning using the Particle Swarm Optimization (PSO) algorithm were implemented. Comparative testing revealed significant performance improvements, with model accuracy increasing from a baseline of 60.87%, to 91.60% after data augmentation, and reaching a peak of 96.00% in the final PSO-optimized model. The final model also demonstrated balanced F1-Scores (above 0.91) across all classes (Healthy, Cercospora Leaf Spot, Tip Burn, and Etiolation). This study proves that the integration of feature engineering, data augmentation, and PSO optimization is an effective strategy for building an accurate and robust SVM model for automated plant disease detection.

***Keywords:** Support Vector Machine, Image Processing, Lettuce Leaf, PSO Optimization, Disease Classification, Data Augmentation.*

KATA PENGANTAR

بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ

Assalamualaikum wr.wb

Puji dan syukur penulis ucapkan terhadap kehadirat Tuhan Yang Maha Esa, yang telah melimpahkan rahmat hidayah dan karunia-Nya pada penulis, sehingga penulis dapat menyelesaikan penulisan skripsi ini yang berjudul "**IDENTIFIKASI JENIS PENYAKIT PADA CITRA DAUN SELADA MENGGUNAKAN SUPPORT VECTOR MACHINE DENGAN OPTIMASI PARTICLE SWARM OPTIMIZATION**". Shalawat beriring salam semoga senantiasa terlimpah kepada Nabi Muhammad SAW, teladan terbaik dan pembawa risalah pencerahan bagi semesta alam, yang telah membimbing umat manusia dari era kebodohan menuju era yang terang benderang oleh ilmu pengetahuan.

Penulis menghaturkan ucapan terima kasih yang mendalam kepada seluruh pihak yang telah memberikan kontribusi, dukungan, dan bimbingan selama proses penyusunan Laporan Tugas Akhir ini, terutama kepada

1. Kedua Orang Tua penulis yang terkasih, Bapak Ismail Saleh dan Ibu Nursyam, atas segala pengorbanan, doa, kasih sayang, dan dukungan tanpa henti yang menjadi sumber kekuatan terbesar.
2. Bapak Dr. Ir. H. Abd. Rakhim Nanda, S.T., M.T., IPU, selaku Rektor Universi tas Muhammadiyah Makassar.
3. Bapak Ir. Muhammad Syafa'at S Kuba, S.T., M.T, selaku Dekan Fakultas Teknik Universitas Muhammadiyah Makassar.
4. Ibu Rizki Yusliana Bakti, S.T., M.T, selaku Ketua Program Studi Informatika Fakultas Teknik Universitas Muhammadiyah Makassar.
5. Ibu Chyquitha Danuputri, S.Kom., M.Kom, selaku Dosen Pembimbing 1 yang telah memberikan arahan dan bimbingan serta saran yang sangat berarti dalam penulisan skripsi ini.
6. Ibu Titin Wahyuni, S.Pd., M.T. selaku Pembimbing Akademik yang senantiasa memberikan bimbingan, arahan, serta motivasi selama penulis menempuh perkuliahan hingga penyusunan tugas akhir ini. Kesabaran, ketelitian, dan

perhatian beliau menjadi dorongan yang sangat berarti bagi penulis dalam menyelesaikan karya ilmiah ini.

7. Bapak dan Ibu dosen pengajar di Program Studi Informatika Fakultas Teknik Universitas Muhammadiyah Makassar yang telah memberikan ilmu serta dorongannya dalam penulisan skripsi ini.
8. Apresiasi tertinggi penulis tujukan kepada Kak Iksan dan keluarga besar Deedad Hidroponik, yang dengan tulus telah membuka pintu dan memberikan kepercayaan penuh kepada penulis untuk menimba ilmu dan data di *Green House* hidroponik yang dikelola. Kelancaran penelitian ini tidak terlepas dari kemurahan hati dan dukungan yang telah diberikan.
9. Ucapan terima kasih tulus dari lubuk hati yang paling dalam penulis sampaikan kepada kakanda tercinta, Albar Arief, S.E., yang senantiasa menjadi pilar kekuatan dan sumber motivasi tiada henti. Dukungan, doa, dan semangat yang tak pernah padam dari beliau menjadi bekal paling berharga bagi penulis untuk sampai pada tahap penyelesaian skripsi ini.
10. Penghargaan tulus juga penulis sampaikan kepada sosok kelahiran 10 Oktober 2003, yang dengan sabar telah menemani setiap langkah, menjadi pendengar terbaik, dan tak pernah lelah memberi semangat di tengah pasang surut pengerjaan skripsi ini.
11. Sebuah apresiasi setinggi-tingginya juga untuk raga dan pikiran ini. Terima kasih karena telah berhasil menaklukkan ego, melawan rasa malas, dan tidak pernah menyerah pada keraguan. Perjalanan ini telah menjadi pelajaran paling berharga tentang arti kegigihan, disiplin, dan pendewasaan
12. Terimakasih Untuk para sahabat dan rekan seperjuangan penulis, Syahril Akbar, Muhammad Adil Syaputra, Risal, Gempar, Nur Alam, Makmur dan Sony, terima kasih telah menjadi teman berbagi dalam suka dan duka selama proses pengerjaan skripsi ini. Setiap diskusi, canda tawa, dan semangat yang kalian berikan merupakan hal yang tak ternilai harganya dan membuat perjalanan ini terasa lebih ringan.
13. Terima kasih yang tulus juga penulis sampaikan kepada seluruh teman-teman seperjuangan di Kelas B Angkatan 2021, Program Studi Informatika

Universitas Muhammadiyah Makassar. Meskipun tidak dapat disebutkan satu per satu, kebersamaan, semangat, dan setiap diskusi yang kita lalui bersama telah menjadi penguat dalam perjalanan akademik ini..

Penulis menyadari bahwa Laporan Tugas Akhir ini masih jauh dari kata sempurna. Untuk itu, kritik serta saran yang konstruktif sangat penulis nantikan guna penyempurnaan di masa mendatang. Harapan penulis, semoga Laporan Tugas Akhir ini dapat memberikan sumbangsih ilmu pengetahuan dan manfaat praktis, khususnya dalam bidang pertanian terkait identifikasi penyakit tanaman selada. Akhir kata, penulis memohon maaf atas segala keterbatasan dan kekhilafan yang terdapat dalam penulisan Laporan Tugas Akhir ini.

Billahi fisabililhaq, fastabiqul khairat.
Wassalamualaikum Wr.Wb.

Makassar, Agustus 2025

Penulis,

MUH. ULIL AMRI

DAFTAR ISI

MOTTO DAN PERSEMBAHAN i
ABSTRAK ii
ABSTRACT iii
KATA PENGANTAR iv
DAFTAR ISI vii
DAFTAR TABEL ix
DAFTAR GAMBAR x
DAFTAR LAMPIRAN xi
DAFTAR ISTILAH xii
BAB I PENDAHULUAN 1
A. Latar Belakang 1
B. Rumusan Masalah 4
C. Tujuan Penelitian 5
D. Manfaat Penelitian 5
E. Ruang Lingkup Penelitian 6
F. Sistematika Penulisan 7
BAB II TINJAUAN PUSTAKA 9
A. Landasan Teori 9
B. Penelitian Terkait 32
C. Kerangka Berfikir 37
BAB III METODE PENELITIAN 38
A. Tempat dan Waktu Penelitian 38
B. Alat dan Bahan 38
C. Perancangan Sistem 39
D. Teknik Pengujian Sistem 46
E. Teknik Analisis Data 48
BAB IV HASIL DAN PEMBAHASAN 49
A. Deskripsi Data Penelitian 49
B. Hasil Implementasi dan Pengujian 50
C. Pembahasan 52

D. Tampilan Antarmuka Sistem..... 61
BAB V PENUTUP..... 63
A. Kesimpulan 63
B. Saran..... 63
DAFTAR PUSTAKA 65

DAFTAR TABEL

Tabel 1. Penelitian Terkait 34
Tabel 2. Skenario Pengujian Penelitian 47
Tabel 3. Komposisi Dataset Induk 49
Tabel 4. Gambar Dataset 50
Tabel 5. Ringkasan Akurasi Keseluruhan Model per Skenario 51
Tabel 6. Rincian Metrik Evaluasi (TP, TN, FP, FN) dari Confusion Matrix 58
Tabel 7. Ringkasan Kesalahan Klasifikasi 59
Tabel 8. Kasus Misklasifikasi Representatif 60

DAFTAR GAMBAR

Gambar 1. Cercospora Beticola (Ningrum dkk., 2024) 11
Gambar 2. Tip Burn (İKiz dkk., 2024) 12
Gambar 3. Etiolasi (Fitrian dkk., 2023) 13
Gambar 4. Fusarium (SUDARMA. 2021) 14
Gambar 5. Bercak Daun Nigrospora (Nursanti dkk., 2021) 14
Gambar 6. Proses Transformasi Citra ke Ruang Warna HSV (Pramudiyah dkk., 2024) 18
Gambar 7. Ilustrasi proses GLCM (Nasution & Andayani, 2017) 19
Gambar 8. Kerangka Berfikir 37
Gambar 9. Flowchar Alur Interaksi Pengguna dengan Sistem Identifikasi Penyakit 40
Gambar 10. Flowchart Tahapan Umum Sistem Identifikasi Penyakit Menggunakan SVM 42
Gambar 11. Flowchart Detail Proses Pelatihan Model SVM 44
Gambar 12. Perbandingan Kinerja Antar Skenario 52
Gambar 13. Laporan Klasifikasi Model Terbaik 56
Gambar 14. Confusion Matrix Model Terbaik 57
Gambar 15. Tampilan Halaman Utama Aplikasi 61
Gambar 16. Tampilan Hasil Prediksi Penyakit 62

DAFTAR LAMPIRAN

Lampiran 1. Surat Permohonan Data Penelitian dari Program Studi 70
Lampiran 2. Surat Permohonan Izin Pelaksanaan Penelitian dari LP3M 71
Lampiran 3. Surat Izin Penelitian Dari Dinas Penanaman Modal dari PTSP Provinsi Sulawesi Selatan 72
Lampiran 4. Dokumentasi Penelitian di DEEDAD Hidroponik Jl. Dg Ngadde no 26 Parang Tambung, Kota Makassar 73
Lampiran 5. Dataset 75
Lampiran 6. Source Code..... 80
Lampiran 7. Surat Keterangan Bebas Plagiat 127
Lampiran 8. Hasil Turnitin Bab 1 - Bab 5 128

DAFTAR ISTILAH

Akurasi	Ukuran kinerja model klasifikasi yang merepresentasikan rasio antara jumlah prediksi yang benar dengan total jumlah data yang diuji.
Augmentasi Data	Teknik untuk meningkatkan jumlah dan variasi data latih secara artifisial dengan cara membuat modifikasi dari data yang sudah ada (misalnya rotasi, pergeseran, zoom, perubahan kontras).
Citra Digital	Representasi visual dari suatu objek yang direkam dalam format digital, terdiri dari kumpulan elemen gambar terkecil yang disebut piksel.
Confusion Matrix	Tabel visualisasi kinerja model klasifikasi yang menyajikan jumlah prediksi *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN).
Data Latih (*Training Data*)	Sebagian dari dataset yang digunakan untuk melatih atau "mengajari" model.
Data Uji (*Test Data*)	Sebagian dari dataset yang terpisah dan tidak pernah dilihat oleh model selama pelatihan, digunakan untuk memberikan evaluasi akhir yang objektif terhadap kinerja model.
Data Validasi (*Validation Data*)	Sebagian dari dataset yang digunakan selama proses pelatihan untuk menyetel *hyperparameter* dan mencegah *overfitting*.

Ekstraksi Fitur	Proses untuk mengidentifikasi dan mengambil informasi kuantitatif yang relevan (fitur) dari data mentah (citra) yang akan digunakan sebagai input bagi model.
F1-Score	Rata-rata harmonik dari *precision* dan *recall*, memberikan metrik tunggal yang menyeimbangkan kedua ukuran tersebut, sangat berguna pada dataset yang tidak seimbang.
Fungsi Kernel (*Kernel Trick*)	Teknik matematis yang digunakan oleh SVM untuk memetakan data ke ruang dimensi yang lebih tinggi agar data yang tidak dapat dipisahkan secara linear menjadi dapat dipisahkan.
Generalisasi	Kemampuan model *machine learning* untuk memberikan prediksi yang akurat pada data baru yang belum pernah dilihat sebelumnya, bukan hanya pada data yang digunakan untuk melatihnya.
***Gray Level Co-occurrence Matrix* (GLCM)**	Metode statistik untuk analisis tekstur citra yang menghitung frekuensi kemunculan pasangan piksel dengan tingkat keabuan tertentu. Dari matriks ini, fitur tekstur seperti Kontras, Homogenitas, Energi, dan Korelasi dapat diekstraksi.
***Hue, Saturation, Value* (HSV)**	Model ruang warna yang mendefinisikan warna berdasarkan tiga komponen: *Hue* (corak warna), *Saturation* (kejenuhan warna), dan *Value* (kecerahan warna).

Hyperparameter	Parameter eksternal pada sebuah model *machine learning* (seperti parameter C dan gamma pada SVM) yang nilainya ditetapkan sebelum proses pelatihan dimulai untuk mengontrol proses pembelajaran.
Hyperplane	Bidang pemisah (dapat berupa garis, bidang, atau dimensi yang lebih tinggi) yang ditemukan oleh algoritma SVM untuk membedakan antar kelas data secara optimal.
Klasifikasi	Proses dalam *machine learning* untuk mengelompokkan atau mengategorikan data ke dalam kelas-kelas yang telah ditentukan sebelumnya.
Konversi Ruang Warna	Proses mengubah representasi warna citra dari satu model ke model lain, misalnya dari RGB (*Red, Green, Blue*) ke HSV atau Grayscale.
Machine Learning	Cabang dari kecerdasan buatan yang memungkinkan sistem komputer untuk belajar dari data dan membuat prediksi atau keputusan tanpa diprogram secara eksplisit.
Margin	Jarak antara *hyperplane* dengan titik data terdekat dari masing-masing kelas. SVM bekerja dengan memaksimalkan margin ini.
Overfitting	Kondisi di mana model *machine learning* memiliki performa sangat baik pada data latih, tetapi performanya buruk pada data baru. Ini terjadi karena model terlalu "menghafal" detail dan *noise* dari data latih.

***Particle Swarm Optimization* (PSO)**	Algoritma optimasi metaheuristik yang terinspirasi dari perilaku sosial kawanan. Digunakan untuk mencari nilai *hyperparameter* terbaik secara efisien dengan menggerakkan "partikel" dalam ruang pencarian.
Piksel	Elemen terkecil dari sebuah citra digital yang memiliki nilai warna dan intensitas tertentu.
Pola	Keteraturan atau hubungan yang berulang dalam data yang dapat dipelajari dan diidentifikasi oleh model *machine learning*.
Pra-pemrosesan (*Preprocessing*)	Tahapan awal dalam pengolahan citra untuk meningkatkan kualitas citra atau mempersiapkannya untuk analisis lebih lanjut, seperti mengubah ukuran, menghilangkan derau (*noise*), atau segmentasi.
Precision	Metrik yang mengukur tingkat ketepatan dari total prediksi positif yang dibuat model (TP / (TP + FP)).
Prediksi	Output atau hasil yang diberikan oleh model *machine learning* setelah menganalisis data input.
***Recall* (Sensitivitas)**	Metrik yang mengukur kemampuan model untuk menemukan kembali semua data positif yang sebenarnya ada (TP / (TP + FN)).
Reduksi Derau (*Noise Reduction*)	Proses menghilangkan atau mengurangi gangguan (*noise*) acak pada citra yang dapat mengganggu analisis.

***Region of Interest* (ROI)**	Bagian spesifik dari sebuah citra yang menjadi fokus analisis, misalnya area daun yang sakit.
Segmentasi Citra	Proses mempartisi citra menjadi beberapa segmen untuk menyederhanakan representasi citra, sering kali untuk memisahkan objek yang diminati (daun) dari latar belakangnya.
Skala Keabuan (*Grayscale*)	Representasi citra di mana setiap piksel hanya memiliki nilai intensitas cahaya (tingkat keabuan), tanpa informasi warna.
Supervised Learning	Paradigma *machine learning* di mana model belajar dari data yang telah diberi label (setiap data input memiliki output yang benar).
***Support Vector Machine* (SVM)**	Algoritma *supervised learning* yang bekerja dengan menemukan *hyperplane* terbaik yang dapat memisahkan data ke dalam kelas-kelas yang berbeda dengan margin maksimal.
Underfitting	Kondisi di mana model terlalu sederhana untuk menangkap pola yang ada pada data latih, sehingga menghasilkan performa yang buruk baik pada data latih maupun data uji.
Validasi Silang (*Cross-Validation*)	Teknik evaluasi model di mana data latih dibagi menjadi beberapa bagian (lipatan/fold) dan model dilatih serta diuji beberapa kali untuk memastikan kinerjanya stabil dan tidak bergantung pada pembagian data tertentu.

BAB I

PENDAHULUAN

A. Latar Belakang

Sektor pertanian memegang peranan strategis dalam perekonomian nasional dan upaya pemenuhan ketahanan pangan di Indonesia. Di antara berbagai komoditas pertanian, hortikultura, khususnya sayuran daun seperti selada (*Lactuca sativa L.*), memiliki kontribusi penting sebagai sumber pangan segar bagi masyarakat. Seiring dengan meningkatnya kesadaran akan konsumsi sayuran, upaya peningkatan produktivitas selada menjadi semakin relevan. Salah satu pendekatan budidaya yang berkembang, terutama di wilayah dengan keterbatasan lahan, adalah sistem hidroponik. Metode ini memungkinkan budidaya tanaman, termasuk selada, tanpa menggunakan media tanah, dengan fokus pada pemberian nutrisi terlarut langsung ke akar tanaman, yang dapat mendukung efisiensi penggunaan lahan dan kontrol lingkungan tumbuh (Huda Dwi Putra & Yulianto, 2024)

Meskipun demikian, upaya peningkatan produktivitas selada, baik secara konvensional maupun hidroponik, seringkali dihadapkan pada tantangan signifikan berupa gangguan kesehatan tanaman. Gangguan ini tidak hanya disebabkan oleh serangan organisme pengganggu tanaman (OPT) yang bersifat patogenik, tetapi juga oleh gangguan fisiologis akibat faktor lingkungan dan nutrisi. Kegagalan dalam membedakan penyebab gangguan ini dapat berujung pada kerugian ekonomi bagi petani (Indah Kusuma dkk., 2024).

Beberapa jenis gangguan kesehatan yang menjadi perhatian dalam penelitian ini antara lain adalah Bercak Daun Cercospora, sebuah penyakit patogenik yang pada selada umumnya disebabkan oleh jamur *Cercospora longissima* dan dapat dicirikan dengan munculnya lesi-lesi spesifik pada daun. Selain itu, terdapat pula ancaman dari gangguan fisiologis seperti *Tip Burn*, suatu kondisi nekrosis pada ujung daun yang menurut İKiz dkk. (2024) disebabkan oleh defisiensi kalsium lokal. Relevansi masalah ini di Indonesia juga ditekankan oleh Sri Setyo dkk. (2023) yang menyoroti pentingnya deteksi dini untuk mencegah kerugian panen. Gangguan fisiologis lainnya adalah Etiolasi, yang manifestasi visualnya berupa pertumbuhan abnormal

dan daun yang pucat akibat kekurangan paparan cahaya, sebuah fenomena yang dijelaskan oleh Fitrian dkk. (2023) sebagai dampak dari penaungan antar tanaman.

Pengelolaan gangguan kesehatan tanaman yang efektif sangat bergantung pada kemampuan untuk melakukan identifikasi secara akurat. Namun, diagnosis secara visual yang dilakukan oleh petani seringkali menghadapi kendala subjektivitas dan inkonsistensi. Penilaian sangat bergantung pada tingkat pengalaman dan kejelian masing-masing individu, sehingga dapat menghasilkan diagnosis yang berbeda-beda (Nursanti dkk., 2021) Kesalahan dalam identifikasi dapat berakibat pada penerapan tindakan pengendalian yang tidak tepat sasaran dan kurang efisien. Oleh karena itu, diperlukan suatu pendekatan yang lebih objektif, sistematis, dan terotomatisasi untuk membantu proses identifikasi.

Perkembangan pesat dalam teknologi pengolahan citra digital *Computer Vision* dan pembelajaran mesin *Machine Learnin*g menawarkan alternatif solusi yang prospektif untuk mengatasi permasalahan ini. Sistem berbasis pengolahan citra mampu menganalisis karakteristik visual dari daun tanaman, seperti pola warna, tekstur, atau bentuk, untuk mendeteksi adanya kelainan akibat penyakit. Teknik seperti ekstraksi fitur menggunakan *Gray Level Co-occurrence Matrix* (GLCM) telah terbukti dapat digunakan untuk mengidentifikasi ciri penyakit, misalnya pada daun tomat (Feiters Tampinongkol dkk. 2023) telah menunjukkan kinerja yang baik dalam berbagai tugas pengenalan pola, metode ini umumnya memerlukan *dataset* citra dalam jumlah yang sangat besar untuk proses pelatihan yang efektif (Sulistiyana & Anardani 2023) Dalam kondisi di mana ketersediaan *dataset* terbatas, seperti yang sering dihadapi pada tahap awal penelitian atau untuk kasus penyakit yang kemunculannya sporadis, algoritma *Machine Learning* klasik dapat menjadi pilihan yang lebih sesuai.

Pendekatan *Machine Learning* klasik dalam konteks ini biasanya melibatkan dua tahap utama pertama, ekstraksi fitur (*feature extraction*) untuk mengkuantifik asi karakteristik visual penting dari citra dan kedua, klasifikasi (*classification)* menggunakan algoritma pembelajaran mesin berdasarkan fitur-fitur yang telah diekstraksi tersebut. Salah satu algoritma klasifikasi yang dikenal *robust* dan efektif, bahkan untuk *dataset* yang tidak terlalu besar asalkan didukung oleh fitur

yang representatif, adalah *Support Vector Machine* (SVM) (Sulistiyana & Anardani 2023) SVM bekerja dengan prinsip mencari *hyperplane* optimal yang dapat memisahkan data antar kelas dengan margin terbesar. Penting untuk digarisbawahi bahwa kinerja SVM sangat dipengaruhi oleh kualitas fitur yang dijadikan input. Oleh karena itu, tahap ekstraksi fitur visual yang relevan seperti parameter warna, metrik tekstur, atau deskriptor bentuk menjadi langkah yang sangat krusial dalam pengembangan model klasifikasi berbasis SVM (Wariyanto Abdullah dkk., 2025)

Untuk melakukan kuantifikasi terhadap karakteristik visual tersebut, teknik ekstraksi fitur memegang peranan fundamental. Perubahan kromatik pada daun, sebagai salah satu indikator patologis yang signifikan, dapat dianalisis secara komprehensif melalui representasi warna dalam ruang *Hue, Saturation, Value* (HSV). Pemilihan ruang warna HSV didasarkan pada kemampuannya untuk melakukan dekomposisi informasi warna (Hue dan Saturation) dari informasi iluminasi (Value), sehingga menghasilkan deskriptor warna yang lebih *robust* terhadap fluktuasi kondisi pencahayaan dibandingkan, sebagai contoh, ruang warna RGB (Fatham dkk. 2023) Lebih lanjut, manifestasi penyakit pada tanaman seringkali terefleksi pula pada alterasi tekstur permukaan daun. Fitur tekstural ini dapat dikuantifikasi melalui metode statistik, salah satunya adalah *Gray Level Co-occurrence Matrix* (GLCM). Metode GLCM menganalisis relasi spasial antar-piksel dalam citra untuk menghasilkan serangkaian deskriptor tekstur yang kaya informasi. Efektivitas GLCM telah terdemonstrasi dalam berbagai domain pengenalan pola, termasuk identifikasi karakteristik penyakit pada citra tanaman (Feiters Tampinongkol dkk. 2023) sekalipun begitu, implementasinya kerap mensyaratkan ketersediaan *dataset* yang representatif guna optimalisasi proses pelatihan (Sulistiyana & Anardani, 2023)

Perkembangan pesat dalam teknologi pengolahan citra digital *Computer Vision* dan pembelajaran mesin *Machine Learning* menawarkan alternatif solusi yang prospektif untuk permasalahan ini. Dalam konteks penelitian dengan ketersediaan data yang terbatas, pendekatan *machine learning* klasik seperti *Support Vector Machine* SVM seringkali menjadi pilihan yang lebih realistis dibandingkan metode *Deep Learning* yang membutuhkan data dalam jumlah masif. Keberhasilan

SVM sangat bergantung pada kualitas fitur yang diekstraksi. Oleh karena itu, penelitian ini akan berfokus pada ekstraksi fitur visual yang relevan, yaitu fitur warna dari ruang HSV dan fitur tekstur dari *Gray-Level Co-occurrence Matrix* (GLCM), sebuah pendekatan yang telah menunjukkan efektivitas dalam studi identifikasi penyakit tanaman (Zikra dkk. (2021). Untuk memastikan model yang dibangun memiliki kemampuan generalisasi yang baik dan tidak *overfitting*, akan diterapkan pula teknik augmentasi data. Teknik ini terbukti secara signifikan dapat meningkatkan akurasi model dengan memperkaya variasi citra secara sintetis, terutama pada skenario *dataset* terbatas (Husen dkk. 2022). Dengan demikian, penelitian ini bertujuan untuk merancang dan mengevaluasi sebuah model klasifikasi yang *robust* untuk mengidentifikasi penyakit utama pada daun selada.

Untuk membangun sebuah sistem identifikasi yang tidak hanya akurat tetapi juga andal, pemilihan *hyperparameter* pada Support Vector Machine (SVM) memegang peranan krusial. Pentingnya proses optimasi ini telah ditegaskan dalam berbagai penelitian di domain pertanian. Sebagai contoh, penelitian oleh Rusman dkk. (2023) berhasil mencapai akurasi tinggi dalam klasifikasi kematangan buah kopi dengan menerapkan optimasi *hyperparameter*.

Untuk mencapai kinerja yang lebih optimal dan efisien, penelitian ini mengusulkan penggunaan algoritma metaheuristik, yaitu *Particle Swarm Optimization* (PSO), untuk proses optimasi parameter SVM. Pendekatan SVM-PSO ini telah terbukti berhasil dalam meningkatkan akurasi klasifikasi pada kasus serupa, seperti yang ditunjukkan oleh Saputro dkk. (2025) dalam penelitian klasifikasi kualitas biji kopi.

Oleh karena itu, untuk membuktikan efektivitas pendekatan ini secara kuantitatif, penelitian ini akan berfokus pada analisis perbandingan kinerja secara komprehensif antara model SVM dengan parameter standar dan model SVM yang telah dioptimalkan menggunakan PSO. Perbandingan ini akan menjadi inti dari evaluasi untuk menentukan metode implementasi SVM yang paling efektif.

B. Rumusan Masalah

Adapun rumusan masalah dalam penelitian adalah sebagai berikut

1. Bagaimana hasil implementasi dan perbandingan kinerja dari tiga skenario model Support Vector Machine (SVM) untuk identifikasi penyakit daun selada, yaitu (a) Model SVM Murni yang dilatih dengan dataset asli (b). Model SVM dengan Augmentasi Data. (c). Model SVM dengan Augmentasi Data dan Optimasi *Particle Swarm Optimization* (PSO).
2. Seberapa signifikan dampak dari masing-masing teknik (augmentasi data dan optimasi PSO) terhadap peningkatan akurasi model klasifikasi?

C. Tujuan Penelitian

Berdasarkan rumusan masalah yang ada, maka tujuan dalam penelitian ini adalah

1. Mengimplementasikan dan mengevaluasi kinerja dari tiga skenario model Support Vector Machine (SVM). (a) Model Murni, (b) Model dengan Augmentasi Data, dan (c) Model dengan Augmentasi Data dan Optimasi PSO.
2. Menganalisis dan membandingkan secara kuantitatif dampak dari penerapan teknik augmentasi data dan optimasi PSO terhadap peningkatan akurasi model dalam mengidentifikasi penyakit daun selada.

D. Manfaat Penelitian

Adapun beberapa manfaat dari penelitian yang dilakukan penulis dan juga bagi pengguna yaitu

1. Manfaat bagi Petani Selada

a. Demonstrasi Potensi Alat Bantu Identifikasi Meskipun dilakukan pada skala data terbatas, penelitian ini bertujuan menunjukkan kelayakan (*proof-of-concept*) bahwa teknik machine learning (SVM) berbasis analisis citra daun dapat digunakan untuk membantu membedakan jenis-jenis penyakit selada. Ini membuka wawasan tentang potensi adanya alat bantu identifikasi di masa depan.
b. Informasi Awal Fitur Pembeda Hasil penelitian ini diharapkan dapat memberikan indikasi awal mengenai fitur visual seperti karakteristik warna atau tekstur yang mungkin paling signifikan dalam membedakan penyakit pada daun selada. Informasi ini, meskipun perlu validasi lebih lanjut, bisa menjadi pengetahuan tambahan bagi petani dalam mengamati tanamannya.

c. Mendorong Penerapan Teknologi Dengan menyajikan contoh konkret penerapan teknologi klasifikasi citra, penelitian ini diharapkan dapat meningkatkan kesadaran dan minat petani terhadap solusi berbasis teknologi untuk mengatasi masalah identifikasi penyakit yang selama ini sulit dilakukan secara manual.

2. Manfaat bagi Penulis

a. Pengembangan Kompetensi Teknis Memberikan kesempatan bagi penulis untuk mengaplikasikan dan memperdalam pengetahuan dalam bidang *Machine Learning* (SVM), Pengolahan Citra Digital ekstraksi fitur, dan pemrograman Python (Scikit-learn, OpenCV, NumPy).
b. Memberikan pengalaman praktis dalam melaksanakan seluruh tahapan penelitian ilmiah, mulai dari perumusan masalah hingga analisis dan pelaporan hasil.
c. Pemenuhan Syarat Akademik Penyelesaian skripsi ini merupakan salah satu syarat kelulusan untuk memperoleh gelar Sarjana Komputer (S.Kom) di Jurusan Teknik Informatika Universitas Muhammadiyah Makassar.
d. Fondasi untuk Pengembangan Lanjutan: Hasil dan pengalaman dari penelitian ini dapat menjadi dasar bagi penulis untuk penelitian lebih lanjut di bidang AI terapan.

E. Ruang Lingkup Penelitian

Dari analisis rumusan masalah di atas, dapat dirumuskan beberapa Batasan masalah sebagai berikut.

1. Tanaman Target dan Organ Penelitian ini secara spesifik difokuskan pada identifikasi penyakit pada tanaman Selada (*Lactuca sativa L.*), dengan analisis citra terbatas hanya pada organ daun. Gejala penyakit pada bagian batang atau akar tidak termasuk dalam cakupan identifikasi penelitian ini.
2. Kategori Identifikasi Kelas yang akan diidentifikasi oleh model terbatas pada kondisi daun sehat dan beberapa jenis penyakit utama yang memiliki gejala visual jelas pada daun yaitu. Bercak Cercospora, Tip Burn, Etiolasi.

3. Data yang digunakan dalam penelitian ini adalah *dataset* citra digital daun selada yang terdiri dari beberapa citra. Seluruh citra ini diperoleh melalui pengambilan gambar secara mandiri oleh peneliti di lokasi budidaya DEEDAD HIDROPONIK (Alamat: Jl. Dg. Ngadde No.26, Parang Tambung, Kec. Tamalate, Kota Makassar, Sulawesi Selatan 90223). *Dataset* ini mencakup citra untuk empat kelas yang telah ditetapkan, yaitu Sehat, Bercak Cercospora, Tip Burn, dan Etiolasi.
4. Input Model & Fitur Citra masukan daun akan distandarisasi ukurannya menjadi 224x224 piksel dan dinormalisasi sebelum proses ekstraksi fitur. Jenis fitur visual yang akan diekstraksi difokuskan pada kombinasi fitur warna berbasis statistik histogram ruang warna HSV (*Hue, Saturation, Value*) dan fitur tekstur berbasis statistik *Gray-Level Co-occurrence Matrix* (GLCM) seperti Kontras, Korelasi, Energi, dan Homogenitas.
5. Fungsi Sistem. Keluaran dari sistem ini murni berupa hasil identifikasi kondisi daun. Sistem tidak dirancang untuk memberikan rekomendasi tindakan perawatan, saran agronomis, atau metode pengobatan untuk penyakit yang terdeteksi.
6. Mode Operasi. Sistem ini dirancang untuk bekerja pada mode analisis gambar tunggal. Fungsionalitas untuk analisis video secara real-time atau deteksi berkelanjutan dari umpan kamera tidak termasuk dalam ruang lingkup proyek ini.
7. Skala Implementasi. Perangkat lunak yang dihasilkan dalam penelitian ini merupakan sebuah prototipe yang bertujuan untuk pembuktian konsep (proof-of-concept). Pengembangan lebih lanjut untuk menjadikannya produk yang siap digunakan secara komersial dengan skalabilitas tinggi berada di luar cakupan penelitian ini.

