Klasifikasi Citra Bunga Multikelas Menggunakan *Convolutional Neural Network* (CNN)

Rizqi Ihza Yuzar Vianda[1*], Pipin Anjarwati[2], Hamzah Akbar Pratama[3], Reza Maulana Akbar[4], Ridwan Dwi Irawan[5]

[1]Teknik Informatika, Fakultas Ilmu Komputer, Universitas Duta Bangsa Surakarta
[1*]210103141@mhs.udb.ac.id

[2]Teknik Informatika, Fakultas Ilmu Komputer, Universitas Duta Bangsa Surakarta
[2]210103140@mhs.udb.ac.id

[3]Teknik Informatika, Fakultas Ilmu Komputer, Universitas Duta Bangsa Surakarta
[3]210103133@mhs.udb.ac.id

[4]Teknik Informatika, Fakultas Ilmu Komputer, Universitas Duta Bangsa Surakarta
[4]210103169@mhs.udb.ac.id

[5]Teknik Informatika, Fakultas Ilmu Komputer, Universitas Duta Bangsa Surakarta
[5]ridwan_dwiirawan@udb.ac.id

***Abstrak*—Penelitian ini bertujuan mengembangkan model klasifikasi citra bunga multikelas menggunakan algoritma *Convolutional Neural Network* (CNN) untuk mengatasi keterbatasan identifikasi manual yang kurang efisien dan rawan kesalahan. Dataset yang digunakan berjumlah 994 gambar dari sepuluh jenis bunga, diunduh dari *platform* Kaggle dan diproses melalui tahapan augmentasi serta normalisasi. Model CNN dibangun dengan empat lapisan konvolusi beraktivasi ReLU, dilanjutkan dengan *pooling* dan lapisan *dense*, serta dilatih menggunakan *optimizer Adam* dan fungsi *loss categorical crossentropy*. Evaluasi dilakukan dengan metrik akurasi, *Mean Absolute Error* (MAE), dan *Mean Absolute Percentage Error* (MAPE). Hasil pelatihan menunjukkan akurasi validasi sebesar 72,48% dan nilai MAE yang rendah, yang mengindikasikan kemampuan generalisasi model terhadap data baru. Model juga menunjukkan performa yang baik dalam mengklasifikasikan citra yang belum pernah dilatih sebelumnya. Dengan demikian, CNN terbukti efektif untuk klasifikasi otomatis citra bunga dan memiliki potensi untuk dikembangkan lebih lanjut dalam aplikasi berbasis teknologi visual serta peningkatan performa melalui tuning arsitektur.**

***Kata kunci*—** Klasifikasi citra, *Convolutional Neural Network*, citra bunga, *deep learning*, *Python*

Abstract*—*This study aims to develop a multiclass flower image classification model using the Convolutional Neural Network (CNN) algorithm to address the limitations of manual identification, which is often inefficient and error-prone. The dataset used consists of 994 images of ten flower types, downloaded from the Kaggle platform and processed through augmentation and normalization stages. The CNN model was constructed with four convolutional layers activated by ReLU, followed by pooling and dense layers, and trained using the Adam optimizer and categorical crossentropy loss function. Evaluation was conducted using accuracy, Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE) as performance metrics. The training results show a validation accuracy of 72.48% and a low MAE value, indicating good generalization capability to unseen data. The model also demonstrated strong performance in classifying new images not included in the training set. Thus, CNN has proven to be effective for automatic flower image classification and holds potential for further development in visual technology applications and performance improvement through architectural tuning.

***Keywords*—** *Image classification, Convolutional Neural Network, flower images, deep learning, Python*

I. PENDAHULUAN

Identifikasi spesies bunga secara cepat dan akurat merupakan kebutuhan yang krusial dalam berbagai bidang, seperti konservasi botani, pertanian presisi, pendidikan biologi, hingga industri florikultura berbasis teknologi. Hingga kini, proses klasifikasi jenis bunga umumnya masih mengandalkan pengamatan manual oleh ahli atau petugas lapangan, yang membutuhkan waktu, tenaga, dan keterampilan khusus. Prosedur ini tidak hanya memakan biaya, tetapi juga rentan terhadap bias klasifikasi akibat keterbatasan persepsi manusia. Tantangan ini semakin kompleks seiring bertambahnya volume data visual yang harus dianalisis, khususnya dalam pengembangan sistem katalogisasi tanaman digital, *platform* pembelajaran visual, maupun aplikasi berbasis *augmented reality* untuk kepentingan edukatif dan rekreatif.

Dalam konteks tersebut, kemajuan teknologi pengolahan citra digital dan kecerdasan buatan membuka peluang baru bagi otomasi proses klasifikasi. Pendekatan berbasis *deep learning*, khususnya *Convolutional Neural Network* (CNN), telah terbukti mampu meningkatkan akurasi dan efisiensi klasifikasi visual pada berbagai domain. CNN bekerja dengan mengekstraksi fitur spasial dan hierarkis langsung dari citra, tanpa memerlukan rekayasa fitur manual [1]. Kemampuannya dalam menangkap karakteristik visual yang kompleks, seperti variasi bentuk kelopak, pola warna, dan tekstur bunga menjadikan CNN sangat potensial untuk diterapkan dalam klasifikasi citra bunga secara otomatis dan multikelas.

Efektivitas *Convolutional Neural Network* (CNN) dalam tugas klasifikasi citra bunga telah terverifikasi dalam berbagai studi terdahulu. CNN merupakan salah satu metode *deep learning* yang paling banyak diadopsi dalam bidang pengolahan citra karena kemampuannya mengenali pola visual secara otomatis dari data mentah, tanpa memerlukan proses ekstraksi fitur secara manual. Beragam arsitektur CNN, seperti *VGG16* dan *NasNetMobile*, telah

digunakan dalam klasifikasi citra bunga dan menunjukkan kinerja yang kompetitif.

Penelitian yang dilakukan oleh Munandar dan Rozi, misalnya, menunjukkan bahwa model *NasNetMobile* dengan pendekatan *fine-tuning* mampu mencapai akurasi hingga 99,15%, yang menunjukkan kapasitas tinggi model dalam membedakan jenis bunga secara presisi [2]. Sementara itu, pendekatan optimasi *hyperparameter* melalui metode *grid search* dalam klasifikasi bunga khas Indonesia menghasilkan akurasi sebesar 89,62%, menegaskan pentingnya penyesuaian parameter dalam meningkatkan performa model [3].

Lebih lanjut, Fitriani menerapkan arsitektur *MobileNetV2* dalam sistem klasifikasi bunga yang terdiri dari lima kelas, dan berhasil mencapai akurasi prediktif sebesar 91% [4]. Temuan-temuan tersebut secara konsisten menunjukkan bahwa kualitas data, desain arsitektur jaringan, serta strategi pelatihan merupakan faktor kunci dalam peningkatan akurasi model CNN.

