



Analisis Ekstraksi Fitur Tekstur dan Warna Untuk Klasifikasi Penyakit Daun Kentang Menggunakan Metode GLCM

Adinda Nadila^{1*}, Mahardika Abdi Prawira Tanjung²

^{1,2}Program Studi Sistem Informasi, Fakultas Ilmu Komputer dan Teknologi Informasi, Universitas Muhammadiyah Sumatera Utara, Medan, Indonesia

Email: ^{1*}adindanadilaa04@gmail.com, ²mahardikaabdiprawira@umsu.ac.id

Abstrak

Penyakit bercak daun dini (*Early Blight*) dan bercak daun lambat (*Late Blight*) merupakan ancaman utama produktivitas tanaman kentang, sedangkan identifikasi visual secara manual bersifat subjektif dan memakan waktu. Penelitian ini bertujuan merancang algoritma fusi fitur tekstur dan warna, menganalisis karakteristik ruang fitur yang dihasilkan, serta mengevaluasi kinerjanya untuk klasifikasi dua kelas penyakit daun kentang. Fitur tekstur diekstraksi menggunakan *Gray Level Co-occurrence Matrix* (GLCM) berupa kontras, korelasi, energi, dan homogenitas pada sudut 0° dengan jarak satu piksel, sedangkan fitur warna berupa nilai rata-rata kanal *Hue*, *Saturation*, dan *Value* (HSV). Kedua kelompok fitur digabungkan melalui konkatenasi vektor satu dimensi menjadi vektor tujuh dimensi, kemudian diklasifikasikan menggunakan *Random Forest* dan *1-Nearest Neighbor* (1-NN) pada citra PlantVillage berukuran 256×256 piksel. Pengujian batch terhadap 80 citra (40 citra per kelas) menunjukkan bahwa *Random Forest* mencapai akurasi 95,00% (76 dari 80 citra benar), dengan akurasi kelas *Early Blight* 97,50% dan *Late Blight* 92,50%, sedangkan 1-NN mencapai akurasi 93,75%. Kesalahan klasifikasi terutama terjadi pada lesi yang sangat kecil atau telah menyatu sehingga nilai warna dan teksturnya menyerupai kelas lain. Sistem diimplementasikan sebagai aplikasi web berbasis Flask. Hasil ini menunjukkan bahwa fusi fitur *handcrafted* yang ringkas layak digunakan sebagai pendekatan awal yang ringan, namun masih memerlukan validasi pada data yang lebih besar dan beragam.

Kata Kunci: Ekstraksi Fitur, Tekstur, Warna, Klasifikasi Penyakit Daun Kentang, GLCM.

PENDAHULUAN

Kentang (*Solanum tuberosum*) merupakan komoditas pangan strategis yang bernilai ekonomi tinggi, namun produktivitasnya kerap terancam oleh penyakit daun. Dua penyakit yang paling sering dijumpai adalah bercak daun dini (*Early Blight*) yang disebabkan oleh jamur *Alternaria solani* (Li dkk., 2024) dan bercak daun lambat (*Late Blight*) yang disebabkan oleh oomisetes *Phytophthora infestans*. Keduanya menimbulkan lesi pada daun yang mengurangi luas jaringan fotosintesis sehingga berdampak pada hasil panen. Identifikasi penyakit secara visual oleh petani maupun penyuluh bersifat subjektif dan bergantung pada pengalaman, sementara pemeriksaan manual cenderung memakan waktu dan rawan kesalahan (Shoaib dkk., 2023). Oleh karena itu, sistem klasifikasi berbasis citra digital dan pembelajaran mesin dibutuhkan sebagai alat bantu diagnosis dini yang cepat dan konsisten.

Pendekatan pembelajaran mendalam (*deep learning*) telah banyak dilaporkan memberikan kinerja tinggi pada klasifikasi penyakit daun kentang. Alhammad dkk. (2025), misalnya, mengombinasikan *transfer learning* dengan *explainable AI* untuk meningkatkan akurasi sekaligus interpretabilitas model. Namun, model semacam itu umumnya menuntut data dan sumber daya komputasi yang besar. Sebagai alternatif, ekstraksi fitur buatan tangan (*handcrafted features*) yang dipadukan dengan pengklasifikasi klasik tetap kompetitif dan lebih ringan. *Gray Level Co-occurrence Matrix* (GLCM) yang diperkenalkan oleh Haralick dkk. (1973) merupakan salah satu deskriptor tekstur yang paling banyak digunakan. Nyasulu dkk. (2023) memanfaatkan fitur tekstur GLCM untuk mengklasifikasikan empat penyakit jamur pada daun tomat dan memperoleh akurasi 94% dengan *Artificial Neural Network*, sedangkan pada daun kentang Muhamad (2024) menggunakan fitur GLCM dengan algoritma *Multi-SVM*. Penambahan informasi warna terbukti bermanfaat: Muchtar (2024) mengintegrasikan fitur warna, tekstur, dan Renyi fraktal dengan *Linear Discriminant Analysis* pada 456 citra daun kentang dan memperoleh akurasi 98,46%, sedangkan Bulan dan Nurtanio (2025) menggabungkan fitur warna dan tekstur dengan *Principal Component Analysis* dan *K-Nearest Neighbor* ($k = 1$) hingga mencapai akurasi 97,78% pada 12 kelas.

Paradigma yang sama, yaitu ekstraksi fitur terstruktur yang diikuti pengklasifikasi klasik dan diterapkan pada sistem berbasis web, juga dimanfaatkan pada bidang lain. Kamila Insani P dan Tanjung (2026) pada jurnal ini membangun klasifikasi ujaran kebencian teks bahasa gaul dengan representasi fitur TF-IDF dan *Support Vector Machine* berkernel linear yang menghasilkan akurasi 99,10% pada 2.663 data uji, lalu mengimplementasikannya pada sistem web untuk mendukung moderasi konten. Meskipun domain datanya berbeda (teks, bukan citra), studi tersebut menunjukkan bahwa kombinasi ekstraksi fitur yang tepat dan algoritma klasik yang ringan dapat dikemas menjadi aplikasi yang bermanfaat secara praktis.

Studi-studi di atas menggunakan dataset, jumlah kelas, dan kombinasi fitur yang berbeda-beda, sehingga perilaku vektor fitur warna dan tekstur yang ringkas pada skenario dua penyakit yang mirip secara visual (*Early Blight* dan *Late Blight*) masih perlu dikaji lebih rinci. Kebaruan penelitian ini terletak pada (1) perancangan vektor fitur fusi yang ringkas, yaitu tujuh dimensi yang terdiri atas empat properti GLCM dan tiga rata-rata kanal HSV, (2) perbandingan *Random Forest* dan 1-*Nearest Neighbor* (1-NN) pada vektor tersebut disertai analisis kasus salah klasifikasi per sampel, serta (3) implementasi pengujian dalam aplikasi web dengan pelaporan pengujian batch otomatis. Permasalahan yang dikaji adalah seberapa baik fusi fitur GLCM dan HSV yang ringkas mampu membedakan *Early Blight* dan *Late Blight*, serta bagaimana kinerja kedua pengklasifikasi tersebut.