F. Sistematika Penulisan

Secara garis besar penulisan Skripsi ini terbagi menjadi beberapa bab yang tersusun sebagai berikut.

BAB 1 PENDAHULUAN

Pada bab ini menerangkan secara singkat dan jelas mengenal latar belakang penulisan penelitian tugas akhir, rumusan masalah, tujuan dan manfaat penelitian, batasan permasalahan metodologi yang di gunakan dan sistematika penulisan.

BAB II TINJAUAN PUSTAKA

Pada bab ini membahas teori-teori yang melandasi penulis dalam melaksanakan penelitian seperti metode dan algoritma yang terkait dengan penelitian ini. Pada bab ini juga berisi penjelasan dari penelitian sebelumnya yang relavan dengan penelitian ini

BAB III METODE PENELITIAN

Pada bab ini membahas tentang metode yang di gunakan dalam melakukan penelitian, waktu dan tempat penelitian, alat dan bahan penelitian, perancangan sistem penelitian dan teknik analisis data

BAB IV HASIL DAN PEMBAHASAN

Pada bab ini menguraikan secara rinci hasil dari implementasi dan pengujian sistem, pembahasan mendalam mengenai kinerja model yang dianalisis dari setiap skenario, serta menyajikan tampilan antarmuka dari web yang telah dikembangkan.

BAB V PENUTUP

Pada bab ini berisi kesimpulan yang ditarik dari keseluruhan hasil penelitian untuk menjawab rumusan masalah dan saran-saran yang dapat digunakan sebagai acuan untuk pengembangan penelitian selanjutnya.

BAB II

TINJAUAN PUSTAKA

A. Landasan Teori

1. Tanaman Selada (*Lactuca sativa L.*)

Tanaman selada (*Lactuca sativa L.*) merupakan salah satu komoditas hortikultura penting yang banyak dibudidayakan dan dikonsumsi secara luas. Kualitas visual dan kesehatan tanaman selada menjadi faktor krusial yang menentukan nilai ekonomisnya. Penelitian yang dilakukan oleh (SUDARMA, 2021) dalam Jurnal Agroekoteknologi Tropika menyoroti pentingnya selada dalam sistem budidaya modern seperti hidroponik. Sistem budidaya ini, yang juga menjadi konteks dalam penelitian sistem pakar oleh (Huda Dwi Putra & Yulianto, 2024) dalam Jurnal Teknologi Informasi (JTI) dan (Florensia dkk., 2025) dalam JITET (Jurnal Informatika dan Teknik Elektro Terapan), meskipun menawarkan efisiensi, tetap tidak terlepas dari ancaman penyakit. Penelitian (Zahra dkk., 2023) dalam Filogeni: Jurnal Mahasiswa Biologi juga secara spesifik membahas budidaya tanaman selada (*Lactuca sativa L.*) secara hidroponik dengan sistem DFT (*Deep Flow Technique*) dan penggunaan nutrisi AB Mix, yang menunjukkan relevansi budidaya modern ini. Varietas selada lokal seperti siomak, sebagaimana dilaporkan dalam penelitian (Pratama & Abadi, 2024) dalam Jurnal HPT, juga menunjukkan kerentanan terhadap penyakit tertentu. Buku Ajar Pengolahan Citra Digital (Dijaya, 2023) dengan fokusnya pada teknik pengolahan citra, secara implisit mendukung pentingnya analisis visual objek biologis seperti daun tanaman.

2. Penyakit Umum Tanaman Selada

Selada merupakan tanaman hortikultura dari *famili Asteraceae*, yang dikenal karena daunnya yang segar dan renyah. Sebagai tanaman semusim, siklus hidupnya relatif pendek, menjadikannya pilihan menarik bagi petani. Studi oleh Tropika dkk., (2021) dalam Jurnal Agroekoteknologi Tropika, yang berfokus pada identifikasi jamur penyebab penyakit utama pada tanaman selada hidroponik, menegaskan bahwa selada adalah komoditas penting dalam sistem budidaya inovatif ini. Secara morfologi, terdapat beragam jenis selada, mulai dari yang membentuk krop padat

seperti tipe *iceberg*, daun bertekstur lembut tipe *butterhead*, hingga daun memanjang dan tegak tipe *romaine*. Tanaman siomak atau selada wangi (*Lactuca sativa L*. var. augustuna), sebagaimana menjadi objek dalam penelitian (Pratama & Abadi, 2024) di Jurnal HPT, merupakan salah satu varietas yang memiliki karakteristik dan potensi pasar tersendiri. Keragaman ini juga mencakup warna daun, dari hijau muda, hijau tua, hingga kemerahan, yang dapat menjadi salah satu aspek visual dalam analisis citra. Selain aspek estetika dan rasa, selada juga dikenal sebagai sumber vitamin A, K, dan serat pangan yang baik untuk kesehatan.

a. Penyakit Utama pada Tanaman Selada yang Diteliti

Penelitian ini memfokuskan pada tiga jenis penyakit yang umum dan berpotensi merusak pada pertanaman selada.

1) Bercak Cercospora (*Cercospora Beticola*)

Penyakit bercak daun yang disebabkan oleh cendawan dari genus Cercospora merupakan salah satu penyakit penting pada selada. (Pratama & Abadi, 2024) dalam Jurnal HPT memberikan kontribusi signifikan dengan mengidentifikasi Cercospora sp. sebagai agen penyebab penyakit bercak daun pada tanaman siomak. Mereka melaporkan bahwa penyakit ini dapat menyebabkan kerusakan daun hingga mencapai 54,19%, yang tentu berdampak besar pada produksi dan pendapatan petani. Gejala visual yang paling menonjol dari penyakit ini adalah munculnya bercak-bercak pada daun. Awalnya, bercak mungkin berukuran kecil dan berwarna lebih gelap. Seiring dengan perkembangan infeksi, bercak tersebut membesar, seringkali menunjukkan bagian tengah yang nekrotik berwarna abu-abu atau coklat muda, sementara tepian bercak tetap berwarna lebih gelap (coklat tua atau kemerahan), menciptakan pola yang dikenal sebagai "mata kodok" (*frog-eye spot*). Pola ini menjadi salah satu ciri visual utama yang dapat dianalisis melalui pengolahan citra. Faktor lingkungan seperti kelembapan udara yang tinggi dan suhu hangat sangat kondusif bagi perkecambahan spora dan penyebaran penyakit. (SU DARMA, 2021) dalam penelitiannya juga menemukan Cercospora sp. sebagai penyebab penyakit bercak daun yang dominan dengan persentase penyakit tertinggi pada selada yang dibudidayakan secara hidroponik di beberapa lokasi, yang menunjukkan adaptabilitas patogen ini pada berbagai sistem budidaya. Identifikasi

makroskopis dan mikroskopis seperti yang dilakukan oleh Pratama & Abadi, yang melihat ciri koloni cendawan dan morfologi spora, adalah metode konvensional yang coba dibantu oleh pendekatan otomatis dalam penelitian ini.

Gambar 1. *Cercospora Beticola* (Ningrum dkk., 2024)

2) Tip Burn

Tip Burn adalah salah satu gangguan fisiologis yang paling merusak dalam budidaya selada hidroponik. Kondisi ini ditandai dengan munculnya nekrosis atau kematian jaringan yang dimulai dari ujung atau tepi daun-daun muda. Menurut penelitian oleh İKiz dkk. (2024) penyebab utama Tip Burn adalah defisiensi kalsium (Ca^{2+}) lokal pada jaringan tersebut, yang seringkali dipicu oleh faktor lingkungan seperti suhu tinggi atau laju pertumbuhan yang terlalu cepat yang menghambat transport kalsium. Secara visual, seperti yang menjadi fokus deteksi oleh Sri Setyo dkk. (2023) gejala Tip Burn adalah munculnya area berwarna coklat hingga hitam seperti terbakar pada tepi daun. Jaringan yang mati menjadi kering, rapuh, dan secara signifikan menurunkan kualitas serta nilai jual selada.

Tantangan utama dalam mengidentifikasi Tip Burn adalah lokasinya yang spesifik dan kemiripannya dengan gejala nekrosis lain. Gejala utamanya adalah munculnya area coklat hingga hitam pekat pada tepi daun. Perubahan warna yang drastis ini dapat dideteksi melalui analisis komponen warna, di mana penggunaan ruang warna HSV terbukti efektif dalam mengisolasi perubahan warna dari pengaruh pencahayaan Sri Setyo dkk. (2023) Secara tekstur, meskipun disebabkan

oleh gangguan fisiologis, efeknya adalah perubahan pada tepi daun yang menjadi kering. Namun, fitur pembeda utamanya bukanlah tekstur yang kompleks, melainkan perubahan warna yang tajam pada area yang terlokalisasi di tepi daun.

Gambar 2. Tip Burn (İKiz dkk., 2024)

3) Etiolasi

Etiolasi adalah sindrom respons fisiologis tanaman yang terjadi akibat kondisi pertumbuhan dengan intensitas cahaya yang tidak memadai, bukan disebabkan oleh patogen. Seperti yang didemonstrasikan dalam penelitian oleh Fitrian dkk. (2023) jarak tanam yang terlalu rapat pada budidaya selada dapat menyebabkan penaungan *shading* antar tanaman, sehingga memicu terjadinya etiolasi. Mekanisme etiolasi diatur secara hormonal, terutama oleh auksin, yang dalam kondisi gelap mendorong pemanjangan sel secara ekstrem. Gejala visualnya meliputi daun yang berwarna pucat atau kekuningan (*klorosis*) karena kegagalan sintesis klorofil, serta pertumbuhan batang yang lemah dan memanjang secara tidak normal (*spindly*).

Berbeda dengan penyakit lain yang menciptakan lesi, Etiolasi memengaruhi seluruh daun secara lebih merata. Karakteristik visual utamanya adalah perubahan warna secara global, di mana daun menjadi pucat atau kekuningan karena kegagalan pembentukan klorofil. Fenomena ini telah didokumentasikan terjadi pada selada yang mengalami kekurangan cahaya akibat jarak tanam yang terlalu rapat (Fitrian et al., 2023). Dalam analisis citra, perubahan ini akan tercermin sebagai pergeseran nilai Hue dari spektrum hijau menuju spektrum kuning, serta penurunan nilai Saturation yang signifikan. Oleh karena itu, fitur pembeda utama untuk Etiolasi diperkirakan bukan berasal dari parameter tekstur (GLCM),

melainkan dari perubahan dominan pada parameter statistik warna di seluruh area daun.

Gambar 3. Etiolasi (Fitrian dkk., 2023)

4) Layu Fusarium

Disebabkan oleh cendawan tanah Fusarium oxysporum f.sp. lactucae. Penelitian SUDARMA (2021) dalam Jurnal Agroekoteknologi Tropika mengidentifikasi penyakit layu yang disebabkan oleh Fusarium sp. sebagai salah satu dari dua penyakit utama yang ditemukan pada selada hidroponik di lokasi penelitian mereka, dengan persentase kejadian yang signifikan (mencapai 23,16% pada daun dan 28,55% pada akar di beberapa sampel). Gejala khas meliputi klorosis (menguning) yang seringkali dimulai pada satu sisi daun atau daun yang lebih tua, diikuti oleh kelayuan tanaman. Ciri diagnostik penting adalah diskolorasi coklat pada jaringan vaskular batang atau akar. Patogen ini bersifat tular tanah dan dapat menjadi masalah serius dalam sistem hidroponik jika sumber air atau media tanam jika digunakan untuk persemaian terkontaminasi.

Gambar 4. Fusarium (SUDARMA. 2021)

5) Bercak Daun Nigrospora

Penyakit bercak daun Nigrospora sp. adalah salah satu penyakit yang cukup sering muncul namun kerap terabaikan dalam budidaya sayuran daun seperti selada, termasuk dalam sistem hidroponik. Padahal, jamur ini bisa menyebar cepat, menyebabkan penurunan kualitas daun, dan jika tidak ditangani bisa berdampak pada nilai jual sayur di pasaran.

Dalam penelitian yang dilakukan oleh Nursanti dkk. (2021) di kawasan pertanian Desa Serang, Purbalingga, Jawa Tengah, Nigrospora sp. ditemukan sebagai salah satu jamur patogen penyebab bercak daun pada selada (*Lactuca sativa*) yang dibudidayakan di dataran tinggi. Penelitian ini menyebutkan bahwa meskipun dominasi infeksi oleh Rhizoctonia sp. sedikit lebih tinggi, Nigrospora sp. tetap tercatat sebagai penyebab signifikan bercak daun dalam sistem pertanian hortikultura lokal.

Gambar 5. Bercak Daun Nigrospora (Nursanti dkk., 2021)

3. Pengolahan Citra Digital untuk Identifikasi Penyakit Tanaman

Diagnosis penyakit tanaman yang cepat dan akurat adalah kunci untuk manajemen pertanian yang efektif. Metode diagnosis konvensional seringkali memiliki keterbatasan. Pengolahan citra digital menawarkan alternatif solusi yang menjanjikan melalui analisis objektif dan kuantitatif terhadap gejala visual penyakit. Sub-bab ini akan menguraikan secara sistematis bagaimana konsep dan teknik pengolahan citra digital, sebagaimana dielaborasi dalam Buku Ajar Pengolahan Citra Digital Dijaya (2023) dapat diaplikasikan secara spesifik untuk identifikasi penyakit pada daun tanaman selada. Tinjauan ini bertujuan untuk membangun landasan metodologis yang kuat untuk penelitian ini, dengan mempelajari tahapan-tahapan esensial dalam sistem identifikasi berbasis citra, dan menunjukkan bagaimana setiap tahapan didukung oleh teori pengolahan citra serta temuan penelitian terkini.

a. Prinsip Fundamental Pengolahan Citra sebagai Alat Diagnosis

Inti dari penggunaan pengolahan citra untuk diagnosis penyakit adalah kemampuannya untuk mengonversi informasi visual menjadi data numerik yang dapat dianalisis. Bab 1 (Pengantar Pengolahan Citra Digital) dan Bab 2 (Dasar Matematika untuk Pengolahan Citra) dalam buku ajar Dijaya (2023) memberikan fondasi teoretis mengenai bagaimana citra digital, sebagai representasi matriks piksel, dapat diproses untuk mengungkapkan pola atau anomali. Dalam konteks penyakit tanaman, gejala visual seperti perubahan warna, pembentukan lesi, atau alterasi tekstur daun merupakan manifestasi yang dapat ditangkap dan dikuantifikasi oleh algoritma komputer. Keberhasilan berbagai studi, seperti yang dilakukan oleh Feiters Tampinongkol dkk. (2023) pada tomat, Zikra dkk. (2021) pada cabai, dan Suhendra & Juliwardi (2022) pada jagung, dalam memanfaatkan citra daun untuk identifikasi penyakit memvalidasi potensi pendekatan ini untuk diterapkan juga pada tanaman selada.

b. Alur Kerja Sistematis dalam Pengembangan Sistem Deteksi Penyakit Berbasis Citra

Pengembangan sistem identifikasi penyakit berbasis citra mengikuti serangkaian tahapan logis, dari akuisisi data hingga evaluasi. Pemahaman alur kerja ini, yang juga relevan dengan praktik umum pengembangan aplikasi berbasis citra seperti yang ada dalam Bab 13-15 Aplikasi Lanjutan dan Kasus Studi bukuDijaya (2023) penting untuk memastikan pendekatan penelitian yang terstruktur.

1) Akuisisi dan Pembentukan *Dataset* Citra Daun Selada

Tahap "Pengambilan Citra" Dijaya (2023) Bab 3 & 4 merupakan langkah awal. Kualitas dan representativitas *dataset* citra sangat krusial. Penelitian ini akan memanfaatkan *dataset* citra daun selada yang mencakup kondisi sehat dan berbagai manifestasi penyakit yang diteliti. Pentingnya kualitas akuisisi, termasuk kontrol pencahayaan dan resolusi, adalah faktor yang sering ditekankan dalam berbagai penelitian misalnya, dalam Feiters Tampinongkol dkk. (2023) di mana variasi *dataset* publik menjadi pertimbangan.

2) Pra-pemrosesan Citra untuk Peningkatan Kualitas dan Standardisasi

Setelah akuisisi, citra daun selada perlu melalui tahap pra-pemrosesan untuk mengoptimalkan kualitasnya sebelum ekstraksi fitur. Sebagaimana diuraikan dalam Bab 4 (Pengambilan dan Praproses Citra), Bab 5 Pengolahan Intensitas dan Transformasi, dan Bab 6 (Filtrasi dalam Domain Spasial) buku Dijaya (2023) tahap ini dapat meliputi

a) Noise Reduction. Penggunaan filter untuk mengurangi derau yang mungkin mengganggu analisis.
b) Image Enhancement. Peningkatan kontras atau penajaman detail gejala visual.
c) Resizing. Standardisasi ukuran citra untuk konsistensi input.
d) Konversi Ruang Warna. Perubahan representasi warna dari RGB(standar akuisisi) ke ruang warna yang lebih sesuai untuk analisis fitur tertentu, seperti HSV yang akan dibahas pada bagian ekstraksi fitur. Penelitian Fatham dkk. (2023) yang mengeksplorasi berbagai color space untuk penyakit busuk selada menunjukkan signifikansi langkah ini. Langkah-langkah pra-pemrosesan ini bertujuan untuk memastikan bahwa informasi visual yang relevan dengan gejala penyakit menjadi lebih menonjol dan mudah dianalisis, serta menghindari kelemahan yang mungkin timbul dari variasi citra yang tidak terkontrol.

3) Segmentasi Citra untuk Isolasi Region of Interest

Analisis dan Segmentasi Citra" serta "Segmentasi Citra" (Dijaya, 2023) Bab 9 & 10 membahas teknik untuk memisahkan objek atau area yang menarik (misalnya, area lesi atau keseluruhan daun) dari latar belakang. Meskipun penelitian ini mungkin akan menerapkan ekstraksi fitur secara global pada citra daun yang sudah di-crop tanpa segmentasi lesi yang sangat detail, pemahaman konsep segmentasi tetap relevan sebagai salah satu teknik fundamental dalam pengolahan citra yang bisa jadi penting dalam skenario lain atau pengembangan sistem di masa depan. Siaulhak dkk. (2022) dalam sistem pendeteksi penyakit daun mereka, juga menyebutkan segmentasi sebagai salah satu langkah.

4) Ekstraksi Fitur Visual sebagai Kunci Pengenalan Pola Penyakit

Inti dari pendekatan machine learning klasik yang digunakan. Setelah citra dipra-proses, fitur-fitur kuantitatif yang merepresentasikan karakteristik warna dan tekstur daun diekstraksi. "Ekstraksi Fitur dan Pengenalan Pola" (Dijaya, 2023, Bab 11 & 12) menyediakan landasan teoretis untuk tahap ini. Pemilihan fitur yang tepat, yang akan dibahas mendalam, sangat krusial untuk keberhasilan klasifikasi. Pendekatan ini memungkinkan kita mempelajari dari penelitian-penelitian sebelumnya mengenai fitur mana yang efektif untuk penyakit tanaman misalnya GLCM dari studi (Feiters Tampinongkol dkk., 2023; Talib dkk., 2024).

4. Ekstraksi Fitur Visual

Ekstraksi fitur visual merupakan tahapan krusial dalam sistem identifikasi penyakit tanaman berbasis citra. Proses ini bertujuan untuk mentransformasi data piksel mentah yang kompleks menjadi sekumpulan deskriptor numerik yang lebih ringkas namun mampu merepresentasikan karakteristik esensial dari citra daun, sehingga dapat membedakan antara kondisi sehat dan sakit. Pemilihan fitur yang tepat dan metode ekstraksi yang efektif sangat menentukan keberhasilan sistem klasifikasi yang dibangun Dijaya (2023) Dalam konteks penelitian ini, penekanan diberikan pada fitur warna dan tekstur, mengingat manifestasi visual penyakit tanaman seringkali melibatkan perubahan pada kedua aspek tersebut.

a. Analisis Fitur Warna Menggunakan Ruang Warna HSV (*Hue, Saturation, Value*)

Perubahan warna daun merupakan salah satu indikator visual primer yang menandakan adanya infeksi penyakit atau gangguan fisiologis pada tanaman. Untuk menganalisis perubahan warna ini secara kuantitatif dan mengurangi sensitivitas terhadap variasi kondisi pencahayaan eksternal, penggunaan ruang warna HSV menjadi pendekatan yang relevan. Ruang warna HSV mendekomposisi warna ke dalam tiga komponen intuitif: Hue (rona warna sebenarnya), Saturation (kejenuhan atau kemurnian warna), dan Value (kecerahan warna).

Gambar 6. Proses Transformasi Citra ke Ruang Warna HSV (Pramudiyah dkk., 2024)

Studi yang dilakukan oleh Fatham dkk. (2023) mengenai analisis color space untuk deteksi penyakit busuk pada tanaman selada secara eksplisit menunjukkan bahwa pemilihan ruang warna memiliki dampak signifikan terhadap performa deteksi. Temuan ini menggarisbawahi pentingnya eksplorasi ruang warna seperti HSV yang mampu memisahkan informasi krominan (H dan S) dari informasi luminan (V), sehingga fitur warna yang diekstraksi menjadi lebih stabil dan diskriminatif. Berbeda dengan ruang warna RGB yang komponennya saling berkorelasi dan sensitif terhadap iluminasi, HSV menawarkan representasi yang lebih mendekati persepsi visual manusia terhadap warna. Dalam penelitian ini, informasi statistik seperti mean, standar deviasi, dan skewness dari histogram masing-masing kanal H, S, dan V akan diekstraksi. Fitur-fitur ini diharapkan mampu menangkap pergeseran warna spesifik yang diakibatkan oleh penyakit Bercak Cercospora, Busuk Daun, dan Embun Tepung pada daun selada,

sebagaimana perubahan warna juga menjadi fokus dalam penelitian identifikasi kematangan buah tomat oleh Alfaruq dkk. (2023) yang juga memanfaatkan fitur dari komponen HSV.

b. Analisis Fitur Tekstur Menggunakan *Gray Level Co-occurrence Matrix* (GLCM)

Selain perubahan warna, penyakit tanaman seringkali menimbulkan alterasi pada tekstur permukaan daun, seperti munculnya lesi, pola bercak, atau perubahan kehalusan permukaan. GLCM merupakan salah satu metode analisis tekstur orde kedua yang paling mapan dan banyak digunakan untuk mengkuantifikasi karakteristik tekstural citra. Metode ini bekerja dengan menganalisis hubungan spasial antar pasangan piksel pada jarak dan orientasi tertentu, yang direpresentasikan dalam sebuah matriks ko-okurensi matriks kejadian bersama.

Gambar 7. Ilustrasi proses GLCM (Nasution & Andayani, 2017)

Dari matriks GLCM, sejumlah deskriptor statistik dapat dihitung untuk merepresentasikan fitur tekstur. Beberapa penelitian yang relevan telah menunjukkan keberhasilan GLCM dalam domain pertanian dan identifikasi objek biologis. Sebagai contoh, (Talib dkk. (2024) berhasil menerapkan GLCM untuk klasifikasi jenis cengkeh berdasarkan fitur tekstur daun, dengan fokus pada parameter seperti Contrast, Correlation, Energy, dan Homogeneity. Serupa dengan itu, (Juniarti dkk. 2025) juga mengimplementasikan GLCM, dikombinasikan dengan SVM dan KNN, untuk mendeteksi penyakit daun pada berbagai tanaman hortikultura, yang menunjukkan bahwa parameter GLCM efektif untuk ekstraksi

fitur tekstur penyakit. Penelitian oleh Zikra dkk. (2021) yang mendeteksi penyakit cabai menggunakan GLCM dan SVM juga melaporkan tingkat akurasi yang tinggi, memperkuat potensi kombinasi ini. Lebih lanjut, (Feiters Tampinongkol dkk. 2023) menggunakan parameter Energy dan Entropy dari GLCM untuk identifikasi penyakit daun tomat dengan SVM. Bahkan dalam konteks yang lebih luas, seperti klasifikasi kematangan buah, (Alfaruq dkk. (2023) dan (Saputra dkk., 2022) juga memanfaatkan GLCM untuk ekstraksi fitur tekstur.

Dalam penelitian ini, fitur-fitur GLCM seperti Kontras (mengukur variasi lokal), Korelasi (mengukur dependensi linear antar piksel), Energi (mengukur keseragaman), dan Homogenitas (mengukur kedekatan distribusi elemen GLCM ke diagonal) akan diekstraksi dari citra daun selada yang telah dikonversi ke grayscale. Pemilihan fitur-fitur ini didasarkan pada kemampuannya yang telah terbukti dalam menangkap variasi tekstur yang disebabkan oleh gejala penyakit seperti bercak, perubahan permukaan akibat embun tepung, atau nekrosis akibat busuk daun. Keberhasilan GLCM dalam studi-studi tersebut, meskipun pada objek yang berbeda, memberikan landasan kuat untuk penerapannya dalam identifikasi penyakit daun selada.

5. SVM (Support Vector Mechine)

Support Vector Machine (SVM) adalah metode pembelajaran mesin (*machine learning*) yang bekerja atas prinsip *Structural Risk Minimization* (SRM) dan banyak digunakan untuk tugas klasifikasi maupun regresi. SVM bertujuan untuk menemukan *hyperplane* bidang pemisah terbaik yang dapat memisahkan data ke dalam kelas-kelas yang berbeda dengan margin atau jarak terbesar antara kelas-kelas tersebut (Andono & Rachmawanto 2021) Dalam teknik ini, SVM berusaha menemukan fungsi pemisah (*klasifier*) terbaik di antara fungsi yang tidak terbatas jumlahnya untuk memisahkan dua macam objek. *Hyperplane* terbaik merupakan *hyperplane* yang terletak di tengah-tengah antara dua objek dari dua kelas.

a. Prinsip Dasar Hyperplane dan Margin

Secara prinsip, SVM adalah *linear classifier*. Jika data memiliki dua kelas, SVM akan membuat *decision boundary* yang akan memisahkan data ke dalam dua kelas

yang berbeda, yang disebut *hyperplane*. Tujuan utama dari SVM adalah menemukan *hyperplane* dengan margin yang maksimal. Margin adalah jarak tegak lurus antara *hyperplane* dengan objek terdekat dari masing-masing kelas. Objek-objek terdekat yang menentukan posisi *hyperplane* ini disebut sebagai *support vectors*. Dengan memaksimalkan margin, SVM dapat mencapai kinerja klasifikasi yang tinggi dan mengurangi kemungkinan kesalahan dalam prediksi.

Diberikan satu set data pelatihan berlabel $(x1, y1),...,(xn, yn)$, di mana $xi \in Rd$ adalah vektor fitur dan $yi \in \{-1, +1\}$ adalah label kelasnya. *Hyperplane* optimal yang memisahkan data dapat didefinisikan dengan persamaan

$$w \cdot x + b = 0 \quad (1)$$

w adalah vektor bobot (normal terhadap hyperplane).

x adalah vektor fitur input.

b adalah bias (parameter skalar yang menentukan offset hyperplane dari titik asal).

Fungsi keputusan untuk klasifikasi data baru adalah

$$\mathrm{F}(x) = \mathrm{sign}(w.x + b) \quad (2)$$

Vektor ***w*** dan bias ***b*** harus memenuhi kondisi berikut untuk semua data pelatihan

$$yi(w \cdot xi + b) \geq 1, \ \forall i \quad (3)$$

Atau secarah terpisah:

$\mathrm{w} \cdot \mathrm{xi} + \mathrm{b} \geq +1$ jika $\mathrm{yi} = +1$

$\mathrm{w} \cdot \mathrm{xi} + \mathrm{b} \leq -1$ jika $\mathrm{yi} = -1$

Proses optimasi SVM bertujuan untuk memaksimalkan margin, yang ekuivalen dengan meminimalkan || w || atau lebih umum 21 || w || 2. Jadi, masalah optimasi SVM dapat ditulis sebagai

Minimize

$21 || w || 2 (4)

b. Kernel Trick untuk Data Non-Linear

SVM pada dasarnya bekerja dengan memetakan data ke *feature space* berdimensi tinggi sehingga titik data dapat dikelompokkan, dan kemudian *hyperplane* pemisah ditemukan. Untuk kasus di mana data tidak dapat dipisahkan secara linear di ruang aslinya, SVM menggunakan teknik yang dikenal sebagai *kernel trick*. Fungsi kernel memungkinkan SVM untuk melakukan pemetaan data ke ruang fitur berdimensi lebih tinggi tanpa secara eksplisit menghitung koordinat data di ruang baru tersebut. Dengan demikian, *hyperplane* pemisah yang linear di ruang berdimensi tinggi tersebut dapat menjadi non-linear di ruang fitur aslinya. Beberapa fungsi kernel yang umum digunakan meliputi (Andono & Rachmawanto, 2021; Saputra dkk., 2022)

Linear = $K(x_i, x_j) = x_i^T x_j$

Polinominal = $K(x_i, x_j) = (\gamma x_i^T x_j + r)^d$ dimana $\gamma > 0$.

Radial Basis Function (RBF) atau Gaussian= $K(x_i, x_j) = \exp(-\gamma |x_i - x_j|^2)$

Dimana $\gamma > 0$. Kernel RBF sering dipilih karena kemampuannya menangani hubungan non-linear.

c. Klasifikasi MultiClass

Meskipun SVM pada dasarnya dirancang untuk klasifikasi biner, SVM dapat diperluas untuk menangani masalah klasifikasi multi-kelas. Dua pendekatan utama yang umum digunakan adalah *One-Against-All* (OAA) / *One-vs-Rest* (OvR) Pada metode ini, untuk masalah klasifikasi k-kelas, dibangun k fungsi pemisah (model SVM). Setiap fungsi pemisah ke-*i* dilatih dengan semua data dari kelas-i sebagai kelas positif (+1) dan semua data dari kelas lain sebagai kelas negatif (-1). Data baru diklasifikasikan ke kelas yang memiliki nilai fungsi keputusan tertinggi. Formulasi optimasi primal untuk fungsi pemisah ke-i adalah:

$$\min_{w^i,b^i,\xi^i} \frac{1}{2}(w^i)^T w^i + C\sum_{j=1}^{l} \xi_j^i \quad (5)$$

Dengan Syarat :

$$w^i\phi(x_j) + b^i \geq 1 - \xi_j^i, \quad \text{jika } y_j = i$$

$$w^i\phi(x_j) + b^i \leq -1 + \xi_j^i, \quad \text{jika } y_j \neq i$$

$$\xi_j^i \geq 0, \quad j = 1,2,\dots,l$$

Dimana $\phi(x_j)$ adalah fungsi pemetaan ke ruang fitur yang lebih tinggi (implisit melalui kernel), ξ_j^i adalah *slack variables* untuk menangani data yang tidak dapat dipisahkan secara sempurna, dan C adalah parameter penalti.

SVM telah terbukti memiliki kemampuan generalisasi yang tinggi, bahkan dengan dimensi ruang input yang tinggi, tanpa memerlukan pengetahuan tambahan mengenai data. Efektivitasnya juga terlihat dalam klasifikasi citra penyakit daun, seperti pada penelitian (Suhendra & Juliwardi, 2022) yang mencapai akurasi 99,5% untuk penyakit daun jagung, dan penelitian (Juniarti dkk. 2025) yang mengimple mentasikan SVM untuk deteksi penyakit pada tanaman holtikultura. Keberhasilan SVM sangat bergantung pada pemilihan fitur yang relevan dan proses tuning parameter kernel serta parameter regularisasi C yang tepat.

d. Pengolahan Citra, Ruang Warna HSV, GLCM, SVM

Menurut Alfaruq dkk., (2023) dalam penelitiannya yang berjudul "Klasifikasi Kematangan Buah Tomat Dengan Metode Support Vector Machine" melakukan klasifikasi tingkat kematangan buah tomat menjadi tiga kelas, yaitu matang, setengah matang, dan muda. Penelitian ini memanfaatkan kombinasi fitur tekstur menggunakan *Gray-Level Co-occurrence* Matrix (GLCM) dan fitur warna menggunakan metode Momen Warna. Ekstraksi fitur warna dilakukan setelah citra dikonversi dari RGB ke ruang warna HSV. Fitur GLCM yang diekstraksi meliputi kontras, homogenitas, energi, dan korelasi. Sedangkan fitur Momen Warna yang diekstraksi dari setiap komponen HSV adalah mean, standard deviation, dan skewness.

