Journal Computer and Technology
Vol. 3, No. 2, Desember 2025, hlm. 127-136
ISSN : 3048-1880 (Online)
DOI : https://doi.org/10.69916/comtechno.v3i2.414

KLASIFIKASI JENIS DAUN TANAMAN OBAT BERDASARKAN CITRA TEKSTUR MENGGUNAKAN *GRAY LEVEL CO OCCURRENCE MATRIX* (GLCM) DAN ALGORITMA *K-NN*

Anakanda Bungsu Panogari Lubis[1], Siti Sundari[*2]

[1,2]Universitas Harapan Medan, Fakultas Teknik dan Komputer, Teknik Informatika
Email: [1]anakanda@gmail.com, [2]sitisundaristth@gmail.com

SEJARAH ARTIKEL
Diterima: 22.12.2025
Direvisi: 30.12.2025
Publish: 31.12.2025

ABSTRAK

Penentuan jenis daun tanaman obat yang cepat dan konsisten penting untuk menjamin mutu pemanfaatan fitofarmaka, sementara identifikasi manual rentan bias dan memakan waktu. Penelitian ini merancang sistem klasifikasi berbasis citra tekstur dengan ekstraksi fitur *Gray Level Co occurrence Matrix* dan pengklasifikasi *K-NN*. Alur kerja mencakup praproses citra menjadi grayscale, penyetaraan ukuran 128×128, pembentukan GLCM pada empat orientasi, serta perhitungan enam properti tekstur sehingga dihasilkan vektor fitur yang merepresentasikan pola permukaan daun. Vektor ini diklasifikasikan menggunakan *K-NN* pada skenario multikelas. Dataset berisi seribu citra dari sepuluh kelas yang dibagi menjadi delapan ratus data latih dan dua ratus data uji. Implementasi dilakukan di Google Colab dengan antarmuka Gradio sehingga pengguna dapat mengunggah citra dan memperoleh hasil secara interaktif. Hasil awal menunjukkan akurasi tiga puluh empat persen yang menandakan sistem telah menangkap sebagian sinyal tekstur namun masih memerlukan peningkatan. Perbaikan yang disarankan meliputi kurasi dan augmentasi data, normalisasi pencahayaan, seleksi serta penimbangan fitur, penalaan parameter *K-NN*, dan eksplorasi pengklasifikasi lanjutan seperti SVM atau CNN agar kinerja meningkat pada variasi citra yang lebih luas. Sistem ini menjadi baseline yang sederhana, transparan, dan mudah direplikasi untuk pengembangan berikutnya.

Kata Kunci: klasifikasi citra, daun, tanaman obat, GLCM, k-nearest neighbor.

ABSTRACT

Determining the type of medicinal plant leaves quickly and consistently is important to ensure the quality of phytopharmaceuticals, while manual identification is prone to bias and time-consuming. This study designed an image texture-based classification system with Gray Level Co-occurrence Matrix feature extraction and K-NN classifier. The workflow includes preprocessing images into grayscale, resizing them to 128×128, forming GLCM in four orientations, and calculating six texture properties to produce feature vectors that represent leaf surface patterns. These vectors are classified using K-NN in a multi-class scenario. The dataset contains a thousand images from ten classes, divided into eight hundred training data and two hundred test data. The implementation was carried out in Google Colab with a Gradio interface so that users can upload images and obtain results interactively. Initial results show an accuracy of thirty-four percent, indicating that the system has captured some of the texture signals but still needs improvement. Suggested improvements include data curation and augmentation, lighting normalization, feature selection and weighting, K-NN parameter tuning, and exploration of advanced classifiers such as SVM or CNN to improve performance on a wider variety of images. This system serves as a simple, transparent, and easily replicable baseline for further development.

Keywords: *image classification, medicinal, plant leaves, GLCM, k-nearest neighbor.*

1. PENDAHULUAN

Pemrosesan citra digital merupakan bidang ilmu yang berkembang pesat, terutama dalam konteks analisis pola visual melalui teknik ekstraksi fitur. Salah satu pendekatan yang banyak digunakan dalam mengenali karakteristik permukaan objek adalah melalui analisis tekstur. Citra tekstur mengandung informasi spasial yang kompleks, dan teknik seperti *Gray Level Co-occurrence Matrix (GLCM*) memungkinkan perhitungan statistik hubungan antar-piksel untuk mendapatkan deskriptor tekstur yang representatif [1]–[4]. *GLCM* mampu mengekstrak fitur seperti energi, entropi, kontras, dan homogenitas, yang sangat berguna dalam proses klasifikasi berbasis citra [5].

Di sisi lain, daun tanaman obat memiliki keragaman morfologi yang kaya dan khas, menjadikannya objek ideal dalam penelitian klasifikasi berbasis visual. Indonesia, sebagai negara dengan keanekaragaman hayati yang tinggi, memiliki ratusan spesies tanaman obat tradisional yang digunakan dalam pengobatan alternatif. Identifikasi jenis daun tanaman obat secara akurat sangat penting untuk menjaga efektivitas dan keamanan penggunaan tanaman tersebut. Namun, proses identifikasi manual oleh pakar botani memerlukan waktu, tenaga, dan pengetahuan yang mendalam, sehingga dibutuhkan pendekatan otomatis yang efisien dan dapat diandalkan [6].

Permasalahan utama yang dihadapi adalah bagaimana mengembangkan sistem klasifikasi otomatis yang mampu mengenali jenis daun tanaman obat berdasarkan citra teksturnya. Kompleksitas tekstur pada permukaan daun, perbedaan kondisi pencahayaan, dan variasi posisi pengambilan gambar menjadi tantangan tersendiri dalam ekstraksi dan klasifikasi fitur citra. Selain itu, banyak metode klasifikasi yang tidak mampu bekerja optimal jika tidak didukung oleh ekstraksi fitur yang akurat dan efisien dari citra daun tersebut.