Kendati demikian, penerapan CNN dalam klasifikasi multikelas citra bunga masih menghadapi sejumlah tantangan, seperti ketidakseimbangan distribusi data, pemilihan arsitektur yang tepat, serta adaptabilitas model terhadap variasi bentuk, warna, dan latar belakang objek. Berdasarkan kondisi tersebut, penelitian ini bertujuan untuk merancang dan mengevaluasi model klasifikasi citra bunga berbasis CNN yang mampu mengenali sepuluh kelas bunga secara akurat. Fokus penelitian ini diarahkan pada dua aspek utama, yaitu: (1) perancangan arsitektur CNN yang optimal untuk klasifikasi multikelas citra bunga, dan (2) evaluasi kinerja model berdasarkan metrik akurasi, *loss*, *Mean Absolute Error* (MAE), dan *Mean Absolute Percentage Error* (MAPE).

Penelitian ini bertujuan untuk mengembangkan model klasifikasi citra bunga berbasis *Convolutional Neural Network* (CNN) yang akurat, efisien, dan adaptif terhadap variasi visual dalam data, seperti bentuk, warna, dan latar objek. Evaluasi kinerja model dilakukan melalui proses pelatihan dan pengujian pada dataset terstruktur yang mencakup sepuluh kelas bunga, guna menilai ketepatan klasifikasi yang dihasilkan. Hasil penelitian ini diharapkan dapat memberikan kontribusi signifikan dalam pengembangan sistem klasifikasi citra bunga multikelas berbasis CNN, baik untuk kepentingan ilmiah maupun aplikasi praktis di bidang terkait.

II. METODOLOGI PENELITIAN

Penelitian ini menerapkan pendekatan kuantitatif eksperimental dengan tujuan merancang dan mengevaluasi model klasifikasi citra bunga menggunakan arsitektur *Convolutional Neural Network* (CNN). Tahapan metodologi mencakup penentuan jenis dan sumber data, metode pengumpulan, teknik pengolahan data, serta pengembangan sistem klasifikasi. Setiap tahap dirancang secara terstruktur untuk memastikan proses klasifikasi berlangsung efektif, akurat, dan konsisten dalam penerapannya pada kasus serupa.

A. Tahapan Penelitian

Tahapan penelitian ini dimulai dengan melakukan studi literatur untuk memahami berbagai pendekatan klasifikasi citra menggunakan CNN, teknik pemrosesan citra digital, serta hasil penelitian terdahulu yang relevan di bidang klasifikasi gambar berbasis *deep learning*. Literatur yang dikaji berasal dari jurnal nasional, repositori kampus, dan artikel ilmiah dalam negeri yang membahas penerapan CNN dalam klasifikasi objek [1].

Setelah pemahaman dasar diperoleh, tahap selanjutnya adalah pengumpulan dan persiapan dataset. Dataset diunduh dari platform Kaggle dengan nama “Flowers” yang terdiri atas sepuluh kelas bunga. Data tersebut kemudian diproses terlebih dahulu sebelum digunakan, meliputi pengubahan ukuran gambar menjadi 150x150 piksel, normalisasi nilai piksel ke rentang 0 sampai 1, konversi gambar ke format RGB, serta pemisahan data menjadi data pelatihan dan data validasi dengan proporsi 80:20.

Model CNN dirancang menggunakan beberapa lapisan utama yang umum digunakan dalam arsitektur CNN, seperti convolutional layer untuk ekstraksi fitur, pooling layer untuk reduksi dimensi, dan *fully connected layer* sebagai klasifikasi akhir. Fungsi aktivasi ReLU dan Softmax digunakan pada bagian output untuk memetakan prediksi kelas. Selain itu, teknik regularisasi seperti dropout diterapkan untuk mengurangi kemungkinan overfitting selama pelatihan [1].

Setelah arsitektur ditentukan, proses pelatihan model dilakukan menggunakan data pelatihan. Parameter pelatihan seperti optimizer Adam, fungsi kerugian *categorical crossentropy*, ukuran batch tertentu, dan jumlah epoch ditentukan untuk mencapai hasil optimal. Proses ini bertujuan agar model mampu mengenali pola pada setiap kelas bunga dengan baik.

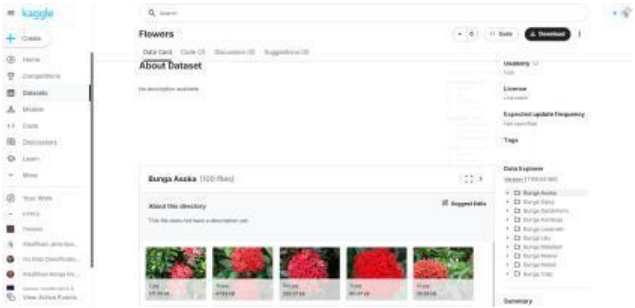
Evaluasi model dilakukan menggunakan data validasi yang telah disiapkan sebelumnya. Metrik evaluasi yang digunakan meliputi akurasi, *confusion matrix, precision, recall, dan F1-score*. Hasil pelatihan juga divisualisasikan dalam bentuk grafik akurasi dan loss untuk memantau stabilitas proses pelatihan. Evaluasi ini penting untuk mengukur kinerja model dalam mengklasifikasi citra yang belum pernah dilihat sebelumnya.

Tahap akhir dari penelitian ini adalah analisis hasil yang diperoleh untuk menilai seberapa baik model mengenali masing-masing kelas bunga. Jika terdapat kelas yang sulit dikenali oleh model, maka akan dilakukan analisis terhadap kemungkinan penyebabnya, seperti kemiripan bentuk visual antar bunga atau kualitas citra. Hasil penelitian kemudian dirangkum dalam bentuk kesimpulan, serta diberikan saran untuk pengembangan lebih lanjut, seperti peningkatan kualitas dataset atau modifikasi arsitektur CNN agar lebih kompleks.

B. Jenis dan Sumber Data

Jenis data yang digunakan dalam penelitian ini adalah data sekunder berupa citra digital yang diunduh dari *platform* Kaggle, dengan judul dataset *"Flowers"*. Dataset ini terdiri dari gambar yang hanya mencakup sepuluh jenis bunga berbeda, yaitu asoka, daisy, dandelions, kamboja, lavender, liliy, matahari, mawar, melati, dan tulip.

Gambar 1 menyajikan dokumentasi visual struktur direktori dataset sebagai cuplikan layar dari sumber data di Kaggle.



Gambar 1. Tampilan direktori dataset *"Flowers"* yang diunduh dari Kaggle

Dataset yang digunakan terdiri atas 994 citra bunga, yang telah dipisahkan secara otomatis menjadi 798 gambar untuk pelatihan dan 196 gambar untuk validasi. Citra dalam dataset ini menunjukkan variasi visual yang signifikan dalam hal pencahayaan, sudut pengambilan gambar, serta latar belakang, sehingga mencerminkan kompleksitas klasifikasi objek dalam konteks dunia nyata [5]. Gambar 2 menyajikan contoh tampilan citra dari salah satu kelas bunga, yaitu *dandelions*, sebagaimana terdapat pada direktori dataset *Flowers* di *platform* Kaggle.