Tujuan penelitian ini adalah (1) merancang algoritma ekstraksi dan fusi fitur tekstur GLCM dan warna HSV pada citra daun kentang, (2) menganalisis karakteristik ruang fitur hasil fusi dalam memisahkan kedua kelas penyakit, dan (3) mengevaluasi kinerja klasifikasi *Random Forest* dan 1-NN menggunakan *confusion matrix*.

METODE

Dataset

Data yang digunakan adalah data sekunder berupa citra daun kentang dari repositori publik PlantVillage (Hughes dan Salathé, 2015), yang banyak dipakai sebagai tolok ukur penelitian klasifikasi penyakit tanaman. Citra berformat RGB dikurasi ke dalam dua kelas dengan jumlah seimbang, yaitu *Early Blight* dan *Late Blight*, masing-masing 150 citra (Tabel 1). *Early Blight* ditandai lesi nekrotik cokelat tua dengan pola cincin konsentris menyerupai target (Gambar 1), sedangkan *Late Blight* ditandai bercak basah tidak beraturan berwarna cokelat keabuan yang menyebar cepat (Gambar 2). Pengujian kinerja dilakukan terhadap 80 citra uji yang terdiri atas 40 citra *Early Blight* dan 40 citra *Late Blight*.

Tabel 1. Spesifikasi dataset citra daun kentang

No	Kelas	Karakteristik visual	Jumlah citra	Sumber
1	<i>Early Blight</i>	Lesi nekrotik cokelat tua dengan pola cincin konsentris	150	PlantVillage
2	<i>Late Blight</i>	Bercak basah tidak beraturan, cokelat keabuan, menyebar luas	150	PlantVillage
Total			300	



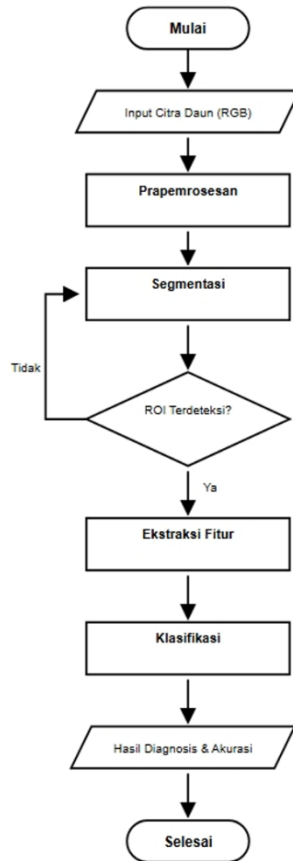
Gambar 1. Contoh daun kentang penyakit *Early Blight*



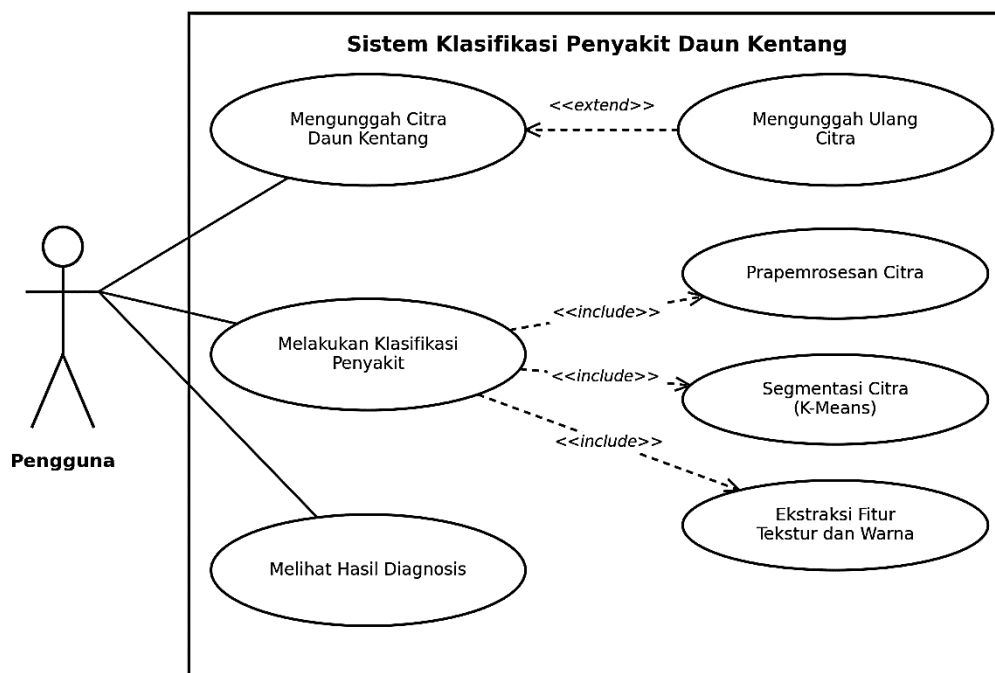
Gambar 2. Contoh daun kentang penyakit *Late Blight*