Sebagai solusi atas permasalahan tersebut, penelitian ini mengusulkan penggunaan kombinasi metode *GLCM* untuk ekstraksi fitur tekstur daun dan algoritma *k-Nearest Neighbor (K-NN)* sebagai klasifikator. Metode ini diimplementasikan secara interaktif menggunakan *Google Colab* untuk keperluan komputasi awan dan Gradio sebagai antarmuka pengguna berbasis web. Dengan pendekatan ini, diharapkan dapat dibangun sebuah sistem klasifikasi daun tanaman obat yang tidak hanya akurat, tetapi juga mudah diakses dan digunakan oleh pengguna tanpa latar belakang teknis yang mendalam [7].

Penelitian sebelumnya dilakukan oleh [8] Masalah utama dalam klasifikasi daun tanaman obat adalah kemiripan bentuk antar jenis yang menyebabkan rendahnya akurasi sistem klasifikasi konvensional. Penelitian ini mengatasi hal tersebut dengan menggunakan metode *GLCM* untuk ekstraksi ciri tekstur dan *K-NN* untuk klasifikasi. Hasil pengujian menunjukkan bahwa dengan nilai $k = 1$, sistem mencapai akurasi tertinggi sebesar 98%, membuktikan efektivitas pendekatan ini dalam meningkatkan akurasi dan mengurangi tingkat kesalahan klasifikasi.

2. METODE PENELITIAN

2.1 Citra

Citra digital adalah representasi visual dari suatu objek atau pemandangan yang disimpan dalam format diskret. Citra ini terdiri dari kumpulan piksel, di mana setiap piksel memiliki nilai intensitas tertentu. Dalam pengolahan citra digital, ilmu komputer dan teknik elektro berfokus pada pemrosesan dan analisis gambar digital menggunakan algoritma matematis. Tujuan utamanya adalah untuk meningkatkan kualitas visual citra, mengekstrak informasi penting, atau mempersiapkan data visual untuk proses lanjutan seperti klasifikasi dan deteksi pola. Secara praktis, pengolahan citra digital biasanya terbagi menjadi tiga tahapan utama yaitu pra-pemrosesan, analisis fitur, dan pasca-pemrosesan. Tahap pra-pemrosesan sangat penting karena berfungsi untuk menghasilkan citra yang lebih bersih dan tajam. Ini melibatkan teknik-teknik seperti pengurangan noise, konversi warna, segmentasi objek, dan peningkatan kontras. Kesalahan atau kekurangan pada tahap awal ini dapat berdampak signifikan pada akurasi hasil akhir. Salah satu aspek krusial dalam pengolahan citra adalah ekstraksi fitur, yaitu proses pengambilan informasi khas dari citra yang dapat digunakan untuk analisis lebih lanjut. Fitur-fitur ini bisa berupa bentuk, warna, tepi, atau tekstur. Misalnya, untuk menganalisis tekstur, metode seperti Gray Level Co-occurrence Matrix (GLCM) digunakan untuk menghitung hubungan spasial antar piksel dan mengekstrak informasi statistik seperti energi, homogenitas, dan kontras. Fitur-fitur ini sangat bermanfaat dalam berbagai aplikasi, termasuk klasifikasi objek dan pengenalan citra [9].

2.2 *GLCM*

Gray Level Co-occurrence Matrix (GLCM) adalah metode statistik yang digunakan dalam pengolahan citra untuk mengekstraksi fitur tekstur dengan cara menganalisis hubungan spasial antara pasangan piksel dalam suatu citra. Teknik ini pertama kali diperkenalkan oleh Haralick et al. (1973) dan menjadi salah satu pendekatan paling umum dalam ekstraksi ciri tekstur karena mampu merepresentasikan pola distribusi dan keteraturan dari intensitas piksel dalam sebuah gambar. Secara teknis, *GLCM* merepresentasikan seberapa sering sepasang piksel dengan intensitas tertentu (nilai abu-abu) muncul dalam jarak dan orientasi tertentu dalam citra. Matriks ini dibentuk dengan menghitung frekuensi pasangan nilai intensitas piksel (i, j) yang berada dalam suatu relasi spasial, seperti pada arah 0°, 45°, 90°, atau 135°, dan jarak tertentu (misalnya 1 piksel). Hasilnya adalah sebuah matriks simetris berukuran $N \times N$, di mana N adalah jumlah tingkat keabuan (gray level) dalam citra [10]. Rumus Umum *GLCM* yaitu :

1. *GLCM* pada posisi relatif $(\Delta x, \Delta y)$ didefinisikan sebagai:

$$P(i, j \mid \Delta x, \Delta y) = \text{jumlah pasangan piksel } (i, j) \text{ yang terpisah oleh } (\Delta x, \Delta y) \qquad (1)$$

2. Nilai matriks kemudian biasanya dinormalisasi menjadi:

$$P_{norm}(i,j) = \frac{P(i,j)}{\sum_{i=0}^{N-1}\sum_{j=0}^{N-1} P(i,j)} \quad (2)$$

Setelah *GLCM* terbentuk, beberapa fitur statistik dapat dihitung darinya:

1. Energy (Angular Second Moment):

$$\text{Energy} = \sum_i \sum_j P\,(i,j)^2 \quad (3)$$

Menunjukkan keseragaman. Nilai tinggi berarti tekstur lebih homogen.

2. Contrast:

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

Mengukur perbedaan lokal dalam citra. Semakin besar nilai, semakin kasar tekstur.