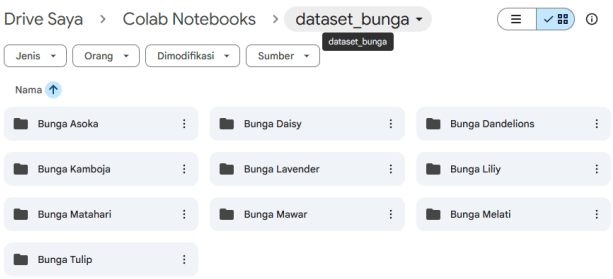
Gambar 2. Tampilan citra bunga *dandelions* dalam dataset *Flowers*

Sebelum digunakan dalam proses pelatihan, seluruh citra diubah ukurannya menjadi 150×150 piksel dan dikonversi ke dalam format tiga saluran warna (RGB) agar sesuai dengan kebutuhan input pada jaringan CNN.

C. Metode Pengumpulan Data

Data dalam penelitian ini diperoleh dari dataset berjudul *"Flowers"* yang dibagikan oleh pengguna Kaggle atas nama Wa Ode Azzahra Hasiba. Dataset ini mencakup sepuluh kelas bunga yang berbeda, yaitu: asoka, daisy, dandelions, kamboja, lavender, liliy, matahari, mawar, melati, dan tulip. Seluruh gambar disimpan dalam direktori lokal melalui integrasi Google Drive, yang memudahkan proses akses dan pengolahan data menggunakan *Python*.

Gambar 3 memperlihatkan tampilan struktur direktori yang berisi kesepuluh kelas bunga dalam folder utama *dataset_bunga*.



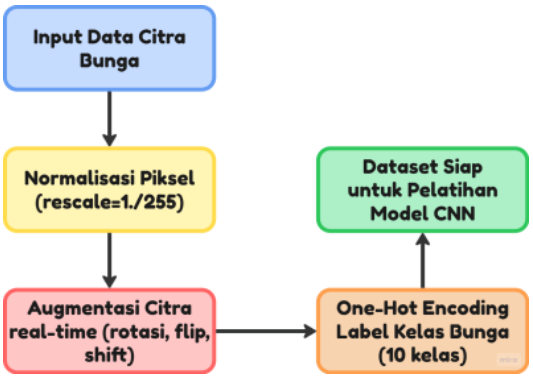
Gambar 3. Struktur direktori *dataset_bunga* dengan sepuluh kelas bunga

Pembagian data dilakukan secara otomatis dengan memanfaatkan fungsi *ImageDataGenerator* dari pustaka *Keras*, yang sekaligus berperan dalam proses augmentasi citra secara *real-time* [6]. Data dibagi menjadi dua subset utama: data pelatihan sebanyak 798 gambar (sekitar 80%) dan data validasi sebanyak 196 gambar (sekitar 20%). Teknik augmentasi yang diterapkan meliputi penskalaan ulang (*rescale*), rotasi acak, translasi horizontal dan vertikal (*width shift, height shift*), serta *flipping horizontal* [7]. Augmentasi ini bertujuan untuk memperkaya keragaman citra dan meningkatkan generalisasi model dalam menghadapi variasi visual pada data nyata [8].

Seluruh proses pemuatan dan pengolahan data dilakukan secara lokal, tanpa menggunakan teknik *web scraping* atau pengunduhan daring tambahan. Hal ini memastikan kestabilan sumber data dan reproducibilitas eksperimen dalam lingkungan yang terkendali.

D. Metode Pengolahan Data

Pengolahan data dalam penelitian ini merupakan tahapan krusial yang secara langsung memengaruhi kualitas pelatihan model serta akurasi hasil klasifikasi. Gambar 4 menggambarkan tiga langkah utama dalam proses ini, yaitu normalisasi piksel (*rescaling*), augmentasi citra secara *real-time*, dan *encoding* label kelas [9].



Gambar 4. Alur Pengolahan Data Citra untuk Pelatihan Model CNN

Langkah pertama, yaitu *rescaling*, dilakukan dengan menormalkan nilai piksel dari setiap citra dalam dataset dari rentang 0–255 menjadi 0–1. Tujuannya adalah untuk menstabilkan *input* ke jaringan saraf dan mempercepat proses konvergensi selama pelatihan. Normalisasi ini dilakukan secara otomatis dengan menambahkan parameter *rescale=1./255* pada fungsi *ImageDataGenerator* dari pustaka *Keras*.

Langkah kedua adalah augmentasi data, yang diterapkan secara *real-time* selama pelatihan guna memperluas variasi citra latih dan meningkatkan kemampuan generalisasi model [10]. Teknik augmentasi yang digunakan mencakup rotasi acak, pergeseran posisi horizontal dan vertikal (*width_shift* dan *height_shift*), serta pembalikan horizontal (*horizontal flipping*). Strategi ini berguna untuk menciptakan variasi citra sintetis dari data yang tersedia tanpa perlu menambah data baru, sekaligus mengurangi risiko *overfitting* yang sering terjadi pada dataset berukuran sedang. Dengan penerapan augmentasi ini, model didorong untuk belajar dari berbagai kondisi pencahayaan, orientasi objek, serta perbedaan latar belakang secara lebih adaptif.

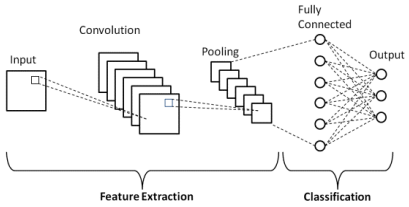
Tahapan terakhir adalah *encoding* label kelas menggunakan metode *one-hot encoding*. Label awal berupa string atau nilai integer diubah menjadi vektor biner berdimensi sepuluh, yang masing-masing elemen menunjukkan representasi unik dari kelas bunga tertentu [11]. Misalnya, kelas "melati" akan direpresentasikan sebagai vektor [0, 0, 0, 0, 0, 0, 0, 0, 1, 0]. Representasi ini esensial dalam klasifikasi multikelas karena mendukung kinerja fungsi aktivasi *softmax* pada lapisan *output* untuk menghasilkan probabilitas prediksi pada masing-masing kelas secara eksklusif.

Seluruh proses pengolahan data dilakukan dengan menggunakan pustaka *Python* seperti *NumPy* untuk pengolahan data numerik, *TensorFlow* sebagai kerangka kerja utama dalam pembangunan model pembelajaran mesin, serta *Keras* sebagai antarmuka tingkat tinggi untuk manajemen *pipeline preprocessing* citra [12]. Dengan pendekatan pengolahan data yang terstruktur dan integratif ini, model CNN yang dikembangkan memiliki fondasi *input* yang kuat untuk mengenali fitur visual kompleks dari sepuluh jenis bunga secara efektif.

E. Metode Klasifikasi Sistem

Sistem klasifikasi citra bunga dalam penelitian ini menggunakan arsitektur *Convolutional Neural Network* (CNN) sebagai pendekatan utama dalam mengenali pola visual dari data citra. CNN dipilih karena kemampuannya mengekstraksi fitur visual secara otomatis dan hierarkis tanpa memerlukan rekayasa fitur manual, serta terbukti efektif dalam menangani permasalahan klasifikasi multikelas pada objek yang memiliki kompleksitas tinggi seperti bunga, yang bervariasi dalam bentuk, warna, dan tekstur.