Desain Penelitian

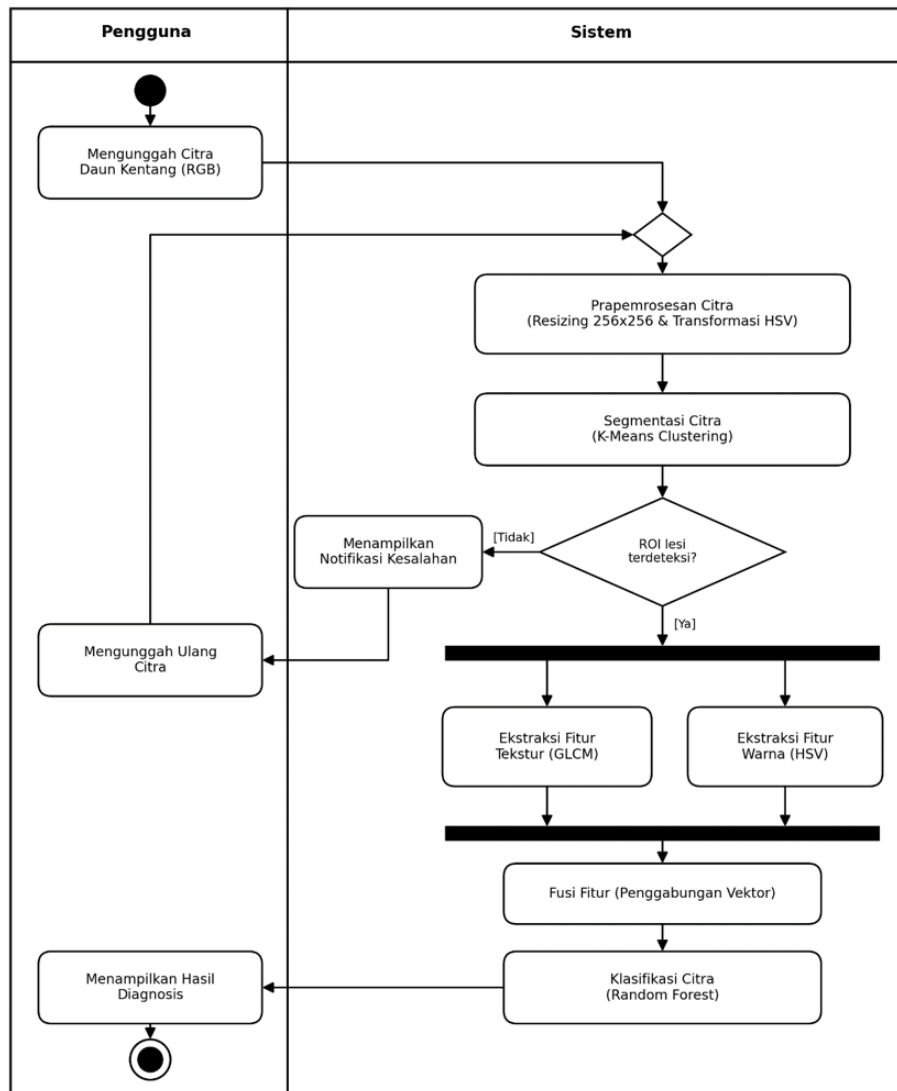
Penelitian ini merupakan penelitian eksperimental kuantitatif yang mengembangkan dan menguji sistem klasifikasi penyakit daun kentang berbasis fusi fitur. Alur penelitian terdiri atas pengumpulan citra, praproses, ekstraksi fitur warna dan tekstur, fusi fitur, pelatihan dan pengujian model klasifikasi, serta evaluasi kinerja. Alur kerja sistem diilustrasikan pada Gambar 3, interaksi pengguna dan sistem pada Gambar 4, dan aktivitas antara pengguna dan sistem pada Gambar 5.



Gambar 3. *Flowchart* alur kerja sistem



Gambar 4. *Use case diagram* sistem



Gambar 5. Activity diagram sistem

Praproses Citra

Seluruh citra dibaca menggunakan pustaka OpenCV, kemudian diubah ukurannya menjadi 256×256 piksel dengan interpolasi bilinear agar panjang vektor fitur dan area perhitungan GLCM seragam untuk setiap sampel. Ukuran tersebut juga menjaga beban komputasi tetap rendah.

Ekstraksi Fitur Warna

Fitur warna diekstraksi pada ruang warna HSV karena komponen *Hue* dan *Saturation* memisahkan informasi kromatik dari kecerahan (*Value*), sehingga lebih tahan terhadap variasi pencahayaan dibandingkan ruang RGB. Setelah kanal R, G, dan B dinormalisasi ke rentang 0–1 (R' , G' , B'), dihitung nilai maksimum $C_{\max}$, minimum $C_{\min}$, dan selisih $\Delta = C_{\max} - C_{\min}$. Komponen HSV dihitung dengan Persamaan (1) sampai (3).

$$V = C_{\max} \quad (1)$$

$$S = 0 \text{ jika } C_{\max} = 0; S = \Delta / C_{\max} \text{ jika } C_{\max} \neq 0 \quad (2)$$

$$H = 60^\circ \times ((G' - B') / \Delta \bmod 6) \text{ jika } C_{\max} = R'; H = 60^\circ \times ((B' - R') / \Delta + 2) \text{ jika } C_{\max} = G'; H = 60^\circ \times ((R' - G') / \Delta + 4) \text{ jika } C_{\max} = B' \quad (3)$$

Pada pustaka OpenCV, sudut *Hue* 0° – 360° dipetakan ke skala 0–179 ($H/2$) agar sesuai dengan tipe data 8-bit. Sebagai contoh, piksel oranye dengan RGB (255, 165, 0) menghasilkan $H = 38,82^\circ$, $S = 1,0$, dan $V = 1,0$, yang pada OpenCV bernilai (19, 255, 255). Fitur warna yang digunakan adalah nilai rata-rata ketiga kanal HSV pada seluruh citra, sehingga menghasilkan tiga fitur.

Ekstraksi Fitur Tekstur dengan GLCM

Citra diubah ke skala abu-abu, kemudian matriks kookurensi tingkat keabuan dibentuk menggunakan fungsi *graycomatrix* pada pustaka scikit-image dengan jarak piksel $d = 1$ dan sudut $\theta = 0^\circ$. Dari matriks yang ternormalisasi $P(i, j)$ berukuran $G \times G$, fungsi *graycoprops* menghitung empat properti

statistik orde dua (Haralick dkk., 1973), yaitu kontras, korelasi, energi, dan homogenitas, sebagaimana Persamaan (4) sampai (7).

$$\text{Kontras} = \sum_i \sum_j (i - j)^2 \cdot P(i, j) \quad (4)$$

$$\text{Korelasi} = \sum_i \sum_j [(i - \mu_i)(j - \mu_j) \cdot P(i, j)] / (\sigma_i \sigma_j) \quad (5)$$

$$\text{Energi} = \sqrt{\sum_i \sum_j P(i, j)^2} \quad (6)$$

$$\text{Homogenitas} = \sum_i \sum_j P(i, j) / (1 + |i - j|) \quad (7)$$

Kontras mengukur variasi intensitas lokal, korelasi mengukur ketergantungan linear antarpiksel bertetangga, energi mengukur keseragaman tekstur, dan homogenitas mengukur kedekatan sebaran elemen matriks terhadap diagonalnya.

Fusi Fitur

Tiga fitur warna HSV dan empat properti GLCM digabungkan dengan teknik konkatenasi vektor menjadi satu larik satu dimensi berukuran tujuh elemen. Vektor fusi ini menjadi masukan bagi kedua pengklasifikasi, sehingga informasi kromatik (perubahan pigmen) dan informasi spasial (kekasaran permukaan lesi) dipertimbangkan secara bersamaan.