3. Homogeneity (Inverse Difference Moment):

$$\text{Homogeneity} = \sum_i \sum_j \frac{P(i,j)}{1+|i-j|} \quad (5)$$

Mengukur keseragaman distribusi nilai piksel. Nilai tinggi menunjukkan tekstur halus.

4. Entropy:

$$\text{Entropy} = -\sum_i \sum_j P\,(i,j) \cdot \log_2(P(i,j) + \varepsilon) \quad (6)$$

Mengukur kompleksitas informasi. Nilai tinggi menunjukkan ketidakteraturan atau kekacauan.

2.3 *K-NN*

K-Nearest Neighbor (K-NN) adalah algoritma klasifikasi non-parametrik yang digunakan dalam pembelajaran mesin (machine learning) untuk mengklasifikasikan objek berdasarkan kedekatan atau kemiripan dengan data lain yang sudah diberi label. *K-NN* termasuk dalam metode pembelajaran berbasis instance (lazy learning), di mana proses pelatihan tidak dilakukan secara eksplisit, melainkan seluruh data latih disimpan dan digunakan langsung saat proses klasifikasi. Konsep dasar dari *K-NN* adalah bahwa suatu objek akan diklasifikasikan ke dalam kelas mayoritas dari k tetangga terdekatnya di ruang fitur. Kedekatan antar data diukur menggunakan fungsi jarak (distance metric), seperti Euclidean distance, Manhattan distance, atau Minkowski distance, tergantung pada kebutuhan dan karakteristik data. Rumus Jarak Euclidean (yang paling umum digunakan dalam *K-NN*):

$$d(p,q) = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2} \quad (7)$$

Keterangan:
p dan q adalah dua vektor data dalam ruang berdimensi n
p_i dan q_i adalah komponen ke-i dari vektor p dan q
$d(p,q)$ adalah jarak Euclidean antara p dan q
.

3. HASIL DAN PEMBAHASAN

Tahap ini menjelaskan hasil implementasi dan pengujian sistem klasifikasi citra daun herbal menggunakan metode *Gray Level Co-occurrence Matrix (GLCM*) dan algoritma *K-Nearest Neighbor (K-NN)* pada platform *Google Colab* dengan antarmuka interaktif Gradio. Proses ini bertujuan untuk mengidentifikasi jenis daun berdasarkan pola tekstur citra grayscale yang telah diproses dan diklasifikasikan secara otomatis. Sistem dikembangkan dengan alur kerja yang terstruktur dari mulai praproses hingga visualisasi evaluasi hasil. Adapun tahapan pelaksanaan sistem sebagai berikut:

1. Praproses Citra

   Tahap praproses citra merupakan langkah awal yang krusial dalam sistem klasifikasi citra, karena bertujuan untuk menyiapkan data visual agar memiliki format dan kualitas yang konsisten sebelum dilakukan ekstraksi fitur. Pada tahap ini, citra daun diolah sedemikian rupa sehingga siap digunakan dalam proses perhitungan tekstur dan klasifikasi.

```
def extract_glcm_features(image):
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    resized = cv2.resize(gray, (128, 128))
```

Gambar 1. Kodingan Praproses Citra

   Potongan kode yang ditampilkan pada gambar 4.1 merupakan bagian dari fungsi extract_*GLCM* _features(image) yang menangani tahap praproses citra dalam sistem klasifikasi daun herbal. Pada tahap ini, citra masukan terlebih dahulu dikonversi dari format BGR (Blue, Green, Red) ke format grayscale menggunakan fungsi cv2.cvtColor(image, cv2.COLOR_BGR2GRAY). Konversi ini bertujuan untuk menyederhanakan informasi visual dari citra dengan menghilangkan komponen warna, sehingga hanya menyisakan intensitas keabuan yang lebih relevan untuk analisis tekstur. Setelah dikonversi ke grayscale, citra kemudian diubah ukurannya menjadi 128×128 piksel menggunakan cv2.resize(gray, (128, 128)). Proses resize ini dilakukan agar

seluruh citra memiliki dimensi yang seragam, sehingga mempermudah dan mengefisienkan proses ekstraksi fitur *GLCM* di tahap berikutnya.

2. Ekstrasi Fitur *GLCM*

Setelah citra melalui tahap praproses, langkah selanjutnya adalah melakukan ekstraksi fitur menggunakan metode *Gray Level Co-occurrence Matrix (GLCM*). Tahap ini bertujuan untuk memperoleh representasi numerik dari tekstur citra grayscale, yang nantinya digunakan sebagai input dalam proses klasifikasi.

```
glcm = graycomatrix(resized, distances=[1], angles=[0, np.pi/4, np.pi/2, 3*np.pi/4],
                    levels=256, symmetric=True, normed=True)
props = ['contrast', 'dissimilarity', 'homogeneity', 'energy', 'correlation', 'ASM']
features = []
for prop in props:
    features.extend(graycoprops(glcm, prop).flatten())
return features, resized
```

Gambar 2. Ekstrasi Fitur *GLCM*

Potongan kode pada gambar 4.2 merupakan bagian dari proses ekstraksi fitur tekstur menggunakan metode *Gray Level Co-occurrence Matrix (GLCM*). Pertama, matriks *GLCM* dihitung dari citra grayscale berukuran 128×128 piksel yang telah melalui tahap praproses, menggunakan fungsi graycomatrix. Parameter distances=[1] menunjukkan jarak antar piksel yang dianalisis, sedangkan angles=[0, np.pi/4, np.pi/2, 3*np.pi/4] menyatakan empat orientasi sudut (0°, 45°, 90°, dan 135°) untuk menangkap pola tekstur dari berbagai arah. *GLCM* kemudian dinormalisasi dan dibuat simetris agar hasil ekstraksi lebih stabil. Setelah *GLCM* dihitung, dilakukan perhitungan enam properti tekstur yang umum digunakan, yaitu: contrast, dissimilarity, homogeneity, energy, correlation, dan ASM (Angular Second Moment). Masing-masing properti dihitung dari *GLCM* menggunakan fungsi graycoprops, lalu hasilnya diratakan dengan flatten() dan ditambahkan ke dalam list features. Proses ini menghasilkan vektor numerik yang mewakili karakteristik tekstur dari citra input. Akhirnya, fungsi mengembalikan dua nilai: array features hasil ekstraksi dan citra grayscale hasil resize, yang dapat digunakan untuk keperluan klasifikasi dan visualisasi.