Alur kerja penelitian ini dimulai dari pengambilan citra, pra-pemrosesan (meliputi cropping dan konversi ruang warna), ekstraksi fitur GLCM dan Momen Warna, hingga klasifikasi menggunakan SVM. Dalam tahap pra-pemrosesan, citra

RGB dikonversi menjadi grayscale untuk ekstraksi GLCM dan menjadi HSV untuk ekstraksi Momen Warna. Data yang digunakan terdiri dari 300 citra untuk pelatihan dan 100 citra untuk pengujian.

Rumus-rumus yang digunakan untuk ekstraksi fitur GLCM dalam penelitian ini adalah sebagai berikut

1) Kontras $\sum \mathrm{P_{ij}}(\mathrm{i}-\mathrm{j})^2$ (6)

2) Homoginitas IDM $= \sum_{\mathrm{i,j=0}}^{\mathrm{N-1}} \frac{\mathrm{P_{ij}}}{1+(\mathrm{i}-\mathrm{j})^2}$ Dirujuk Pada IDM $= \sum_{\mathrm{i,j}} \frac{\mathrm{P_{ij}}}{1+(\mathrm{i}-\mathrm{j})^2}$ (7)

3) Korelasi $\sum_{\mathrm{i,j=0}}^{\mathrm{N-1}} \mathrm{P_{i,j}} \left[\frac{(\mathrm{i}-\mu_\mathrm{i})(\mathrm{j}-\mu_\mathrm{j})}{\sqrt{(\sigma_\mathrm{i}^2)(\sigma_\mathrm{j}^2)}}\right]$ (8)

Untuk fitur Momen Warna, rumus yang digunakan adalah:

1) Mean $E_i = \frac{1}{N}\sum_{j=1}^{N} p_{ij}$ (9)

2) Standard Deviation $\sigma_i = \sqrt{\frac{1}{N}\sum_{j=1}^{N}(p_{ij} - E_i)^2}$ (10)

3) Skewness $\mathrm{s_i} = \sqrt[3]{\frac{1}{\mathrm{N}}\sum_{\mathrm{j=1}}^{\mathrm{N}}(\mathrm{p_{ij}} - \mathrm{E_i})^3}$ (11)

Menurut Yana & Nafi'iyah, (2021) dalam penelitian berjudul "Klasifikasi Jenis Pisang Berdasarkan Fitur Warna, Tekstur, Bentuk Citra Menggunakan SVM dan KNN" melakukan klasifikasi terhadap 7 jenis pisang. Penelitian ini menggunakan tiga jenis fitur, yaitu fitur warna nilai rata-rata RGB, standar deviasi RGB, skewness RGB, dan entropy RGB, fitur tekstur (nilai rata-rata citra grayscale, standar deviasi grayscale, dan GLCM yang meliputi kontras, energi, korelasi, homogeneity), serta fitur bentuk (*area*, *perimeter*, *metric*, *major axis*, *minor axis*, *eccentricity*). Algoritma klasifikasi yang digunakan adalah SVM dan KNN.

Alur kerja penelitian ini terdiri dari tahap pengambilan fitur citra dan tahap klasifikasi. Pada tahap pengambilan fitur, citra RGB digunakan untuk ekstraksi fitur warna; citra grayscale untuk fitur tekstur; dan citra biner untuk fitur bentuk.

Proses klasifikasi kemudian dilakukan menggunakan algoritma SVM terhadap fitur-fitur yang telah diekstraksi.

Rumus-rumus yang digunakan untuk ekstraksi fitur statistik dan GLCM dalam penelitian ini adalah

1) Rata-rata ($\bar{x}$): $\bar{x} = \frac{\sum f(i)}{N}$ (12)

2) Standar Daviasi (S): $S = \sqrt{\frac{\sum_{i=1}^{N} |f(i) - \bar{x}|^2}{N-1}}$ (13)

3) Skewness = $\frac{E(x-\bar{x})^3}{\sigma^3}$ (14)

4) Entropi $= -\sum p(i) \cdot \log_2 p(i)$ (15)

5) Kontras (GLCM)= *Kontras* $\sum_{i,j} |i - j|^2 p(i,j)$ (16)

6) Energi (GLCM)= *Energi* $\sum_{i,j} p(i,j)^2$ (17)

7) Korelasi (GLCM)= *Correlation* $\frac{\sum_{i,j}(i-\bar{x}_i)(j-\bar{x}_j)p(i,j)}{\sigma_i \sigma_j}$ (18)

8) Homogeneity (GLCM)= *Homogeneity* $\sum_{i,j} \frac{p(i,j)}{1+|i-j|}$ (19)

6. Evaluasi Kinerja dan Teknik Pengujian Model Klasifikasi

Setelah sebuah model klasifikasi seperti *Support Vector Machine* (SVM) berhasil dibangun, langkah selanjutnya yang tidak kalah penting adalah melakukan evaluasi untuk mengukur kinerjanya secara objektif. Proses evaluasi ini bertujuan untuk memvalidasi kemampuan model dalam melakukan generalisasi, yaitu kemampuannya untuk memberikan prediksi yang akurat pada data baru yang belum pernah dilihat sebelumnya selama proses pelatihan. Evaluasi yang *robust* sangat penting untuk memastikan bahwa model yang dikembangkan dapat diandalkan untuk aplikasi di dunia nyata, seperti identifikasi penyakit tanaman. Pentingnya evaluasi yang tepat juga ditekankan dalam berbagai penelitian identifikasi penyakit, di mana metrik yang jelas menjadi tolak ukur keberhasilan sistem (Zikra dkk. 2021). Bagian ini akan menguraikan konsep-konsep fundamental dalam teknik pengujian dan metrik evaluasi kinerja model klasifikasi, yang menjadi standar dalam domain machine learning terapan.

a. Confusion Matrix

Confusion Matrix adalah alat fundamental yang digunakan untuk memvisualis asikan dan meringkas kinerja sebuah algoritma klasifikasi. Matriks ini menyajikan perbandingan antara kelas aktual dari data dengan kelas yang diprediksi oleh model. Untuk masalah klasifikasi biner, matriks ini berukuran 2x2. Namun, dalam penelitian ini yang melibatkan klasifikasi multi-kelas (Sehat, Bercak Cercospora, Tip Burn, dan Etiolasi), matriks akan berukuran N x N, di mana N adalah jumlah kelas.

Setiap sel dalam matriks ini merepresentasikan jumlah data. Sumbu vertikal biasanya mewakili kelas aktual (*Ground Truth*), sedangkan sumbu horizontal mewakili kelas yang diprediksi oleh model (*Predicted* Label). Dari matriks ini, kita dapat mengekstrak empat istilah kunci untuk setiap kelas, yang menjadi dasar perhitungan metrik kinerja selanjutnya (Saputra dkk., 2022)

1) *True Positive* (TP) Jumlah data yang berhasil diklasifikasikan dengan benar sebagai kelas positif.
2) *True Negative* (TN) Jumlah data yang berhasil diklasifikasikan dengan benar sebagai kelas negatif.
3) *False Positive* (FP) / Kesalahan Tipe I Jumlah data yang salah diklasifikasikan sebagai kelas positif (alarm palsu).
4) *False Negative (FN)* / Kesalahan Tipe II Jumlah data yang salah diklasifikasikan sebagai kelas negatif (kasus yang terlewatkan).

b. Metrik Evaluasi Kinerja (Performance Metrics)

Berdasarkan nilai-nilai TP, TN, FP, dan FN dari confusion matrix, beberapa metrik kuantitatif dapat dihitung untuk mengevaluasi berbagai aspek kinerja model. Pemilihan metrik yang tepat sangat bergantung pada tujuan aplikasi dan karakteristik *dataset*.

1) Akurasi (*Accuracy*)

Metrik ini mengukur proporsi total prediksi yang benar dari keseluruhan data.

$$Akurasi = (TP + TN)/(TP + TN + FP + FN)$$

Akurasi sering digunakan sebagai metrik utama dalam banyak penelitian klasifikasi, seperti pada identifikasi penyakit daun jagung yang mencapai akurasi tinggi Suhendra & Juliwardi (2022). Namun, metrik ini bisa menjadi sangat menyesatkan jika *dataset* tidak seimbang (imbalanced *dataset*), di mana jumlah data pada satu kelas jauh lebih dominan daripada kelas lainnya (Andono & Rachmawanto, 2021)

2) Presisi (*Precision*)

Presisi mengukur akurasi dari prediksi positif. Metrik ini menjawab pertanyaan "Dari semua instance yang diprediksi sebagai kelas X, berapa persen yang sebenarnya benar-benar kelas X?"

$Presisi = TP/(TP + FP)$

Dalam konteks diagnosis penyakit, presisi tinggi penting untuk memastikan bahwa ketika sistem mengidentifikasi suatu penyakit, diagnosis tersebut kemungkinan besar benar, sehingga petani tidak melakukan tindakan pengendalian yang tidak perlu dan boros sumber daya (Wariyanto Abdullah dkk., 2025)

3) Recall (*Sensitivity* atau *True Positive Rate*)

Recall mengukur seberapa baik model dapat mengidentifikasi semua instance positif yang relevan.

$Recall = TP/(TP + FN)$

Recall tinggi menunjukkan bahwa model memiliki tingkat kasus yang "terlewatkan" yang rendah. Ini sangat krusial dalam diagnosis penyakit, karena kegagalan mendeteksi penyakit (*False Negative*) dapat menyebabkan penyebaran penyakit yang tidak terkendali dan kerugian panen yang signifikan, sebuah aspek yang menjadi perhatian dalam sistem deteksi dini (Fatham dkk., 2023)

4) F1-Score

F1-Score adalah rata-rata harmonik (*harmonic mean*) dari Presisi dan Recall. Metrik ini memberikan skor tunggal yang menyeimbangkan antara kesalahan False Positive dan False Negative.

$F1 - Score = 2 * (Presisi * Recall)/(Presisi + Recall)$

F1-Score sangat berguna untuk evaluasi pada *dataset* yang tidak seimbang, karena mempertimbangkan kedua metrik (presisi dan recall) secara bersamaan, memberikan gambaran kinerja yang lebih holistik (Mulyana & Wibowo, 2023)

7. Validasi Model dan Evaluasi Kinerja

Validasi model merupakan tahapan krusial dalam pengembangan sistem berbasis pembelajaran mesin yang bertujuan untuk memberikan estimasi yang andal mengenai kinerja model pada data baru yang independen. Proses ini esensial untuk memastikan bahwa model yang dibangun memiliki kemampuan generalisasi yang baik, yaitu mampu bekerja secara akurat pada data yang belum pernah dilihat sebelumnya, bukan sekadar "menghafal" data pelatihan. Kegagalan dalam melakukan validasi yang tepat dapat menyebabkan masalah *overfitting*, di mana model menunjukkan performa sangat tinggi pada data latih namun sangat buruk pada data uji, sehingga tidak dapat diandalkan dalam aplikasi dunia nyata (Nti dkk. 2021)

a. Proses Validasi dengan *K-Fold Cross-Validation*

Untuk mengatasi risiko *overfitting* dan mendapatkan estimasi kinerja yang stabil, terutama pada kondisi *dataset* yang terbatas, metode *K-Fold Cross-Validation* menjadi pendekatan yang sangat relevan. Prosedur ini bekerja dengan cara mempartisi *dataset* menjadi *K* bagian atau "lipatan" (*folds*) yang berukuran kurang lebih sama. Proses pelatihan dan pengujian kemudian diulang sebanyak *K* kali. Pada setiap iterasi, satu lipatan unik akan digunakan sebagai data validasi, sementara *K-1* lipatan lainnya digabungkan untuk menjadi data pelatihan. Dengan cara ini, setiap sampel data memiliki kesempatan untuk menjadi bagian dari set validasi tepat satu kali. Hasil kinerja dari *K* iterasi tersebut kemudian dirata-ratakan untuk menghasilkan satu skor evaluasi akhir yang lebih representatif dan kurang rentan terhadap bias pembagian data (Nti dkk. 2021)

b. Metrik Evaluasi Kinerja

Untuk mengukur performa model secara kuantitatif pada setiap iterasi validasi, digunakan beberapa metrik standar yang diturunkan dari *confusion matrix*. *Confus ion matrix* adalah sebuah tabel yang memvisualisasikan kinerja model dengan

membandingkan kelas aktual dengan kelas yang diprediksi oleh model. Metrik-metrik yang umum digunakan antara lain (Husen dkk. 2022)

1) Akurasi (*Accuracy*) Mengukur proporsi total dari prediksi yang benar (baik positif maupun negatif) terhadap keseluruhan data. Meskipun intuitif, metrik ini bisa memberikan gambaran yang menyesatkan jika *dataset* tidak seimbang (imbalanced).
2) Presisi (*Precision*) Mengukur tingkat ketepatan dari prediksi positif. Metrik ini menjawab pertanyaan: "Dari semua data yang diprediksi sebagai kelas X, berapa persen yang sebenarnya benar-benar kelas X?". Presisi tinggi penting untuk meminimalkan kesalahan *False Positive*.
3) Akurasi Recall (*Sensitivity*) Mengukur kemampuan model untuk menemukan semua data positif yang relevan. Metrik ini menjawab pertanyaan Dari semua data yang seharusnya masuk ke dalam kelas X, berapa persen yang berhasil di prediksi dengan benar oleh model?". *Recall* tinggi penting untuk meminimalkan kesalahan *False Negative*.
4) F1-Score. Merupakan rata-rata harmonik dari presisi dan recall. Metrik ini memberikan skor tunggal yang menyeimbangkan kedua metrik tersebut, sehingga sangat berguna untuk mengevaluasi model pada *dataset* yang tidak seimbang

8. Augmentasi Data

Menurut Fadli Gunardi (2022) dalam makalahnya, augmentasi data atau augmentasi citra adalah sebuah metode yang digunakan untuk mengatasi permasalahan kurangnya jumlah citra pada suatu dataset. Kinerja sebuah model klasifikasi sangat bergantung pada jumlah data training yang digunakan. Jumlah data yang sedikit memiliki kecenderungan menyebabkan overfitting, yaitu kondisi di mana model menghafal fitur detail dari data training sehingga tidak dapat digeneralisasi dengan baik pada data baru. Sebaliknya, penambahan jumlah data training dapat meningkatkan akurasi dari model tersebut. Metode augmentasi data menjadi solusi yang paling tepat untuk masalah ini karena dapat menciptakan citra artifisial melalui berbagai cara pemrosesan, seperti rotasi acak atau peregangan kontras, tanpa mengubah label dari citra tersebut.

a. Dampak Augmentasi Data terhadap Kinerja Model

Menurut Fadli Gunardi (2022) juga menjelaskan bahwa penerapan augmentasi data terbukti dapat meningkatkan nilai akurasi sebuah model secara signifikan. Peningkatan ini terjadi karena model dapat mengenali lebih banyak objek dari beragam kondisi yang ada pada citra. Sebagai contoh, jika sebelum augmentasi sebuah model tidak mampu mengklasifikasikan citra yang mengalami blur atau memiliki kontras rendah, maka setelah proses augmentasi model tersebut akan mampu mengklasifikasikannya

b. Jenis-jenis Teknik Augmentasi Data

Teknik augmentasi data menjadi dua pendekatan utama, yaitu melalui proses *filtering* dan transformasi geometri.

1) Transformasi Geometri

Transformasi geometri, sebagaimana diuraikan oleh Fadli Gunardi (2022) merupakan operasi yang diterapkan untuk menggeser, merotasi, atau mengubah skala sebuah citra. Teknik-teknik yang termasuk dalam kategori ini antara lain.

a) Rotasi (*Rotation*) Operasi untuk menggeser citra pada suatu sudut tertentu.
b) Translasi (*Translation*) Operasi untuk menggeser citra pada sumbu x atau y.
c) Penskalaan (*Scaling*) Operasi untuk memperbesar atau memperkecil tampilan suatu citra.

2) Proses Filtering (Perbaikan Kualitas Citra)

Filtering atau *image enhancement*, yang bertujuan untuk memanipulasi kualitas citra untuk mengatasi gangguan seperti blur, derau (*noise*), atau kontras yang rendah. Teknik-teknik yang relevan untuk augmentasi dalam kategori ini meliputi.

a) Peregangan Kontras (*Contrast Stretching*): Sebuah metode untuk memperbaiki citra yang memiliki kontras rendah dengan cara meningkatkan rentang nilai-nilai keabuannya.
b) Pencerahan Citra (*Image Brightening*): Proses untuk menerangkan piksel-piksel yang gelap dengan menambahkan sebuah konstanta pada setiap piksel dalam citra.
c) Penajaman Citra (*Image Sharpening*): Proses untuk meningkatkan intensitas pada bagian tepi dari suatu objek dalam citra menggunakan high-pass filter.

d) Pelembutan Citra (*Image Smoothing*): Proses untuk menghasilkan efek blur atau mengurangi derau pada citra dengan menekan komponen berfrekuensi tinggi

9. Optimasi Parameter dengan *Particle Swarm Optimization* (PSO)

Kinerja dari model Support Vector Machine (SVM), khususnya yang menggunakan kernel RBF, sangat dipengaruhi oleh pemilihan *hyperparameter*-nya, yaitu parameter regularisasi C dan parameter kernel gamma. Menemukan kombinasi optimal dari kedua parameter ini secara manual adalah proses yang sulit dan tidak efisien. Untuk mengatasi tantangan ini, digunakanlah algoritma optimasi.

Particle Swarm Optimization (PSO) adalah salah satu algoritma optimasi *metaheuristik* yang terinspirasi dari perilaku sosial kawanan burung atau ikan dalam mencari makanan. PSO bekerja dengan sebuah populasi disebut swarm yang terdiri dari sejumlah solusi kandidat disebut partikel. Setiap partikel terbang di dalam ruang pencarian untuk menemukan solusi terbaik (Rusman dkk., 2023)

Proses kerja PSO secara umum adalah sebagai berikut

a. Inisialisasi Sejumlah partikel disebar secara acak di dalam ruang pencarian. Setiap partikel memiliki posisi yang merepresentasikan satu set solusi, misalnya [C, gamma] dan kecepatan awal.

b. Evaluasi Fitness Posisi setiap partikel dievaluasi menggunakan fungsi fitness. Dalam konteks optimasi SVM, nilai fitness adalah akurasi yang dihasilkan oleh model SVM saat menggunakan parameter dari posisi partikel tersebut.

c. Pembaruan Posisi Dalam setiap iterasi, setiap partikel memperbarui kecepatannya berdasarkan dua informasi utama

1) Personal Best (pbest) Posisi terbaik yang pernah dicapai oleh partikel itu sendiri.
2) Global Best (gbest) Posisi terbaik yang pernah dicapai oleh seluruh partikel dalam swarm.

d. Terminasi Proses iterasi ini berlanjut hingga kriteria berhenti terpenuhi (misalnya, jumlah iterasi maksimum tercapai atau tidak ada lagi peningkatan signifikan pada gbest). Partikel dengan posisi gbest terakhir akan menjadi solusi optimal.

Pendekatan optimasi menggunakan PSO untuk menemukan parameter SVM telah terbukti efektif dalam berbagai penelitian klasifikasi. Sebagai contoh,

penelitian oleh Rusman dkk. (2023) berhasil meningkatkan akurasi klasifikasi kualitas biji kopi dengan menerapkan SVM-PSO. Demikian pula, meskipun menggunakan metode optimasi yang berbeda, penelitian oleh Rusman dkk. (2023) juga menegaskan pentingnya proses optimasi *hyperparameter* untuk mencapai kinerja SVM yang maksimal

B. Penelitian Terkait

Hasil penelitian terkait mencakup ringkasan dari studi-studi sebelumnya yang telah dipublikasikan, baik kelebihan maupun kelemahannya. Data ini dikumpulkan dari berbagai sumber seperti jurnal, laporan penelitian, atau karya ilmiah lainnya, dan bertujuan untuk menjadi bahan pertimbangan bagi peneliti dalam menyusun kerangka penelitian.

1. (Juniarti dkk., 2025) (dalam Technology, Health, and Agriculture Nexus: Conference Series) berfokus pada implementasi *Gray Level Co-Occurrence Matrix* (GLCM) untuk mendeteksi penyakit daun pada berbagai tanaman hortikultura, yaitu jagung, kentang, dan tomat. Mereka menggunakan SVM dan *K-Nearest Neighbor* (KNN) sebagai metode klasifikasi. Hasil penelitian menunjukkan bahwa kinerja kedua algoritma bervariasi antar *dataset* tanaman, dengan KNN terkadang memberikan akurasi lebih tinggi untuk jagung (79%), sementara SVM lebih baik untuk kentang (43% vs 52% untuk KNN), dan keduanya mencapai 30% untuk tomat. Studi ini menyoroti pentingnya pemilihan parameter K pada KNN dan kualitas *dataset*, serta menunjukkan bahwa GLCM efektif untuk ekstraksi fitur tekstur pada penyakit tanaman.
2. (Zikra dkk., 2021) dalam Seminar Nasional Hasil Penelitian dan Pengabdian Masyarakat merancang sistem untuk mendeteksi penyakit pada tanaman cabai (meliputi virus kuning, keriting, dan bercak daun). Penelitian ini memanfaatkan citra daun yang kemudian diekstraksi fiturnya menggunakan metode GLCM dengan parameter kontras, korelasi, energi, dan homogeniti. Klasifikasi dilakukan menggunakan *Support Vector Machine* (SVM) dengan kernel polinomial dan strategi multiclass one-against-one (OAO), yang berhasil mencapai tingkat akurasi sebesar 95%. Keberhasilan ini menunjukkan potensi

besar kombinasi GLCM dan SVM untuk identifikasi penyakit pada tanaman hortikultura.

3. (Feiters Tampinongkol dkk., 2023) dalam Techno Xplore juga relevan dari sisi metodologi, di mana mereka mengidentifikasi penyakit pada daun tomat. Fitur tekstur diekstraksi menggunakan GLCM (khususnya parameter Energy dan Entropy) dan kemudian diklasifikasikan menggunakan SVM. Studi ini, meskipun tidak menyebutkan akurasi spesifik dalam abstrak yang tersedia, memperkuat penggunaan GLCM dan SVM dalam analisis citra penyakit daun.
4. (Fatham dkk., 2023)dalam Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer melakukan analisis *color space* (termasuk RGB, HSV, LAB, dan YCbCr) untuk sistem deteksi penyakit busuk pada tanaman selada. Fokus mereka pada eksplorasi ruang warna untuk mendapatkan fitur diskriminatif sangat relevan dengan penelitian ini, meskipun metode klasifikasi akhir yang mereka gunakan mungkin berbeda atau tidak secara spesifik disebutkan sebagai SVM dalam informasi yang tersedia. Penelitian ini menekankan pentingnya tahap analisis warna dalam identifikasi penyakit selada.
5. (Mulyana & Wibowo, 2023) dalam JINTEKS (Jurnal Informatika Teknologi dan Sains) mengimplementasikan GLCM dan SVM untuk menentukan tingkat kematangan buah Monk, dan berhasil mencapai akurasi 89% dengan parameter C=50 pada SVM. Ini sekali lagi menunjukkan keberhasilan GLCM-SVM pada klasifikasi objek biologis berdasarkan tekstur visualnya.
6. (Sri Setyo dkk., 2023) dalam Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer mengimplementasikan metode segmentasi thresholding pada ruang warna HSV untuk mendeteksi gangguan fisiologis Tip Burn pada selada hidroponik. Sistem yang diuji dengan 18 data citra ini berhasil mencapai akurasi sebesar 94%. Penelitian ini secara langsung mendukung pemilihan fitur warna HSV sebagai prediktor kunci dan menegaskan relevansi deteksi Tip Burn sebagai masalah signifikan pada budidaya selada.
7. (Fitrian dkk., 2023) dalam Berkala Ilmiah PERTANIAN melakukan penelitian eksperimental untuk menginvestigasi pengaruh perbedaan jarak tanam terhadap pertumbuhan selada. Hasil studi mereka menunjukkan secara empiris bahwa

jarak tanam yang terlalu rapat menyebabkan penaungan yang memicu terjadinya Etiolasi, yang ditandai dengan pertumbuhan abnormal dan daun berwarna pucat

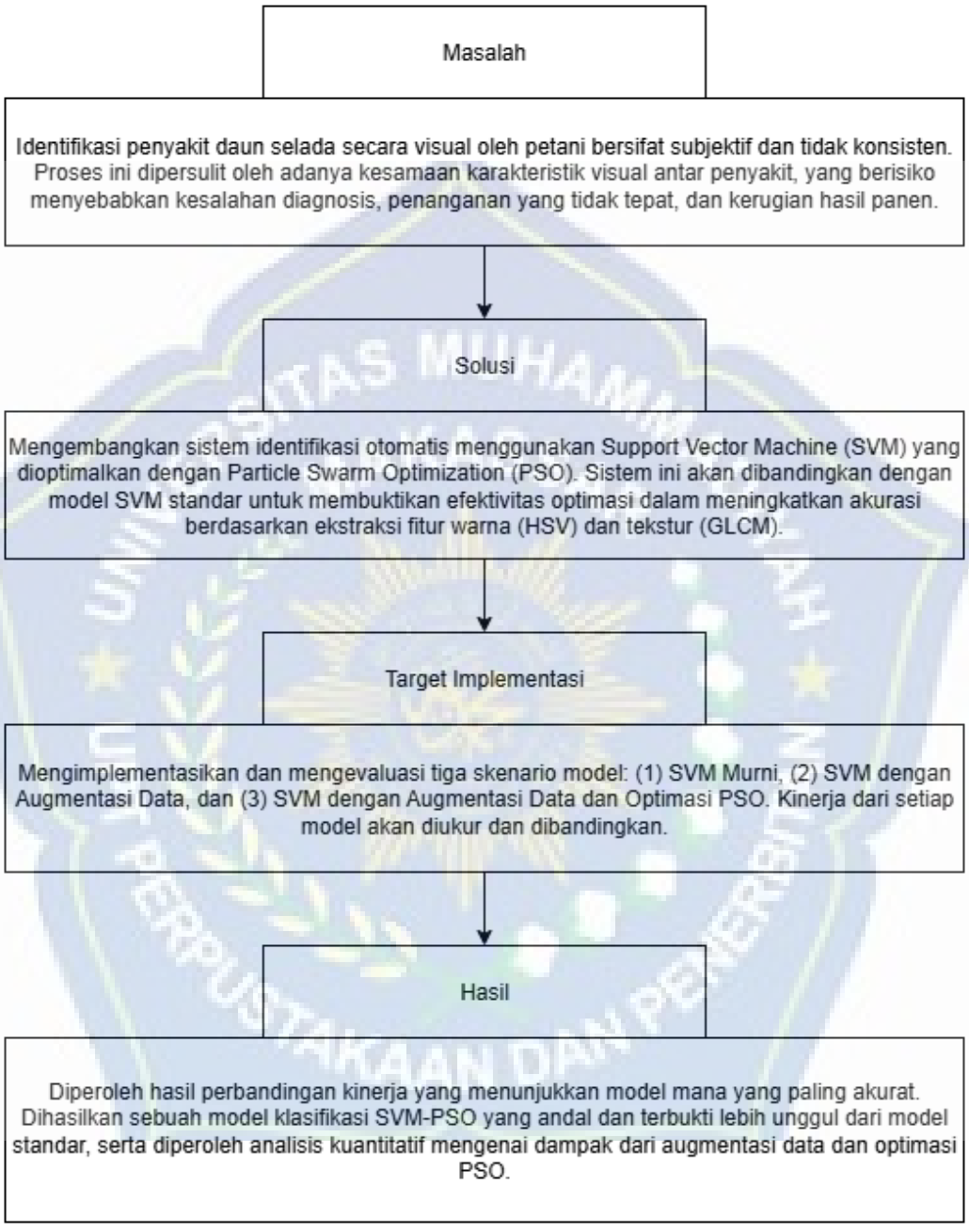
Tabel 1. Penelitian Terkait

No	Penelitian	Kontribusi
1.	Juniarti dkk. (2025) Implementasi Gray Level Co-Occurrence Matrix (GLCM) Untuk Mendeteksi Penyakit Daun Pada Tanaman Holtikultura	Menunjukkan efektivitas GLCM untuk ekstraksi fitur tekstur penyakit pada berbagai tanaman hortikultura (jagung, kentang, tomat) dan membandingkan kinerja SVM serta KNN, memberikan dasar metodologis untuk ekstraksi fitur tekstur dan pemilihan klasifikator.
2.	Zikra dkk. (2021) Deteksi Penyakit Cabai Berdasarkan Citra Daun Menggunakan Metode Gray Level Co- Occurrence Matrix Dan Support Vector Machine	Mengaplikasikan GLCM (kontras, korelasi, energi, homogeniti) dan SVM (kernel polinomial, OAO) untuk deteksi penyakit cabai dengan akurasi tinggi (95%), mendukung pemilihan kombinasi GLCM-SVM untuk penyakit tanaman hortikultura.
3.	Suhendra & Juliwardi (2022) Identifikasi dan Klasifikasi Penyakit Daun Jagung Menggunakan Support Vector Machine	Mencapai akurasi sangat tinggi (99,5%) menggunakan SVM dengan fitur gabungan warna dan tekstur untuk penyakit daun jagung, menunjukkan potensi besar kombinasi fitur visual dan SVM.
4.	Sri Setyo dkk. (2023) Sistem Deteksi Tip Burn pada Selada Hidroponik menggunakan Metode Thresholding pada Warna Hue Saturation Value.	Secara spesifik mengembangkan sistem deteksi untuk pada selada. Mengonfirmasi efektivitas ruang warna HSV untuk analisis perubahan warna daun, yang

No	Penelitian	Kontribusi
		memperkuat pemilihan fitur warna dal am penelitian ini.
5.	Andono & Rachmawanto (2021)Evaluasi Ekstraksi Fitur GLC M dan LBP Menggunakan Mul tikernel SVM untuk Klasifikasi Batik	Mengevaluasi berbagai kernel SVM (polinomial, linear, gaussian) dan parameter GLCM (jarak) untuk klasifikasi pola visual (batik), memberikan wawasan mengenai optimasi parameter SVM dan GLCM yang relevan untuk analisis tekstur.
6.	SUDARMA (2021) Identifikasi Jamur Penyebab Penyakit Utam a pada Tanaman Selada (Lactuca sativa L.) Hidroponik	Mengidentifikasi Cercospora sp. dan F usarium sp. sebagai penyebab penyakit utama pada selada hidroponik di Bali, memberikan landasan empiris mengen ai jenis penyakit jamur yang relevan dan umum menyerang tanaman selada, yang menjadi target identifikasi dalam penelitian ini.
7.	Pratama & Abadi (2024) Penya kit Bercak Daun Pada Tanaman Siomak (Lactuca sativa L. var. a ugustuna) Serta Pengendalianny a Di Komunitas Pertanian Orga nik Brenjonk	Secaraspesifik mengidentifikasi Cercos pora sp. sebagai penyebab bercak daun pada varietas selada siomak dan melaporkan dampak kerusakannya, me mperkuat urgensi penelitian terhadap penyakit Bercak Cercospora.

No	Penelitian	Kontribusi
8.	Fitrian dkk (2023) Respon Pertumbuhan dan Hasil Tanaman Selada Romaine Terhadap Perbedaan Jarak Tanam.	Memberikan landasan empiris untuk mendefinisikan kelas Etiolasi. Menunjukkan bahwa kerapatan tanam yang tinggi menyebabkan gejala pertumbuhan abnormal dan klorosis, yang menjadi ciri visual pembeda untuk klasifikasi.
9.	İKiz dkk. (2024), Mitigating Tip Burn True Foliar Calcium Application in Indoor Hydroponicly Grown Mini Cos Lettuce.	Memberikan validasi ilmiah internasional mengenai Tip Burn sebagai gangguan fisiologis akibat defisiensi kalsium. Memperkuat justifikasi dan urgensi pemilihan Tip Burn sebagai salah satu kelas target dalam penelitian.
10.	ULIL, (2025) Identifikasi Jenis Penyakit pada Citra Daun Selada Menggunakan Support Vector Mechine Dengan Optimasi Particle Swarm Optimization	Mengembangkan dan mengevaluasi secara sistematis sebuah kerangka kerja untuk identifikasi penyakit selada. Kontribusi utama adalah analisis kuantitatif terhadap dampak augmentasi data dan optimasi PSO, yang terbukti meningkatkan akurasi model dari 60,87% (dasar) menjadi 96,00% (final). Penelitian ini menyajikan metodologi terpadu yang efektif untuk mencapai kinerja tinggi pada skenario dataset terbatas.