Klasifikasi

Dua algoritma dibandingkan, yaitu *Random Forest* (RF) dan *1-Nearest Neighbor* (1-NN), yang dibangun menggunakan pustaka scikit-learn (Pedregosa dkk., 2011). RF (Breiman, 2001) merupakan ansambel pohon keputusan yang dilatih pada subset data hasil *bootstrap* dengan pemilihan subset fitur acak pada setiap percabangan; keputusan akhir diambil melalui pemungutan suara mayoritas, dan proporsi suara pohon digunakan sebagai tingkat keyakinan prediksi. Algoritma 1-NN (Cover dan Hart, 1967) menetapkan label citra uji sesuai label citra latih terdekat berdasarkan jarak Euclidean pada Persamaan (8), dengan q adalah vektor fitur citra uji, p adalah vektor fitur citra latih, dan $n = 7$ adalah jumlah dimensi fitur.

$$d(p, q) = \sqrt{\sum_{k=1}^n (q_k - p_k)^2} \quad (8)$$

Evaluasi Kinerja

Kinerja diukur menggunakan *confusion matrix* yang memuat *true positive* (TP), *true negative* (TN), *false positive* (FP), dan *false negative* (FN). Metrik yang dihitung meliputi akurasi, presisi, *recall*, dan *F1-score* seperti pada Persamaan (9) sampai (12). Untuk klasifikasi dua kelas ini, presisi, *recall*, dan *F1-score* dihitung per kelas dan dirata-ratakan secara makro.

$$\text{Akurasi} = (TP + TN) / (TP + TN + FP + FN) \quad (9)$$

$$\text{Presisi} = TP / (TP + FP) \quad (10)$$

$$\text{Recall} = TP / (TP + FN) \quad (11)$$

$$F1\text{-score} = 2 \times \text{Presisi} \times \text{Recall} / (\text{Presisi} + \text{Recall}) \quad (12)$$

Implementasi dan Lingkungan Eksperimen

Sistem diimplementasikan dalam bahasa Python dengan pustaka yang dirangkum pada Tabel 2. Aplikasi web dibangun menggunakan *framework* Flask: berkas web.py menangani rute, ekstraksi fitur fusi, dan pemuatan model terlatih (model_rf.pkl), berkas index.html menyediakan antarmuka unggah citra, dan skrip batch_test_to_pdf.py menjalankan pengujian massal serta menulis hasilnya ke berkas laporan_pengujian_batch.csv dan laporan PDF.

Tabel 2. Perangkat lunak dan pustaka yang digunakan

Kategori	Pustaka	Fungsi dalam penelitian
Bahasa pemrograman	Python	Menjalankan seluruh skrip praproses, ekstraksi fitur, pelatihan, dan evaluasi
Pengolahan citra	OpenCV	Pembacaan citra, perubahan ukuran, dan konversi ruang warna RGB ke HSV
Pengolahan citra	scikit-image	Pembentukan matriks GLCM dan ekstraksi properti tekstur (van der Walt dkk., 2014)
Pembelajaran mesin	scikit-learn	Model <i>Random Forest</i> dan 1-NN serta perhitungan metrik evaluasi
Komputasi numerik	NumPy	Operasi larik dan perhitungan rata-rata kanal warna
Struktur data	Pandas	Pembacaan dan analisis berkas laporan pengujian batch
Aplikasi web dan pelaporan	Flask, ReportLab	Antarmuka web dan pembuatan laporan PDF pengujian batch

HASIL DAN PEMBAHASAN

Hasil Implementasi Sistem

Sistem berhasil diimplementasikan sebagai aplikasi web. Berkas web.py memuat model *Random Forest* dan 1-NN yang telah disimpan, membaca citra yang diunggah, mengubah ukurannya menjadi 256×256 piksel, menghitung rata-rata kanal HSV dan empat properti GLCM, lalu menggabungkannya menjadi vektor fusi sebelum diklasifikasikan (Gambar 6). Antarmuka pada berkas index.html menyediakan formulir unggah citra JPG/JPEG dan menampilkan citra yang diunggah berdampingan dengan tabel perbandingan prediksi kedua model, termasuk tingkat keyakinan *Random Forest* (Gambar 7).

```
Untitled

@app.route('/', methods=)
def index():
    if request.method == 'POST':
        if 'file' not in request.files:
            return redirect(request.url)
        file = request.files['file']
        if file.filename == '':
            return redirect(request.url)
        if file:
            filepath = os.path.join(app.config, file.filename)
            file.save(filepath)

            # Ekstraksi fusi fitur
            features = extract_features(filepath)

            # Prediksi menggunakan Random Forest dan 1-NN
            pred_rf = model_rf.predict(features)
            prob_rf = np.max(model_rf.predict_proba(features)) *

100

            pred_1nn = model_1nn.predict(features)

            return render_template('index.html',
                                   filename=file.filename,
                                   pred_rf=pred_rf,
                                   prob_rf=f"{prob_rf:.2f}",
                                   pred_1nn=pred_1nn)

    return render_template('index.html')

if __name__ == '__main__':
    app.run(debug=True)
```

Gambar 6. Implementasi kode ekstraksi fitur citra dan inisialisasi Flask

```
Untitled

<!DOCTYPE html>
<html lang="id">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Sistem Deteksi Penyakit Daun Kentang</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">
</head>
<body class="bg-light">
    <div class="container py-5">
        <div class="text-center mb-4">
            <h1>Sistem Deteksi Penyakit Daun Kentang</h1>
            <h2>Fusi Fitur HSV & GLCM</h2>
        </div>
        <div class="row justify-content-center">
            <div class="col-md-6">
                <div class="card shadow-sm p-4 bg-white rounded">
                    <form class="form" method="post" enctype="multipart/form-data">
                        <div class="mb-3">
                            <label for="file" class="form-label fw-bold">Unggah Citra Daun Kentang</label>
                            <input class="form-file-input form-control" type="file" name="file" id="file" required>
                        </div>
                        <button type="submit" class="btn btn-success w-100">Deteksi Penyakit</button>
                    </form>
                </div>
            </div>
        </div>
        <div class="row justify-content-center">
            <div class="col-md-6">
                <div class="card shadow-sm p-4 bg-white rounded">
                    <h3>Hasil Analisis Citra</h3>
                    <div class="row align-items-center">
                        <div class="col-md-6 text-center">
                            
                            <p class="text-muted small">Nama File: {{ filename }}</p>
                        </div>
                        <div class="col-md-6">
                            <table class="table table-bordered">
                                <thead class="table-dark">
                                    <tr>
                                        <th>Metode Klasifikasi</th>
                                        <th>Hasil Prediksi</th>
                                    </tr>
                                </thead>
                            </table>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </div>
</body>
</html>
```