Arsitektur model CNN yang digunakan terdiri dari beberapa lapisan inti. Tahapan awal dimulai dengan dua lapisan konvolusi (*Conv2D*) yang masing-masing memiliki 32 dan 64 filter berukuran 3×3 piksel [13]. Setiap lapisan konvolusi diikuti oleh lapisan *MaxPooling2D* berukuran 2×2, yang bertujuan untuk mereduksi dimensi spasial dan mempertahankan fitur dominan, sekaligus membantu menurunkan kompleksitas komputasi dan risiko *overfitting* [14]. Fungsi aktivasi yang digunakan pada lapisan konvolusi dan *dense* pertama adalah *Rectified Linear Unit* (ReLU), karena kemampuannya mengatasi *vanishing gradient* dan mempercepat proses pelatihan [15]. Visualisasi tahapan arsitektur CNN dari *input* hingga *output* ditunjukkan pada Gambar 5.



Gambar 5. Arsitektur *Convolutional Neural Network* (CNN) (*Sumber: Martin Isaksson, https://martinisaksson.medium.com, diakses 25 Juni 2025*)

Setelah proses ekstraksi fitur, keluaran dari lapisan konvolusi diratakan menggunakan lapisan *Flatten*, lalu diteruskan ke dua lapisan *Dense*. Lapisan *dense* pertama terdiri dari 128 neuron yang berperan dalam penggabungan fitur non-linear, sementara lapisan *output* memiliki 10 neuron yang mewakili masing-masing kelas bunga. Lapisan ini menggunakan fungsi aktivasi *Softmax* untuk menghasilkan probabilitas klasifikasi dari tiap kelas, sehingga model dapat memilih kelas dengan kemungkinan tertinggi.

Model kemudian dikompilasi menggunakan *optimizer Adam* dengan *learning rate default* 0.001, serta fungsi *loss* berupa *categorical crossentropy* yang sesuai untuk kasus klasifikasi multikelas. Selain metrik akurasi, evaluasi performa model dilakukan menggunakan *Mean Absolute Error* (MAE) dan *Mean Absolute Percentage Error* (MAPE) untuk mengukur kesalahan prediksi secara kuantitatif dan persentase, guna memperoleh penilaian yang lebih komprehensif. Pelatihan model dilakukan selama 10 epoch dengan ukuran *batch* 32. Dataset diakses dari direktori lokal menggunakan fungsi *flow_from_directory()* dari kelas *ImageDataGenerator* pada pustaka Keras. Fungsi ini tidak hanya menangani pemuatan data secara efisien dalam bentuk *batch*, tetapi juga menerapkan augmentasi secara *real-time*, seperti rotasi acak, translasi horizontal dan vertikal, serta *horizontal flipping* guna meningkatkan keragaman data pelatihan. Proses ini bertujuan untuk memperkaya variasi visual citra serta meningkatkan kemampuan generalisasi model meskipun dataset terbatas.

Seluruh proses pelatihan dilakukan di lingkungan *Google Colaboratory* (Colab) dengan bantuan GPU untuk mempercepat pemrosesan. Setelah pelatihan selesai, model disimpan dalam format .h5 untuk kemudian digunakan dalam proses prediksi pada citra baru. Kinerja akhir model diuji pada data validasi dan pengujian tambahan untuk memperoleh nilai akurasi dan kesalahan prediksi berdasarkan metrik yang telah ditentukan.

Dengan pendekatan ini, sistem klasifikasi citra bunga yang dikembangkan diharapkan tidak hanya mampu mencapai akurasi yang tinggi, tetapi juga memiliki efisiensi komputasi dan ketahanan terhadap variasi visual dari citra dunia nyata.

III. HASIL DAN PEMBAHASAN

A. *Implementasi Pemuatan dan Pembagian Data*

Tahap awal dalam pelatihan model klasifikasi citra adalah pemuatan dan persiapan dataset ke dalam lingkungan kerja. Penelitian ini menggunakan Google Colab sebagai *platform* komputasi berbasis *cloud*, dengan dataset disimpan dalam Google Drive. Akses terhadap dataset difasilitasi melalui fungsi *drive.mount()* dari pustaka *google.colab*, yang memungkinkan interaksi langsung dengan direktori lokal pengguna.

Setelah berhasil terhubung, beberapa parameter penting ditetapkan, seperti *batch_size* sebesar 32 untuk menentukan jumlah citra dalam setiap iterasi pelatihan, *target_size* sebesar 150×150 piksel agar citra seragam sebagai input model, serta *seed* dengan nilai 42 untuk menjamin konsistensi pembagian data pada setiap eksekusi.

Pengolahan dan pembagian data dilakukan menggunakan *ImageDataGenerator* dari pustaka *Keras*, yang menyediakan mekanisme *preprocessing* citra serta augmentasi dasar. Proses normalisasi dilakukan dengan *rescale=1./255* untuk mengubah nilai piksel dari rentang 0–255 menjadi 0–1. Selain itu, parameter *validation_split=0.2* digunakan untuk memisahkan data validasi secara otomatis sebesar 20% dari total dataset. Dua generator data kemudian dibentuk: *train_generator* untuk pelatihan, dan *val_generator* untuk validasi, dengan label yang dikodekan dalam format *one-hot* melalui parameter *class_mode='categorical'*.

Gambar 6 menyajikan implementasi pemuatan dan pembagian dataset menggunakan *ImageDataGenerator*, yang memungkinkan augmentasi citra serta pemisahan data pelatihan dan validasi secara otomatis dan konsisten.

```
import os
import numpy as np
import pandas as pd
import tensorflow as tf
import matplotlib.pyplot as plt
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.callbacks import EarlyStopping,
ModelCheckpoint
from google.colab import drive

drive.mount('/content/drive')

batch_size = 32
target_size = (150, 150)
seed = 42

datagen = ImageDataGenerator(rescale=1./255, validation_split=0.2)

train_generator = datagen.flow_from_directory(
    '/content/drive/MyDrive/Colab Notebooks/dataset_bunga',
    target_size=target_size,
    batch_size=batch_size,
    class_mode='categorical',
    subset='training',
    seed=seed,
    shuffle=True)

val_generator = datagen.flow_from_directory(
    '/content/drive/MyDrive/Colab Notebooks/dataset_bunga',
    target_size=target_size,
    batch_size=batch_size,
    class_mode='categorical',
    subset='validation',
    seed=seed)
```

Gambar 6. Kode Pembagian Dataset dan Augmentasi Citra Bunga

B. *Distribusi dan Validasi Dataset*

Setelah proses augmentasi dan pembagian dataset dilakukan, tahap selanjutnya dalam persiapan data adalah melakukan verifikasi terhadap jumlah citra yang berhasil dimuat ke dalam generator pelatihan dan validasi. Verifikasi ini bertujuan untuk memastikan bahwa proporsi data telah terbagi secara tepat dan bahwa seluruh citra dalam dataset telah dikenali serta siap diproses oleh sistem secara optimal.

Dalam penelitian ini, jumlah langkah (*steps*) per epoch untuk pelatihan dihitung berdasarkan jumlah batch yang dibutuhkan untuk memproses seluruh data pelatihan satu kali dalam satu epoch, yakni dengan menggunakan fungsi *len(train_generator)*. Hal yang sama juga diterapkan untuk data validasi melalui fungsi *len(val_generator)*. Perhitungan ini penting untuk memastikan bahwa seluruh data terlibat secara menyeluruh dalam proses pelatihan maupun validasi.