Gambar 3. Hasil Ekstrasi GLCM

Gambar itu menunjukkan daun yang sudah disegmentasi dari latar, kemudian diubah ke grayscale dan ditingkatkan kontrasnya (kemungkinan memakai CLAHE). Latar belakang hitam menandakan mask bekerja—hanya area daun yang dipertahankan, sehingga tekstur lamina (butiran epidermis) dan pola tulang daun (midrib/urat utama melintang di tengah dan urat sekunder yang menyirip) tampak jelas. Distribusi abu-abu yang bervariasi di sepanjang jaringan urat menonjolkan kontras lokal dan keseragaman/ketidakseragaman tekstur; kondisi ini ideal untuk ekstraksi fitur berbasis GLCM (mis. contrast, homogeneity, correlation) dan juga fitur bentuk/pola seperti HOG. Sorotan tipis di tepi daun serta gradasi pada permukaan menunjukkan adanya relief/ketebalan yang ikut menangkap pencahayaan, yang sering membantu membedakan spesies berurat tegas vs halus.

3. Pelatihan dan Pengujian Model *K-NN*

Setelah proses ekstraksi fitur selesai dilakukan, tahap berikutnya adalah pelatihan dan pengujian model klasifikasi menggunakan algoritma *K-Nearest Neighbor (K-NN)*. Dalam implementasi ini, nilai parameter k ditentukan sebesar 3, artinya model akan menentukan kelas dari citra uji berdasarkan tiga tetangga terdekat

dalam ruang fitur. Dataset yang telah diekstraksi dibagi menjadi dua bagian, yaitu data latih (training) dan data uji (testing). Model dilatih menggunakan data latih untuk membentuk representasi hubungan antar fitur dan label, lalu diuji terhadap data uji untuk mengevaluasi kemampuannya dalam melakukan klasifikasi jenis daun.

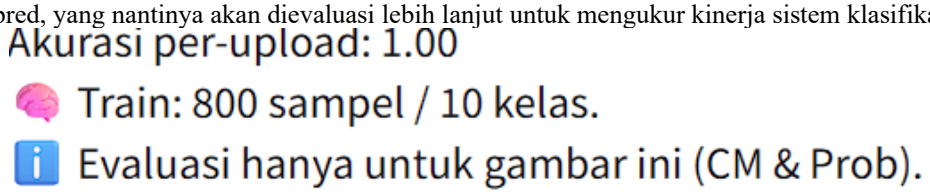
```
train_path = '/content/dataset/DATASET TANAMAN HERBAL/Data Training'
test_path = '/content/dataset/DATASET TANAMAN HERBAL/Data Testing'

X_train, y_train = load_dataset_from_folder(train_path)
X_test, y_test = load_dataset_from_folder(test_path)

# === 8. Train KNN Model ===
model = KNeighborsClassifier(n_neighbors=3)
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
```

Gambar 4. Kodingan Pelatihan dan Pengujian Model *K-NN*

Potongan kode pada gambar 4.4 menunjukkan proses pelatihan (training) dan pengujian (testing) model klasifikasi menggunakan algoritma *K-Nearest Neighbor (K-NN)*. Pertama, ditentukan jalur direktori untuk data latih (train_path) dan data uji (test_path) yang berada di dalam folder dataset tanaman herbal. Data kemudian dimuat dari masing-masing folder menggunakan fungsi load_dataset_from_folder, yang sebelumnya telah didefinisikan untuk membaca citra daun dan mengekstraksi fitur teksturnya menggunakan metode *GLCM* . Setelah data latih (X_train, y_train) dan data uji (X_test, y_test) tersedia, model *K-NN* dibentuk menggunakan KNeighborsClassifier dengan parameter n_neighbors=3, yang berarti model akan memprediksi label suatu data uji berdasarkan tiga tetangga terdekatnya dalam ruang fitur. Model kemudian dilatih menggunakan data latih melalui fungsi model.fit(), dan selanjutnya digunakan untuk memprediksi label data uji dengan model.predict(). Hasil prediksi ini disimpan dalam variabel y_pred, yang nantinya akan dievaluasi lebih lanjut untuk mengukur kinerja sistem klasifikasi.



Gambar 5. Hasil Pelatihan

Gambar 4.5 prediksi model benar dengan akurasi per upload sama dengan 1,00. Model dilatih menggunakan delapan ratus sampel yang mewakili sepuluh kelas daun. Keterangan bahwa evaluasi hanya untuk gambar ini menegaskan bahwa confusion matrix dan probabilitas yang tampil hanya mencerminkan kinerja pada satu citra sehingga tidak dapat dijadikan dasar untuk menilai kemampuan model secara umum. Untuk menilai ketahanan model terhadap variasi data diperlukan pengujian pada himpunan uji yang terpisah dari data latih atau setidaknya pengujian batch berisi banyak gambar unggahan beserta label kebenaran agar metrik seperti akurasi rata rata macro F1 dan confusion matrix penuh dapat dihitung dengan lebih jujur.