Gambar 7. Kode antarmuka web (HTML/Jinja2) sistem deteksi penyakit daun kentang

Untuk pengujian massal, skrip `batch_test_to_pdf.py` membaca seluruh citra uji secara berurutan, mengekstraksi fitur fusi, memprediksi kelas dengan kedua model, dan mencatat nama berkas, label sebenarnya (*ground truth*) yang ditentukan dari pola nama berkas, prediksi, tingkat keyakinan, serta status kebenaran ke berkas CSV. Berkas tersebut kemudian dikonversi menjadi laporan PDF menggunakan ReportLab (Gambar 8). Analisis akurasi dilakukan dengan skrip Pandas yang membandingkan kolom *ground truth* dengan kolom prediksi (Gambar 9).

```
import os
import csv
import cv2
import pickle
import numpy as np
from skimage.feature import graycomatrix, graycoprops
from reportlab.lib.pagesizes import letter
from reportlab.platypus import SimpleDocTemplate, Paragraph, Spacer, Table, TableStyle
from reportlab.lib.styles import getSampleStyleSheet, ParagraphStyle
from reportlab.lib import colors

# Konfigurasi Jalur Direktori
TEST_DIR = "static/test_images/"
CSV_FILE = "laporan_pengujian_batch.csv"
PDF_FILE = "laporan_pengujian_batch.pdf"

# Memuat Model Klasifikasi Terlatih
with open('model_rf.pkl', 'rb') as f:
    model_rf = pickle.load(f)
with open('model_inn.pkl', 'rb') as f:
    model_inn = pickle.load(f)

def extract_fused_features(image_path):
    img = cv2.imread(image_path)
    if img is None:
        return None
    img_resized = cv2.resize(img, (256, 256))

    # Ekstraksi Fitur Warna HSV
    img_hsv = cv2.cvtColor(img_resized, cv2.COLOR_BGR2HSV)
    h_mean = np.mean(img_hsv[:, :, 0])
    s_mean = np.mean(img_hsv[:, :, 1])
    v_mean = np.mean(img_hsv[:, :, 2])

    # Ekstraksi Fitur Tekstur GLCM
    img_gray = cv2.cvtColor(img_resized, cv2.COLOR_BGR2GRAY)
    glcm = graycomatrix(img_gray, distances=[1], angles=[0], levels=256, symmetric=True, normed=True)
    contrast = graycoprops(glcm, 'contrast')
    correlation = graycoprops(glcm, 'correlation')
    energy = graycoprops(glcm, 'energy')
    homogeneity = graycoprops(glcm, 'homogeneity')

    return [h_mean, s_mean, v_mean, contrast, correlation, energy, homogeneity]

# 1. Bagian Eksekusi Pengujian Batch dan Penulisan ke CSV
def run_batch_test():
    with open(CSV_FILE, mode='w', newline='') as file:
        writer = csv.writer(file)
        writer.writerow()

        for filename in sorted(os.listdir(TEST_DIR)):
            if filename.lower().endswith(['.jpg', '.jpeg', '.png']):
                ground_truth = "Early Blight" if "Early-B" in filename else "Late Blight"
                filepath = os.path.join(TEST_DIR, filename)

                features = extract_fused_features(filepath)
                if features is None:
                    continue

                features_arr = np.array(features).reshape(1, -1)

                # Prediksi menggunakan Random Forest
                pred_rf = model_rf.predict(features_arr)
                prob_rf = np.max(model_rf.predict_proba(features_arr)) * 100

                # Prediksi menggunakan 1-NN
                pred_inn = model_inn.predict(features_arr)

                status_rf = "BENAR" if pred_rf == ground_truth else "SALAH"
                writer.writerow([filename, ground_truth, pred_rf, f"{prob_rf:.2f}", pred_inn, status_rf])

# 2. Bagian Pembuatan CSV dan Konversi Menjadi PDF Basic
def export_csv_to_pdf():
    doc = SimpleDocTemplate(PDF_FILE, pagesize=letter, rightMargin=30, leftMargin=30, topMargin=30, bottomMargin=30)
    styles = getSampleStyleSheet()
    title_style = ParagraphStyle(
        'TitleStyle', parent=styles['Section1'], fontName='serif', fontStyle='italic', textColor=colors.HexColor('#808080'), alignment='center'
    )
    normal_style = styles['Normal']

    story.append(Paragraph("LAPORAN HASIL PENGUJIAN BATCH SISTEM KLASIFIKASI", title_style))
    story.append(Paragraph("Analisis Pengujian Batch Sistem Klasifikasi Daun Kentang Berbasis Fitur HSV & GLCM", ParagraphStyle('Sub', parent=title_style, fontName='serif', fontStyle='italic', fontColor=colors.gray)))
    story.append(Spacer(1, 10))

    # Membaca Data dari CSV
    table_data = []
    with open(CSV_FILE, mode='r') as file:
        reader = csv.reader(file)
        headers = next(reader)
        table_data.append(Paragraph("\n".join(headers), normal_style))
        for row in reader:
            table_data.append(Paragraph("\n".join(row), normal_style))

    # Membuat Tabel dalam PDF
    report_table = Table(table_data, colWidths=[100, 100, 100, 100, 100, 100])
    report_table.setStyle(TableStyle([
        ('colspan', 6, 1, 6, 'Table-Header'),
    ]))
    story.append(report_table)
    doc.build(story)

if __name__ == "__main__":
    run_batch_test()
    export_csv_to_pdf()
```

Gambar 8. Implementasi evaluasi massal citra daun kentang dan pembuatan laporan PDF

```

import pandas as pd

# Membaca file hasil pengujian batch
df = pd.read_csv('laporan_pengujian_batch.csv')

# Menghitung akurasi Random Forest
total_samples = len(df)
correct_rf = df.value_counts().get('BENAR', 0)
accuracy_rf = (correct_rf / total_samples) * 100

# Menghitung akurasi 1-NN
correct_1nn = (df == df['Prediksi 1-NN']).sum()
accuracy_1nn = (correct_1nn / total_samples) * 100

print(f"Total Sampel Uji: {total_samples}")
print(f"Prediksi Benar RF: {correct_rf} ({accuracy_rf:.2f}%)")
print(f"Prediksi Benar 1-NN: {correct_1nn} ({accuracy_1nn:.2f}%)")
    
```

Gambar 9. Implementasi kode evaluasi kinerja dan akurasi pengujian batch

Hasil Pengujian Batch

Berkas laporan_pengujian_batch.csv merekam 80 citra uji, terdiri atas 40 citra *Early Blight* dan 40 citra *Late Blight*. Model *Random Forest* memprediksi 76 citra dengan benar sehingga akurasinya 95,00% (galat 5,00%), sedangkan 1-NN memprediksi 75 citra dengan benar sehingga akurasinya 93,75% (galat 6,25%). Tabel 3 menyajikan cuplikan 14 baris laporan tersebut, yang memuat seluruh citra yang salah diklasifikasikan oleh sedikitnya satu model.