C. Kerangka Berfikir



Gambar 8. Kerangka Berfikir

BAB III

METODE PENELITIAN

A. Tempat dan Waktu Penelitian

Penelitian ini dijadwalkan berlangsung mulai bulan Mei 2025 hingga Juli 2025. Proses pengumpulan data primer berupa citra digital daun selada dilaksanakan secara langsung di lokasi budidaya DEEDAD HIDROPONIK, Jl. Dg. Ngadde No.26, Parang Tambung, Kec. Tamalate, Kota Makassar, Sulawesi Selatan 90223. Sementara itu, seluruh kegiatan komputasi meliputi pra-pemrosesan data, implementasi algoritma ekstraksi fitur, pengembangan dan pelatihan model SVM, pengujian, serta analisis hasil.

B. Alat dan Bahan

Adapun Alat dan Bahan yang digunakan dalam penelitian ini adalah.

1. Perangkat Keras (Hardware)

a. Ponsel Cerdas (Smartphone) Redmi Note 11 Pro (RAM 8GB, Penyimpanan 128GB)
b. Laptop ASUS TUF Gaming F15 (Intel® Core™ i5-10300H, RAM 16GB, GPU NVIDIA® GeForce® GTX 1650 4GB GDDR6, SSD 512GB)

2. Perangkat Lunak (Software)

a. Sistem Operasi
   1) Windows 11
   2) Visual Studio Code
b. Bahasa Pemrograman: Python, JavaScript (next.js)
c. Library
   1) Scikit-learn,
   2) OpenCV & Scikit-image,
   3) NumPy,
   4) Pyswarms,
   5) Matplotlib & Seaborn,
   6) Flask & Werkzeug,
   7) Joblib

C. Perancangan Sistem

Perancangan sistem dalam penelitian ini bertujuan untuk membangun sebuah alur kerja komputasional yang mampu mengidentifikasi jenis penyakit pada daun selada berdasarkan analisis citra digital. Sistem ini dirancang dengan pendekatan *machine learning* klasik, menggunakan *Support Vector Machine* (SVM) sebagai algoritma klasifikasi utama, yang bekerja berdasarkan fitur-fitur visual yang diekstraksi dari citra.

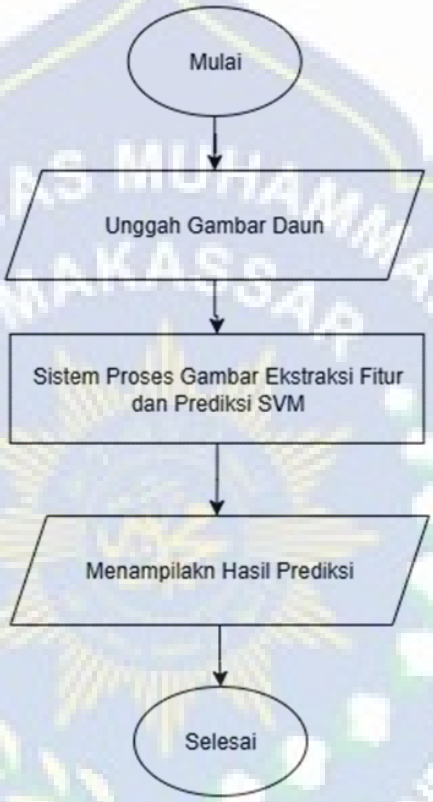
Secara garis besar, sistem ini dirancang untuk menerima input berupa citra daun selada dan menghasilkan output berupa prediksi kelas kondisi daun tersebut (Sehat, Bercak Cercospora, Tip Burn, atau Etiolosi). Modul Input Citra Menerima masukan berupa file citra digital daun selada.

1. Modul Pra-pemrosesan Citra Melakukan standardisasi pada citra input (Resizing, Normalisasi, Konversi Ruang Warna jika perlu).
2. Modul Ekstraksi Fitur Visual Mengkuantifikasi karakteristik visual penting, warna dengan statistik histogram HSV dan Tekstur dengan statistik GLCM menjadi vektor fitur numerik.
3. Modul Klasifikasi SVM Menggunakan model SVM yang telah dilatih pada vektor fitur untuk memprediksi kelas kondisi daun.
4. Modul Output Prediksi Menyajikan hasil klasifikasi (label kelas) sebagai output akhir.

Flowchart Sistem

1. **Flowchart Alur Interaksi Pengguna dengan Sistem Identifikasi Penyakit**

Untuk memberikan gambaran umum mengenai cara kerja sistem dari sudut pandang pengguna, pada Gambar 9 disajikan diagram alir yang mengilustrasikan alur interaksi.



Gambar 9. Flowchar Alur Interaksi Pengguna dengan Sistem Identifikasi Penyakit

Penjelasan Flowchart 1

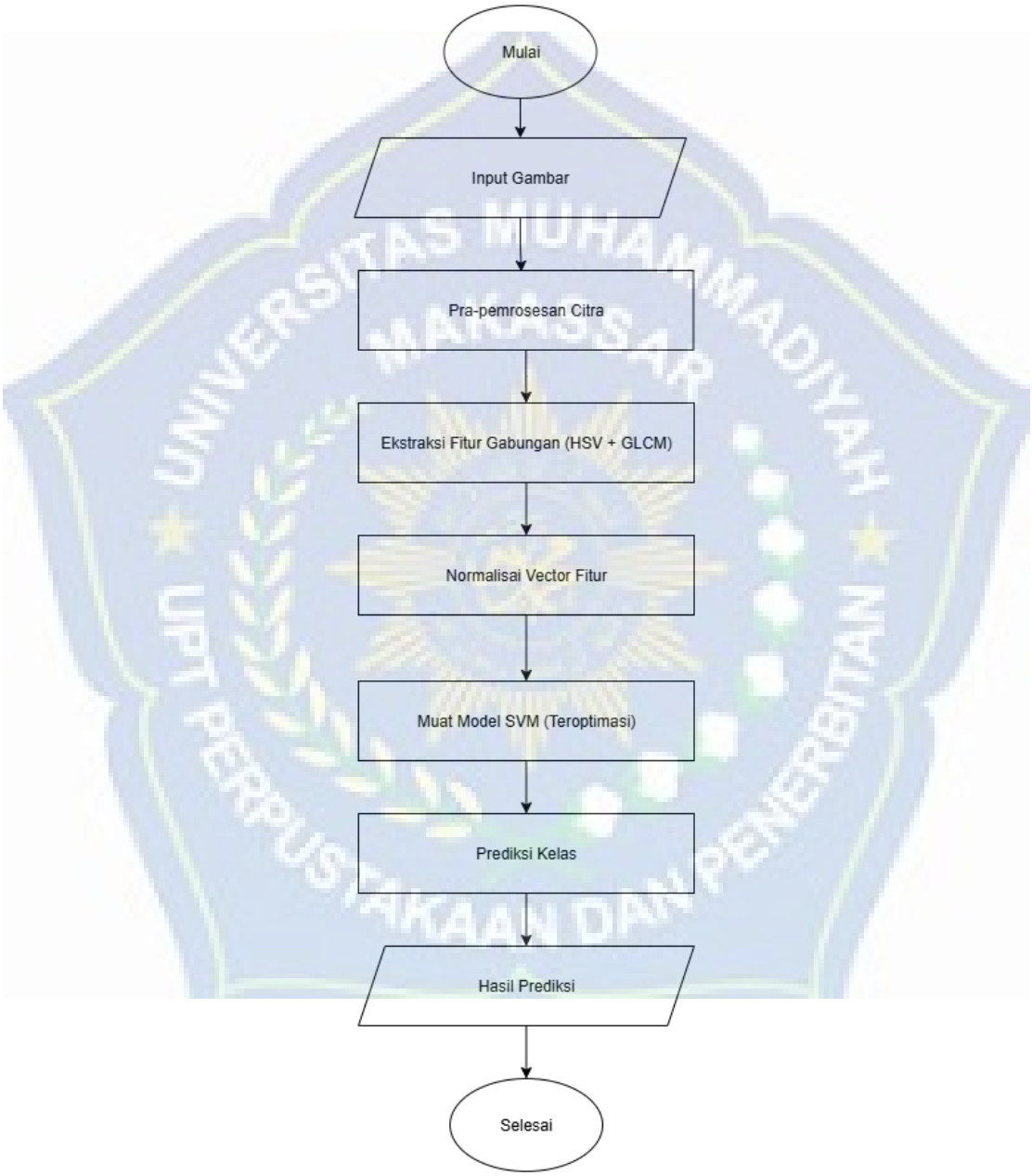
1. Proses interaksi dimulai ketika pengguna berinisiatif untuk menggunakan sistem, ditandai dengan tahap Mulai.
2. Pengguna melakukan aksi utama dengan Unggah citra Daun selada yang ingin diidentifikasi kondisinya.
3. Setelah citra berhasil diunggah dan diterima oleh sistem, tahap Sistem Proses Gambar akan berjalan.

a. Dalam tahap ini, sistem secara otomatis akan melakukan Ekstraksi Fitur visual yang relevan dari citra daun tersebut. Fitur ini mencakup karakteristik warna (misalnya, dari histogram HSV) dan tekstur (misalnya, dari statistik GLCM) .
b. Selanjutnya, berdasarkan vektor fitur yang telah diekstraksi, model *Support Vector Machine* (SVM) yang telah dilatih akan melakukan Prediksi SVM untuk menentukan kelas kondisi atau jenis penyakit daun.

4. Setelah proses prediksi selesai, Sistem Menampilkan Hasil Prediksi kepada pengguna. Hasil ini akan menunjukkan apakah daun selada tersebut sehat atau teridentifikasi mengidap salah satu penyakit yang telah ditentukan (Bercak Cercospora, Tip Burn, atau Embun Etiolasi).
5. Siklus interaksi pengguna untuk identifikasi satu gambar daun selada kemudian Selesai .

2. Flowchart Tahapan Umum Sistem Identifikasi Penyakit Menggunakan SVM

Untuk menjelaskan alur kerja internal sistem saat melakukan prediksi, Gambar 10 menyajikan diagram alir yang merinci tahapan-tahapan teknisnya.



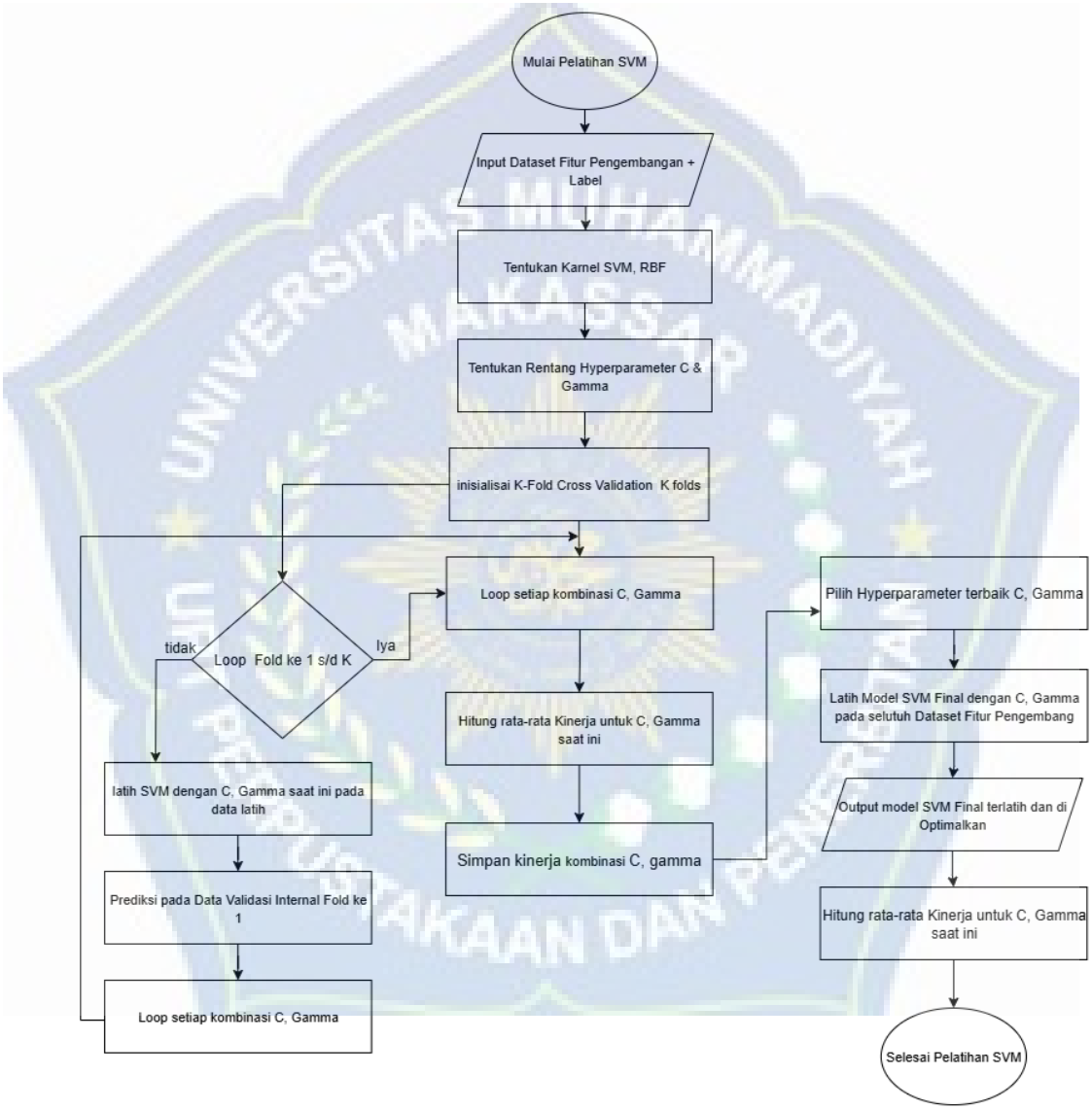
Gambar 10. Flowchart Tahapan Umum Sistem Identifikasi Penyakit Menggunakan SVM

Penjelasan Flowchart 2

1. Mulai. Ini adalah titik awal dari proses prediksi, yang dipicu ketika pengguna mengunggah sebuah gambar.
2. Input Gambar. Sistem menerima masukan berupa sebuah gambar digital daun selada yang akan dianalisis.
3. Pra-pemrosesan Citra. Gambar input yang diterima kemudian melalui tahap standardisasi untuk memastikan konsistensi. Proses ini dapat mencakup penyesuaian ukuran gambar (*resizing*) ke dimensi yang sama dengan data latih (224x224 piksel).
4. Ekstraksi Fitur Gabungan. Pada tahap ini, sistem menganalisis gambar yang telah diproses untuk mengekstrak ciri-ciri visual penting dan mengubahnya menjadi data numerik (vektor fitur). Sesuai dengan metodologi penelitian, fitur yang diekstraksi adalah fitur warna dari ruang HSV dan fitur tekstur dari GLCM.
5. Normalisasi Vektor Fitur. Vektor fitur yang dihasilkan dari tahap sebelumnya kemudian dinormalisasi. Proses ini menyesuaikan skala nilai dari setiap fitur ke dalam rentang antara 0 dan 1 menggunakan skaler yang telah disimpan dari proses pelatihan. Ini adalah langkah krusial untuk memastikan bahwa data input memiliki skala yang sama dengan data yang digunakan untuk melatih model.
6. Muat Model SVM Terbaik (Teroptimasi). Sistem memuat model machine learning yang telah dilatih, divalidasi, dan dioptimalkan pada tahap pengembangan model dari Skenario 3. Model ini sudah berisi semua "pengetahuan" yang diperlukan untuk melakukan klasifikasi.
7. Prediksi Kelas. Vektor fitur yang sudah dinormalisasi dimasukkan sebagai input ke dalam model SVM yang telah dimuat. Model kemudian memproses data ini untuk membuat prediksi, menentukan gambar tersebut masuk ke dalam kategori (kelas) penyakit mana (Sehat, *Cercospora beticola*, Tip Burn, atau Etiolasi).
8. Hasil Prediksi. Hasil dari prediksi kelas, beserta tingkat keyakinan (confidence score), ditampilkan sebagai output akhir kepada pengguna melalui antarmuka.
9. Selesai. Ini menandakan bahwa proses identifikasi untuk satu gambar telah selesai.

3. Flowchart Detail Proses Pelatihan Model SVM

Keberhasilan sistem klasifikasi sangat bergantung pada kualitas model yang dilatih. Gambar 11 menyajikan diagram alir terperinci mengenai proses pelatihan dan optimasi model *Support Vector Machine* (SVM) yang dilakukan pada tahap pengembangan.



Gambar 11. Flowchart Detail Proses Pelatihan Model SVM

Penjelasan Flowchart 3

1. Input: *Dataset* Fitur Pengembangan + Label

   Dataset yang digunakan terdiri dari

a. Fitur (X) variabel independen atau input.
b. Label (Y) variabel target yang ingin diprediksi.

Dataset ini digunakan untuk melatih dan mengevaluasi performa model.

2. Tentukan kernel SVM. RBF

Model SVM yang digunakan memakai Radial Basis Function (RBF) sebagai kernel. Kernel ini cocok untuk menangani data non-linear dengan memetakan data ke dimensi yang lebih tinggi.

3. Tentukan Rentang *Hyperparameter* C dan gamma

Sebelum pelatihan dimulai, ditentukan terlebih dahulu rentang nilai

a. C. parameter regulasi (penyeimbang antara kesalahan klasifikasi dan margin).
b. gamma (γ) parameter kernel yang mengontrol seberapa jauh pengaruh satu data pelatihan.

Nilai-nilai ini akan diuji satu per satu untuk mencari kombinasi terbaik.

4. Inisialisasi *K-Fold Cross-Validation*

Dataset dibagi menjadi K bagian (folds). Pada setiap iterasi

a. Satu fold digunakan sebagai data validasi.
b. Sisa fold digunakan sebagai data latih.

Teknik ini membantu menilai performa model secara lebih menyeluruh dan mengurangi overfitting.

5. Loop Setiap Kombinasi C, gamma

Dilakukan pencarian grid (grid search) untuk menguji semua kombinasi nilai C dan gamma yang telah ditentukan.

6. Loop Fold ke-1 s/d K

Untuk setiap kombinasi C dan gamma:

a. Dilakukan proses pelatihan dan evaluasi menggunakan *K-Fold Cross-Validation.*

Langkah-langkahnya

1.) Latih SVM dengan C, gamma saat ini pada Data Latih Internal

Model SVM dilatih menggunakan data latih dari fold ke-i.

2.) Prediksi pada Data Validasi Internal Fold ke-i

Model memprediksi data validasi untuk menguji performa.

3.) Hitung dan Simpan Kinerja Fold ke-i

Dihitung metrik kinerja (misalnya: akurasi, precision, recall) dan disimpan.

7. Hitung Rata-rata Kinerja untuk C, gamma Saat Ini

   Setelah semua fold diuji, dihitung nilai rata-rata kinerja sebagai representasi performa untuk kombinasi C dan gamma tersebut.

8. Simpan Kinerja Kombinasi C, gamma

   Hasil rata-rata disimpan untuk dibandingkan dengan kombinasi lainnya.

9. Pilih *Hyperparameter* Terbaik C, gamma

   Dipilih kombinasi C dan gamma yang memberikan hasil terbaik berdasarkan rata-rata kinerja.

10. Latih Model SVM Final dengan C, gamma

    Model akhir dilatih menggunakan seluruh *dataset* dengan kombinasi *hyperparameter* terbaik. Ini memastikan model belajar dari semua data yang tersedia.

11. Output: Model SVM Final Terlatih dan Dioptimalkan

    Model akhir yang telah terlatih dengan optimal siap digunakan untuk prediksi.

12. Simpan Model SVM Final

    Model disimpan dalam bentuk file atau objek agar dapat digunakan kembali pada proses inferensi atau deployment.

13. Selesai Pelatihan SVM

    Proses pelatihan selesai. Model siap digunakan untuk klasifikasi atau prediksi data baru.

D. Teknik Pengujian Sistem

Teknik pengujian sistem dirancang untuk mengevaluasi dan membandingkan kinerja model secara objektif sesuai dengan tujuan penelitian. Proses ini akan melibatkan pembagian dataset dan pengujian berbasis skenario untuk analisis komparatif.

1. Pembagian Dataset

Dataset yang telah melalui tahap pra-pemrosesan dan ekstraksi fitur akan dibagi menjadi dua bagian 80% sebagai data latih (*training set*) dan 20% sebagai data uji

(*testing set*). Proses optimasi parameter dan pelatihan model akan dilakukan pada data latih, sementara evaluasi akhir akan dilakukan pada data uji untuk mengukur kemampuan generalisasi model.

2. Desain Skenario Pengujian

Untuk menjawab rumusan masalah mengenai perbandingan kinerja, pengujian akan dilakukan dalam 3 skenario utama. Pendekatan komparatif ini bertujuan untuk menganalisis dampak dari proses optimasi parameter terhadap kinerja model SVM.

Tabel 2. Skenario Pengujian Penelitian

Skenario	Deskripsi Metode	Tujuan
A : SVM Murni	Model SVM dilatih menggunakan fitur gabungan (HSV + GLCM) dari dataset asli. Parameter yang digunakan adalah parameter *default* dari pustaka Scikit-learn tanpa optimasi.	Mengukur kinerja *baseline* (dasar) model sebagai titik awal perbandingan.
B : SVM dengan Augmentasi Data	Model SVM dilatih menggunakan fitur gabungan (HSV + GLCM) dengan parameter *default* dari pustaka Scikit-learn.	Mengukur kinerja *baseline* sebagai dasar perbandingan.
C: SVM Teroptimasi dengan Augmentasi Data dan (PSO)	Model SVM dilatih menggunakan fitur gabungan (HSV + GLCM), dan parameternya dioptimalkan oleh *Particle Swarm Optimization* (PSO).	Menganalisis dan membuktikan efektivitas proses optimasi dalam meningkatkan kinerja model.

3. Metrik Evaluasi

Kinerja dari kedua skenario akan dievaluasi dan dibandingkan menggunakan metrik standar klasifikasi, yaitu Akurasi, Presisi, *Recall*, dan F1-*Score* yang dihitung dari *confusion matrix*. Hasil perbandingan ini akan menjadi jawaban kuantitatif untuk rumusan masalah penelitian.

Gambaran output akhir dari sistem ini akan disajikan kepada pengguna dalam bentuk sebuah tampilan informasi visual yang terstruktur dan informatif. Setelah pengguna mengunggah gambar daun selada, kartu ini akan menampilkan kembali gambar tersebut di bagian atas, lengkap dengan label "*Confidence*" yang menunjukkan tingkat keyakinan model dalam bentuk persentase. Di bawah gambar, akan ditampilkan hasil diagnosis utama berupa nama penyakit ("*Cercospora Beticola*, Tip Burn, Etiolosi") sebagai judul, diikuti oleh nama ilmiah patogennya (*Bremia lactucae*). Secara keseluruhan, output ini dirancang untuk memberikan laporan diagnosis yang komprehensif, transparan, dan mudah dipahami dalam satu tampilan yang rapi.

E. Teknik Analisis Data

Analisis data dalam penelitian ini akan difokuskan pada interpretasi kuantitatif dari metrik-metrik kinerja yang diperoleh dari proses pengujian model *Support Vector Machine* (SVM) serta pemahaman terhadap pola klasifikasi yang dihasilkan. Data utama yang dianalisis adalah nilai akurasi, presisi, recall, dan F1-score yang dihitung dari evaluasi pada set pengujian akhir. Akurasi akan memberikan gambaran umum efektivitas model, sementara presisi, recall, dan F1-score akan dianalisis untuk setiap kelas penyakit dan kondisi sehat guna memahami kemampuan model dalam mengenali masing-masing kategori secara spesifik dan keseimbangan antara kesalahan prediksi positif palsu dengan negatif palsu. Lebih lanjut, analisis mendalam terhadap *confusion matrix* akan dilakukan untuk mengidentifikasi jenis-jenis misklasifikasi yang paling sering terjadi antar kelas, serta untuk mengetahui kelas mana yang paling sulit atau paling mudah untuk diidentifikasi oleh model. Temuan dari analisis metrik dan *confusion matrix* ini akan digunakan untuk menarik kesimpulan mengenai keberhasilan model SVM berbasis fitur visual dalam mengidentifikasi penyakit daun selada dan untuk memberikan dasar rekomendasi penelitian selanjutnya.

BAB IV

HASIL DAN PEMBAHASAN

A. Deskripsi Data Penelitian

Dataset yang digunakan dalam penelitian ini merupakan data primer yang diperoleh melalui pengambilan gambar secara langsung di lokasi budidaya DEEDAD HIDROPONIK, Makassar. Dataset ini secara spesifik berfokus pada citra daun selada untuk mengidentifikasi empat kondisi kesehatan yang berbeda.

1. Komposisi Dataset Induk

Dataset induk (sebelum augmentasi) terdiri dari 230 citra asli yang dikumpulkan dan telah dilabeli oleh peneliti. Kumpulan data ini menjadi dasar untuk semua skenario pengujian. Dataset dibagi menjadi empat kelas kategori, yaitu

a. Sehat. Citra daun selada yang tidak menunjukkan gejala penyakit.
b. *Cercospora beticola*. Daun selada yang terinfeksi penyakit Bercak Daun Cercospora.
c. Tip Burn. Daun selada yang menunjukkan gejala gosong pada tepi daun.
d. Etiolasi. Daun selada yang menunjukkan gejala pertumbuhan pucat akibat kekurangan cahaya.

Jumlah total citra dalam dataset induk adalah 230 gambar, dengan rincian sebagai berikut

Tabel 3. Komposisi Dataset Induk

No	Kelas Kategori	Jumlah Gambar
1.	Sehat	49
2.	Cercospora beticola	79
3.	Tip Burn	54
4.	Etiolasi	48
	Total	230

Tabel 4. Gambar Dataset

a. Sehat	b. Cercospora beticola
	
c. Tip Burn	**d. Etiolasi**
	

Berikut adalah contoh visual untuk setiap kelas dalam dataset

B. Hasil Implementasi dan Pengujian

1. Skenario 1 (SVM Murni) Ekstraksi Fitur

Pada Model SVM dilatih hanya menggunakan dataset asli (226 gambar) tanpa augmentasi, menggunakan parameter standar (C=1.0, gamma='scale'). Skenario ini menjadi *baseline* paling dasar.

a. Akurasi Keseluruhan = 60.87%

2. Skenario 2 (SVM + Augmentasi)

Model SVM dilatih menggunakan dataset yang telah diperkaya dengan augmentasi data dinamis, namun tanpa optimasi hyperparameter.

a. Akurasi Keseluruhan : 91.60%

3. Skenario 3 (SVM + Augmentasi + PSO)

Skenario final di mana dataset diperkaya dengan augmentasi, dan hyperparameter SVM (C dan gamma) dioptimalkan menggunakan Particle Swarm Optimization (PSO) untuk performa maksimal.

a. Akurasi Keseluruhan : 96.00%

Ringkasan akurasi keseluruhan dari ketiga skenario disajikan pada Tabel 5

Tabel 5. Ringkasan Akurasi Keseluruhan Model per Skenario

Skenario Pengujian	**Deskripsi Singkat**	**Akurasi Keseluruhan**
Skenario 1 (SVM Murni)	Model dilatih hanya dengan fitur dari dataset asli.	60.87%
Skenario 2 (SVM + Augmentasi)	Model dilatih dengan dataset yang diperkaya augmentasi data dinamis.	91.60%
Skenario 3 (SVM + Augmentasi + PSO)	Model dari Skenario 2 dioptimalkan lebih lanjut menggunakan PSO.	96.00%

Perhitungan manual untuk setiap skenario di atas adalah sebagai berikut.

1) Skenario 1 (SVM Murni).

   Akurasi = (28 Prediksi Benar / 46 Total Data Uji) x 100% = 60.87%

2) Skenario 2 (SVM + Augmentasi).

   Akurasi = (229 Prediksi Benar / 250 Total Data Uji) x 100% = 91.60%

3) Skenario 3 (SVM + Augmentasi + PSO).

   Akurasi = (240 Prediksi Benar / 250 Total Data Uji) x 100% = 96.00%

Jumlah data uji pada Skenario 2 dan 3 lebih banyak. Hal ini disebabkan karena pada kedua skenario tersebut, proses augmentasi data dilakukan di awal sebelum

pembagian data latih dan uji, sehingga total dataset yang dibagi menjadi jauh lebih besar.

C. Pembahasan

Bagian ini menganalisis dan menginterpretasikan hasil dari ketiga skenario pengujian untuk menjawab rumusan masalah.

1. Analisis Perbandingan Kinerja Antar Skenario

Perbandingan akurasi dari ketiga skenario divisualisasikan pada Gambar 12 menunjukkan dampak yang jelas dari setiap teknik yang diterapkan terhadap kinerja model.

Gambar 12. Perbandingan Kinerja Antar Skenario

Analisis dari grafik tersebut mengarah pada dua temuan utama.

a. Kinerja Baseline: Skenario 1 (SVM Murni) menghasilkan akurasi 60.87%. Angka ini menjadi titik awal yang menunjukkan kinerja model pada kondisi paling dasar tanpa perlakuan khusus.

b. Dampak Krusial Augmentasi Data: Lompatan kinerja terbesar terjadi pada Skenario 2. Dengan menerapkan augmentasi data pada dataset, akurasi melonjak drastis menjadi 91.60%. Peningkatan lebih dari 30% ini membuktikan secara konklusif bahwa augmentasi data adalah strategi paling vital untuk meningkatkan kemampuan generalisasi model.

c. Peran Optimasi PSO sebagai Penyempurna: Skenario 3 menunjukkan bahwa optimasi hyperparameter menggunakan PSO memberikan peningkatan lebih lanjut, mencapai akurasi puncak 96.00%. Ini menunjukkan bahwa proses fine-tuning berhasil menyempurnakan model untuk mencapai performa terbaiknya.

2. Analisis Kinerja Model Terbaik (Skenario 3)

Model terbaik (Skenario 3) dengan akurasi 96.00% dievaluasi lebih lanjut menggunakan Laporan Klasifikasi (Gambar 13) dan Confusion Matrix (Gambar 14) untuk menganalisis pola prediksi secara mendetail.