4. Visualisasi Hasil

   Setelah model selesai dilatih dan diuji, tahap selanjutnya adalah melakukan visualisasi hasil prediksi dan evaluasi performa sistem secara menyeluruh. Visualisasi ini bertujuan untuk memberikan gambaran yang lebih intuitif terhadap kinerja model, sementara evaluasi kuantitatif dilakukan untuk menilai sejauh mana model mampu mengklasifikasikan citra daun dengan akurat.

```python
# === 13. Gradio Interface ===
def classify_leaf(image):
    image_bgr = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)
    features, resized = extract_glcm_features(image_bgr)
    prediction = model.predict([features])[0]

    # Output info
    summary = f"""
📌 **Jenis Daun Terklasifikasi:** {prediction}

📦 **Jumlah Data:**
- Data Training : {len(X_train)}
- Data Testing  : {len(X_test)}

🎯 **Akurasi Model:** {acc*100:.2f}%
"""

    # Generate images
    grayscale_img = cv2.cvtColor(resized, cv2.COLOR_GRAY2RGB)
    knn_plot = plot_knn_prediction(features, prediction)
    report_img = plot_report_table(report_df.iloc[:-3])  # exclude avg rows
    conf_img = plot_conf_matrix(conf_matrix, np.unique(y_test))

    return summary, image, grayscale_img, knn_plot, report_img, conf_img

gr.Interface(
    fn=classify_leaf,
    inputs=gr.Image(type="numpy", label="Unggah Citra Daun"),
    outputs=[
        gr.Textbox(label="📄 Ringkasan Hasil"),
        gr.Image(label="🖼️ Citra Asli"),
        gr.Image(label="🔍 Citra Grayscale (128x128)"),
        gr.Image(label="📈 Visualisasi PCA K-NN"),
        gr.Image(label="📊 Tabel Laporan Klasifikasi"),
        gr.Image(label="🔢 Confusion Matrix")
    ],
    title="🌿 Klasifikasi Daun Herbal dengan GLCM + KNN",
    description="Model ini mengklasifikasikan jenis daun herbal berdasarkan tekstur citra menggunakan GLCM dan K-Nearest Neighbor (K=3)."
).launch()
```

Gambar 6. Kodingan Gradio

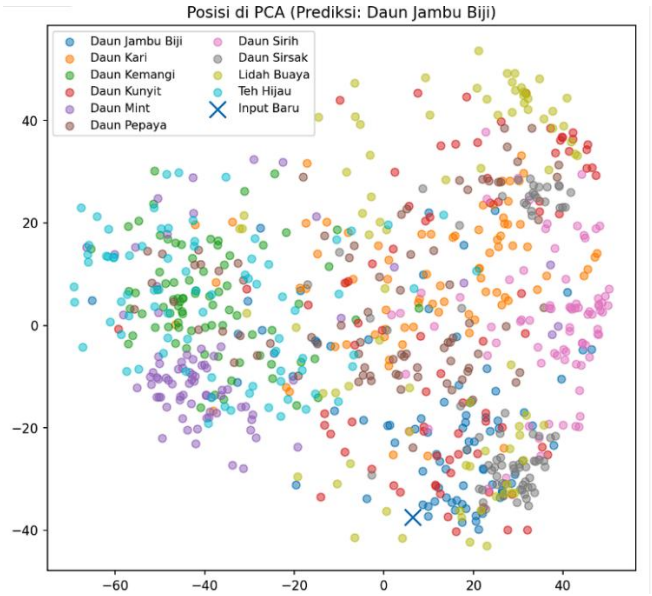
Potongan kode pada gambar 4.6 menampilkan bagian akhir dari sistem klasifikasi citra daun herbal, yaitu implementasi antarmuka interaktif menggunakan Gradio. Fungsi classify_leaf(image) dirancang untuk menerima input berupa citra daun dari pengguna. Citra tersebut pertama-tama dikonversi dari format RGB ke BGR menggunakan OpenCV agar sesuai dengan format pemrosesan sebelumnya. Selanjutnya, citra diproses melalui fungsi extract_*GLCM* _features untuk dilakukan ekstraksi fitur tekstur, lalu hasilnya digunakan oleh model *K-Nearest Neighbor (K-NN)* untuk memprediksi jenis daun. Setelah prediksi diperoleh, sistem menyiapkan beberapa informasi penting sebagai output, seperti label daun hasil klasifikasi, jumlah data training dan testing, serta akurasi model dalam bentuk persentase. Selain itu, untuk mendukung pemahaman visual pengguna, sistem menghasilkan beberapa gambar, yaitu: citra grayscale hasil resize, visualisasi distribusi data dan hasil prediksi dengan PCA, tabel evaluasi dalam bentuk classification report, serta confusion matrix dalam bentuk heatmap. Semua elemen ini kemudian diintegrasikan ke dalam antarmuka Gradio menggunakan gr.Interface(). Antarmuka ini menyediakan kolom unggah gambar, menampilkan hasil prediksi dalam bentuk teks dan visualisasi, serta menyertakan penjelasan mengenai fungsi sistem

```python
pca = PCA(n_components=2)
X_train_2D = pca.fit_transform(X_train)

def plot_knn_prediction(new_feat_1D, label_pred):
    new_2D = pca.transform([new_feat_1D])
    plt.figure(figsize=(6, 6))
    for i, label in enumerate(np.unique(y_train)):
        idx = y_train == label
        plt.scatter(X_train_2D[idx, 0], X_train_2D[idx, 1], label=label, alpha=0.5)
    plt.scatter(new_2D[:, 0], new_2D[:, 1], color='black', marker='x', s=100, label='Input Baru')
    plt.title(f"Visualisasi K-NN (Prediksi: {label_pred})")
    plt.legend()
    tmpfile = tempfile.NamedTemporaryFile(suffix='.png', delete=False)
    plt.savefig(tmpfile.name)
    plt.close()
    return tmpfile.name
```