Tabel 3. Cuplikan hasil pengujian batch (laporan_pengujian_batch.csv)

No	Berkas (ID dan nomor citra)	Label aktual	Prediksi RF	Keyakinan RF (%)	Prediksi 1-NN	Status RF	Status 1-NN
1	2cc3eb24 (Early-B-6829)	Early Blight	Early Blight	85,00	Early Blight	Benar	Benar
2	2cc3eb24 (Early-B-6829)	Early Blight	Early Blight	93,00	Early Blight	Benar	Benar
3	2cce6cd6 (Early-B-8999)	Early Blight	Early Blight	87,00	Early Blight	Benar	Benar
4	2ce7e8d7 (Early-B-8609)	Early Blight	Early Blight	81,00	Late Blight	Benar	Salah
5	2d149f7a (Early-B-9143)	Early Blight	Late Blight	51,00	Early Blight	Salah	Benar
6	2d344be7 (Early-B-7173)	Early Blight	Early Blight	94,00	Early Blight	Benar	Benar
7	3ab17a89 (Early-B-8636)	Early Blight	Early Blight	53,00	Late Blight	Benar	Salah
8	b4e52186 (LB-2655)	Late Blight	Late Blight	54,00	Late Blight	Benar	Benar
9	b9cde1e2 (LB-2610)	Late Blight	Early Blight	54,00	Late Blight	Salah	Benar
10	bb19c5f3 (LB-4950)	Late Blight	Late Blight	100,00	Late Blight	Benar	Benar
11	c58de13c (LB-3116)	Late Blight	Early Blight	53,00	Late Blight	Salah	Benar
12	d4463dcB (LB-2788)	Late Blight	Early Blight	54,00	Early Blight	Salah	Salah
13	d98195f0 (LB-4790)	Late Blight	Late Blight	54,00	Early Blight	Benar	Salah
14	dd9b9122 (LB-5290)	Late Blight	Late Blight	75,00	Early Blight	Benar	Salah

Catatan: ID berkas disingkat dari delapan karakter pertama nama berkas PlantVillage. Status 1-NN ditentukan dengan membandingkan label aktual dan prediksi 1-NN.

Confusion matrix kedua model disajikan pada Tabel 4. *Random Forest* salah mengklasifikasikan satu citra *Early Blight* sebagai *Late Blight* dan tiga citra *Late Blight* sebagai *Early Blight*. Model 1-NN salah mengklasifikasikan dua citra *Early Blight* dan tiga citra *Late Blight*. Perhitungan metrik per kelas berdasarkan Tabel 4 disajikan pada Tabel 5.

Tabel 4. *Confusion matrix* komparatif *Random Forest* dan 1-NN pada 80 citra uji

Algoritma	Kelas aktual	Prediksi <i>Early Blight</i>	Prediksi <i>Late Blight</i>	Total aktual	Akurasi kelas (%)
<i>Random Forest</i>	<i>Early Blight</i>	39	1	40	97,50
	<i>Late Blight</i>	3	37	40	92,50
1-NN	<i>Early Blight</i>	38	2	40	95,00
	<i>Late Blight</i>	3	37	40	92,50

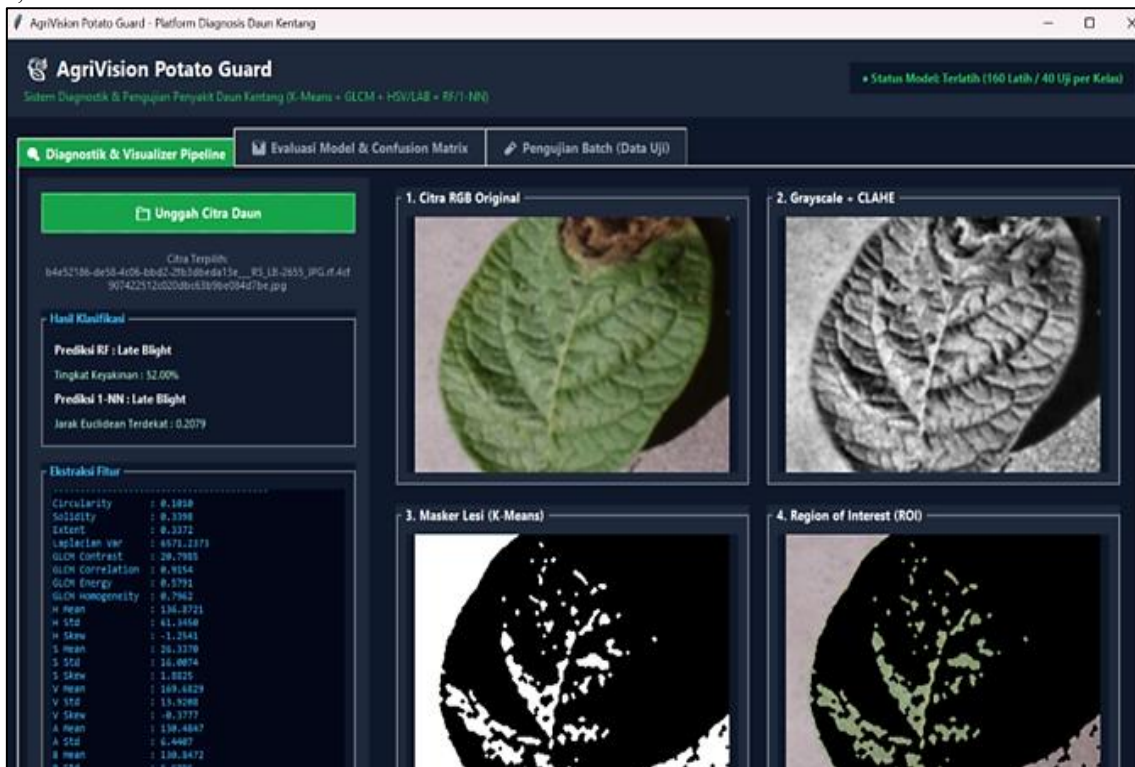
Tabel 5. Presisi, *recall*, dan *F1-score* per kelas (dihitung dari Tabel 4)

Algoritma	Kelas	Presisi	<i>Recall</i>	<i>F1-score</i>
<i>Random Forest</i>	<i>Early Blight</i>	92,86%	97,50%	95,12%
	<i>Late Blight</i>	97,37%	92,50%	94,87%
	Rata-rata makro	95,11%	95,00%	95,00%
1-NN	<i>Early Blight</i>	92,68%	95,00%	93,83%
	<i>Late Blight</i>	94,87%	92,50%	93,67%
	Rata-rata makro	93,78%	93,75%	93,75%

Dengan interval kepercayaan Wilson 95%, akurasi *Random Forest* berada pada rentang 87,8% sampai 98,0%, sedangkan akurasi 1-NN berada pada rentang 86,2% sampai 97,3%. Kedua interval saling tumpang tindih hampir seluruhnya.

Hasil Pengujian pada Antarmuka Sistem

Pengujian langsung pada antarmuka aplikasi menggunakan sampel citra daun terinfeksi menunjukkan bahwa kedua model memberikan prediksi yang sama, yaitu *Late Blight* (Gambar 10 dan Tabel 6). *Random Forest* memberikan keyakinan 94,00%, sedangkan 1-NN melaporkan jarak Euclidean 0,0000.