Sebelum menganalisis hasilnya, penting untuk memahami bagaimana metrik-metrik evaluasi utama dihitung dari nilai True Positive (TP), True Negative (TN), False Positive (FP), dan False Negative (FN). Metrik ini dihitung untuk setiap kelas guna memahami kinerja model pada masing-masing kategori.

a. Akurasi

$$\frac{79 + 54 + 51 + 56}{(79 + 1 + 0 + 2) + (0 + 54 + 0 + 0) + (1 + 0 + 51 + 3) + (2 + 0 + 1 + 56)}$$

$$= \frac{240}{250} = 0.960 = 96.0\%$$

b. Presisi Perkelas

1) Cercospora beticola (CB)

$$= \frac{79}{79 + 3} = \frac{79}{82} = 0.9634146 \approx 0.963\ (96.34\%)$$

2) Etiolasi (Et)

$$= \frac{54}{54 + 1} = \frac{54}{55} = 0.9818182 \approx 0.982\ (98.18\%)$$

3) Sehat (Sh)

$$= \frac{51}{51+1} = \frac{51}{52} = 0.9807692 = 0.981\ (98.08\%)$$

4) Tip Burn (TB)

$$= \frac{56}{56+5} = \frac{56}{61} = 0.9180328 = 0.918\ (91.80\%)$$

Rata-rata Presisi (Macro)

$$= \frac{0.9634146 + 0.9818182 + 0.9807692 + 0.9180328}{4}$$

$$= 0.9610087 = 0.961\ (96.10\%)$$

c. Recall Perkelas

1) Cercospora beticola (CB)

$$= \frac{79}{79+3} = \frac{79}{82} = 0.9634146 = 0.963\ (96.34\%)$$

2) Etiolasi (Et)

$$= \frac{54}{54+0} = \frac{54}{54} = 1.0000000 = 1.000\ (100.00\%)$$

3) Sehat (Sh)

$$= \frac{51}{51+4} = \frac{51}{55} = 0.9272727 = 0.927\ (92.73\%)$$

4) Tip Burn (TB)

$$= \frac{56}{56+3} = \frac{56}{59} = 0.9491525 = 0.949\ (94.92\%)$$

Rata-rata Recall (macro)

$$= \frac{0.9634146 + 1.0000000 + 0.9272727 + 0.9491525}{4}$$

$$= 0.95995998 = 0.960\ (96.00\%)$$

d. F1-Score per kelas (substitusi langsung bentuk 2TP/(2TP+FP+FN))

1) Cercospora beticola (CB)

$$= \frac{2 \times 79}{2 \times 79 + 3 + 3} = \frac{158}{164} = 0.9634146 \approx 0.963$$

2) Etiolasi (Et)

$$= \frac{2 \times 54}{2 \times 54 + 1 + 0} = \frac{108}{109} = 0.9908257 \approx 0.991$$

3) Sehat (Sh)

$$= \frac{2 \times 51}{2 \times 51 + 1 + 4} = \frac{102}{107} = 0.9532710 \approx 0.953$$

4) Tip Burn (TB)

$$= \frac{2 \times 56}{2 \times 56 + 5 + 3} = \frac{112}{120} = 0.9333333 \approx 0.933$$

Rata-rata F1 (macro)

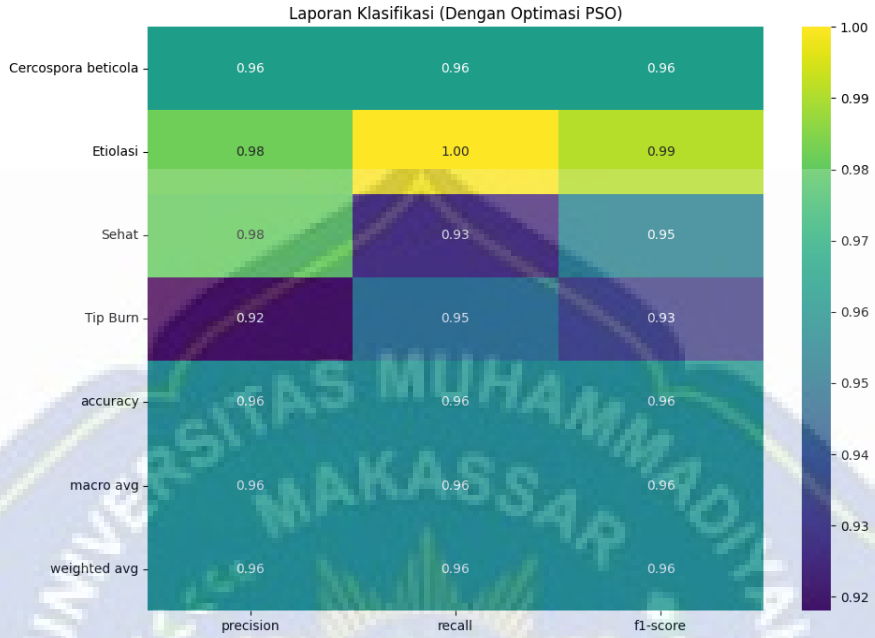
$$= \frac{0.9634146 + 0.9908257 + 0.9532710 + 0.9333333}{4}$$

$$= 0.96021117 = 0.960\ (96.02\%)$$

e. Rekap akhir

1) Akurasi = 0.960 → 96.0%
2) Presisi (per kelas) = CB 96.34%, Et 98.18%, Sh 98.08%, TB 91.80%
   Presisi rata-rata (macro) = 96.10%
3) Recall (per kelas) = CB 96.34%, Et 100.00%, Sh 92.73%, TB 94.92%
   Recall rata-rata (macro) = 96.00%
4) F1 (per kelas) = CB 96.34%, Et 99.08%, Sh 95.33%, TB 93.33%
   F1 rata-rata (macro) = 96.02%

Perhitungan inilah yang menghasilkan nilai-nilai yang akan ditampilkan pada laporan klasifikasi pada gambar 13



Gambar 13. Laporan Klasifikasi Model Terbaik

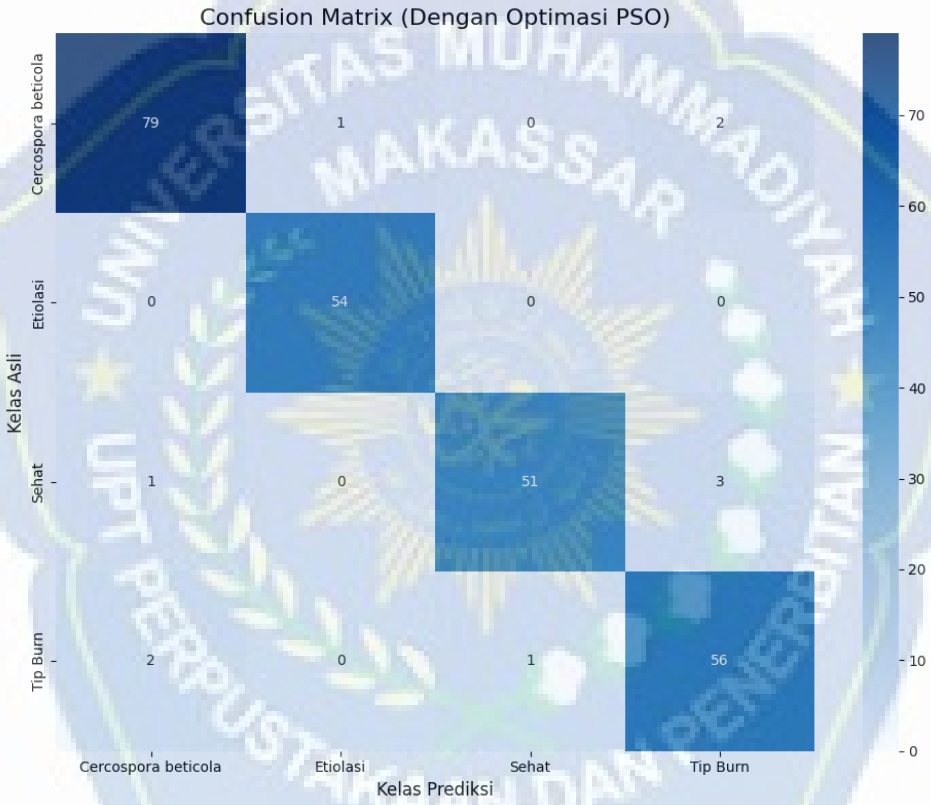
Dari laporan klasifikasi, terlihat bahwa model mencapai skor presisi, recall, dan f1-score yang sangat tinggi (umumnya di atas 0.91) untuk semua kelas, yang menandakan kinerja model sangat baik dan seimbang dalam mengenali setiap kategori. Berikut adalah rinciannya.

a. Cercospora beticola. Presisi = 0.96, Recall = 0.96, F1-Score = 0.96.
   Ini menunjukkan kinerja yang sangat seimbang dan akurat. Model sangat andal dalam memprediksi kelas ini dan berhasil mengidentifikasi hampir semua kasus yang sebenarnya.
b. Etiolasi. Presisi = 0.98, Recall = 1.00, F1-Score = 0.99.
   Ini adalah kinerja terbaik. Recall 1.00 berarti model berhasil menemukan semua kasus Etiolasi tanpa ada yang terlewat. Presisi yang sangat tinggi juga berarti prediksinya hampir selalu benar.
c. Sehat, Presisi = 0.98, Recall = 0.93, F1-Score = 0.95. Model sangat presisi, jika ia memprediksi 'Sehat', kemungkinan besar itu benar. Namun, ada beberapa

kasus 'Sehat' yang terlewat dan salah diklasifikasikan sebagai penyakit lain (recall sedikit lebih rendah).

d. Tip Burn. Presisi = 0.92, Recall = 0.95, F1-Score = 0.93. Kinerja yang sangat baik. Model berhasil mengidentifikasi sebagian besar kasus Tip Burn (recall tinggi), meskipun ada sedikit kasus di mana kelas lain salah diprediksi sebagai Tip Burn (presisi sedikit lebih rendah).

Selanjutnya, Confusion Matrix (Gambar 14) digunakan untuk menganalisis pola kesalahan secara lebih spesifik.



Gambar 14. Confusion Matrix Model Terbaik

Analisis dari matriks tersebut menunjukkan bahwa model memiliki tingkat prediksi benar yang sangat tinggi di semua kelas (angka di sepanjang diagonal). Untuk menganalisis pola kesalahan secara lebih spesifik, dapat dilihat beberapa poin berikut

a. Sebagian besar prediksi (240 dari 250) berada di diagonal utama, yang mengonfirmasi akurasi model sebesar 96.00%.

b. Kelas Etiolasi menunjukkan kinerja sempurna dari sisi recall, di mana tidak ada satupun gambar Etiolasi yang salah diklasifikasikan.
c. Terdapat beberapa misklasifikasi yang menjadi catatan, misalnya. 3 kasus Seh at salah diprediksi sebagai Tip Burn.2 kasus Tip Burn salah diprediksi sebagai *Cercospora beticola*. 2 kasus *Cercospora beticola* salah diprediksi sebagai Ti p Burn.
d. Pola kesalahan ini, terutama antara kelas Sehat, Tip Burn, dan Cercospora, kemungkinan disebabkan oleh kemiripan visual pada tahap awal penyakit atau variasi pencahayaan yang membuat fitur-fitur tertentu tumpang tindih. Analisis ini memberikan wawasan berharga untuk perbaikan model di masa mendatang.

Dari *confusion matrix* ini, dapat ditarik rincian nilai True Positive (TP), True Negative (TN), False Positive (FP), dan False Negative (FN) untuk setiap kelas, yang disajikan pada Tabel 6

Tabel 6. Rincian Metrik Evaluasi (TP, TN, FP, FN) dari Confusion Matrix

Kelas Kategori	True Positive (TP)	True Negatif (TN)	False Positive (FP)	False Negatif (FN)
Cercospora Beticola	79	165	3	3
Etiolasi	54	195	1	0
Tip Burn	51	194	1	4
Sehat	56	186	5	3

3. Analisis Detail Kesalahan Klasifikasi

Meskipun model menunjukkan kinerja yang sangat kuat secara keseluruhan, analisis yang lebih dalam terhadap kasus-kasus di mana model melakukan kesalahan dapat memberikan wawasan berharga. Berdasarkan hasil pengujian pada model usulan.

a. Total Sampel Data Uji = 250

b. Total Kesalahan Klasifikasi= 10

c. Tingkat Kesalahan (Error Rate) = 4.0% (10 dari 250)

Kesalahan-kesalahan tersebut terdistribusi seperti yang ditunjukkan pada Tabel 7 Ringkasan Kesalahan Klasifikasi

Tabel 7. Ringkasan Kesalahan Klasifikasi

Kelas Aktual	**Kelas Prediksi**	**Jumlah Kasus**
Cercospora beticola	Etiolasi	1
Cercospora beticola	Tip Burn	2
Sehat	Cercospora beticola	1
Sehat	Tip Burn	3
Tip Burn	Cercospora beticola	2
Tip Burn	Sehat	1

Dari tabel 7, terlihat bahwa tidak ada kesalahan klasifikasi sama sekali untuk kelas Etiolasi. Pola kesalahan paling umum terjadi antara kelas Sehat dan Tip Burn, serta antara *Cercospora beticola* dan *Tip Burn*, yang mengindikasikan adanya kemiripan fitur visual di antara kelas-kelas tersebut yang masih menjadi tantangan bagi model.

Untuk memberikan gambaran visual yang lebih jelas mengenai pola kesalahan yang teridentifikasi, berikut adalah rincian visual untuk beberapa kasus misklasifikasi yang representatif dalam tabel 8. Berikut adalah rincian visual untuk setiap gambar yang salah diklasifikasikan.

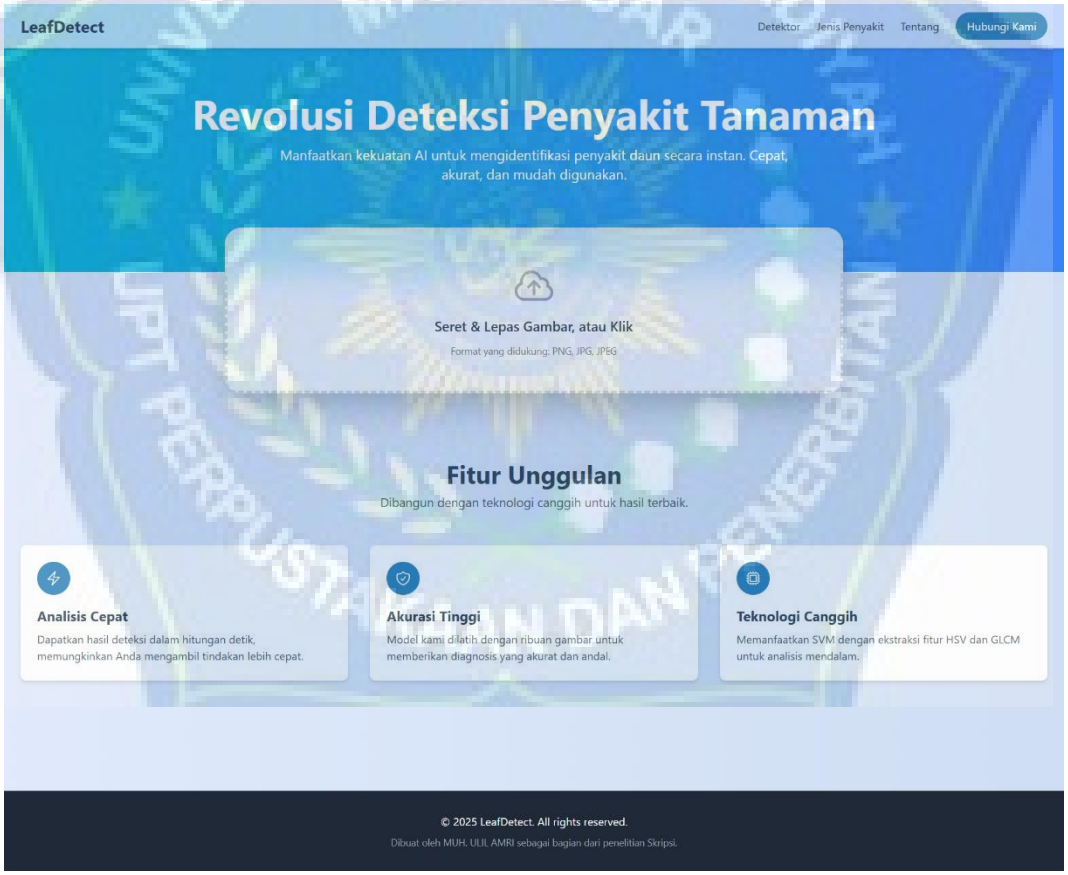
Tabel 8. Kasus Misklasifikasi Representatif

No	Gambar	Kelas Sebenarnya	Prediksi Model
1.		Cercospora Beticola	Etiolasi
2.		Cercospora Beticola	Sehat
4.		Sehat	Cercospora Beticola
5.		Tip Burn	Cercospora Beticola

D. Tampilan Antarmuka Sistem

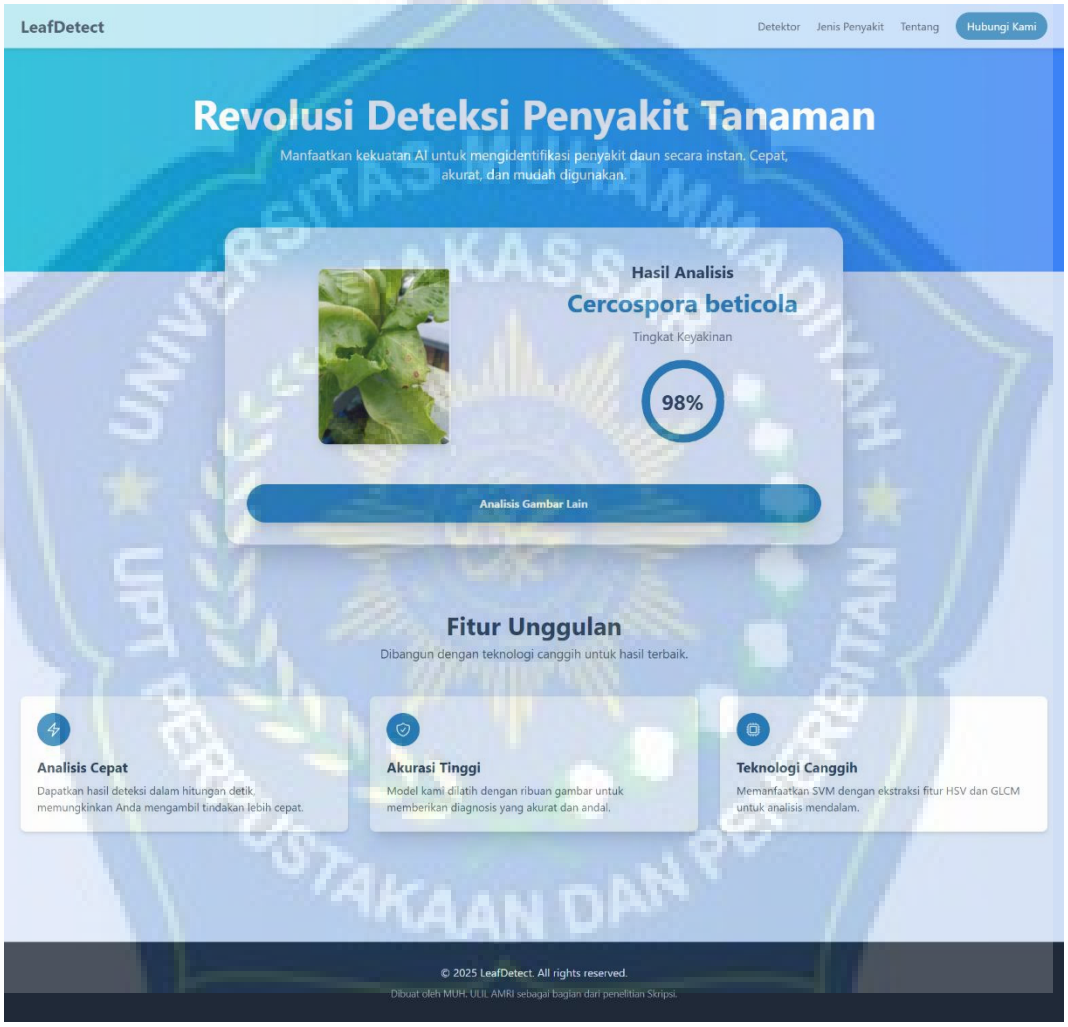
Sebagai bukti konsep dan implementasi praktis dari model yang telah dikembangkan, sebuah antarmuka pengguna berbasis web telah dibuat. Antarmuka ini memungkinkan pengguna untuk berinteraksi dengan model klasifikasi secara mudah dan intuitif, menjembatani kesenjangan antara penelitian teknis dan aplikasi dunia nyata.

Gambar 15 menunjukkan halaman utama dari aplikasi "LeafDetect". Pada halaman ini, pengguna disambut dengan area unggah yang interaktif, di mana mereka dapat menyeret dan melepas (*drag and drop*) atau mengklik untuk memilih file gambar daun selada yang ingin diidentifikasi. Desain yang bersih dan *user frie ndly* memastikan pengalaman yang mulus bagi pengguna.



Gambar 15. Tampilan Halaman Utama Aplikasi

Setelah pengguna mengunggah gambar, gambar tersebut dikirim ke backend API untuk diproses oleh model SVM terbaik (Skenario 3). Hasil prediksi kemudian ditampilkan kembali kepada pengguna, seperti yang terlihat pada Gambar 16 Tampilan hasil menyajikan informasi penting secara jelas, yaitu gambar yang dianalisis, nama penyakit yang diprediksi dalam contoh ini "Cercospora beticola", dan tingkat keyakinan (*confidence*) dari model dalam bentuk persentase.



Gambar 16. Tampilan Hasil Prediksi Penyakit

Implementasi antarmuka ini menunjukkan bahwa model yang dihasilkan tidak hanya akurat secara teoretis, tetapi juga berhasil dioperasionalkan menjadi sebuah sistem yang fungsional dan mudah digunakan, memberikan nilai tambah praktis dari penelitian ini.

BAB V

PENUTUP

A. Kesimpulan

Berdasarkan hasil implementasi, pengujian, dan analisis yang telah dilakukan, dapat ditarik beberapa kesimpulan sebagai berikut

1. Implementasi dan Perbandingan Kinerja Model. Implementasi tiga skenario model Support Vector Machine (SVM) berhasil dilakukan. Perbandingan kinerjanya menunjukkan bahwa. Model SVM Murni menghasilkan akurasi dasar 60.87%. Penambahan augmentasi data secara signifikan meningkatkan akurasi menjadi 91.60%. Model final yang menggunakan augmentasi data dan optimasi Particle Swarm Optimization (PSO) mencapai akurasi puncak sebesar 96.00%.
2. Signifikansi Dampak Augmentasi dan Optimasi. *Augmentasi data* terbukti memberikan dampak paling signifikan, dengan peningkatan akurasi lebih dari 30%, yang menegaskan perannya sebagai strategi vital untuk meningkatkan generalisasi model. Optimasi PSO memberikan peningkatan kinerja lebih lanjut yang esensial, menyempurnakan model untuk mencapai akurasi final 96.00%, membuktikan bahwa kombinasi kedua teknik sangat efektif dan andal

B. Saran

Berdasarkan kesimpulan dan keterbatasan yang ada dalam penelitian ini, berikut adalah beberapa saran untuk pengembangan di masa depan

1. Pengembangan Pengembangan Dataset dan Analisis Fitur. Mengingat pentingnya data, disarankan untuk memperkaya dataset dengan lebih banyak variasi gambar asli (kondisi pencahayaan, latar belakang, dan tingkat keparahan penyakit). Selain itu, melakukan analisis signifikansi fitur dapat memberikan wawasan tentang fitur mana yang paling dominan, yang berpotensi menyederhanakan model.
2. Eksplorasi Model Alternatif (Deep Learning). Sebagai pembanding dari pendekatan rekayasa fitur manual + SVM, penelitian selanjutnya dapat mengimplementasikan model berbasis Deep Learning (seperti Convolutional

Neural Networks/CNN). Model CNN dapat mempelajari fitur secara otomatis dari gambar dan berpotensi memberikan tingkat akurasi yang lebih tinggi lagi.

3. Pengembangan Aplikasi Berbasis Pengguna. Melanjutkan pengembangan antarmuka yang telah dibuat menjadi aplikasi *full-stack* yang matang. Aplikasi ini dapat diuji coba oleh pengguna akhir (petani atau praktisi) untuk mendapatkan umpan balik mengenai fungsionalitas, usabilitas, dan manfaat praktisnya di lapangan.

DAFTAR PUSTAKA

Alfaruq, B. M., Erwanto, D., & Yanuartanti, I. (2023). Klasifikasi Kematangan Buah Tomat Dengan Metode Support Vector Machine. Dalam *Generation Journal* (Vol. 7, Nomor 3).

Andono, P. N., & Rachmawanto, E. H. (2021). Terakreditasi SINTA Peringkat 2 Evaluasi Ekstraksi Fitur GLCM dan LBP Menggunakan Multikernel SVM untuk Klasifikasi Batik. *masa berlaku mulai*, *1*(3), 1–9. https://doi.org/10.29207/resti.v5i1.265

Dijaya, R. (2023). *Buku Ajar Pengolahan Citra Digital*. Umsida Press. https://doi.org/10.21070/2023/978-623-464-075-5

Fadli Gunardi, M. (2022). *Implementasi Augmentasi Citra pada Suatu Dataset*.

Fatham, M., Akbar, M., Fitriyah, H., & Akbar, S. R. (2023). *Analisis Color Space untuk Spesifikasi Perancangan Perangkat Lunak pada Embedded System Deteksi Penyakit Busuk Selada* (Vol. 7, Nomor 5). http://j-ptiik.ub.ac.id

Feiters Tampinongkol, F., Herdian, C., Basri, H., & Halim, L. (2023). *Techno Xplore Jurnal Ilmu Komputer dan Teknologi Informasi Identifikasi Penyakit Daun Tomat Menggunakan Gray Level Co-occurrence Matrix (GLCM) dan Support Vector Machine (SVM)*. https://www.kaggle.com/

Fitrian, A., Bafdal, N., Dwiratna, S., & Perwitasari, N. (2023). *Respon Pertumbuhan dan Hasil Tanaman Selada Romaine (Lactuca sativa L. var. Longifolia) Terhadap Perbedaan Jarak Tanam Pada Smart Watering System SWU 02 Growth Response and Yield of Romaine Lettuce (Lactuca sativa L. var. Longifolia) Against Differences in Planting Spacing in Smart Watering System SWU 02*.

Florensia, N. P., -, R.-, Patimah, Y.-, & Chusnul Khotimah, Y. N. (2025). ANALISIS PENGUJIAN SISTEM PAKAR PENYAKIT SELADA

MENGGUNAKAN METODE BLACK BOX & WHITE BOX TESTING. *Jurnal Informatika dan Teknik Elektro Terapan*, *13*(2). https://doi.org/10.23960/jitet.v13i2.6394

Huda Dwi Putra, K., & Yulianto, F. (2024). *ANALISIS PENYAKIT PADA TUMBUHAN HIDROPONIK SELADA MENGGUNAKAN METODE FORWARD CHAINING*. *18*(2). https://doi.org/https://doi.org/10.47111/jti.v18i2.14957

Husen, D., Kusrini, K., & Kusnawi, K. (2022). Deteksi Hama Pada Daun Apel Menggunakan Algoritma Convolutional Neural Network. *JURNAL MEDIA INFORMATIKA BUDIDARMA*, *6*(4), 2103. https://doi.org/10.30865/mib.v6i4.4667

İKiz, B., Daşgan, H. Y., & Öz, B. C. (2024). Mitigating Tipburn True Foliar Calcium Application in Indoor Hydroponicly Grown Mini Cos Lettuce. *BIO Web of Conferences*, *85*. https://doi.org/10.1051/bioconf/20248501003

Indah Kusuma, M., Orbit, J., Safira, W. I., Zuhri, N. M., Maulida, N., & Ayomi, S. (2024). Kerusakan Tanaman Selada (Lactuca sativa L) oleh OPT pada Budidaya Hidroponik di Kota Semarang. Dalam *Journal of Integrated Agricultural Socio Economics and Entrepreneurial Research* (Vol. 2, Nomor 2).

Juniarti, P. M., Arrahman, H. H. M., & Sulistianingsih, N. (2025). *Implementasi Gray Level Co-Occurrence Matrix (GLCM) Untuk Mendeteksi Penyakit Daun Pada Tanaman Holtikultura* (Vol. 1, Nomor 1). https://journal.ummat.ac.id/index.php/tcs

Mulyana, I. D., & Wibowo, R. D. (2023). *IMPLEMENTASI TINGKAT KEMATANGAN BUAH MONK DENGAN MENGGUNAKAN EKSTRAKSI GRAY-LEVEL CO-OCCURRENCE MATRIX (GLCM) DAN SUPPORT VECTOR MACHINE (SVM)*.

Nasution, T. H., & Andayani, U. (2017). Recognition of roasted coffee bean levels using image processing and neural network. *IOP Conference Series: Materials Science and Engineering*, *180*(1). https://doi.org/10.1088/1757-899X/180/1/012059

Ningrum, S. S., Nurdina Widanti, Sri Wiji Lestari, Ahmad Hafizh, & Yose Efraim Sitorus Pane. (2024). SMART GREENHOUSE PADA KELOMPOK USAHA MIKRO CAHAYA FARM DI CARANG PULANG. *JUARA: Jurnal Wahana Abdimas Sejahtera*, 1–11. https://doi.org/10.25105/h8mt2g79

Nti, I. K., Nyarko-Boateng, O., & Aning, J. (2021). Performance of Machine Learning Algorithms with Different K Values in K-fold CrossValidation. *International Journal of Information Technology and Computer Science*, *13*(6), 61–71. https://doi.org/10.5815/ijitcs.2021.06.05

Nursanti, A., Sucianto, E. T., & Mumpuni, A. (2021). Identifikasi Jamur Patogen dan Tingkat Persentase Penyakit pada Tanaman Selada (Lactuca sativa L.) di Sentra Tanaman Sayur Desa Serang, Kecamatan Karangreja, Kabupaten Purbalingga. *Jurnal Ilmiah Biologi Unsoed*, *3*(1), 9–19.

Pramudiyah, R., Asyraq, C., Kadafi, A., & Putra, S. R. (2024). *ANALISIS GAMBAR MENGGUNAKAN METODE GRAYSCALE DAN HSV (HUE, SATURATION, VALUE)*.

Pratama, A. N. S., & Abadi, A. L. (2024). PENYAKIT BERCAK DAUN PADA TANAMAN SIOMAK (Lactuca sativa L. var. augustuna) SERTA PENGENDALIANNYA DI KOMUNITAS PERTANIAN ORGANIK BRENJONK MOJOKERTO. *Jurnal Hama dan Penyakit Tumbuhan*, *12*(3), 158–172. https://doi.org/10.21776/ub.jurnalhpt.2024.012.3.4

Rusman, J., Haryati, B. Z., & Michael, A. (2023). Optimisasi Hiperparameter Tuning pada Metode Support Vector Machine untuk Klasifikasi Tingkat Kematangan Buah Kopi. *Jurnal Komputer dan Informatika*, *11*(2), 195–202. https://doi.org/10.35508/jicon.v11i2.12571

Saputra, R. A., Puspitasari, D., & Baidawi, T. (2022). Deteksi Kematangan Buah Melon dengan Algoritma Support Vector Machine Berbasis Ekstraksi Fitur GLCM. Dalam *Jurnal Infortech* (Vol. 4, Nomor 2). http://ejournal.bsi.ac.id/ejurnal/index.php/infortech200

Saputro, W. A., Andono, P. N., & Soeleman, M. A. (2025). Klasifikasi SVM Menggunakan Optimasi PSO Untuk Kelayakan Biji Kopi Dengan Level Medium Roast. *Techno.Com*, *24*(2), 543–555. https://doi.org/10.62411/tc.v24i2.12657

Siaulhak, Irsan Jaya, & Safwan Kasma. (2022). *SISTEM PENDETEKSI DAN KLASIFIKASI PENYAKIT DAUN MENGGUNAKAN IMAGE PROCESSING PENGELOLAAN CITRA DIGITAL PADA PEMBELAJARAN BIOLOGI SMAN 5 PALOPO*.

Sri Setyo, A., Fitriyah, H., & Primananda, R. (2023). *Sistem Deteksi Tip Burn pada Selada Hidroponik menggunakan Metode Thresholding pada Warna Hue Saturation Value* (Vol. 7, Nomor 3). http://j-ptiik.ub.ac.id

SUDARMA, M. I. W. S. A. N. G. R. P. N. (2021). *Identifikasi Jamur Penyebab Penyakit Utama pada Tanaman Selada (Lactuca sativa L.) Hidroponik NI PUTU RATIH SUDIARTINI GUSTI NGURAH ALIT SUSANTA WIRYA *) I MADE SUDARMA*. *10*(3). https://ojs.unud.ac.id/index.php/JAT308

Suhendra, R., & Juliwardi, I. (2022). *Identifikasi dan Klasifikasi Penyakit Daun Jagung Menggunakan Support Vector Machine*. *1*(1), 29–35. http://jurnal.utu.ac.id/JTI

Sulistiyana, F., & Anardani, S. (2023). *Seminar Nasional Teknologi Informasi dan Komunikasi-2023 "Exploring the Intersection of Big Data, Cyber Security, Aplikasi Deteksi Penyakit Tanaman Jagung Dengan Convolutional Neural Network dan Support Vector Machine*.

Talib, S., Sudin, S., Dzikrullah Suratin, M., Ahmad Dahlan No, J. K., Kec Ternate Selatan, S., & Ternate Maluku Utara, K. (2024). PENERAPAN METODE SUPPORT VECTOR MACHINE (SVM) PADA KLASIFIKASI JENIS CENGKEH BERDASARKAN FITUR TEKSTUR DAUN. *Jurnal RESTIA*, *2*(1).

Tropika, J. A., Studi, P., & Fakultas, A. (2021). *Identifikasi Jamur Penyebab Penyakit Utama pada Tanaman Selada (Lactuca sativa L.) Hidroponik NI PUTU RATIH SUDIARTINI GUSTI NGURAH ALIT SUSANTA WIRYA *) I MADE SUDARMA*. *10*(3). https://ojs.unud.ac.id/index.php/JAT308

Wariyanto Abdullah, R., Kusumastuti, R., & Handoko. (2025). Analisis Pengolahan Ekstraksi Fitur Citra Untuk Klasifikasi Jenis Apel Menggunakan Scikit-Learn Dengan Algoritma K-Nearest Neighbor. *Jurnal Teknologi Informasi dan Ilmu Komputer*, *12*(1), 165–174. https://doi.org/10.25126/jtiik.2025129149

Yana, Y. E., & Nafi'iyah, N. (2021). Classification of Banana Types Based on Color, Texture, Image Shape Features Using SVM and KNN. Dalam *Research : Journal of Computer* (Vol. 4, Nomor 1).