Gambar 7. Kodingan PCA

Potongan gambar 4.8 merupakan bagian dari proses visualisasi hasil klasifikasi *K-NN* menggunakan metode Principal Component Analysis (PCA). PCA digunakan untuk mereduksi dimensi fitur hasil ekstraksi *GLCM* menjadi dua dimensi (2D), sehingga dapat divisualisasikan dalam bentuk grafik scatter plot. Variabel X_train_2D menyimpan hasil transformasi data latih yang telah direduksi oleh PCA. Fungsi plot_*K-NN*_prediction() kemudian digunakan untuk memvisualisasikan distribusi data pelatihan berdasarkan labelnya, serta menampilkan posisi data uji (input baru) yang sedang diprediksi. Di dalam fungsi tersebut, data input yang baru (new_feat_1D) juga ditransformasikan ke dalam ruang 2D menggunakan PCA. Selanjutnya, dengan menggunakan matplotlib, fungsi ini memplot seluruh data latih berdasarkan label kelasnya, dan memberi warna serta label berbeda untuk tiap kelas. Titik input baru yang sedang diuji ditampilkan secara mencolok dengan warna hitam dan simbol "x". Judul grafik menyertakan hasil prediksi label untuk input tersebut. Setelah selesai, visualisasi ini disimpan sementara dalam file gambar .png dan dikembalikan sebagai output fungsi.



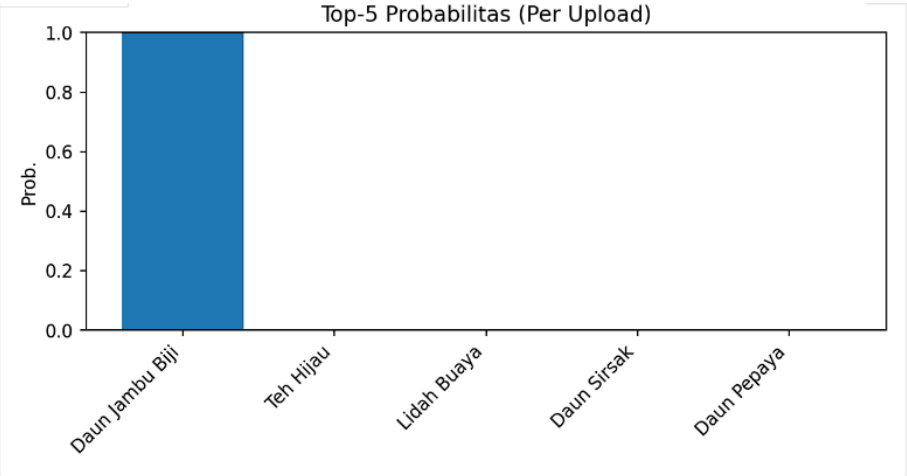
Gambar 8. Visualisasi PCA *K-NN*

Gambar 4.9 ditampilkan merupakan hasil proyeksi data fitur daun ke dalam dua dimensi menggunakan metode Principal Component Analysis atau PCA. Setiap titik berwarna merepresentasikan satu citra daun dari kelas tertentu, sementara warna yang berbeda sesuai dengan sepuluh kelas yang ada, seperti daun jambu biji, kari, kemangi, kunyit, mint, pepaya, sirih, sirsak, lidah buaya, dan teh hijau. Posisi titik biru besar berbentuk tanda silang menandai citra baru yang diuji. Pada plot ini model mengklasifikasikan citra tersebut sebagai daun jambu biji karena posisinya lebih dekat dengan kelompok titik berwarna biru muda yang memang mewakili kelas tersebut. Visualisasi ini tidak menggambarkan ruang asli berdimensi tinggi melainkan hasil reduksi sehingga ada informasi yang hilang. Meski demikian, pola kedekatan atau keterpisahan antar cluster masih memberi gambaran intuitif tentang seberapa mirip fitur dari daun uji dengan kelas tertentu.

```
def plot_report_table(df):
    fig, ax = plt.subplots(figsize=(8, len(df)*0.5))
    ax.axis('off')
    tbl = pd.plotting.table(ax, df.round(2), loc='center')
    tbl.auto_set_font_size(False)
    tbl.set_fontsize(10)
    tbl.scale(1, 1.5)
    tmp = tempfile.NamedTemporaryFile(delete=False, suffix=".png")
    plt.savefig(tmp.name, bbox_inches='tight')
    plt.close()
    return tmp.name
```

Gambar 9. Kodingan Klasifikasi Report

Potongan kode pada gambar menunjukkan fungsi plot_report_table(df) yang digunakan untuk menampilkan hasil evaluasi klasifikasi dalam bentuk tabel dan menyimpannya sebagai file gambar .png. Fungsi ini menerima parameter df, yaitu dataframe yang berisi hasil evaluasi dari model, seperti precision, recall, dan f1-score untuk masing-masing kelas. Pertama, fungsi membuat objek plot kosong menggunakan plt.subplots() dengan tinggi dinamis berdasarkan jumlah baris dalam dataframe. Sumbu (axis) dihilangkan untuk menampilkan tabel secara bersih tanpa garis koordinat. Tabel dibentuk menggunakan pd.plotting.table() dan diletakkan di tengah area plot. Nilai-nilai pada tabel dibulatkan menjadi dua angka di belakang koma, font diatur manual agar tidak terlalu kecil, dan skala tabel diperbesar agar lebih terbaca. Selanjutnya, tabel disimpan ke file gambar .png sementara menggunakan modul tempfile, sehingga hasilnya bisa langsung digunakan untuk ditampilkan di antarmuka Gradio. Setelah file disimpan, plot ditutup untuk menghemat memori, dan nama file dikembalikan sebagai output. Fungsi ini berperan penting dalam visualisasi laporan evaluasi model secara rapi dan mudah dibaca oleh pengguna.