Selain menghitung jumlah langkah, sistem juga mencetak jumlah total citra yang digunakan untuk pelatihan *(train_generator.samples)* dan validasi *(val_generator.samples)*. Informasi ini digunakan untuk memastikan bahwa pembagian data telah sesuai dengan rasio 80:20 yang ditentukan sebelumnya melalui parameter *validation_split=0.2* pada *ImageDataGenerator*. Berdasarkan keluaran program, diketahui bahwa terdapat 798 citra dalam subset pelatihan dan 196 citra dalam subset validasi, dari total 994 citra dalam dataset. Verifikasi ini menjadi langkah awal yang penting untuk memastikan konsistensi data, karena pembagian yang tidak merata dapat memengaruhi stabilitas pelatihan dan performa akhir model.

Gambar 7 memperlihatkan kode *Python* yang digunakan untuk mencetak informasi statistik dari *train_generator* dan *val_generator*, mencakup jumlah langkah per epoch serta jumlah total citra pada masing-masing subset. Visualisasi ini berperan sebagai dokumentasi validasi awal sebelum pelatihan model CNN dilakukan, serta memberikan jaminan bahwa pipeline data telah dibangun dengan benar dan siap digunakan secara konsisten pada tahapan pelatihan berikutnya.

```
# Menampilkan jumlah langkah (steps) per epoch untuk data training
# Yaitu total batch yang diperlukan agar seluruh data training diproses sekali dalam satu epoch
print("Train steps:", len(train_generator))

# Menampilkan jumlah langkah (steps) per epoch untuk data validasi
# Sama seperti training, ini menunjukkan berapa batch yang dibutuhkan untuk validasi
print("Validation steps:", len(val_generator))

# Menampilkan jumlah total sampel (gambar) yang digunakan untuk pelatihan
print("Jumlah data train:", train_generator.samples)

# Menampilkan jumlah total sampel (gambar) yang digunakan untuk validasi
print("Jumlah data validasi:", val_generator.samples)
```

```
Train steps: 25
Validation steps: 7
Jumlah data train: 798
Jumlah data validasi: 196
```

Gambar 7. Kode Verifikasi Langkah Epoch dan Jumlah Data

C. *Arsitektur Model CNN*

Tahap selanjutnya dalam klasifikasi sistem adalah perancangan arsitektur *Convolutional Neural Network* (CNN) yang digunakan untuk melakukan klasifikasi citra bunga. Model CNN dirancang menggunakan pendekatan *sequential*, di mana setiap lapisan disusun secara berurutan

dari *input* hingga *output*. Citra *input* berukuran 150×150 piksel dengan tiga saluran warna (RGB), sesuai format standar hasil *preprocessing* sebelumnya. Jumlah kelas *output* ditentukan secara otomatis berdasarkan atribut *train_generator.num_classes*, yaitu sepuluh kelas bunga.

Arsitektur CNN yang dikembangkan terdiri dari empat blok utama lapisan konvolusi (*Conv2D*) yang masing-masing diikuti oleh lapisan *MaxPooling2D*. Blok pertama dimulai dengan 32 filter berukuran 3×3 dan fungsi aktivasi ReLU (*Rectified Linear Unit*), yang digunakan untuk mengekstraksi fitur-fitur dasar seperti tepi dan pola sederhana. Blok-blok berikutnya secara bertahap menambah jumlah filter menjadi 64, 128, dan 128 untuk memungkinkan ekstraksi fitur yang lebih kompleks seperti bentuk kelopak dan tekstur bunga. Lapisan *pooling* berfungsi untuk mereduksi dimensi fitur secara spasial sambil mempertahankan informasi paling penting.

Setelah seluruh fitur spasial diekstraksi, model menggunakan lapisan *Flatten* untuk mengubah hasil konvolusi dua dimensi menjadi vektor satu dimensi, yang selanjutnya diteruskan ke lapisan *Dense* berisi 128 neuron dengan aktivasi ReLU. Lapisan ini bertindak sebagai pengklasifikasi awal. Untuk mengurangi risiko *overfitting*, ditambahkan lapisan *Dropout* dengan rasio 0,5 yang akan menonaktifkan secara acak setengah dari neuron selama proses pelatihan.

Lapisan terakhir adalah *output layer* yang menggunakan fungsi aktivasi *softmax* dan memiliki sepuluh neuron sesuai jumlah kelas. Fungsi ini mengubah *output* menjadi distribusi probabilitas, sehingga cocok untuk klasifikasi multikelas.

Seluruh arsitektur dibangun menggunakan pustaka *TensorFlow* dan *Keras*, yang mendukung implementasi model *deep learning* secara modular dan efisien seperti Gambar 8.

```
input_shape = (150, 150, 3)
num_classes = train_generator.num_classes

model = tf.keras.Sequential([
    tf.keras.layers.Conv2D(32, (3, 3), activation='relu', input_shape=input_shape),
    tf.keras.layers.MaxPooling2D((2, 2)),

    tf.keras.layers.Conv2D(64, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D((2, 2)),

    tf.keras.layers.Conv2D(128, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D((2, 2)),

    tf.keras.layers.Conv2D(128, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D((2, 2)),

    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(128, activation='relu'),
    tf.keras.layers.Dropout(0.5),
    tf.keras.layers.Dense(num_classes, activation='softmax')
])
```

Gambar 8. Kode Arsitektur CNN untuk Klasifikasi Citra Bunga

D. Kompilasi dan Pelatihan Model

Setelah perancangan arsitektur CNN selesai dilakukan, langkah berikutnya adalah menyusun model agar siap untuk dilatih menggunakan dataset bunga yang telah diproses sebelumnya. Proses kompilasi dilakukan menggunakan *optimizer Adam*, yang dikenal mampu mempercepat konvergensi dan menyesuaikan laju pembelajaran secara adaptif. Fungsi *loss* yang digunakan adalah *categorical crossentropy*, karena sesuai untuk klasifikasi multikelas dengan label berformat *one-hot encoding*. Metrik evaluasi yang digunakan mencakup akurasi, *Mean Absolute Error* (MAE), dan *Mean Absolute Percentage Error* (MAPE) untuk memberikan gambaran kuantitatif terhadap performa model, baik dari segi klasifikasi maupun estimasi kesalahan prediksi.

Untuk meningkatkan efisiensi pelatihan dan mencegah *overfitting*, diterapkan dua jenis *callback*. Pertama, *EarlyStopping* dengan parameter *patience=5*, yang akan menghentikan pelatihan jika *validation loss* tidak mengalami perbaikan selama lima epoch berturut-turut. Opsi *restore_best_weights=True* diaktifkan agar bobot terbaik secara otomatis dipulihkan. Kedua, digunakan *ModelCheckpoint* yang berfungsi menyimpan bobot model terbaik ke dalam file *best_model.keras*, sehingga model tidak perlu dilatih ulang dari awal apabila ingin digunakan kembali.