Zahra, N., Muthiadin, C., & Ferial, F. (2023). Budidaya tanaman selada (Lactuca sativa L.) secara hidroponik dengan sistem DFT di BBPP Batangkaluku. *Filogeni: Jurnal Mahasiswa Biologi*, *3*(1), 18–22. https://doi.org/10.24252/filogeni.v3i1.29922

Zikra, F., Usman, K., & Patmasari, R. (2021). *Deteksi Penyakit Cabai Berdasarkan Citra Daun Menggunakan Metode Gray Level Co-Occurence Matrix Dan Support Vector Machine*.

Lampiran 1. Surat Permohonan Data Penelitian dari Program Studi

MAJELIS PENDIDIKAN TINGGI PIMPINAN PUSAT MUHAMMADIYAH
UNIVERSITAS MUHAMMADIYAH MAKASSAR
FAKULTAS TEKNIK
PROGRAM STUDI INFORMATIKA

بسم الله الرحمن الرحيم

Nomor : 049/05/FT/C.4-VI/VII/47/2025
Lamp. : -
Hal : **Permohonan Data Penelitian**

Makassar, 13 Muharram 1447 H
8 Juli 2025 M

Kepada yang Terhormat,
Ketua LP3M Unismuh Makassar
Di –
Tempat

Assalamu 'Alaikum Warahmatullahi Wabarakatuh

Dengan Rahmat Allah SWT, Semoga aktivitas kita bernilai ibadah di Sisi–Nya. Dalam rangka penyelesaian Tugas Akhir pada Program Studi Informatika dengan judul "Identifikasi Jenis Penyakit pada Citra Daun Selada Menggunakan *Support Vector Machine* (SVM) Berbasis Ekstraksi Fitur Visual". Bersama ini kami sampaikan mahasiswa:

Stambuk	Nama
105 84 11062 21	Muh. Ulil Amri

Sehubungan dengan hal tersebut, maka kami memohon dibuatkan surat pengantar pada instansi di bawah ini.
Nama Instansi : Deedad Hidroponik
Alamat : Jl. Dg. Ngadde No. 26, Paran Tambung, Kec. Tamalate, Kota Makassar

Demikian surat kami atas perhatian dan kerja samanya kami haturkan banyak terima kasih.
Jazakumullah Khaeran Katsiran
Wassalamu 'Alaikum warahmatullah Wabarakatuh

Ketua Program Studi
Informatika,

Muhyiddin A.M Hayat, S.Kom., M.T.
NBM. 1504 577

Tembusan:
1. Dekan Fakultas Teknik
2. Arsip

Lampiran 2. Surat Permohonan Izin Pelaksanaan Penelitian dari LP3M

UNIVERSITAS MUHAMMADIYAH MAKASSAR
LEMBAGA PENELITIAN PENGEMBANGAN DAN PENGABDIAN KEPADA MASYARAKAT
Jl. Sultan Alauddin No. 259 Telp. 866972 Fax. (0411) 865588 Makassar 90221 *e-mail:* lp3m@unismuh.ac.id

بسم الله الرحمن الرحيم

Nomor : 145/LP3M/05/C.4-VIII/VII/1447/2025
Lampiran : 1 (satu) rangkap proposal
Hal : Permohonan Izin Pelaksanaan Penelitian

Kepada Yth:
Bapak Gubernur Provinsi Sulawesi Selatan
Cq. Kepala Dinas Penanaman Modal & PTSP Provinsi Sulawesi Selatan
di-
Makassar

Assalamu Alaikum Wr. Wb

Berdasarkan surat:Dekan Fakultas Program Studi Informatika, nomor: 049 tanggal: 17 Juli 2025, menerangkan bahwa mahasiswa dengan data sebagai berikut.

Nama	: MUH ULIL AMRI
Nim	: 105841106221
Fakultas	: Teknik
Prodi	: Informatika

Bermaksud melaksanakan penelitian/pengumpulan data dalam rangka penulisan laporan tugas akhir Skripsi dengan judul: :

"IDENTIFIKASI JENIS PENYAKIT PADA CITRA DAUN SELADA MENGGUNAKAN SUPPORT VECTOR MACHINE (SVM) BERBASIS EKSTRAKSI FITUR VISUAL"

Yang akan dilaksanakan dari tanggal 24 Juli 2025 s/d 24 September 2025.
Sehubungan dengan maksud di atas, kiranya Mahasiswa tersebut diberikan izin untuk melakukan penelitian sesuai ketentuan yang berlaku.

Demikian, atas perhatian dan kerjasamanya diucapkan jazakumullahu khaeran katziraa.
Billahi Fii Sabilil Haq, Fastabiqul Khaerat.
Wassalamu Alaikum Wr. Wb.

Makassar 22 Muharram 1447
18 Juli 2025

Ketua LP3M Unismuh Makassar,

Dr. Muh. Arief Muhsin, M.Pd.
NBM. 112 7761

Lampiran 3. Surat Izin Penelitian Dari Dinas Penanaman Modal dari PTSP Provinsi Sulawesi Selatan

PEMERINTAH PROVINSI SULAWESI SELATAN

DINAS PENANAMAN MODAL DAN PELAYANAN TERPADU SATU PINTU

Jl.Bougenville No.5 Telp. (0411) 441077 Fax. (0411) 448936
Website : http://simap-new.sulselprov.go.id Email : ptsp@sulselprov.go.id
Makassar 90231

Nomor : 16374/S.01/PTSP/2025
Lampiran : -
Perihal : Izin penelitian

Kepada Yth.
Kepala Dinas Ketahanan Pangan, Hortikultura dan Perkebunan Prov. Sulsel

di-
Tempat

Berdasarkan surat Ketua LP3M Unismuh Makassar Nomor : 145/LP3M/05/C.4-VIII/VII/1447/2025 tanggal 18 Juli 2025 perihal tersebut diatas, mahasiswa/peneliti dibawah ini:

N a m a	: MUH. ULIL AMRI
Nomor Pokok	: 105841106221
Program Studi	: Teknik Informatika
Pekerjaan/Lembaga	: Mahasiswa (S1)
Alamat	: Jl. Slt Alauddin No. 259 Makassar

Bermaksud untuk melakukan penelitian di daerah/kantor saudara dalam rangka menyusun SKRIPSI, dengan judul :

" IDENTIFIKASI JENIS PENYAKIT PADA CITRA DAUN SELADA MENGGUNAKAN SUPPORT VECTOR MACHINE (SVM) BERBASIS EKSTRAKSI FITUR VISUAL "

Yang akan dilaksanakan dari : Tgl. ***24 Juli s/d 24 Agustus 2025***

Sehubungan dengan hal tersebut diatas, pada prinsipnya kami ***menyetujui*** kegiatan dimaksud dengan ketentuan yang tertera di belakang surat izin penelitian.

Demikian Surat Keterangan ini diberikan agar dipergunakan sebagaimana mestinya.

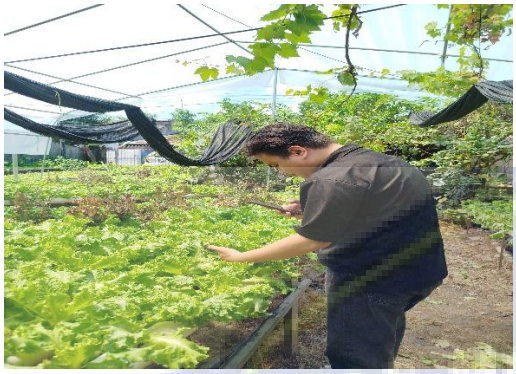
Diterbitkan di Makassar
Pada Tanggal 24 Juli 2025

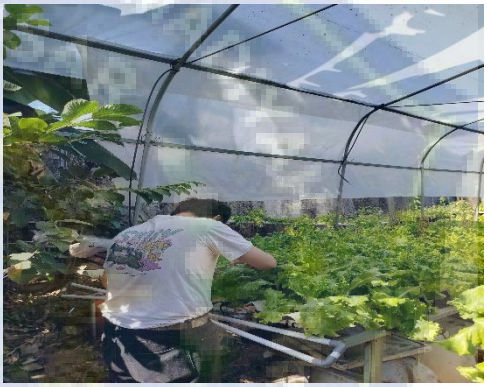
KEPALA DINAS PENANAMAN MODAL DAN PELAYANAN TERPADU SATU PINTU PROVINSI SULAWESI SELATAN

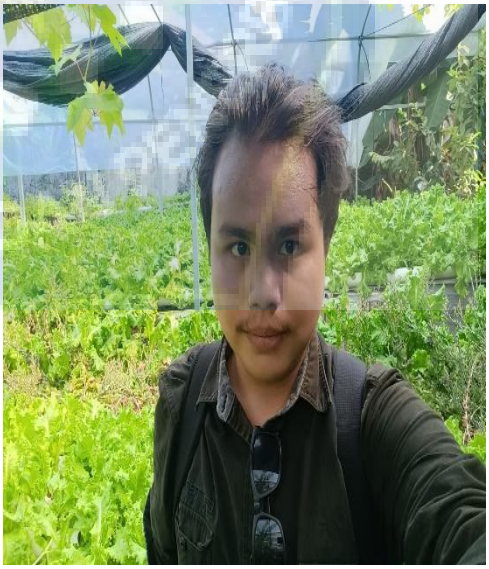
ASRUL SANI, S.H., M.Si.
Pangkat : PEMBINA UTAMA MUDA (IV/c)
Nip : 19750321 200312 1 008

Tembusan Yth
1. Ketua LP3M Unismuh Makassar di Makassar;
2. *Pertinggal.*

Lampiran 4. Dokumentasi Penelitian di DEEDAD Hidroponik Jl. Dg Ngadde no 26 Parang Tambung, Kota Makassar

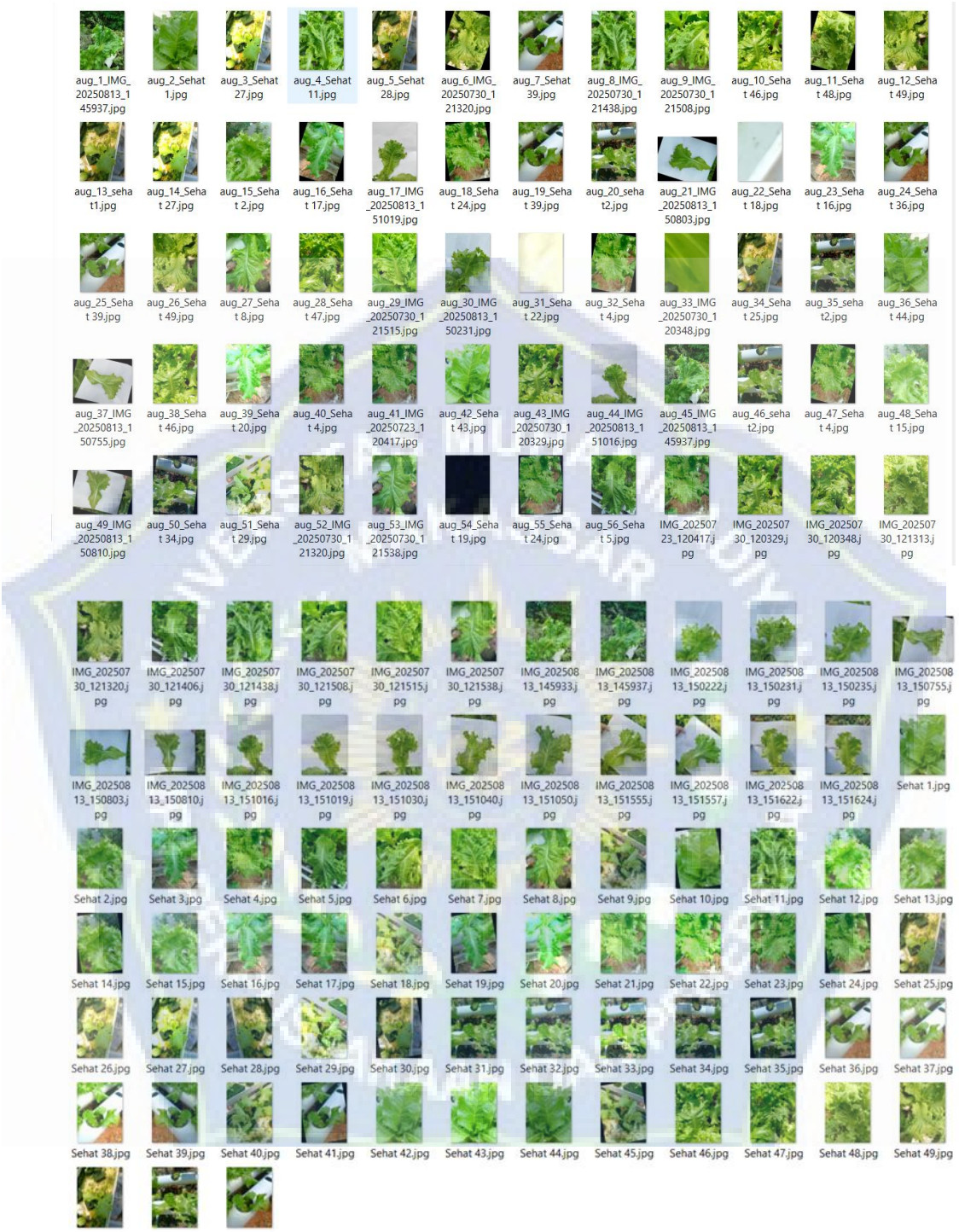


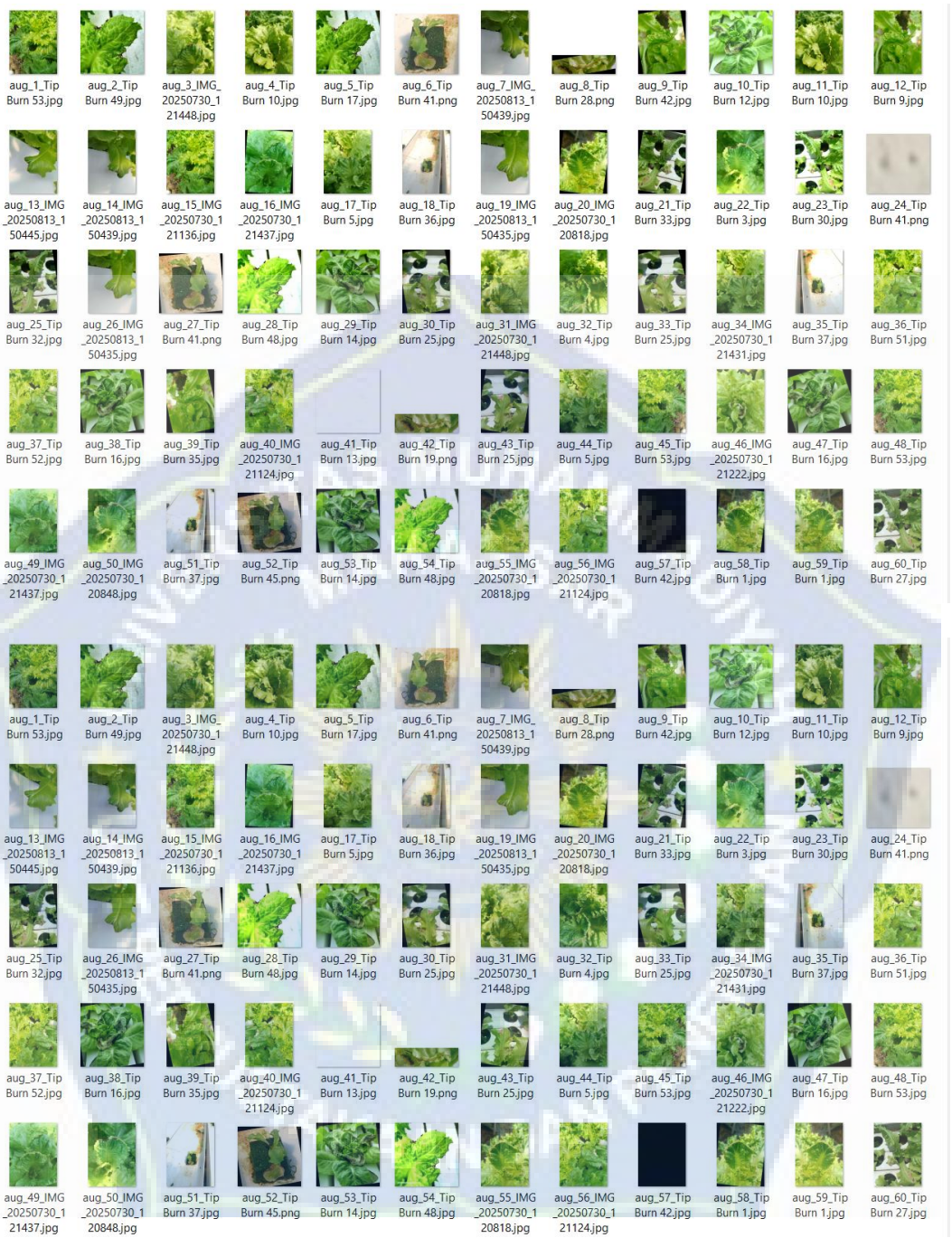



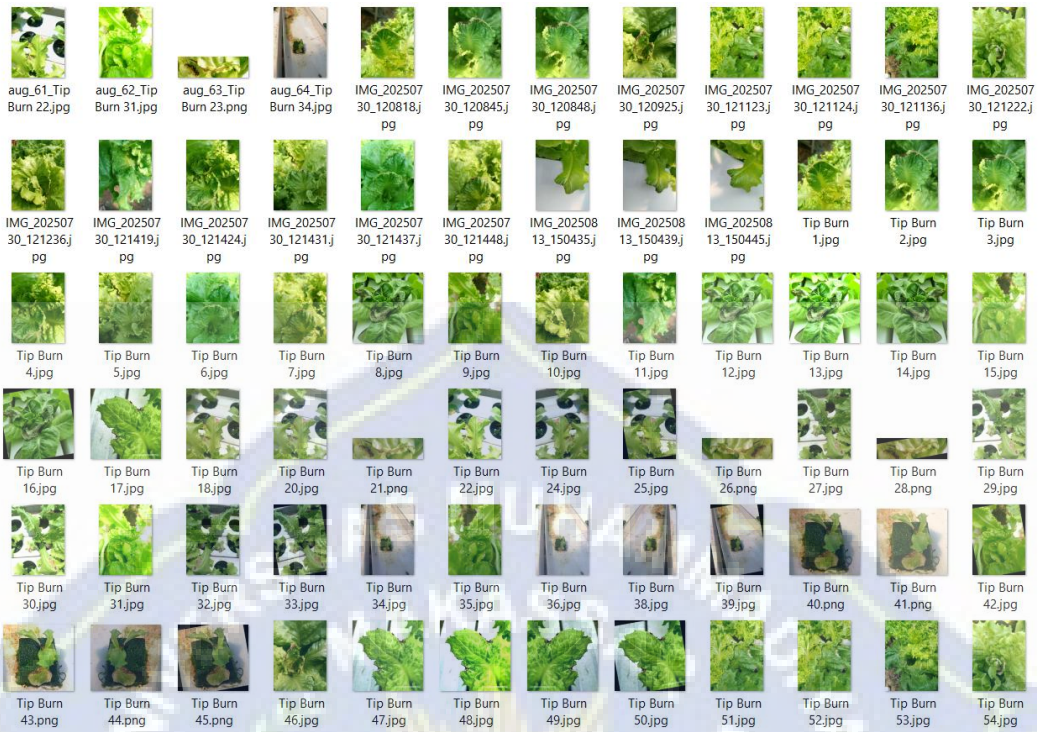


Lampiran 5. Dataset


aug_1_Etiolasi 3.jpg
aug_2_Etiolasi 40.jpg
aug_3_Etiolasi 33.jpg
aug_4_Etiolasi 14.jpg
aug_5_Etiolasi 39.jpg
aug_6_Etiolasi 42.jpg
aug_7_Etiolasi 35.jpg
aug_8_Etiolasi 14.jpg
aug_9_IMG_20250730_122059.jpg
aug_10_Etiolasi 30.jpg
aug_11_Etiolasi 31.jpg
aug_12_Etiolasi 24.jpg
aug_13_Etiolasi 24.jpg
aug_14_Etiolasi 43.jpg
aug_15_Etiolasi 44.jpg
aug_16_Etiolasi 24.jpg
aug_17_Etiolasi 11.jpg
aug_18_Etiolasi 35.jpg
aug_19_Etiolasi 7.jpg
aug_20_Etiolasi 24.jpg
aug_21_Etiolasi 8.jpg
aug_22_IMG_20250730_121815.jpg
aug_23_Etiolasi 31.jpg
aug_24_IMG_20250730_120619.jpg
aug_25_Etiolasi 39.jpg
aug_26_Etiolasi 31.jpg
aug_27_Etiolasi 12.jpg
aug_28_Etiolasi 21.jpg
aug_29_Etiolasi 10.jpg
aug_30_Etiolasi 23.jpg
aug_31_Etiolasi 23.jpg
aug_32_Etiolasi 32.jpg
aug_33_Etiolasi 9.jpg
aug_34_Etiolasi 33.jpg
aug_35_Etiolasi 10.jpg
aug_36_Etiolasi 38.jpg
aug_37_Etiolasi 41.jpg
aug_38_Etiolasi 31.jpg
aug_39_Etiolasi 26.jpg
aug_40_Etiolasi 42.jpg
aug_41_Etiolasi 11.jpg
aug_42_Etiolasi 36.jpg
aug_43_Etiolasi 38.jpg
aug_44_Etiolasi 1.jpg
aug_45_Etiolasi 24.jpg
aug_46_Etiolasi 5.jpg
aug_47_Etiolasi 39.jpg
aug_48_Etiolasi 34.jpg
aug_49_Etiolasi 37.jpg
aug_50_IMG_20250730_122049.jpg
aug_51_Etiolasi 42.jpg
aug_52_Etiolasi 17.jpg
aug_53_Etiolasi 3.jpg
aug_54_IMG_20250730_121109.jpg
aug_55_IMG_20250730_121109.jpg
aug_56_Etiolasi 5.jpg
aug_57_Etiolasi 23.jpg
aug_58_IMG_20250730_121945.jpg
aug_59_Etiolasi 43.jpg
aug_60_Etiolasi 48.jpg
aug_61_Etiolasi 31.jpg
aug_62_Etiolasi 22.jpg
aug_63_Etiolasi 12.jpg
aug_64_IMG_20250730_122049.jpg
aug_65_Etiolasi 33.jpg
aug_66_Etiolasi 33.jpg
aug_67_Etiolasi 2.jpg
aug_68_IMG_20250730_121945.jpg
aug_69_Etiolasi 33.jpg
aug_70_Etiolasi 38.jpg
aug_71_Etiolasi 16.jpg
aug_72_Etiolasi 40.jpg
aug_73_IMG_20250730_121925.jpg
aug_74_Etiolasi 13.jpg
aug_75_Etiolasi 9.jpg
aug_76_Etiolasi 13.jpg
aug_77_IMG_20250730_122049.jpg
aug_78_Etiolasi 34.jpg
aug_79_Etiolasi 28.jpg
Etiolasi 1.jpg
Etiolasi 2.jpg
Etiolasi 3.jpg
Etiolasi 4.jpg
Etiolasi 5.jpg
Etiolasi 6.jpg
Etiolasi 7.jpg
Etiolasi 8.jpg
Etiolasi 9.jpg
Etiolasi 10.jpg
Etiolasi 11.jpg
Etiolasi 12.jpg
Etiolasi 13.jpg
Etiolasi 14.jpg
Etiolasi 15.jpg
Etiolasi 16.jpg
Etiolasi 17.jpg
Etiolasi 18.jpg
Etiolasi 19.jpg
Etiolasi 20.jpg
Etiolasi 21.jpg
Etiolasi 22.jpg
Etiolasi 23.jpg
Etiolasi 24.jpg
Etiolasi 25.jpg
Etiolasi 26.jpg
Etiolasi 27.jpg
Etiolasi 28.jpg
Etiolasi 29.jpg
Etiolasi 30.jpg
Etiolasi 31.jpg
Etiolasi 32.jpg
Etiolasi 33.jpg
Etiolasi 34.jpg
Etiolasi 35.jpg
Etiolasi 36.jpg
Etiolasi 37.jpg
Etiolasi 38.jpg
Etiolasi 39.jpg
Etiolasi 40.jpg
Etiolasi 41.jpg
Etiolasi 42.jpg
Etiolasi 43.jpg
Etiolasi 44.jpg
Etiolasi 45.jpg
Etiolasi 46.jpg
Etiolasi 47.jpg
Etiolasi 48.jpg
IMG_20250730_120619.jpg
IMG_20250730_121106.jpg
IMG_20250730_121109.jpg
IMG_20250730_121815.jpg
IMG_20250730_121925.jpg
IMG_20250730_121945.jpg
IMG_20250730_122049.jpg
IMG_20250730_122059.jpg


aug_1_IMG_20250813_145937.jpg
aug_2_Sehat 1.jpg
aug_3_Sehat 27.jpg
aug_4_Sehat 11.jpg
aug_5_Sehat 28.jpg
aug_6_IMG_20250730_121320.jpg
aug_7_Sehat 39.jpg
aug_8_IMG_20250730_121438.jpg
aug_9_IMG_20250730_121508.jpg
aug_10_Sehat 46.jpg
aug_11_Sehat 48.jpg
aug_12_Sehat 49.jpg
aug_13_sehat1.jpg
aug_14_Sehat 27.jpg
aug_15_Sehat 2.jpg
aug_16_Sehat 17.jpg
aug_17_IMG_20250813_151019.jpg
aug_18_Sehat 24.jpg
aug_19_Sehat 39.jpg
aug_20_sehat2.jpg
aug_21_IMG_20250813_150803.jpg
aug_22_Sehat 18.jpg
aug_23_Sehat 16.jpg
aug_24_Sehat 36.jpg
aug_25_Sehat 39.jpg
aug_26_Sehat 49.jpg
aug_27_Sehat 8.jpg
aug_28_Sehat 47.jpg
aug_29_IMG_20250730_121515.jpg
aug_30_IMG_20250813_150231.jpg
aug_31_Sehat 22.jpg
aug_32_Sehat 4.jpg
aug_33_IMG_20250730_120348.jpg
aug_34_Sehat 25.jpg
aug_35_sehat2.jpg
aug_36_Sehat 44.jpg
aug_37_IMG_20250813_150755.jpg
aug_38_Sehat 46.jpg
aug_39_Sehat 20.jpg
aug_40_Sehat 4.jpg
aug_41_IMG_20250723_120417.jpg
aug_42_Sehat 43.jpg
aug_43_IMG_20250730_120329.jpg
aug_44_IMG_20250813_151016.jpg
aug_45_IMG_20250813_145937.jpg
aug_46_sehat2.jpg
aug_47_Sehat 4.jpg
aug_48_Sehat 15.jpg
aug_49_IMG_20250813_150810.jpg
aug_50_Sehat 34.jpg
aug_51_Sehat 29.jpg
aug_52_IMG_20250730_121320.jpg
aug_53_IMG_20250730_121538.jpg
aug_54_Sehat 19.jpg
aug_55_Sehat 24.jpg
aug_56_Sehat 5.jpg
IMG_20250723_120417.jpg
IMG_20250730_120329.jpg
IMG_20250730_120348.jpg
IMG_20250730_121313.jpg
IMG_20250730_121320.jpg
IMG_20250730_121406.jpg
IMG_20250730_121438.jpg
IMG_20250730_121508.jpg
IMG_20250730_121515.jpg
IMG_20250730_121538.jpg
IMG_20250813_145933.jpg
IMG_20250813_145937.jpg
IMG_20250813_150222.jpg
IMG_20250813_150231.jpg
IMG_20250813_150235.jpg
IMG_20250813_150755.jpg
IMG_20250813_150803.jpg
IMG_20250813_150810.jpg
IMG_20250813_151016.jpg
IMG_20250813_151019.jpg
IMG_20250813_151030.jpg
IMG_20250813_151040.jpg
IMG_20250813_151050.jpg
IMG_20250813_151555.jpg
IMG_20250813_151557.jpg
IMG_20250813_151622.jpg
IMG_20250813_151624.jpg
Sehat 1.jpg
Sehat 2.jpg
Sehat 3.jpg
Sehat 4.jpg
Sehat 5.jpg
Sehat 6.jpg
Sehat 7.jpg
Sehat 8.jpg
Sehat 9.jpg
Sehat 10.jpg
Sehat 11.jpg
Sehat 12.jpg
Sehat 13.jpg
Sehat 14.jpg
Sehat 15.jpg
Sehat 16.jpg
Sehat 17.jpg
Sehat 18.jpg
Sehat 19.jpg
Sehat 20.jpg
Sehat 21.jpg
Sehat 22.jpg
Sehat 23.jpg
Sehat 24.jpg
Sehat 25.jpg
Sehat 26.jpg
Sehat 27.jpg
Sehat 28.jpg
Sehat 29.jpg
Sehat 30.jpg
Sehat 31.jpg
Sehat 32.jpg
Sehat 33.jpg
Sehat 34.jpg
Sehat 35.jpg
Sehat 36.jpg
Sehat 37.jpg
Sehat 38.jpg
Sehat 39.jpg
Sehat 40.jpg
Sehat 41.jpg
Sehat 42.jpg
Sehat 43.jpg
Sehat 44.jpg
Sehat 45.jpg
Sehat 46.jpg
Sehat 47.jpg
Sehat 48.jpg
Sehat 49.jpg


aug_1_Tip Burn 53.jpg
aug_2_Tip Burn 49.jpg
aug_3_IMG_20250730_121448.jpg
aug_4_Tip Burn 10.jpg
aug_5_Tip Burn 17.jpg
aug_6_Tip Burn 41.png
aug_7_IMG_20250813_150439.jpg
aug_8_Tip Burn 28.png
aug_9_Tip Burn 42.jpg
aug_10_Tip Burn 12.jpg
aug_11_Tip Burn 10.jpg
aug_12_Tip Burn 9.jpg
aug_13_IMG_20250813_150445.jpg
aug_14_IMG_20250813_150439.jpg
aug_15_IMG_20250730_121136.jpg
aug_16_IMG_20250730_121437.jpg
aug_17_Tip Burn 5.jpg
aug_18_Tip Burn 36.jpg
aug_19_IMG_20250813_150435.jpg
aug_20_IMG_20250730_120818.jpg
aug_21_Tip Burn 33.jpg
aug_22_Tip Burn 3.jpg
aug_23_Tip Burn 30.jpg
aug_24_Tip Burn 41.png
aug_25_Tip Burn 32.jpg
aug_26_IMG_20250813_150435.jpg
aug_27_Tip Burn 41.png
aug_28_Tip Burn 48.jpg
aug_29_Tip Burn 14.jpg
aug_30_Tip Burn 25.jpg
aug_31_IMG_20250730_121448.jpg
aug_32_Tip Burn 4.jpg
aug_33_Tip Burn 25.jpg
aug_34_IMG_20250730_121431.jpg
aug_35_Tip Burn 37.jpg
aug_36_Tip Burn 51.jpg
aug_37_Tip Burn 52.jpg
aug_38_Tip Burn 16.jpg
aug_39_Tip Burn 35.jpg
aug_40_IMG_20250730_121124.jpg
aug_41_Tip Burn 13.jpg
aug_42_Tip Burn 19.png
aug_43_Tip Burn 25.jpg
aug_44_Tip Burn 5.jpg
aug_45_Tip Burn 53.jpg
aug_46_IMG_20250730_121222.jpg
aug_47_Tip Burn 16.jpg
aug_48_Tip Burn 53.jpg
aug_49_IMG_20250730_121437.jpg
aug_50_IMG_20250730_120848.jpg
aug_51_Tip Burn 37.jpg
aug_52_Tip Burn 45.png
aug_53_Tip Burn 14.jpg
aug_54_Tip Burn 48.jpg
aug_55_IMG_20250730_120818.jpg
aug_56_IMG_20250730_121124.jpg
aug_57_Tip Burn 42.jpg
aug_58_Tip Burn 1.jpg
aug_59_Tip Burn 1.jpg
aug_60_Tip Burn 27.jpg
aug_1_Tip Burn 53.jpg
aug_2_Tip Burn 49.jpg
aug_3_IMG_20250730_121448.jpg
aug_4_Tip Burn 10.jpg
aug_5_Tip Burn 17.jpg
aug_6_Tip Burn 41.png
aug_7_IMG_20250813_150439.jpg
aug_8_Tip Burn 28.png
aug_9_Tip Burn 42.jpg
aug_10_Tip Burn 12.jpg
aug_11_Tip Burn 10.jpg
aug_12_Tip Burn 9.jpg
aug_13_IMG_20250813_150445.jpg
aug_14_IMG_20250813_150439.jpg
aug_15_IMG_20250730_121136.jpg
aug_16_IMG_20250730_121437.jpg
aug_17_Tip Burn 5.jpg
aug_18_Tip Burn 36.jpg
aug_19_IMG_20250813_150435.jpg
aug_20_IMG_20250730_120818.jpg
aug_21_Tip Burn 33.jpg
aug_22_Tip Burn 3.jpg
aug_23_Tip Burn 30.jpg
aug_24_Tip Burn 41.png
aug_25_Tip Burn 32.jpg
aug_26_IMG_20250813_150435.jpg
aug_27_Tip Burn 41.png
aug_28_Tip Burn 48.jpg
aug_29_Tip Burn 14.jpg
aug_30_Tip Burn 25.jpg
aug_31_IMG_20250730_121448.jpg
aug_32_Tip Burn 4.jpg
aug_33_Tip Burn 25.jpg
aug_34_IMG_20250730_121431.jpg
aug_35_Tip Burn 37.jpg
aug_36_Tip Burn 51.jpg
aug_37_Tip Burn 52.jpg
aug_38_Tip Burn 16.jpg
aug_39_Tip Burn 35.jpg
aug_40_IMG_20250730_121124.jpg
aug_41_Tip Burn 13.jpg
aug_42_Tip Burn 19.png
aug_43_Tip Burn 25.jpg
aug_44_Tip Burn 5.jpg
aug_45_Tip Burn 53.jpg
aug_46_IMG_20250730_121222.jpg
aug_47_Tip Burn 16.jpg
aug_48_Tip Burn 53.jpg
aug_49_IMG_20250730_121437.jpg
aug_50_IMG_20250730_120848.jpg
aug_51_Tip Burn 37.jpg
aug_52_Tip Burn 45.png
aug_53_Tip Burn 14.jpg
aug_54_Tip Burn 48.jpg
aug_55_IMG_20250730_120818.jpg
aug_56_IMG_20250730_121124.jpg
aug_57_Tip Burn 42.jpg
aug_58_Tip Burn 1.jpg
aug_59_Tip Burn 1.jpg
aug_60_Tip Burn 27.jpg