Gambar 10. Hasil diagnosis penyakit daun kentang menggunakan model *Random Forest* dan 1-NN pada antarmuka aplikasi

Tabel 6. Keluaran prediksi model pada citra sampel di antarmuka aplikasi

Algoritma	Prediksi kelas	Keyakinan atau jarak	Status
<i>Random Forest</i>	<i>Late Blight</i>	Probabilitas 94,00%	Valid
1-NN	<i>Late Blight</i>	Jarak Euclidean 0,0000	Valid

Pembahasan: Fusi Fitur Warna dan Tekstur

Akurasi 95,00% pada vektor fitur yang hanya berdimensi tujuh menunjukkan bahwa informasi warna rata-rata dan tekstur GLCM sudah cukup informatif untuk membedakan *Early Blight* dan *Late Blight* pada data yang diuji. Secara konseptual, kedua jenis fitur saling melengkapi. Fitur HSV menangkap perubahan pigmen akibat nekrosis dan klorosis, dengan komponen *Value* yang terpisah dari kromatisitas sehingga mengurangi pengaruh bayangan, tetapi pada tahap awal infeksi kedua penyakit sama-sama tampak kecokelatan sehingga rentang warnanya dapat tumpang tindih. Fitur GLCM menambahkan informasi spasial: lesi *Early Blight* dengan cincin konsentris cenderung meningkatkan kontras dan menurunkan homogenitas, sedangkan lesi basah *Late Blight* cenderung berpola lebih halus. Pola ini sejalan dengan temuan penelitian terdahulu yang melaporkan manfaat penambahan fitur warna pada fitur tekstur untuk klasifikasi penyakit daun kentang (Bulan dan Nurtanio, 2025; Muchtar, 2024).

Perlu ditegaskan bahwa penelitian ini tidak membandingkan secara langsung kinerja fitur warna saja, fitur tekstur saja, dan fitur gabungan (*ablation study*). Dengan demikian, kontribusi masing-masing kelompok fitur terhadap akurasi 95,00% belum dapat dikuantifikasi, dan keunggulan fusi dibandingkan fitur tunggal masih berupa dugaan yang didukung literatur, bukan hasil yang dibuktikan secara eksperimental dalam studi ini. Perbandingan langsung dengan akurasi studi lain (misalnya 97,78% pada Bulan dan Nurtanio, 2025, atau 98,46% pada Muchtar, 2024) juga tidak setara karena perbedaan dataset, jumlah kelas, dan jumlah fitur.

Pembahasan: Perbandingan Random Forest dan 1-NN

Random Forest memperoleh akurasi 95,00% dan 1-NN 93,75%, atau selisih satu citra dari 80 citra uji. Selisih sekecil ini berada jauh di dalam rentang ketidakpastian yang ditunjukkan oleh interval Wilson di atas, sehingga belum cukup untuk menyatakan bahwa salah satu algoritma lebih unggul secara statistik. Secara metodologis, *Random Forest* memiliki keunggulan teoretis berupa pemungutan suara mayoritas dari banyak pohon yang menurunkan varians, sedangkan 1-NN hanya bertumpu pada satu tetangga terdekat sehingga lebih sensitif terhadap pencilon lokal (Breiman, 2001; Cover dan Hart, 1967). Pola kesalahan kedua model juga berbeda: *Random Forest* dan 1-NN hanya sama-sama salah pada satu citra (d4463dcb), sementara 1-NN memperbaiki tiga kesalahan *Random Forest* (2d149f7a, b9cde1e2, dan c58de13c) tetapi menambah empat kesalahan lain (2ce7e8d7, 3ab17a89, d98195f0, dan dd9b9122). Hal ini menunjukkan bahwa kedua model menangkap batas keputusan yang sedikit berbeda, sehingga penampilan keduanya berdampingan pada antarmuka dapat membantu pengguna menilai kepercayaan terhadap suatu diagnosis.

Pembahasan: Analisis Kasus Salah Klasifikasi

Keempat kesalahan *Random Forest* memiliki tingkat keyakinan yang rendah, yaitu 51,00% sampai 54,00% (Tabel 3), yang mengindikasikan bahwa sampel tersebut berada di dekat batas keputusan. Citra 2d149f7a (*Early Blight*) diprediksi sebagai *Late Blight* dengan keyakinan 51,00%; berdasarkan penelusuran visual, serangan pada daun tersebut cukup parah sehingga bercak konsentris menyatu, kehilangan pola cincin yang menjadi ciri tekstur kasar *Early Blight*, dan warna daun bergeser ke arah abu-abu kecokelatan. Pada tiga citra *Late Blight* (b9cde1e2, c58de13c, dan d4463dcb), lesi diduga masih kecil sehingga belum cukup dominan mengubah nilai rata-rata HSV pada citra 256×256 piksel, sementara kontras GLCM dari lesi kecil menyerupai bintik-bintik pada *Early Blight*. Keterbatasan ini wajar karena fitur rata-rata dan GLCM pada seluruh citra bersifat global dan tidak membedakan area lesi dari latar maupun jaringan sehat.

Perlu dicatat pula bahwa tingkat keyakinan rendah tidak secara otomatis berarti salah, karena prediksi yang benar pada citra 3ab17a89 (53,00%) dan b4e52186 (54,00%) juga berkeyakinan rendah. Walaupun demikian, seluruh kesalahan *Random Forest* terkumpul pada rentang keyakinan rendah ini, sehingga ambang keyakinan (misalnya di bawah 60%) dapat dimanfaatkan sebagai penanda agar diagnosis diverifikasi ulang oleh penyuluh atau ahli. Penelusuran lanjutan melalui segmentasi area lesi (*region of interest*) sebelum ekstraksi fitur sangat disarankan untuk mengurangi pengaruh latar belakang dan jaringan sehat.

Pembahasan: Efisiensi dan Implikasi Praktis

Ekstraksi fitur pada penelitian ini hanya melibatkan perhitungan rata-rata kanal HSV dan empat properti GLCM pada citra 256×256 piksel, ditambah inferensi model yang berukuran kecil (berkas model_rf.pkl kurang dari dua megabita). Operasi sederhana ini memungkinkan sistem dijalankan pada komputer berspesifikasi rendah tanpa GPU, berbeda dengan pendekatan *deep learning* yang umumnya membutuhkan sumber daya komputasi lebih besar (Alhammad dkk., 2025). Namun, penelitian ini tidak

mengukur waktu komputasi secara formal, sehingga klaim efisiensi di sini bersifat kualitatif. Pengemasan model klasik menjadi aplikasi web dengan pelaporan pengujian batch otomatis sejalan dengan pendekatan implementasi web pada klasifikasi teks yang dilaporkan Kamila Insani P dan Tanjung (2026): model yang ringan dan antarmuka sederhana memudahkan pemanfaatan oleh pengguna non-teknis, dalam kasus ini petani dan penyuluh pertanian.