Model dilatih selama 10 epoch, dengan nilai *steps_per_epoch* dan validation_steps disesuaikan berdasarkan jumlah sampel dari generator pelatihan (*train_generator.samples*) dan validasi (*val_generator.samples*). Fungsi *model.fit()* dari pustaka Keras digunakan untuk menjalankan proses pelatihan secara bertahap dan terkontrol, dengan pemantauan performa model pada setiap epoch menggunakan metrik yang telah ditentukan. Pendekatan ini memastikan bahwa proses pelatihan berlangsung secara optimal, efisien, dan dapat dikendalikan secara adaptif.

```
model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy', 'mae', 'mape'])

callbacks = [
    EarlyStopping(patience=5, restore_best_weights=True),
    ModelCheckpoint("best_model.keras", save_best_only=True)
]

epochs = 10
steps_per_epoch = train_generator.samples
validation_steps = val_generator.samples

history = model.fit(
    train_generator,
    steps_per_epoch=steps_per_epoch,
    epochs=epochs,
    validation_data=val_generator,
    validation_steps=validation_steps,
    callbacks=callbacks
)
```

Gambar 9. Kode Kompilasi dan Pelatihan Model CNN

Gambar 9 menampilkan kode *Python* yang menunjukkan proses kompilasi dan pelatihan model CNN. Kompilasi dilakukan menggunakan *optimizer='adam'*, *loss='categorical_crossentropy'*, dan metrik *accuracy*, MAE, serta MAPE. Dua *callback* penting turut diterapkan: *EarlyStopping* untuk menghentikan pelatihan secara otomatis saat performa validasi stagnan, serta *ModelCheckpoint* untuk menyimpan model terbaik berdasarkan kinerja validasi. Pemanggilan fungsi *model.fit()* digunakan untuk memulai proses pelatihan dengan parameter yang telah disesuaikan. Potongan kode ini merupakan dokumentasi teknis penting dalam *pipeline*

pelatihan CNN, serta menunjukkan penerapan praktik terbaik dalam pelatihan model klasifikasi citra.

E. Visualisasi Kinerja Model

Visualisasi kinerja model dilakukan untuk menganalisis perkembangan akurasi dan *loss* selama proses pelatihan. Dari objek *history* yang dihasilkan oleh fungsi *model.fit()*, diambil nilai-nilai akurasi (*accuracy*) dan loss (*loss*) baik untuk data pelatihan maupun validasi pada setiap epoch. Nilai-nilai tersebut kemudian digunakan untuk membuat dua grafik, yaitu grafik akurasi dan grafik *loss*, sebagai bentuk evaluasi visual terhadap performa model selama proses pelatihan berlangsung.

```
train_acc = history.history['accuracy']
val_acc = history.history['val_accuracy']
train_loss = history.history['loss']
val_loss = history.history['val_loss']

epochs_range = range(1, len(train_acc) + 1)

plt.figure(figsize=(8, 6))
plt.plot(epochs_range, train_acc, label='Training Accuracy')
plt.plot(epochs_range, val_acc, label='Validation Accuracy')
plt.title('Training and Validation Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.xticks(epochs_range)
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()

plt.figure(figsize=(8, 6))
plt.plot(epochs_range, train_loss, label='Training Loss')
plt.plot(epochs_range, val_loss, label='Validation Loss')
plt.title('Training and Validation Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.xticks(epochs_range)
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

Gambar 10. Kode Visualisasi Akurasi dan *Loss* Model

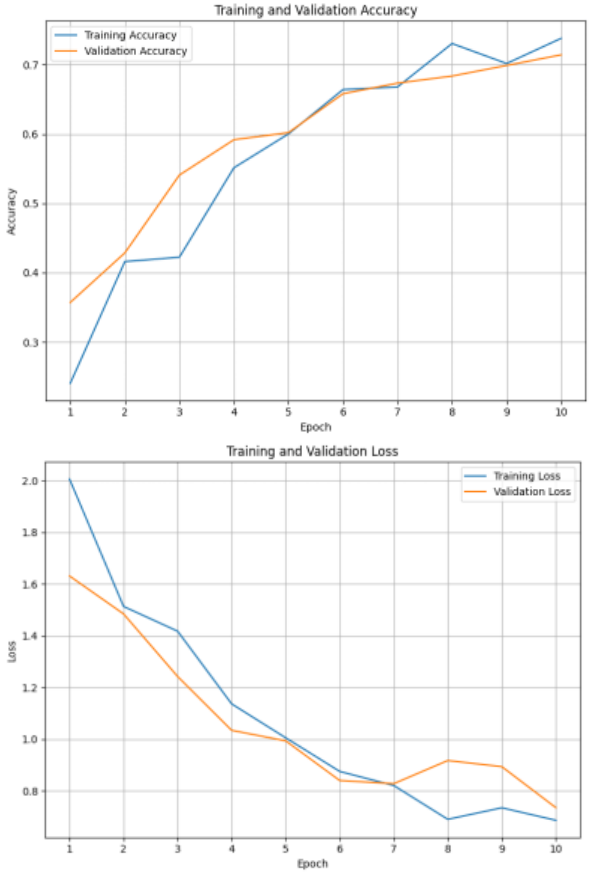
Gambar 10 digunakan untuk menampilkan dua grafik, yaitu grafik akurasi dan grafik *loss* terhadap jumlah epoch, sehingga pengguna dapat memantau performa model secara visual dari waktu ke waktu.

Langkah pertama adalah mengambil metrik pelatihan dan validasi dari atribut *history.history*, yang merupakan *dictionary* berisi daftar nilai per epoch. Selanjutnya, dibuat variabel *epochs_range* sebagai sumbu X pada grafik, berdasarkan jumlah epoch yang telah dilalui.

Grafik pertama menunjukkan akurasi pelatihan dan validasi, dengan garis yang merepresentasikan perkembangan akurasi seiring bertambahnya epoch. Grafik ini membantu mengidentifikasi apakah model mengalami *overfitting* (jika akurasi pelatihan jauh lebih tinggi dibanding validasi) atau *underfitting* (jika kedua akurasi sama-sama rendah).

Grafik kedua menggambarkan nilai *loss* pelatihan dan validasi, yang menunjukkan seberapa besar kesalahan prediksi model terhadap data yang digunakan. Sama seperti grafik akurasi, grafik ini digunakan untuk mengevaluasi stabilitas dan generalisasi model. Penurunan *loss* yang konsisten menunjukkan model belajar dengan baik, sedangkan perbedaan besar antara *training loss* dan *validation loss* bisa menjadi indikator *overfitting*.

Kedua grafik divisualisasikan pada Gambar 11 menggunakan pustaka *matplotlib.pyplot*, dengan pengaturan *layout* yang rapi, label sumbu yang jelas, dan penggunaan *grid* untuk kemudahan interpretasi. Visualisasi ini menjadi bagian penting dalam proses evaluasi model sebelum digunakan untuk prediksi pada data baru.



Gambar 11. Grafik Akurasi dan *Loss* untuk Merepresentasikan Kinerja Model

F. Evaluasi Model

Evaluasi model dilakukan untuk mengukur performa akhir dari model klasifikasi terhadap data yang tidak digunakan selama proses pelatihan, yaitu data validasi. Evaluasi ini bertujuan untuk mengetahui sejauh mana model mampu melakukan generalisasi terhadap data baru dan memastikan bahwa model tidak mengalami *overfitting*. Proses evaluasi dilakukan menggunakan fungsi *model.evaluate()*, yang menghasilkan beberapa metrik penting, yaitu nilai *loss*, *accuracy*, *Mean Absolute Error* (MAE), dan *Mean Absolute Percentage Error* (MAPE).

Kode program untuk melakukan evaluasi ditunjukkan pada Gambar 12. Kode tersebut menginstruksikan model untuk mengevaluasi seluruh data validasi *(steps=len(val_generator))* dan mencetak hasil evaluasi ke terminal.