Lampiran 6. Source Code

1. Augment dataset

Memperbanyak jumlah gambar dalam dataset secara fisik dengan membuat beberapa variasi dari setiap gambar asli. Skrip ini berguna untuk pengujian awal augmentasi

```
# augment_dataset.py
import os
import cv2
import numpy as np

def augment_and_save(image_path, output_dir):
    """
    Membaca sebuah gambar, membuat beberapa versi augmentasi,
    dan menyimpannya ke direktori output.
    """
    # Baca gambar asli
    image = cv2.imread(image_path)
    if image is None:
        print(f"  - Gagal membaca {os.path.basename(image_path)},
dilewati.")
        return 0

    # Dapatkan nama file dasar tanpa ekstensi
    filename, extension =
os.path.splitext(os.path.basename(image_path))

    created_count = 0

    # --- Daftar Augmentasi ---

    # 1. Horizontal Flip (Cermin)
    flipped = cv2.flip(image, 1)
    new_filename_flip = os.path.join(output_dir,
f"{filename}_aug_flip{extension}")
```

```
        cv2.imwrite(new_filename_flip, flipped)
        created_count += 1

        # 2. Brightness (Lebih Cerah)
        bright = cv2.convertScaleAbs(image, alpha=1.0, beta=40)
        new_filename_bright = os.path.join(output_dir,
f"{filename}_aug_bright{extension}")
        cv2.imwrite(new_filename_bright, bright)
        created_count += 1

        # 3. Contrast (Kontras Lebih Tinggi)
        contrast = cv2.convertScaleAbs(image, alpha=1.5, beta=0)
        new_filename_contrast = os.path.join(output_dir,
f"{filename}_aug_contrast{extension}")
        cv2.imwrite(new_filename_contrast, contrast)
        created_count += 1

        # 4. Rotation (Rotasi Sedikit)
        (h, w) = image.shape[:2]
        center = (w // 2, h // 2)
        M = cv2.getRotationMatrix2D(center, angle=10, scale=1.0)
        rotated = cv2.warpAffine(image, M, (w, h))
        new_filename_rotated = os.path.join(output_dir,
f"{filename}_aug_rotated{extension}")
        cv2.imwrite(new_filename_rotated, rotated)
        created_count += 1

        return created_count

def main():
    """
    Fungsi utama untuk menjalankan proses augmentasi di seluruh
dataset.
    """
    dataset_dir = 'dataset'
```

```
    total_new_images = 0

    print("="*50)
    print("Memulai Proses Augmentasi Fisik Dataset")
    print(f"Direktori Target: {dataset_dir}")
    print("="*50)

    if not os.path.isdir(dataset_dir):
        print(f"Error: Direktori '{dataset_dir}' tidak ditemukan.")
        return

    # Iterasi melalui setiap subdirektori (kelas) di dalam folder
dataset
    for class_name in os.listdir(dataset_dir):
        class_dir = os.path.join(dataset_dir, class_name)
        if os.path.isdir(class_dir):
            print(f"\nMemproses kelas: {class_name}")

            # Dapatkan daftar gambar asli sebelum augmentasi
            original_images = [f for f in os.listdir(class_dir) if
f.lower().endswith(('.png', '.jpg', '.jpeg')) and '_aug_' not in f]

            if not original_images:
                print("  - Tidak ada gambar asli untuk di-
augmentasi.")
                continue

            for image_name in original_images:
                image_path = os.path.join(class_dir, image_name)
                print(f"  - Meng-augmentasi: {image_name}")
                count = augment_and_save(image_path, class_dir)
                total_new_images += count

    print("\n" + "="*50)
    print("Proses Augmentasi Selesai.")
```

```
    print(f"Total gambar baru yang berhasil dibuat:
{total_new_images}")
    print("="*50)

if __name__ == '__main__':
    main()
```

2. balance_dataset

   Menyeimbangkan jumlah gambar di setiap kelas dengan cara melakukan *oversampling* (augmentasi) pada kelas-kelas yang jumlah gambarnya lebih sedikit (kelas minoritas

```
import os
import cv2
import numpy as np
import random
from tqdm import tqdm

def apply_random_augmentation(image):
    """Menerapkan satu jenis augmentasi acak pada sebuah gambar."""
    augmentation_type = random.choice(['rotate', 'flip',
'brightness', 'zoom', 'blur'])

    if augmentation_type == 'rotate':
        angle = random.uniform(-20, 20)
        h, w = image.shape[:2]
        M = cv2.getRotationMatrix2D((w / 2, h / 2), angle, 1)
        return cv2.warpAffine(image, M, (w, h))

    if augmentation_type == 'flip':
        return cv2.flip(image, 1) # Flip horizontal

    if augmentation_type == 'brightness':
        value = random.randint(-20, 20)
```

```
            hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
            h, s, v = cv2.split(hsv)
            v = cv2.add(v, value)
            v[v > 255] = 255
            v[v < 0] = 0
            final_hsv = cv2.merge((h, s, v))
            return cv2.cvtColor(final_hsv, cv2.COLOR_HSV2BGR)

        if augmentation_type == 'zoom':
            zoom_factor = random.uniform(0.85, 1.15)
            h, w = image.shape[:2]
            # Crop the center
            ch, cw = h // 2, w // 2
            crop_h, crop_w = int(h / zoom_factor), int(w / zoom_factor)

            y1 = ch - crop_h // 2
            y2 = y1 + crop_h
            x1 = cw - crop_w // 2
            x2 = x1 + crop_w

            cropped = image[y1:y2, x1:x2]
            return cv2.resize(cropped, (w, h))

        if augmentation_type == 'blur':
            return cv2.GaussianBlur(image, (5, 5), random.uniform(0,
1.5))

        return image

    def balance_dataset(dataset_dir):
        """
        Menyeimbangkan jumlah gambar per kelas dengan melakukan
oversampling (augmentasi)
        pada kelas-kelas minoritas.
        """
```

```
    print(f"Memulai proses penyeimbangan dataset di: {dataset_dir}")

    class_paths = [d for d in os.listdir(dataset_dir) if
os.path.isdir(os.path.join(dataset_dir, d))]
    if not class_paths:
        print("Error: Tidak ada folder kelas yang ditemukan.")
        return

    # Hitung jumlah gambar di setiap kelas
    class_counts = {}
    image_files = {}
    for class_name in class_paths:
        class_dir = os.path.join(dataset_dir, class_name)
        # Hanya hitung file gambar asli, bukan yang sudah di-
augmentasi
        files = [f for f in os.listdir(class_dir) if
f.lower().endswith(('.png', '.jpg', '.jpeg')) and not
f.startswith('aug_')]
        image_files[class_name] = files
        class_counts[class_name] = len(files)

    if not class_counts:
        print("Error: Tidak ada gambar yang ditemukan di folder
kelas.")
        return

    # Tentukan jumlah target (jumlah gambar di kelas mayoritas)
    target_count = max(class_counts.values())
    print(f"Distribusi kelas saat ini: {class_counts}")
    print(f"Target jumlah gambar per kelas (oversampling):
{target_count}\n")

    # Lakukan augmentasi pada kelas minoritas
    for class_name, current_count in class_counts.items():
        if current_count == 0:
```

```
            print(f"Peringatan: Kelas '{class_name}' tidak memiliki
gambar asli, dilewati.")
            continue

        if current_count < target_count:
            num_to_generate = target_count - current_count
            print(f"Kelas '{class_name}': Perlu {num_to_generate}
gambar baru. Memulai augmentasi...")

            original_images_in_class = image_files[class_name]
            class_dir = os.path.join(dataset_dir, class_name)

            for i in tqdm(range(num_to_generate), desc=f"Augmenting
{class_name}"):
                # Pilih gambar acak dari file asli untuk di-
augmentasi
                random_image_name =
random.choice(original_images_in_class)
                image_path = os.path.join(class_dir,
random_image_name)
                image = cv2.imread(image_path)

                if image is not None:
                    augmented_image =
apply_random_augmentation(image)

                    # Buat nama file baru yang unik
                    new_filename = f"aug_{i+1}_{random_image_name}"
                    new_filepath = os.path.join(class_dir,
new_filename)
                    cv2.imwrite(new_filepath, augmented_image)
                else:
                    print(f"Warning: Gagal membaca {image_path} saat
augmentasi.")
```

```
    print("\nProses penyeimbangan dataset selesai.")
    # Verifikasi hasil akhir
    print("Verifikasi jumlah file setelah augmentasi:")
    for class_name in class_paths:
        class_dir = os.path.join(dataset_dir, class_name)
        final_count = len([f for f in os.listdir(class_dir) if
f.lower().endswith(('.png', '.jpg', '.jpeg'))])
        print(f"  - Kelas '{class_name}': {final_count} gambar")


if __name__ == '__main__':
    # Path ini harus sesuai dengan lokasi dataset Anda
    DATASET_DIRECTORY = 'C:/SKRIPSI FIKS SEMPRO  DAN SELESAI TAHUN
IN/BACKUP PROGRAM/PROGRAM/PELATIHAN/dataset'
    balance_dataset(DATASET_DIRECTORY)
```

3. create_segmented_dataset

Melakukan segmentasi gambar untuk memisahkan objek utama (daun) dari latar belakangnya. Tujuannya adalah agar model lebih fokus pada fitur daun itu sendiri

```
import cv2
import numpy as np
import os
import shutil

def segment_image(image_path, output_path):
    """Melakukan segmentasi pada satu gambar menggunakan GrabCut dan
menyimpannya."""
```

```
    try:
        img = cv2.imread(image_path)
        if img is None:
            print(f"  - Gagal membaca
{os.path.basename(image_path)}, dilewati.")
            return False

        # Buat mask awal untuk GrabCut
        mask = np.zeros(img.shape[:2], np.uint8)

        # Tentukan rectangle di tengah gambar sebagai area
kemungkinan foreground
        # Kita ambil 90% dari lebar dan tinggi gambar sebagai area
ROI
        rect_margin = int(img.shape[0] * 0.05) # 5% margin
        rect = (rect_margin, rect_margin, img.shape[1]
2*rect_margin, img.shape[0] - 2*rect_margin)

        # Alokasi memori untuk model background dan foreground
        bgdModel = np.zeros((1, 65), np.float64)
        fgdModel = np.zeros((1, 65), np.float64)

        # Jalankan GrabCut
        cv2.grabCut(img, mask, rect, bgdModel, fgdModel, 5,
cv2.GC_INIT_WITH_RECT)

        # Buat mask akhir dimana area foreground (daun) adalah 1 dan
background adalah 0
        final_mask = np.where((mask == 2) | (mask == 0), 0,
1).astype('uint8')

        # Terapkan mask ke gambar asli
        img_segmented = img * final_mask[:, :, np.newaxis]

        # Simpan gambar hasil segmentasi
```

```
            cv2.imwrite(output_path, img_segmented)
            return True
        except Exception as e:
            print(f"  - Error saat memproses
{os.path.basename(image_path)}: {e}")
            return False

def main():
    """Fungsi utama untuk membuat dataset yang sudah
tersegmentasi."""
    source_dir = 'C:/SKRIPSI FIKS SEMPRO  DAN SELESAI TAHUN
IN/PROGRAM/PELATIHAN/dataset'
    dest_dir = 'C:/SKRIPSI FIKS SEMPRO  DAN SELESAI TAHUN
IN/PROGRAM/PELATIHAN/dataset_segmented'

    print("="*50)
    print(f"Membuat dataset tersegmentasi di: {dest_dir}")
    print("="*50)

    if os.path.exists(dest_dir):
        print("Menghapus direktori lama...")
        shutil.rmtree(dest_dir)

    os.makedirs(dest_dir)

    total_processed = 0
    total_success = 0

    for class_name in os.listdir(source_dir):
        source_class_dir = os.path.join(source_dir, class_name)
        dest_class_dir = os.path.join(dest_dir, class_name)

        if os.path.isdir(source_class_dir):
            os.makedirs(dest_class_dir, exist_ok=True)
            print(f"\nMemproses kelas: {class_name}")
```

```
            for image_name in os.listdir(source_class_dir):
                if image_name.lower().endswith(('.png', '.jpg',
'.jpeg')):
                    total_processed += 1
                    source_image_path =
os.path.join(source_class_dir, image_name)
                    dest_image_path = os.path.join(dest_class_dir,
image_name)

                    if segment_image(source_image_path,
dest_image_path):
                        total_success += 1

    print("\n" + "="*50)
    print("Proses Segmentasi Selesai.")
    print(f"Total gambar diproses: {total_processed}")
    print(f"Total gambar berhasil disegmentasi: {total_success}")
    print("="*50)

if __name__ == '__main__':
    main()
```

4. find_duplicates & delete_duplicates

Mengidentifikasi dan menghapus file gambar yang duplikat (identik secara isi) dari dalam dataset untuk memastikan tidak ada redundansi data.

```
import os
import hashlib
from collections import defaultdict

def get_file_hash(filepath):
```

```
    """Calculates the SHA256 hash of a file."""
    h = hashlib.sha256()
    try:
        with open(filepath, 'rb') as f:
            while True:
                data = f.read(65536)  # Read in 64k chunks
                if not data:
                    break
                h.update(data)
        return h.hexdigest()
    except (IOError, OSError) as e:
        print(f"Could not read file: {filepath} ({e})")
        return None

def find_duplicate_images(directory):
    """Finds duplicate image files in a directory."""
    hashes = defaultdict(list)
    for dirpath, _, filenames in os.walk(directory):
        for filename in filenames:
            if filename.lower().endswith(('.png', '.jpg', '.jpeg')):
                filepath = os.path.join(dirpath, filename)
                file_hash = get_file_hash(filepath)
                if file_hash:
                    hashes[file_hash].append(filepath)

    duplicates = {k: v for k, v in hashes.items() if len(v) > 1}
    return duplicates

def delete_duplicates(duplicate_files):
    """Deletes duplicate files, keeping one copy from each set."""
    if not duplicate_files:
        print("No duplicates to delete.")
        return

    total_deleted = 0
```

```
    for file_list in duplicate_files.values():
        # Keep the first file, delete the rest
        file_to_keep = file_list[0]
        files_to_delete = file_list[1:]

        print(f"\n--- Processing Set ---")
        print(f"Keeping: {file_to_keep}")

        for filepath in files_to_delete:
            try:
                os.remove(filepath)
                print(f"Deleted: {filepath}")
                total_deleted += 1
            except OSError as e:
                print(f"Error deleting file {filepath}: {e}")

    print(f"\nTotal files deleted: {total_deleted}")


if __name__ == "__main__":
    dataset_dir = r"C:\SKRIPSI ULIL\SKRIPSI ULIL\dataset"
    print("Finding duplicate images...")
    duplicate_files = find_duplicate_images(dataset_dir)

    if not duplicate_files:
        print("No duplicate images found.")
    else:
        print(f"Found {len(duplicate_files)} sets of duplicate
images.")
        delete_duplicates(duplicate_files)
import os
import hashlib
from collections import defaultdict

def get_file_hash(filepath):
```

```
    """Calculates the SHA256 hash of a file."""
    h = hashlib.sha256()
    try:
        with open(filepath, 'rb') as f:
            while True:
                data = f.read(65536)  # Read in 64k chunks
                if not data:
                    break
                h.update(data)
        return h.hexdigest()
    except (IOError, OSError) as e:
        print(f"Could not read file: {filepath} ({e})")
        return None

def find_duplicate_images(directory):
    """Finds duplicate image files in a directory."""
    hashes = defaultdict(list)
    # Use os.walk to recursively find all files
    for dirpath, _, filenames in os.walk(directory):
        for filename in filenames:
            if filename.lower().endswith(('.png', '.jpg', '.jpeg')):
                filepath = os.path.join(dirpath, filename)
                file_hash = get_file_hash(filepath)
                if file_hash:
                    hashes[file_hash].append(filepath)

    duplicates = {k: v for k, v in hashes.items() if len(v) > 1}
    return duplicates

if __name__ == "__main__":
    dataset_dir = r"C:\SKRIPSI ULIL\SKRIPSI ULIL\dataset"
    duplicate_files = find_duplicate_images(dataset_dir)

    if not duplicate_files:
        print("No duplicate images found.")
```

```
    else:
        print("Found the following duplicate images:")
        for file_list in duplicate_files.values():
            print("\n--- Duplicate Set ---")
            for filepath in file_list:
                print(filepath)
```

5. prepare_gatekeeper_dataset

Membuat dataset khusus untuk melatih "Model Penjaga Gerbang" (*Gatekeeper Model*). Model ini bertugas untuk membedakan antara gambar selada dan gambar bukan selada.

```
import os
import shutil
import glob

def prepare_gatekeeper_dataset(base_dir):
    """Creates and populates the dataset for the gatekeeper
model."""
    print("Mempersiapkan dataset untuk Model Penjaga Gerbang...")

    # --- Path Konfigurasi ---
    gatekeeper_dir = os.path.join(base_dir, 'dataset_gatekeeper')
    lettuce_dir = os.path.join(gatekeeper_dir, 'lettuce')
    not_lettuce_dir = os.path.join(gatekeeper_dir, 'not_lettuce')

    original_dataset_dir = os.path.join(base_dir, 'dataset')
    test_data_dir = os.path.join(base_dir, 'DATA UJI')
    root_images = glob.glob(os.path.join(base_dir, '*.jpg')) +
glob.glob(os.path.join(base_dir, '*.webp'))

    # --- 1. Membuat Struktur Folder ---
    os.makedirs(lettuce_dir, exist_ok=True)
    os.makedirs(not_lettuce_dir, exist_ok=True)
    print(f"Folder telah dibuat di: {gatekeeper_dir}")
```

```
    # --- 2. Menyalin Gambar Selada ---
    lettuce_count = 0
    lettuce_sources = glob.glob(os.path.join(original_dataset_dir,
'**', '*.*'), recursive=True)
    for src_path in lettuce_sources:
        if src_path.lower().endswith(('.png', '.jpg', '.jpeg')):
            try:
                shutil.copy(src_path, lettuce_dir)
                lettuce_count += 1
            except Exception as e:
                print(f"Gagal menyalin {src_path}: {e}")
    print(f"- Berhasil menyalin {lettuce_count} gambar ke folder
'lettuce'.")

    # --- 3. Menyalin Gambar Bukan Selada ---
    not_lettuce_count = 0
    # Ambil semua dari DATA UJI dan gambar non-selada di root
    not_lettuce_sources = glob.glob(os.path.join(test_data_dir,
'*.*'))

    # Tambahkan file spesifik seperti merica.jpg dan kol.webp
    for img_file in root_images:
        filename = os.path.basename(img_file).lower()
        if 'merica' in filename or 'kol' in filename:
            not_lettuce_sources.append(img_file)

    for src_path in not_lettuce_sources:
        if src_path.lower().endswith(('.png', '.jpg', '.jpeg',
'.webp')):
            try:
                shutil.copy(src_path, not_lettuce_dir)
                not_lettuce_count += 1
            except Exception as e:
                print(f"Gagal menyalin {src_path}: {e}")
```

```
    print(f"- Berhasil menyalin {not_lettuce_count} gambar ke folder
'not_lettuce'.")

    print("\nPersiapan dataset selesai.")

if __name__ == '__main__':
    # Menentukan direktori utama proyek
    project_base_dir = os.path.dirname(os.path.abspath(__file__))
    prepare_gatekeeper_dataset(project_base_dir)
```

6. Ekstraksi Fitur

Mengekstrak serangkaian fitur kuantitatif yang komprehensif dari sebuah gambar untuk menangkap karakteristik visual utamanya, yaitu warna, tekstur, dan bentuk

```
import cv2
import numpy as np
from skimage.feature import graycomatrix, graycoprops,
local_binary_pattern
import warnings

# Mengabaikan peringatan yang tidak relevan dari skimage
warnings.filterwarnings("ignore", category=UserWarning,
module='skimage')

def extract_features(image_path_or_data, include_shape=True,
lbp_radius=1, lbp_points=8):
    """
    Mengekstrak fitur canggih: Warna (HSV), Tekstur (GLCM + LBP),
dan Bentuk (Hu Moments).
    Input bisa berupa path file (string) atau data gambar (numpy
array).
    """
    try:
        if isinstance(image_path_or_data, str):
```

```
            img = cv2.imread(image_path_or_data)
            if img is None:
                print(f"Peringatan: Tidak dapat membaca gambar di
{image_path_or_data}")
                return None
        else:
            img = image_path_or_data

        # Ukuran standar untuk konsistensi
        img_resized = cv2.resize(img, (256, 256))
        gray_img = cv2.cvtColor(img_resized, cv2.COLOR_BGR2GRAY)

        # --- 1. Fitur Warna (HSV) ---
        hsv_img = cv2.cvtColor(img_resized, cv2.COLOR_BGR2HSV)
        h, s, v = cv2.split(hsv_img)
        color_features = [
            np.mean(h), np.std(h), # Rata-rata & Deviasi Standar Hue
            np.mean(s), np.std(s), # Rata-rata & Deviasi Standar
Saturation
            np.mean(v), np.std(v)  # Rata-rata & Deviasi Standar
Value
        ]

        # --- 2. Fitur Tekstur (GLCM) ---
        # distances dan angles yang lebih beragam untuk menangkap
lebih banyak info
        glcm = graycomatrix(gray_img, distances=[1, 3, 5],
angles=[0, np.pi/4, np.pi/2], levels=256, symmetric=True,
normed=True)
        texture_features_glcm = [
            graycoprops(glcm, 'contrast').mean(),
            graycoprops(glcm, 'correlation').mean(),
            graycoprops(glcm, 'energy').mean(),
            graycoprops(glcm, 'homogeneity').mean(),
            graycoprops(glcm, 'ASM').mean(),
```

```
            graycoprops(glcm, 'dissimilarity').mean()
        ]

        # --- 3. Fitur Tekstur (LBP) - BARU! ---
        # LBP sangat baik untuk detail tekstur halus dan invarian
terhadap pencahayaan
        lbp = local_binary_pattern(gray_img, P=lbp_points,
R=lbp_radius, method='uniform')
        (hist, _) = np.histogram(lbp.ravel(), bins=np.arange(0,
lbp_points + 3), range=(0, lbp_points + 2))
        hist = hist.astype("float")
        hist /= (hist.sum() + 1e-6) # Normalisasi histogram
        texture_features_lbp = hist.tolist()

        # --- 4. Fitur Bentuk (Hu Moments) ---
        shape_features = [0] * 7 # Default jika tidak ada kontur
        if include_shape:
            # Menggunakan Canny edge detection untuk hasil yang
lebih bersih sebelum mencari kontur
            edged = cv2.Canny(gray_img, 30, 150)
            contours, _ = cv2.findContours(edged, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)

            if contours:
                main_contour = max(contours, key=cv2.contourArea)
                moments = cv2.moments(main_contour)
                hu_moments = cv2.HuMoments(moments)
                # Log-transform untuk menstabilkan nilai
                shape_features = -np.sign(hu_moments) *
np.log10(np.abs(hu_moments) + 1e-7)
                shape_features = shape_features.flatten().tolist()

        # --- Menggabungkan Semua Fitur Menjadi Satu Vektor ---
        # Pastikan semua list fitur memiliki panjang yang sama jika
diperlukan (padding)
```

```
        # Di sini kita menggabungkannya langsung
        all_features = np.array(
            color_features +
            texture_features_glcm +
            texture_features_lbp +
            shape_features
        )
        return all_features

    except Exception as e:
        # Cetak error yang lebih spesifik
        import traceback
        print(f"Error saat memproses gambar: {e}")
        traceback.print_exc()
        return None
```

7. SVM tanpa Optimasi

Melatih model SVM dalam kondisi paling dasar (*baseline*).

```
import os
import cv2
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC
from sklearn.metrics import classification_report, accuracy_score
import joblib
import warnings

from feature_extractor import extract_features

warnings.filterwarnings("ignore", category=UserWarning,
module='skimage')
warnings.filterwarnings("ignore", category=FutureWarning)
```

```
def load_dataset_simple(dataset_dir):
    """Loads a dataset without augmentation and extracts
features."""
    features, labels, class_names = [], [], sorted([d for d in
os.listdir(dataset_dir) if os.path.isdir(os.path.join(dataset_dir,
d))])
    if not class_names:
        print(f"Error: No classes found in '{dataset_dir}'")
        return None, None, None

    print(f"Classes found: {class_names}\nExtracting features...")

    for label_name in class_names:
        class_dir = os.path.join(dataset_dir, label_name)
        for image_name in os.listdir(class_dir):
            image_path = os.path.join(class_dir, image_name)
            if image_path.lower().endswith(('.png', '.jpg',
'.jpeg')):
                image = cv2.imread(image_path)
                if image is None:
                    print(f"Warning: Failed to read {image_path},
skipping.")
                    continue

                feats = extract_features(image, include_shape=True)
                if feats is not None:
                    features.append(feats)
                    labels.append(label_name)

    print(f"\nTotal dataset size: {len(features)} samples")
    return np.array(features), np.array(labels), class_names

if __name__ == '__main__':
    # Adjust this path to your dataset location
```

```
    DATASET_PATH = 'C:/SKRIPSI FIKS SEMPRO  DAN SELESAI TAHUN
IN/PROGRAM/PELATIHAN/dataset'
    SAVED_MODEL_DIR = 'C:/SKRIPSI FIKS SEMPRO  DAN SELESAI TAHUN
IN/PROGRAM/SVM-saja/saved_model'

    print(f"--- Starting SVM-Only Model Training ---")
    print(f"Dataset source: {DATASET_PATH}")
    print(f"Output will be saved in: {SAVED_MODEL_DIR}")

    if not os.path.exists(SAVED_MODEL_DIR):
        os.makedirs(SAVED_MODEL_DIR)

    # Load data
    X, y, class_names = load_dataset_simple(DATASET_PATH)

    if X is not None and X.shape[0] > 0:
        # Split data
        X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42, stratify=y)
        print(f"\nTraining data size: {X_train.shape[0]}, Testing
data size: {X_test.shape[0]}")

        # Scale features
        scaler = StandardScaler()
        X_train_scaled = scaler.fit_transform(X_train)
        X_test_scaled = scaler.transform(X_test)

        # Train a standard SVM model
        print("\nTraining the SVM model...")
        model = SVC(kernel='rbf', C=1.0, gamma='scale',
probability=True, random_state=42)
        model.fit(X_train_scaled, y_train)
        print("Model training complete.")

        # Save the model, scaler, and class names
```

```
        joblib.dump(model, os.path.join(SAVED_MODEL_DIR,
'svm_model_only.pkl'))
        joblib.dump(scaler, os.path.join(SAVED_MODEL_DIR,
'scaler_only.pkl'))
        joblib.dump(class_names, os.path.join(SAVED_MODEL_DIR,
'class_names_only.pkl'))
        print("Model, scaler, and class names have been saved.")

        # Evaluate on the test data
        print("\n--- Test Data Evaluation Results ---")
        y_pred = model.predict(X_test_scaled)
        accuracy = accuracy_score(y_test, y_pred)

        print(f"Accuracy: {accuracy:.4f}")
        print("\nClassification Report:\n",
classification_report(y_test, y_pred, target_names=class_names))

    else:
        print("No data to process. Halting.")
```