Keterbatasan Penelitian

Beberapa keterbatasan perlu dipertimbangkan dalam menafsirkan hasil. Pertama, penelitian hanya melibatkan dua kelas penyakit (*Early Blight* dan *Late Blight*) tanpa kelas daun sehat, dan hanya satu sumber data. Citra PlantVillage diambil dalam kondisi terkontrol (Hughes dan Salathé, 2015), sehingga kemampuan generalisasi pada citra lapangan dengan latar kompleks belum teruji. Kedua, jumlah citra uji hanya 80 dan evaluasi dilakukan pada satu pembagian data tanpa validasi silang, sehingga estimasi akurasi memiliki ketidakpastian yang cukup lebar. Ketiga, jarak Euclidean 0,0000 pada pengujian antarmuka (Tabel 6) berarti vektor fitur citra uji identik dengan vektor sebuah citra latih; hal ini mengindikasikan bahwa citra tersebut kemungkinan juga terdapat pada data latih, sehingga hasil pada sampel tersebut tidak dapat dianggap sebagai uji independen, dan kemungkinan tumpang tindih serupa pada citra uji lainnya perlu dikendalikan pada penelitian berikutnya. Keempat, tidak dilakukan studi ablasi untuk fitur tunggal versus fitur gabungan, dan hanya satu sudut GLCM (0°) yang digunakan. Kelima, segmentasi area lesi tidak diterapkan pada sistem yang diimplementasikan; fitur dihitung dari seluruh citra.

KESIMPULAN

Penelitian ini telah merancang dan mengimplementasikan algoritma fusi fitur tekstur GLCM (kontras, korelasi, energi, dan homogenitas pada sudut 0° dengan jarak satu piksel) dan warna HSV (rata-rata kanal *Hue*, *Saturation*, dan *Value*) melalui konkatenasi vektor satu dimensi menjadi vektor tujuh dimensi, yang dijalankan pada aplikasi web berbasis Flask. Pada pengujian 80 citra daun kentang (40 *Early Blight* dan 40 *Late Blight*), *Random Forest* mencapai akurasi 95,00% (akurasi kelas *Early Blight* 97,50% dan *Late Blight* 92,50%) dan 1-NN mencapai akurasi 93,75%. Selisih satu citra antara kedua model belum cukup untuk menyimpulkan keunggulan salah satunya, dan kesalahan klasifikasi terkonsentrasi pada citra dengan lesi sangat kecil atau lesi yang telah menyatu serta berkeyakinan rendah (51,00%–54,00%). Hasil ini menunjukkan bahwa fusi fitur ringkas yang ringan layak dijadikan pendekatan awal untuk klasifikasi *Early Blight* dan *Late Blight*, dengan catatan bahwa kontribusi fusi terhadap fitur tunggal belum diuji dan generalisasi pada kondisi lapangan belum dibuktikan.

Penelitian selanjutnya disarankan untuk (1) menambahkan kelas daun sehat dan penyakit lain serta menguji pada citra lapangan, (2) memperbesar dan memisahkan data latih dan uji secara ketat serta menerapkan validasi silang, (3) melakukan studi ablasi fitur dan menambah sudut GLCM, (4) menerapkan segmentasi area lesi sebelum ekstraksi fitur, (5) mengevaluasi seleksi atau reduksi fitur seperti PCA dan model ansambel lain, serta (6) mengembangkan aplikasi seluler yang dapat berjalan luring untuk penggunaan di lahan pertanian.

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada Program Studi Sistem Informasi, Fakultas Ilmu Komputer dan Teknologi Informasi, Universitas Muhammadiyah Sumatera Utara, atas dukungan akademik dan fasilitas selama pelaksanaan penelitian ini.

DAFTAR PUSTAKA

- Alhammad, S. M., Khafaga, D. S., El-hady, W. M., Samy, F. M., & Hosny, K. M. (2025). Deep learning and explainable AI for classification of potato leaf diseases. *Frontiers in Artificial Intelligence*, 7, 1449329. <https://doi.org/10.3389/frai.2024.1449329>
- Breiman, L. (2001). Random forests. *Machine Learning*, 45(1), 5–32. <https://doi.org/10.1023/A:1010933404324>
- Bulan, A. S., & Nurtanio, I. (2025). Classification of leaf disease in potato plants based on color and texture features using the K nearest neighbor (KNN) method. *Proceedings of the 1st International Conference on Research and Innovations in Information and Engineering Technology (RITECH 2025)*, 61–67. SCITEPRESS. <https://doi.org/10.5220/0014272800004928>
- Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1), 21–27. <https://doi.org/10.1109/TIT.1967.1053964>
- Haralick, R. M., Shanmugam, K., & Dinstein, I. (1973). Textural features for image classification. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-3(6), 610–621. <https://doi.org/10.1109/TSMC.1973.4309314>

- Hughes, D. P., & Salathé, M. (2015). An open access repository of images on plant health to enable the development of mobile disease diagnostics. *arXiv*. <https://doi.org/10.48550/arXiv.1511.08060>
- Kamila Insani P, & Tanjung, M. A. P. (2026). Klasifikasi ujaran kebencian teks bahasa gaul terkait isu Israel menggunakan Support Vector Machine. *Indonesian Journal of Multidisciplinary Scientific Studies*, 4(5), 336–344. <https://doi.org/10.33151/ijomss.v4i5.1116>
- Li, Q., Feng, Y., Li, J., Hai, Y., Si, L., Tan, C., Peng, J., Hu, Z., Li, Z., Li, C., Hao, D., & Tang, W. (2024). Multi-omics approaches to understand pathogenicity during potato early blight disease caused by *Alternaria solani*. *Frontiers in Microbiology*, 15, 1357579. <https://doi.org/10.3389/fmicb.2024.1357579>
- Muchtar, R. A. (2024). Integrasi fitur warna, tekstur dan Renyi fraktal untuk klasifikasi penyakit daun kentang menggunakan Linear Discriminant Analysis. *Jurnal Mnemonic*, 7(1), 77–84. <https://doi.org/10.36040/mnemonic.v7i1.9258>
- Muhamad, H. (2024). Identifikasi penyakit daun kentang menggunakan fitur GLCM dan algoritma Multi-SVM. *CONTEN: Computer and Network Technology*, 4(1), 52–57. <https://doi.org/10.31294/conten.v4i1.3465>
- Nyasulu, C., Diattara, A., Traore, A., Ba, C., Diedhiou, P. M., Sy, Y., Raki, H., & Peluffo-Ordóñez, D. H. (2023). A comparative study of machine learning-based classification of tomato fungal diseases: Application of GLCM texture features. *Heliyon*, 9(11), e21697. <https://doi.org/10.1016/j.heliyon.2023.e21697>
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830. <https://jmlr.org/papers/v12/pedregosa11a.html>
- Shoaib, M., Shah, B., El-Sappagh, S., Ali, A., Ullah, A., Alenezi, F., Gechev, T., Hussain, T., & Ali, F. (2023). An advanced deep learning models-based plant disease detection: A review of recent research. *Frontiers in Plant Science*, 14, 1158933. <https://doi.org/10.3389/fpls.2023.1158933>
- van der Walt, S., Schönberger, J. L., Nunez-Iglesias, J., Boulogne, F., Warner, J. D., Yager, N., Gouillart, E., & Yu, T. (2014). scikit-image: Image processing in Python. *PeerJ*, 2, e453. <https://doi.org/10.7717/peerj.453>