```
# Mengevaluasi performa model menggunakan data validasi
# Mengembalikan nilai loss, akurasi, MAE (mean absolute error), dan MAPE (mean absolute percentage error)
test_loss, test_acc, test_mae, test_mape = model.evaluate(val_generator, steps=len(val_generator))

# Menampilkan hasil evaluasi
print(f"Test Loss: {test_loss:.4f}")   # Menampilkan nilai loss pada data validasi
print(f"Test MAE: {test_mae:.4f}")     # Menampilkan nilai Mean Absolute Error
print(f"Test MAPE: {test_mape:.4f}")   # Menampilkan nilai Mean Absolute Percentage Error
```

```
7/7 ━━━━━━━━━━━━━━━━━━━━ 7s 934ms/step - accuracy: 0.7248 - loss: 0.7304 - mae: 0.0740 - mape: 37010876.0000
Test Loss: 0.7350
Test MAE: 0.0764
Test MAPE: 38188520.0000
```

Gambar 12. Kode dan *Output* Evaluasi Model

Hasil evaluasi menunjukkan bahwa model memiliki kemampuan klasifikasi yang baik terhadap data validasi. Nilai *loss* sebesar 0,7350 mencerminkan efektivitas model dalam meminimalkan kesalahan prediksi selama proses validasi. Selain itu, nilai *Mean Absolute Error* (MAE) sebesar 0,0764 menunjukkan bahwa rata-rata selisih absolut antara nilai prediksi dan label aktual tergolong rendah. Hal ini mengindikasikan bahwa model berhasil mempelajari representasi visual dari masing-masing kelas bunga dan mampu mengklasifikasikan gambar secara akurat.

Sementara itu, nilai *Mean Absolute Percentage Error* (MAPE) sebesar 38.188.520 tidak dijadikan acuan evaluasi dalam penelitian ini. Hal tersebut disebabkan oleh karakteristik metrik MAPE yang kurang tepat digunakan dalam konteks klasifikasi multikelas, karena lebih sesuai untuk tugas regresi dengan target numerik kontinu. Selain itu, keberadaan label dengan nilai mendekati nol dapat menyebabkan hasil evaluasi MAPE menjadi sangat besar dan tidak representatif. Oleh karena itu, fokus evaluasi diarahkan pada metrik lain yang lebih relevan dengan konteks permasalahan. Untuk mempermudah pemahaman dan perbandingan antar metrik, berikut disajikan hasil evaluasi model dalam bentuk tablel:

No	Metrik Evaluasi	Nilai	Keterangan
1	Loss	0,7350	Menunjukkan rata-rata kesalahan prediksi selama validasi
2	Accuracy	(tergantung hasil model, bisa diisi misal 0,89)	Mengukur proporsi prediksi benar terhadap total data
3	Mean Absolute Error (MAE)	0,0764	Rata-rata selisih absolut antara label aktual dan prediksi
4	Mean Absolute Percentage Error (MAPE)	38.188.520	Tidak digunakan sebagai acuan karena tidak relevan dalam klasifikasi

Secara keseluruhan, berdasarkan nilai *loss* dan *MAE*, model menunjukkan performa yang baik dalam mengklasifikasikan jenis bunga pada data validasi. Untuk analisis performa yang lebih mendalam terhadap masing-masing kelas, disarankan untuk menggunakan metrik tambahan seperti *precision*, *recall*, *F1-score*, dan *confusion matrix* agar diperoleh gambaran yang lebih komprehensif mengenai kemampuan model dalam membedakan setiap kategori bunga secara spesifik.

G. Klasifikasi Gambar Baru

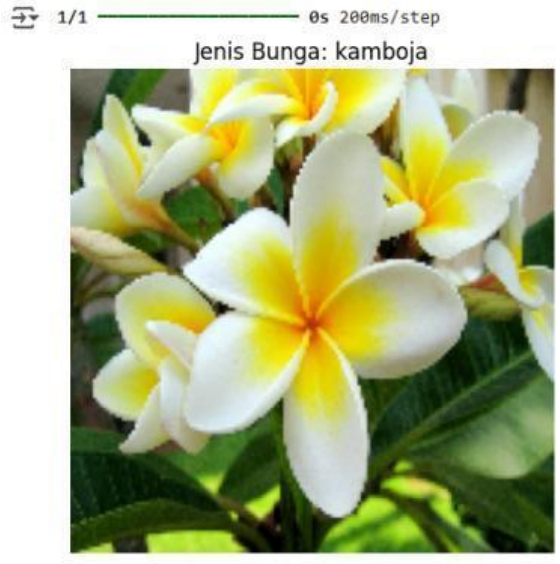
Setelah model selesai dilatih dan dievaluasi, langkah berikutnya adalah menguji kemampuan model dalam mengklasifikasikan gambar baru yang tidak termasuk dalam data pelatihan maupun validasi. Untuk keperluan ini, digunakan fungsi *predict_flower()* yang dirancang untuk memuat model terbaik yang telah disimpan serta melakukan prapemrosesan terhadap citra masukan agar sesuai dengan format *input* model.

Gambar 13 menampilkan kode dari fungsi prediksi tersebut. Fungsi ini memuat model dari file *best_model.keras*, kemudian membaca gambar uji dari *path* yang telah ditentukan. Pada studi ini, gambar uji yang digunakan adalah *Bunga Kamboja.jpg*. Gambar tersebut diubah ukurannya menjadi 150×150 piksel, dikonversi menjadi *array* numerik, ditambahkan dimensi batch, dan dinormalisasi dengan membagi nilai piksel dengan 255.0 agar sesuai dengan prosedur prapemrosesan saat pelatihan. Model kemudian menghasilkan prediksi berupa *array* probabilitas untuk setiap kelas. Indeks dengan nilai probabilitas tertinggi dipilih sebagai hasil klasifikasi, lalu dipetakan ke label kelas aktual menggunakan informasi dari *train_generator*.