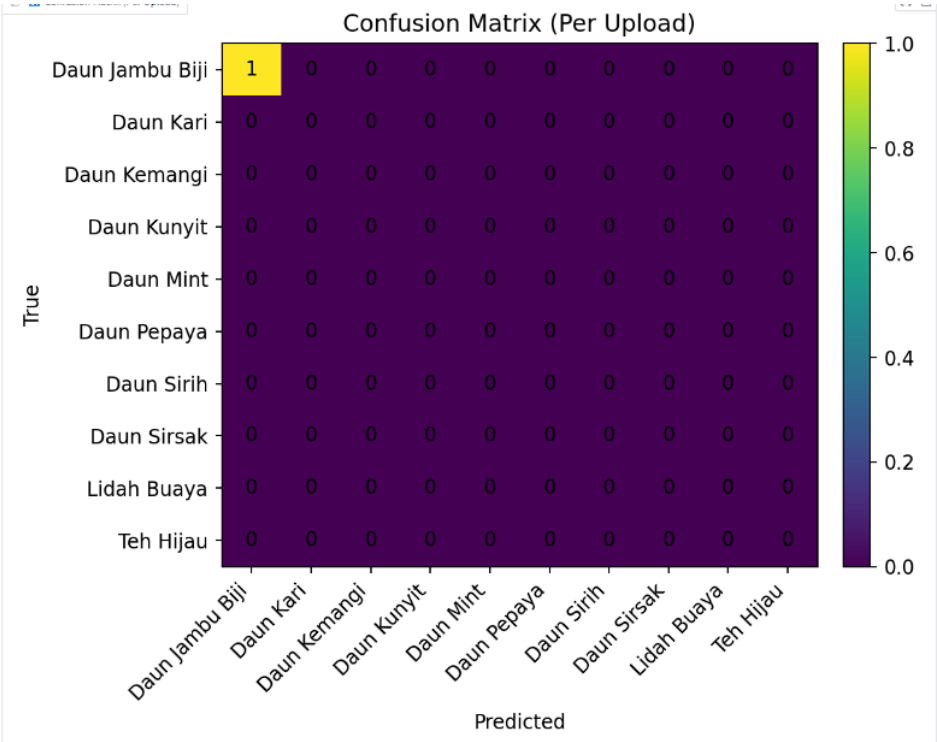
Gambar 10. Grafik Probalitias

Grafik tersebut memperlihatkan distribusi Top-5 probabilitas prediksi model untuk citra daun yang diunggah. Sumbu X menampilkan lima kelas daun yang paling mungkin menurut model, sedangkan sumbu Y menunjukkan nilai probabilitas. Pada hasil ini terlihat bahwa model memberikan probabilitas hampir sempurna, mendekati satu, pada kelas Daun Jambu Biji, sementara kelas lain seperti teh hijau, lidah buaya, daun sirsak, dan daun pepaya memiliki probabilitas nol. Hal ini berarti model sangat yakin bahwa citra yang diuji memang berasal dari daun jambu biji, dan tidak ada ambiguitas signifikan terhadap kelas lain. Keadaan ini mencerminkan bahwa fitur citra yang diekstraksi memiliki kesesuaian yang tinggi dengan pola tekstur, warna, dan bentuk yang selama pelatihan dipelajari dari kelas daun jambu biji.

```
def plot_conf_matrix(cm, labels):
    plt.figure(figsize=(8, 6))
    sns.heatmap(cm, annot=True, fmt='d', xticklabels=labels, yticklabels=labels, cmap="YlGnBu")
    plt.xlabel('Predicted')
    plt.ylabel('True')
    plt.title("Confusion Matrix")
    tmp = tempfile.NamedTemporaryFile(delete=False, suffix=".png")
    plt.savefig(tmp.name)
    plt.close()
    return tmp.name
```

Gambar 12. Kodingan Confusion Matrix

Potongan kode pada gambar 4.12 menunjukkan fungsi plot_conf_matrix(cm, labels), yang digunakan untuk membuat dan menyimpan visualisasi confusion matrix dalam bentuk gambar. Fungsi ini menerima dua parameter: cm yang merupakan confusion matrix hasil evaluasi model, dan labels yang merupakan daftar nama kelas daun yang digunakan sebagai label pada sumbu X dan Y grafik. Di dalam fungsi, dibuatlah visualisasi menggunakan seaborn.heatmap() dengan ukuran gambar 8x6 inci. Matriks divisualisasikan dalam bentuk peta panas (heatmap), di mana setiap kotak mewakili jumlah prediksi yang sesuai (atau salah) antara kelas sebenarnya (True) dan kelas yang diprediksi (Predicted). Nilai dalam setiap kotak ditampilkan secara eksplisit (annot=True) dengan format bilangan bulat (fmt='d'), dan skema warna YlGnBu digunakan untuk memberikan efek visual yang kontras. Judul dan label sumbu ditambahkan untuk memperjelas isi grafik. Setelah grafik selesai dibentuk, gambar disimpan sebagai file sementara berformat .png menggunakan modul tempfile, dan nama file dikembalikan sebagai output fungsi. Visualisasi confusion matrix ini sangat berguna untuk mengevaluasi performa klasifikasi model secara visual, terutama dalam melihat pola kesalahan antar kelas dan mengidentifikasi kelas mana yang sering tertukar.