8. Model dengan Augmentasi Tanpa PSO.

Melatih model SVM menggunakan dataset yang diperbanyak dengan augmentasi, namun tanpa optimasi hyperparameter.

```
import os
import cv2
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC
from sklearn.metrics import classification_report, confusion_matrix,
accuracy_score
import joblib
import warnings
import pandas as pd
```

```
import matplotlib.pyplot as plt
import seaborn as sns
import sys

# Path ke feature_extractor.py di folder PELATIHAN
sys.path.append(os.path.abspath(os.path.join(os.path.dirname(__file_
_), '..', 'PELATIHAN')))
from feature_extractor import extract_features

warnings.filterwarnings("ignore", category=UserWarning,
module='skimage')
warnings.filterwarnings("ignore", category=FutureWarning)

def load_and_augment_dataset(dataset_dir):
    """Memuat dataset, melakukan augmentasi acak, dan mengekstrak
fitur."""
    features, labels, class_names = [], [], sorted([d for d in
os.listdir(dataset_dir) if os.path.isdir(os.path.join(dataset_dir,
d))])
    if not class_names:
        print(f"Error: Tidak ada kelas di '{dataset_dir}'")
        return None, None, None

    print(f"Kelas ditemukan: {class_names}\nMemulai augmentasi acak
dan ekstraksi fitur...")

    for label_name in class_names:
        class_dir = os.path.join(dataset_dir, label_name)
        image_count = 0
        for image_name in os.listdir(class_dir):
            image_path = os.path.join(class_dir, image_name)
            if image_path.lower().endswith(('.png', '.jpg',
'.jpeg')):
                image = cv2.imread(image_path)
                if image is None:
```

```
                    print(f"Warning: Gagal membaca {image_path},
dilewati.")
                    continue

                images_to_process = [image]
                images_to_process.append(cv2.flip(image, 1))
                for _ in range(2):
                    angle = np.random.uniform(-15, 15)
                    M = cv2.getRotationMatrix2D((image.shape[1]//2,
image.shape[0]//2), angle, 1.0)
                    images_to_process.append(cv2.warpAffine(image,
M, (image.shape[1], image.shape[0])))
                for _ in range(2):
                    alpha = np.random.uniform(0.85, 1.15)
                    beta = np.random.randint(-10, 10)
                    images_to_process.append(cv2.convertScaleAbs(ima
ge, alpha=alpha, beta=beta))

                for img in images_to_process:
                    feats = extract_features(img,
include_shape=True)
                    if feats is not None:
                        features.append(feats)
                        labels.append(label_name)
                image_count += 1
            print(f"  - Kelas '{label_name}': {image_count} gambar asli
diproses.")

    print(f"\nTotal dataset setelah augmentasi: {len(features)}
sampel")
    return np.array(features), np.array(labels), class_names


if __name__ == '__main__':
```

```
    DATASET_PATH = 'C:/SKRIPSI FIKS SEMPRO  DAN SELESAI TAHUN
IN/PROGRAM/PELATIHAN/dataset'
    SAVED_MODEL_DIR = '.'

    print(f"--- MEMULAI PELATIHAN DAN EVALUASI (AUGMENTASI SAJA) ---
")
    print(f"Dataset diambil dari: {DATASET_PATH}")
    print(f"Hasil akan disimpan di:
{os.path.abspath(SAVED_MODEL_DIR)}")

    if not os.path.exists(SAVED_MODEL_DIR):
        os.makedirs(SAVED_MODEL_DIR)

    X, y, class_names = load_and_augment_dataset(DATASET_PATH)

    if X is not None and X.shape[0] > 0:
        X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42, stratify=y)
        print(f"\nUkuran data latih: {X_train.shape[0]}, Ukuran data
uji: {X_test.shape[0]}")

        print("\nMelatih model dengan parameter standar pada data
yang sudah di-augmentasi...")
        scaler = StandardScaler()
        X_train_scaled = scaler.fit_transform(X_train)
        X_test_scaled = scaler.transform(X_test)

        model = SVC(C=100, gamma='scale', kernel='rbf',
probability=True)
        model.fit(X_train_scaled, y_train)

        joblib.dump(model, os.path.join(SAVED_MODEL_DIR,
'svm_model_augmentasi_saja.pkl'))
        joblib.dump(scaler, os.path.join(SAVED_MODEL_DIR,
'scaler_augmentasi_saja.pkl'))
```

```
        joblib.dump(class_names, os.path.join(SAVED_MODEL_DIR,
'class_names_augmentasi_saja.pkl'))
        print("Model, scaler, dan nama kelas (versi augmentasi saja)
berhasil disimpan.")

        print("\n--- Hasil Evaluasi pada Data Uji ---")
        y_pred_test = model.predict(X_test_scaled)
        accuracy_test = accuracy_score(y_test, y_pred_test)
        report_dict_test = classification_report(y_test,
y_pred_test, target_names=class_names, output_dict=True)

        print(f"Akurasi (Data Uji): {accuracy_test:.4f}")
        print("Laporan Klasifikasi (Data Uji):\n",
classification_report(y_test, y_pred_test,
target_names=class_names))

        cm = confusion_matrix(y_test, y_pred_test,
labels=class_names)
        cm_img_path = os.path.join(SAVED_MODEL_DIR,
'confusion_matrix_augmentasi_saja.png')
        plt.figure(figsize=(10, 8))
        sns.heatmap(cm, annot=True, fmt='d', cmap='Greens',
xticklabels=class_names, yticklabels=class_names)
        plt.title('Confusion Matrix (Augmentasi Saja)', fontsize=16)
        plt.ylabel('Kelas Asli', fontsize=12)
        plt.xlabel('Kelas Prediksi', fontsize=12)
        plt.tight_layout()
        plt.savefig(cm_img_path)
        plt.close()
        print(f"Confusion matrix disimpan di: {cm_img_path}")

        report_img_path = os.path.join(SAVED_MODEL_DIR,
'classification_report_augmentasi_saja.png')
        plt.figure(figsize=(10, 7))
```

```
        sns.heatmap(pd.DataFrame(report_dict_test).iloc[:-1, :].T,
annot=True, cmap='viridis', fmt='.2f')
        plt.title('Laporan Klasifikasi (Augmentasi Saja)')
        plt.tight_layout()
        plt.savefig(report_img_path)
        plt.close()
        print(f"Heatmap laporan klasifikasi disimpan di:
{report_img_path}")



    else:
        print("Tidak ada data untuk diproses. Proses dihentikan.")
```

9. Model dengan Augmentasi dan Menggunakan Optimasi PSO

Melatih model SVM final yang diusulkan dalam penelitian, yang menggabungkan augmentasi data dengan optimasi hyperparameter menggunakan PSO.

```
import os
import cv2
import numpy as np
from sklearn.model_selection import train_test_split, KFold,
cross_val_score
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC
from sklearn.pipeline import make_pipeline
from sklearn.metrics import classification_report, confusion_matrix,
accuracy_score
import joblib
import warnings
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import pyswarms as ps
```

```
# Impor fungsi ekstraksi fitur yang sudah disempurnakan
from feature_extractor import extract_features

# Mengabaikan beberapa peringatan untuk menjaga output tetap bersih
warnings.filterwarnings("ignore", category=UserWarning,
module='skimage')
warnings.filterwarnings("ignore", category=FutureWarning)

def load_dataset(dataset_dir):
    """Memuat dataset, melakukan augmentasi acak, dan mengekstrak
fitur."""
    features, labels, class_names = [], [], sorted([d for d in
os.listdir(dataset_dir) if os.path.isdir(os.path.join(dataset_dir,
d))])
    if not class_names:
        print(f"Error: Tidak ada kelas di '{dataset_dir}'")
        return None, None, None

    print(f"Kelas ditemukan: {class_names}\nMemulai augmentasi acak
dan ekstraksi fitur...")

    for label_name in class_names:
        class_dir = os.path.join(dataset_dir, label_name)
        image_count = 0
        for image_name in os.listdir(class_dir):
            image_path = os.path.join(class_dir, image_name)
            if image_path.lower().endswith(('.png', '.jpg',
'.jpeg')):
                image = cv2.imread(image_path)
                if image is None:
                    print(f"Warning: Gagal membaca {image_path},
dilewati.")
                    continue
```

```
                # TAMBAHAN: Ubah ukuran gambar secara dinamis jika
terlalu besar untuk menghemat memori
                MAX_PIXELS = 2073600  # Batas resolusi (setara
~1920x1080)
                h, w, _ = image.shape
                if h * w > MAX_PIXELS:
                    # Menjaga rasio aspek saat mengubah ukuran
                    scale_factor = np.sqrt(MAX_PIXELS / (h * w))
                    new_w = int(w * scale_factor)
                    new_h = int(h * scale_factor)
                    image = cv2.resize(image, (new_w, new_h))
                    print(f"Info: Gambar
{os.path.basename(image_path)} di-resize ke ({new_w}, {new_h}) untuk
hemat memori.")

                # Augmentasi dinamis dinonaktifkan karena dataset
sudah seimbang.
                # Langsung ekstrak fitur dari gambar.
                feats = extract_features(image, include_shape=True,
include_lbp=True)
                if feats is not None:
                    features.append(feats)
                    labels.append(label_name)
                image_count += 1
            print(f"  - Kelas '{label_name}': {image_count} gambar asli
diproses.")

    print(f"\nTotal dataset setelah augmentasi: {len(features)}
sampel")
    return np.array(features), np.array(labels), class_names


# --- Fungsi untuk Optimasi PSO ---
def pso_objective_function(particles, X_train, y_train):
```

```
    n_particles = particles.shape[0]
    costs = np.zeros(n_particles)
    for i in range(n_particles):
        C, gamma = particles[i]
        pipeline = make_pipeline(StandardScaler(), SVC(C=C,
gamma=gamma, kernel='rbf'))
        kf = KFold(n_splits=3, shuffle=True, random_state=42)
        accuracy = cross_val_score(pipeline, X_train, y_train,
cv=kf, n_jobs=-1).mean()
        costs[i] = 1.0 - accuracy
    return costs

if __name__ == '__main__':
    DATASET_PATH = 'C:/SKRIPSI FIKS SEMPRO  DAN SELESAI TAHUN
IN/BACKUP PROGRAM/PROGRAM/PELATIHAN/dataset'
    SAVED_MODEL_DIR = 'saved_model_selada'

    print(f"--- MEMULAI PELATIHAN DAN OPTIMASI MODEL (PSO +
AUGMENTASI ACAK) ---")
    print(f"Dataset diambil dari: {DATASET_PATH}")
    print(f"Hasil akan disimpan di: {SAVED_MODEL_DIR}")

    if not os.path.exists(SAVED_MODEL_DIR):
        os.makedirs(SAVED_MODEL_DIR)

    X, y, class_names = load_dataset(DATASET_PATH)

    if X is not None and X.shape[0] > 0:
        X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42, stratify=y)
        print(f"\nUkuran data latih: {X_train.shape[0]}, Ukuran data
uji: {X_test.shape[0]}")

        # --- Optimasi Hyperparameter dengan PSO ---
        print("\nMemulai Optimasi Hyperparameter dengan PSO...")
```

```
        options = {'c1': 0.5, 'c2': 0.3, 'w': 0.9}
        bounds = (np.array([1, 0.0001]), np.array([5000, 0.1]))
        optimizer = ps.single.GlobalBestPSO(n_particles=20,
dimensions=2, options=options, bounds=bounds)

        # Menjalankan optimasi
        cost, pos = optimizer.optimize(pso_objective_function,
iters=50, X_train=X_train, y_train=y_train)

        best_C, best_gamma = pos
        print(f"\nParameter terbaik ditemukan oleh PSO:
C={best_C:.4f}, gamma={best_gamma:.4f}")

        # --- Melatih Model Final dengan Parameter Terbaik ---
        print("\nMelatih model final dengan parameter
teroptimasi...")
        scaler = StandardScaler()
        X_train_scaled = scaler.fit_transform(X_train)
        X_test_scaled = scaler.transform(X_test)

        best_model = SVC(C=best_C, gamma=best_gamma, kernel='rbf',
probability=True)
        best_model.fit(X_train_scaled, y_train)

        # Menyimpan model, scaler, dan nama kelas
        joblib.dump(best_model, os.path.join(SAVED_MODEL_DIR,
'svm_model_final.pkl'))
        joblib.dump(scaler, os.path.join(SAVED_MODEL_DIR,
'scaler_final.pkl'))
        joblib.dump(class_names, os.path.join(SAVED_MODEL_DIR,
'class_names.pkl'))
        print("Model, scaler, dan nama kelas (versi final
teroptimasi) berhasil disimpan.")
```

```
        # --- Evaluasi pada Data Latih (untuk analisis overfitting)
---
        print("\n--- Hasil Evaluasi pada Data Latih ---")
        y_pred_train = best_model.predict(X_train_scaled)
        accuracy_train = accuracy_score(y_train, y_pred_train)
        print(f"Akurasi (Data Latih): {accuracy_train:.4f}")
        print("Laporan Klasifikasi (Data Latih):\n",
classification_report(y_train, y_pred_train,
target_names=class_names))

        # --- Evaluasi pada Data Uji ---
        print("\n--- Hasil Evaluasi pada Data Uji ---")
        y_pred_test = best_model.predict(X_test_scaled)
        accuracy_test = accuracy_score(y_test, y_pred_test)
        report_dict_test = classification_report(y_test,
y_pred_test, target_names=class_names, output_dict=True)

        print(f"Akurasi (Data Uji): {accuracy_test:.4f}")
        print("Laporan Klasifikasi (Data Uji):\n",
classification_report(y_test, y_pred_test,
target_names=class_names))

        # --- Membuat dan Menyimpan Visualisasi Hasil Uji ---
        cm = confusion_matrix(y_test, y_pred_test,
labels=class_names)
        cm_img_path = os.path.join(SAVED_MODEL_DIR,
'confusion_matrix.png')
        plt.figure(figsize=(10, 8))
        sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
xticklabels=class_names, yticklabels=class_names)
        plt.title('Confusion Matrix (Model Final - PSO)',
fontsize=16)
        plt.ylabel('Kelas Asli', fontsize=12)
        plt.xlabel('Kelas Prediksi', fontsize=12)
        plt.tight_layout()
```

```
        plt.savefig(cm_img_path)
        plt.close()
        print(f"Confusion matrix disimpan di: {cm_img_path}")

        report_img_path = os.path.join(SAVED_MODEL_DIR,
'classification_report.png')
        plt.figure(figsize=(10, 7))
        sns.heatmap(pd.DataFrame(report_dict_test).iloc[:-1, :].T,
annot=True, cmap='viridis', fmt='.2f')
        plt.title('Laporan Klasifikasi (Model Final - PSO)')
        plt.tight_layout()
        plt.savefig(report_img_path)
        plt.close()
        print(f"Heatmap laporan klasifikasi disimpan di:
{report_img_path}")



        # --- ANALISIS KESALAHAN ---
        import shutil

        def get_original_paths(dataset_dir):
            paths, labels = [], []
            for label_name in class_names:
                class_dir = os.path.join(dataset_dir, label_name)
                for image_name in os.listdir(class_dir):
                    image_path = os.path.join(class_dir, image_name)
                    if image_path.lower().endswith(('.png', '.jpg',
'.jpeg')):
                        paths.append(image_path)
                        labels.append(label_name)
            return paths, labels

        print("\n--- Memulai Analisis Kesalahan ---")
        error_analysis_dir = 'analisis_kesalahan'
```

```
        if not os.path.exists(error_analysis_dir):
            os.makedirs(error_analysis_dir)

        # Dapatkan path gambar asli dan bagi dengan cara yang sama
untuk mendapatkan path data uji
        original_paths, original_labels =
get_original_paths(DATASET_PATH)
        _, X_test_paths, _, y_test_original_labels =
train_test_split(original_paths, original_labels, test_size=0.2,
random_state=42, stratify=original_labels)

        error_count = 0
        # Asumsi: y_test (dari data augmentasi) dan
y_test_original_labels memiliki urutan yang berhubungan
        # Ini adalah asumsi yang kuat, tetapi diperlukan untuk
melacak kesalahan kembali ke file asli

        # Buat set fitur unik dari data uji untuk dibandingkan
        unique_test_features = {tuple(row) for row in X_test}

        # Loop melalui gambar asli di set pengujian
        for i in range(len(X_test_paths)):
            original_path = X_test_paths[i]
            true_label = y_test_original_labels[i]

            # Ekstrak fitur dari gambar asli
            original_features = extract_features(original_path,
include_shape=True, include_lbp=True)
            if original_features is None:
                continue

            # Scaling dan prediksi
            features_scaled =
scaler.transform(original_features.reshape(1, -1))
            predicted_label = best_model.predict(features_scaled)[0]
```

```
            if predicted_label != true_label:
                error_count += 1
                print(f"Kesalahan #{error_count}:")
                print(f"  - Gambar Asli:
{os.path.basename(original_path)}")
                print(f"  - Kelas Aktual: {true_label}")
                print(f"  - Kelas Prediksi: {predicted_label}")

                new_filename =
f"AKTUAL_{true_label}_PREDIKSI_{predicted_label}__{os.path.basename(
original_path)}"
                destination_path = os.path.join(error_analysis_dir,
new_filename)
                shutil.copy(original_path, destination_path)

        if error_count > 0:
            print(f"\nTotal {error_count} kesalahan ditemukan dan
disalin ke '{error_analysis_dir}'")
        else:
            print("Tidak ada kesalahan yang ditemukan pada data
uji.")

    else:
        print("Tidak ada data untuk diproses. Proses dihentikan.")
```

10. Pendeteksi Citra di luar daun Selada

Melatih model "Penjaga Gerbang" yang tugasnya hanya untuk membedakan apakah sebuah gambar adalah gambar selada atau bukan.

```
import os
import numpy as np
import joblib
```

```
from sklearn.model_selection import train_test_split,
GridSearchCV, KFold
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC
from sklearn.metrics import classification_report, accuracy_score
import warnings

# Impor fungsi ekstraksi fitur
from feature_extractor import extract_features

# Abaikan peringatan
warnings.filterwarnings("ignore", category=UserWarning,
module='skimage')
warnings.filterwarnings("ignore", category=FutureWarning)

import cv2

def load_gatekeeper_data(dataset_dir):
    """Memuat data dari folder dataset_gatekeeper dan melakukan
augmentasi pada kelas minoritas."""
    features, labels = [], []
    class_names = sorted([d for d in os.listdir(dataset_dir) if
os.path.isdir(os.path.join(dataset_dir, d))])

    print(f"Kelas ditemukan: {class_names}")
    for label_name in class_names:
        class_dir = os.path.join(dataset_dir, label_name)
        for image_name in os.listdir(class_dir):
            image_path = os.path.join(class_dir, image_name)
            if image_path.lower().endswith(('.png', '.jpg',
'.jpeg', '.webp')):
                image = cv2.imread(image_path)
                if image is None:
                    continue
```

```
                # Terapkan augmentasi untuk semua kelas untuk
meningkatkan keragaman data
                images_to_process = [
                    image,
                    cv2.flip(image, 1), # Flip horizontal
                    cv2.rotate(image, cv2.ROTATE_90_CLOCKWISE), #
Rotasi 90 derajat
                    cv2.convertScaleAbs(image, alpha=1.2, beta=20)
# Kecerahan/kontras
                ]

                for img in images_to_process:
                    feats = extract_features(img,
include_shape=True)
                    if feats is not None:
                        features.append(feats)
                        labels.append(label_name)

    print(f"Total sampel setelah augmentasi: {len(features)}")
    return np.array(features), np.array(labels)

if __name__ == '__main__':
    # --- Konfigurasi Path ---
    DATASET_PATH = 'dataset_gatekeeper'
    SAVED_MODEL_DIR = 'saved_model'
    if not os.path.exists(SAVED_MODEL_DIR):
        os.makedirs(SAVED_MODEL_DIR)

    # --- Memuat Data ---
    X, y = load_gatekeeper_data(DATASET_PATH)

    if X is not None and X.shape[0] > 0:
        # --- Pembagian Data ---
```

```
        X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42, stratify=y)
        print(f"Ukuran data latih: {X_train.shape[0]}, Ukuran data
uji: {X_test.shape[0]}")

        # --- Penskalaan Fitur ---
        scaler = StandardScaler()
        X_train_scaled = scaler.fit_transform(X_train)
        X_test_scaled = scaler.transform(X_test)

        # --- Pelatihan Model SVM dengan GridSearchCV ---
        print("\nMemulai pelatihan Model Penjaga Gerbang
(Gatekeeper) dengan GridSearchCV...")
        param_grid = {
            'C': [0.1, 1, 10, 100, 1000],
            'gamma': [1, 0.1, 0.01, 0.001, 0.0001],
            'kernel': ['rbf']
        }
        # Menggunakan 3-fold cross-validation
        grid = GridSearchCV(SVC(probability=True,
random_state=42), param_grid, refit=True, verbose=2,
cv=KFold(n_splits=3, shuffle=True, random_state=42))
        grid.fit(X_train_scaled, y_train)

        model = grid.best_estimator_
        print(f"Pelatihan selesai. Parameter terbaik ditemukan:
{grid.best_params_}")

        # --- Menyimpan Model dan Scaler ---
        joblib.dump(model, os.path.join(SAVED_MODEL_DIR,
'gatekeeper_model.pkl'))
        joblib.dump(scaler, os.path.join(SAVED_MODEL_DIR,
'gatekeeper_scaler.pkl'))
```

```
        print("Model Penjaga Gerbang dan scaler berhasil
disimpan.")

        # --- Evaluasi Model ---
        print("\n--- Hasil Evaluasi Model Penjaga Gerbang pada
Data Uji ---")
        y_pred = model.predict(X_test_scaled)
        accuracy = accuracy_score(y_test, y_pred)

        print(f"Akurasi: {accuracy:.4f}")
        print("Laporan Klasifikasi:\n",
classification_report(y_test, y_pred))
    else:
        print("Tidak ada data untuk diproses. Proses dihentikan.")
```

11. Program untuk Predict

Prediksi kelas penyakit dan skor kepercayaan, serta visualisasi gambar jika cukup yakin.

```
import cv2
import joblib
import numpy as np
import os
import matplotlib.pyplot as plt
import warnings
import argparse

# Impor fungsi ekstraksi fitur yang sudah konsisten
from feature_extractor import extract_features

# Mengabaikan beberapa peringatan untuk menjaga output tetap bersih
warnings.filterwarnings("ignore", category=UserWarning,
module='skimage')
warnings.filterwarnings("ignore", category=FutureWarning)
```

```
def predict_single_image(image_path, model, scaler, class_names):
    """
    Melakukan prediksi pada satu gambar menggunakan logika ekstraksi
fitur terpusat.
    Mengembalikan tuple: (label_prediksi, kepercayaan) atau (None,
None) jika gagal.
    """
    if not os.path.exists(image_path):
        print(f"Error: File tidak ditemukan di '{image_path}'")
        return None, None

    # Memanggil fungsi extract_features dari feature_extractor.py
    features = extract_features(image_path, include_shape=True)

    if features is None:
        print("Error: Gagal mengekstrak fitur dari gambar.")
        return None, None

    try:
        # Reshape fitur agar sesuai dengan input scaler
        features_reshaped = features.reshape(1, -1)

        # Lakukan penskalaan fitur dan prediksi
        features_scaled = scaler.transform(features_reshaped)
        prediction_proba = model.predict_proba(features_scaled)

        # Dapatkan kelas dengan probabilitas tertinggi
        best_class_index = np.argmax(prediction_proba)
        prediction_label = class_names[best_class_index]
        confidence = prediction_proba[0][best_class_index] * 100

        return prediction_label, confidence

    except Exception as e:
```

```
        print(f"Terjadi error saat prediksi: {e}")
        return None, None

def display_prediction(image_path, label, confidence):
    """
    Menampilkan gambar beserta hasil prediksinya menggunakan
matplotlib.
    """
    img = cv2.imread(image_path)
    img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

    plt.figure(figsize=(8, 6))
    plt.imshow(img_rgb)

    title_text = f"Prediksi: {label} ({confidence:.2f}%)"
    plt.title(title_text, fontsize=14)

    plt.axis('off')
    plt.show()

def main():
    """
    Fungsi utama untuk menjalankan prediksi interaktif.
    """
    parser = argparse.ArgumentParser(description="Prediksi penyakit
daun menggunakan model SVM (KHUSUS SELADA).")
    parser.add_argument(
        '--threshold',
        type=float,
        default=80.0,
        help="Ambang batas kepercayaan minimum (0-100) untuk
menampilkan hasil prediksi. Default: 80.0"
    )
    args = parser.parse_args()
    CONFIDENCE_THRESHOLD = args.threshold
```

```
    # --- Konfigurasi Path ---
    # DIUBAH: Path menunjuk ke model khusus selada
    SAVED_MODEL_DIR = 'saved_model_selada'
    MODEL_PATH = os.path.join(SAVED_MODEL_DIR,
'svm_model_final.pkl')
    SCALER_PATH = os.path.join(SAVED_MODEL_DIR, 'scaler_final.pkl')
    CLASS_NAMES_PATH = os.path.join(SAVED_MODEL_DIR,
'class_names.pkl')

    print(f"--- MENGGUNAKAN MODEL DETEKSI KHUSUS SELADA ---")
    print(f"Model dimuat dari: '{SAVED_MODEL_DIR}'")
    print(f"Ambang batas kepercayaan: {CONFIDENCE_THRESHOLD}%")

    # --- Memuat Model, Scaler, dan Nama Kelas ---
    try:
        model = joblib.load(MODEL_PATH)
        scaler = joblib.load(SCALER_PATH)
        class_names = joblib.load(CLASS_NAMES_PATH)
        print("\nModel, scaler, dan nama kelas berhasil dimuat.")
        print(f"Kelas yang dikenali: {class_names}")
    except FileNotFoundError as e:
        print(f"Error: Gagal memuat file - {e.filename}.")
        print(f"Pastikan direktori '{SAVED_MODEL_DIR}' ada dan
berisi model yang sudah dilatih.")
        return

    # --- Loop Interaktif untuk Prediksi ---
    while True:
        try:
            image_path = input("\nMasukkan path gambar selada (atau
'keluar' untuk berhenti): ").strip().strip('"')

            if image_path.lower() == 'keluar':
                break
```

```
            if not image_path:
                continue

            label, confidence = predict_single_image(image_path,
model, scaler, class_names)

            if label and confidence is not None:
                if confidence >= CONFIDENCE_THRESHOLD:
                    print(f"--> Hasil Analisis: Prediksi '{label}'
dengan kepercayaan {confidence:.2f}%")
                    display_prediction(image_path, label,
confidence)
                else:
                    print(f"--> Model tidak cukup yakin
(Kepercayaan: {confidence:.2f}%). Ambang batas saat ini:
{CONFIDENCE_THRESHOLD}%")
                    print("    Coba turunkan threshold jika perlu,
contoh: --threshold 50")

        except (KeyboardInterrupt, EOFError):
            print("\nKeluar dari program.")
            break
        except Exception as e:
            print(f"Terjadi kesalahan pada loop utama: {e}")

if __name__ == '__main__':
    main()
```

12. Kode Backend

Menjalankan sebuah server web lokal yang menyediakan API (Application Programming Interface) untuk menerima gambar dari frontend, memprosesnya dengan model, dan mengembalikan hasil prediksi.

```
import os
```

```
import numpy as np
from flask import Flask, request, jsonify, render_template
import joblib
from werkzeug.utils import secure_filename
from flask_cors import CORS

# Impor fungsi ekstraksi fitur yang sudah konsisten
from feature_extractor import extract_features

# Inisialisasi aplikasi Flask
app = Flask(__name__)
CORS(app) # Aktifkan CORS
app.config['UPLOAD_FOLDER'] = 'static/uploads/'

# Pastikan folder upload ada
os.makedirs(app.config['UPLOAD_FOLDER'], exist_ok=True)

# Muat model, scaler, dan nama kelas
try:
    model = joblib.load(os.path.join('saved_model_selada',
'svm_model_final.pkl'))
    scaler = joblib.load(os.path.join('saved_model_selada',
'scaler_final.pkl'))
    class_names = joblib.load(os.path.join('saved_model_selada',
'class_names.pkl'))
    print("Model, scaler, dan nama kelas berhasil dimuat.")
except Exception as e:
    print(f"Error saat memuat model atau scaler: {e}")
    model = None
    scaler = None
    class_names = None

@app.route('/')
def index():
    # Halaman ini tidak lagi diperlukan karena kita fokus pada API
```

```
    return jsonify({"message": "Backend server untuk deteksi
penyakit daun aktif."})

@app.route('/predict', methods=['POST'])
def predict():
    if model is None or scaler is None or class_names is None:
        return jsonify({'error': 'Model atau scaler tidak dimuat
dengan benar.'}), 500

    if 'file' not in request.files:
        return jsonify({'error': 'Tidak ada file yang dikirim'}),
400

    file = request.files['file']

    if file.filename == '':
        return jsonify({'error': 'Tidak ada file yang dipilih'}),
400

    if file:
        filename = secure_filename(file.filename)
        filepath = os.path.join(app.config['UPLOAD_FOLDER'],
filename)
        file.save(filepath)

        try:
            # Ekstrak fitur dari gambar yang diunggah
            features = extract_features(filepath,
include_shape=True)

            if features is not None:
                # Scaling fitur
                features_reshaped = features.reshape(1, -1)
                scaled_features =
scaler.transform(features_reshaped)
```

```
                # Lakukan prediksi
                predicted_class = model.predict(scaled_features)[0]
                probabilities =
model.predict_proba(scaled_features)[0]
                # Cari indeks kelas yang diprediksi untuk
mendapatkan confidence score yang benar
                class_index = np.where(model.classes_ ==
predicted_class)[0][0]
                confidence = probabilities[class_index]

                return jsonify({
                    'prediction': predicted_class,
                    'confidence': f"{confidence:.2%}"
                })
            else:
                return jsonify({'error': 'Gagal mengekstrak fitur
dari gambar.'}), 400
        except Exception as e:
            return jsonify({'error': f'Terjadi kesalahan saat
prediksi: {e}'}), 500
        finally:
            if os.path.exists(filepath):
                os.remove(filepath)

if __name__ == '__main__':
    # Jalankan di port 5001 untuk menghindari konflik dengan
frontend
    app.run(host='0.0.0.0', port=5001, debug=True)
```

Lampiran 7. Surat Keterangan Bebas Plagiat

MAJELIS PENDIDIKAN TINGGI PIMPINAN PUSAT MUHAMMADIYAH
UNIVERSITAS MUHAMMADIYAH MAKASSAR
UPT PERPUSTAKAAN DAN PENERBITAN
Alamat kantor: Jl.Sultan Alauddin NO.259 Makassar 90221 Tlp.(0411) 866972,881593, Fax.(0411) 865588

بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ

SURAT KETERANGAN BEBAS PLAGIAT

UPT Perpustakaan dan Penerbitan Universitas Muhammadiyah Makassar,
Menerangkan bahwa mahasiswa yang tersebut namanya di bawah ini:

Nama : Muh. Ulil Amri
Nim : 105841106221
Program Studi : Teknik Informatika

Dengan nilai:

No	Bab	Nilai	Ambang Batas
1	Bab 1	5%	10 %
2	Bab 2	15%	25 %
3	Bab 3	8%	10 %
4	Bab 4	7%	10 %
5	Bab 5	3%	5 %

Dinyatakan telah lulus cek plagiat yang diadakan oleh UPT- Perpustakaan dan Penerbitan Universitas Muhammadiyah Makassar Menggunakan Aplikasi Turnitin.

Demikian surat keterangan ini diberikan kepada yang bersangkutan untuk dipergunakan seperlunya.

Makassar, 26 Agustus 2025
Mengetahui,

Kepala UPT- Perpustakaan dan Pernerbitan,

Nursinah, S.Hum.,M.I.P
NBM. 964 591

Jl. Sultan Alauddin no 259 makassar 90222
Telepon (0411)866972,881 593,fax (0411)865 588
Website: www.library.unismuh.ac.id
E-mail : perpustakaan@unismuh.ac.id

Lampiran 8. Hasil Turnitin Bab 1 - Bab 5

Bab I MUH. ULIL AMRI 105841106221

ORIGINALITY REPORT

5%	5%	0%	2%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	digilibadmin.unismuh.ac.id Internet Source	3%
2	repository.ub.ac.id Internet Source	2%

Exclude quotes Off

Exclude bibliography On

Exclude matches

Bab II MUH. ULIL AMRI 105841106221

by Tahap Tutup

Bab II MUH. ULIL AMRI 105841106221

ORIGINALITY REPORT

15%	12%	9%	7%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Forum Perpustakaan Perguruan Tinggi Indonesia Jawa Timur Student Paper	2%
2	informatika.stei.itb.ac.id Internet Source	1%
3	papersmai.mercubuana-yogya.ac.id Internet Source	1%
4	id.123dok.com Internet Source	1%
5	Sadri Talib, Sakina Sudin, Muhammad Dzikrullah Suratin. "PENERAPAN METODE SUPPORT VECTOR MACHINE (SVM) PADA KLASIFIKASI JENIS CENGKEH BERDASARKAN FITUR TEKSTUR DAUN", PROSISKO: Jurnal Pengembangan Riset dan Observasi Sistem Komputer, 2024 Publication	1%
6	123dok.com Internet Source	1%
7	Submitted to Perpustakaan Student Paper	1%
8	jurnal.darmajaya.ac.id Internet Source	<1%
9	Submitted to Universitas Jember Student Paper	<1%

Bab III MUH. ULIL AMRI 105841106221

by Tahap Tutup

Bab III MUH. ULIL AMRI 105841106221

ORIGINALITY REPORT

8%	6%	3%	2%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Forum Perpustakaan Perguruan Tinggi Indonesia Jawa Timur II Student Paper	1%
2	text-id.123dok.com Internet Source	1%
3	123dok.com Internet Source	1%
4	repository.ar-raniry.ac.id Internet Source	1%
5	Debi Gusmaliza, Siti Aminah. "Sistem Identifikasi Kualitas Biji Kopi Robusta berbasis Image Processing dengan Support Vector Machine", Edumatic: Jurnal Pendidikan Informatika, 2024 Publication	1%
6	www.bios-mods.com Internet Source	1%
7	hendriyan.wordpress.com Internet Source	1%
8	ejournal.gunadarma.ac.id Internet Source	<1%

Bab IV MUH. ULIL AMRI 105841106221

by Tahap Tutup

Bab IV MUH. ULIL AMRI 105841106221

ORIGINALITY REPORT

7%	6%	3%	3%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to polteksn Student Paper	1%
2	www.mdpi.com Internet Source	1%
3	journal.thamrin.ac.id Internet Source	1%
4	Submitted to Universitas Muhammadiyah Surakarta Student Paper	1%
5	www.neliti.com Internet Source	1%
6	123dok.com Internet Source	1%
7	arxiv.org Internet Source	1%
8	di.uqo.ca Internet Source	1%
9	journals.uhd.edu.iq Internet Source	1%
10	Marwa Sulehu, Wisda Wisda, First Wanita, Markani Markani. "Optimasi Prediksi Kelulusan Mahasiswa Menggunakan Random Forest untuk Meningkatkan Tingkat Retensi", Jurnal Minfo Polgan, 2025 Publication	<1%

Bab V MUH. ULIL AMRI 105841106221

by Tahap Tutup

Bab V MUH. ULIL AMRI 105841106221

ORIGINALITY REPORT

3%	3%	0%	0%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	repositori.uin-alauddin.ac.id Internet Source	3%

Exclude quotes	Off	Exclude matches	< 2%
Exclude bibliography	Off		