Gambar 13. Confusion Matrix

Confusion matrix pada gambar tersebut menggambarkan hasil evaluasi untuk satu citra daun yang diuji. Matriks menampilkan kelas sebenarnya pada sumbu vertikal dan kelas hasil prediksi pada sumbu horizontal. Tampak bahwa hanya sel yang berkorespondensi dengan Daun Jambu Biji yang terisi dengan nilai satu, sedangkan seluruh sel lainnya bernilai nol. Hal ini berarti model berhasil mengklasifikasikan citra uji tersebut dengan benar ke kelas daun jambu biji, tanpa ada kesalahan ke kelas lain. Warna ungu yang mendominasi menandakan nilai nol, sementara kotak kuning dengan angka satu menunjukkan prediksi tepat pada posisi diagonal. Namun penting dicatat bahwa karena confusion matrix ini dihitung hanya dari satu gambar, informasi yang diperoleh masih sangat terbatas. Angka satu di diagonal memang menegaskan prediksi benar, tetapi tidak dapat digunakan untuk menilai performa model secara keseluruhan. Evaluasi yang lebih adil memerlukan banyak citra uji sehingga matriks terisi dengan distribusi nilai yang lebih beragam

4. KESIMPULAN

Berdasarkan hasil implementasi dan evaluasi sistem klasifikasi citra daun herbal, dapat disimpulkan bahwa ekstraksi fitur tekstur citra daun tanaman obat menggunakan metode *Gray Level Co-occurrence Matrix* (GLCM) berhasil dilakukan dengan menghasilkan 24 fitur numerik yang mampu merepresentasikan karakteristik tekstur setiap citra daun. Proses praproses berupa konversi citra ke grayscale dan penyeragaman ukuran menjadi 128×128 piksel terbukti mendukung konsistensi serta efektivitas ekstraksi fitur. Selanjutnya, penerapan algoritma *k-Nearest Neighbor* (K-NN) dengan parameter k=3 mampu mengklasifikasikan jenis daun tanaman obat berdasarkan fitur GLCM yang telah diekstraksi. Hasil evaluasi terhadap 1.000 citra yang terdiri dari 800 data latih dan 200 data uji menunjukkan akurasi sebesar 34%, yang mengindikasikan bahwa metode K-NN dapat berfungsi sebagai klasifikator dasar, namun masih memerlukan pengembangan lebih lanjut, baik dari sisi kualitas dataset maupun metode klasifikasi, agar kinerja sistem dapat ditingkatkan.

DAFTAR PUSTAKA

[1] K. Evandari, M. A. Soeleman, and R. A. Pramunendar, "BPNN Optimization With Genetic Algorithm For Classification of," *Rekayasa Sist. dan Teknol. Inf.*, vol. 5, no. 158, pp. 5–12, 2023.

[2] R. Mujahiddin, Zaeniah, and B. Imran, "Rancang Bangun Sistem Pakar Diagnosa Penyakit Pada Tanaman Cabai Dengan Metode Certainty Factor," *J. Kecerdasan Buatan dan Teknol. Inf.*, vol. 2, no. 1, pp. 11–19, 2023.

[3] D. Hananta Firdaus, B. Imran, L. Darmawan Bakti, and E. Suryadi, "Klasifikasi Penyakit Katarak Pada Mata Menggunakan Metode Convolutional Neural Network (Cnn) Berbasis Web," *J. Kecerdasan Buatan dan Teknol. Inf.*, vol. 1, no. 3, pp. 18–26, 2022.

[4] E. Wahyudi, B. Imran, S. Erniwati, M. N. Karim, I. Pemerintahan, and D. Negeri, "FINE-TUNING RESNET50V2 WITH ADAMW AND ADAPTIVE TRANSFER LEARNING FOR SONGKET

CLASSIFICATION IN LOMBOK," *Pilar Nusa Mandiri*, vol. 21, no. 1, pp. 82–91, 2025, doi: 10.33480/pilar.v21i1.6485.

[5] F. Marpaung, F. Aulia, and R. C. Nabila, "Computer Vision Dan Pengolahan Citra Digital." PUSTAKA AKSARA, 2022.

[6] A. Azizah, "Pengembangan Media Flashcard dalam Memperkenalkan Keanekaragaman Hayati Tanaman Obat." Jakarta: FITK UIN Syarif Hidayatullah Jakarta, 2022.

[7] A. Z. Alfarizi and E. I. Sela, "Klasifikasi Rimpang Menggunakan Metode K-Nearest Neighbor dan Ekstraksi Ciri Gray Level Co-occurrence Matrix," *J. FASILKOM*, vol. 14, no. 1, pp. 88–94, 2024.

[8] S. A. Rosiva Srg, M. Zarlis, and W. Wanayumini, "Identifikasi Citra Daun dengan GLCM (Gray Level Co-Occurence) dan K-NN (K-Nearest Neighbor)," *MATRIK J. Manajemen, Tek. Inform. dan Rekayasa Komput.*, vol. 21, no. 2, pp. 477–488, 2022, doi: 10.30812/matrik.v21i2.1572.

[9] A. S. Sinaga, A. Damayanti, and S. Febriyanti, "Analisis Pengurangan Derau Pada Restorasi Citra Ulos," *J. SAINTIKOM (Jurnal Sains Manaj. Inform. dan Komputer)*, vol. 24, no. 1, pp. 39–47, 2025.

[10] R. F. Naibaho and I. P. Sari, "Implementasi Metode Gray Level Co-Occurrence Matrix Menganalisis Tekstur Kulit Wajah," *sudo J. Tek. Inform.*, vol. 3, no. 4, pp. 172–182, 2024.