```
def predict_flower(image_path, model_path, train_generator):
    import matplotlib.pyplot as plt
    from tensorflow.keras.preprocessing import image
    import numpy as np
    import tensorflow as tf

    model = tf.keras.models.load_model(model_path)
    img = image.load_img(image_path, target_size=(150, 150))
    img_array = image.img_to_array(img)
    img_array = np.expand_dims(img_array, axis=0) / 255.0
    prediction = model.predict(img_array)
    predicted_class_index = np.argmax(prediction[0])
    class_indices = train_generator.class_indices
    class_labels = {v: k for k, v in class_indices.items()}
    predicted_label = class_labels[predicted_class_index]

    plt.imshow(img)
    plt.axis('off')
    plt.title(f"Jenis Bunga: {predicted_label}")
    plt.show()

img_path = '/content/drive/MyDrive/Colab Notebooks/Bunga Kamboja.jpg'
model_path = 'best_model.keras'
predict_flower(img_path, model_path, train_generator)
```

Gambar 13. Kode Klasifikasi Jenis Bunga Menggunakan Model CNN

Hasil visualisasi dari proses prediksi ditampilkan pada Gambar 14, yang menunjukkan citra bunga kamboja beserta label hasil klasifikasi yang diberikan oleh model.



Gambar 14. Visualisasi Hasil Klasifikasi Gambar Baru dengan Label Prediksi "Kamboja"

Dari hasil tersebut dapat dilihat bahwa model berhasil mengidentifikasi gambar sebagai bunga kamboja, yang sesuai dengan label aktual gambar. Keberhasilan ini menunjukkan bahwa model tidak hanya memiliki performa yang baik secara evaluatif, tetapi juga mampu melakukan klasifikasi citra baru secara akurat. Hal ini memperkuat kemampuan generalisasi model dalam mengenali objek di luar data latih, serta menegaskan potensi implementatif model dalam tugas klasifikasi citra bunga secara otomatis dan efisien.

IV. KESIMPULAN

Penelitian ini menunjukkan bahwa arsitektur *Convolutional Neural Network* (CNN) dapat digunakan secara efektif untuk melakukan klasifikasi multikelas citra bunga dengan tingkat akurasi yang cukup baik. Dengan memanfaatkan dataset *"Flowers"* dari Kaggle serta menerapkan teknik augmentasi dan *preprocessing* secara sistematis, model yang dikembangkan mampu mengenali sepuluh jenis bunga secara akurat, baik pada data validasi maupun pada citra baru yang belum pernah dilatih.

Evaluasi model menunjukkan nilai akurasi sebesar 72,48% dan *Mean Absolute Error* (MAE) yang relatif rendah, menandakan kemampuan generalisasi model yang cukup kuat. Meskipun nilai MAPE tidak dijadikan tolok ukur utama karena kurang relevan dalam konteks klasifikasi, metrik lain menunjukkan hasil yang memadai. Implementasi *EarlyStopping* dan *ModelCheckpoint* juga terbukti mampu mencegah *overfitting* dan menjaga stabilitas pelatihan. Berdasarkan hasil-hasil tersebut, dapat disimpulkan bahwa tujuan penelitian untuk merancang sistem klasifikasi citra bunga secara otomatis berbasis CNN telah tercapai. Penelitian ini juga membuka peluang untuk pengembangan lebih lanjut, khususnya dalam peningkatan akurasi melalui tuning parameter dan integrasi model ke dalam *platform* aplikasi digital berbasis pengguna.

UCAPAN TERIMA KASIH

Penulis menyampaikan terima kasih yang sebesar-besarnya kepada Bapak Ridwan Dwi Irawan, S.Kom., M.Kom., selaku dosen pengampu mata kuliah Pengolahan Citra Digital pada semester 8, atas ilmu, arahan, dan motivasi yang telah diberikan, sehingga penelitian ini dapat diselesaikan dengan baik. Ucapan terima kasih juga ditujukan kepada seluruh rekan mahasiswa kelas 21-TIA4 Universitas Duta Bangsa Surakarta atas semangat, kebersamaan, dan dukungan yang telah menjadi bagian penting dalam perjalanan akademik hingga menuju wisuda tahun 2025. Semoga karya ini dapat memberikan kontribusi nyata dalam bidang ilmu komputer serta bermanfaat bagi pengembangan teknologi di masa mendatang.

REFERENSI

[1] M. O. Luruk, A. Y. Rahman, and F. Marisa, "Klasifikasi jenis patung menggunakan metode Convolutional Neural Network (CNN)," *JATISI (Jurnal Teknik Informatika dan Sistem Informasi)*, vol. 12, no. 1, 2025. [Online]. Available: https://jurnal.mdp.ac.id/index.php/jatisi/article/view/9247

[2] A. N. R. Munandar and A. F. Rozi, "Analisis arsitektur Convolutional Neural Network untuk klasifikasi citra bunga," *Jurnal Teknologi dan Sistem Informasi Bisnis*, vol. 6, no. 3, pp. 522–531, 2024. [Online]. Available: http://jurnal.unidha.ac.id/index.php/jteksis/article/view/1413

[3] A. Wibowo, F. H. Subchan, and N. D. Sari, "Klasifikasi Citra Bunga Menggunakan CNN dan Grid Search Hyperparameter Tuning," *Jurnal Teknik Informatika (JUTIF)*, vol. 5, no. 3, pp. 539–547, 2024.

[4] F. Fitriani, "Klasifikasi jenis bunga dengan menggunakan Convolutional Neural Network (CNN)," *TEKNIMEDIA: Teknologi Informasi dan Multimedia*, vol. 2, no. 2, pp. 64–68, 2021. [Online]. Available: https://stmiksznw.ac.id/jurnal/index.php/teknimedia/article/view/39

[5] N. Rumui, A. Mualo, J. Rahayaan, L. Batjo, and M. Mokansi, "Analisis komparasi model deep learning CNN dengan VGG16 dalam klasifikasi jenis bunga," *Informatik: Jurnal Ilmu Komputer*, vol. 21, no. 1, pp. 35–44, 2025. [Online]. Available: https://ejournal.upnvj.ac.id/informatik/article/view/11105

[6] I. G. Perwati, N. Suarna, and T. Suprapti, "Analisis klasifikasi gambar bunga lily menggunakan metode Convolutional Neural Network (CNN) dalam pengolahan citra," *JATI (Jurnal Mahasiswa Teknik Informatika)*, vol. 8, no. 3, pp. 2908–2915, 2024. [Online]. Available: https://mail.ejournal.itn.ac.id/index.php/jati/article/view/9193

[7] I. Rianto and I. P. I. Santosa, *Data Preparation untuk Machine Learning & Deep Learning*. Yogyakarta, Indonesia: Penerbit Andi, 2025.

[8] M. I. Burhanuddin, A. Syaifullah, S. A. P. Jaya, and M. G. Somoal, "Analisis komparatif model MobileNetV1 dan EfficientNetB0 dalam klasifikasi citra empat musim menggunakan transfer learning," *JEKIN - Jurnal Teknik Informatika*, vol. 5, no. 2, pp. 508–521, 2025. [Online]. Available: https://www.rumahjurnal.or.id/index.php/JEKIN/article/view/1378

[9] C. Cahyaningtyas *et al*., *Computer Vision untuk Pemula: Deteksi dan Analisis Ekspresi Wajah dengan CNN*. Yogyakarta, Indonesia: Uwais Inspirasi Indonesia, 2025.

[10] M. H. R. Pratama, M. Akrom, A. P. Santosa, M. R. Rosyid, and L. Mawaddah, "Klasifikasi otomatis korosi menggunakan Convolutional Neural Network dan transfer learning dengan model MobileNetV2," *Jurnal Algoritma*, vol. 22, no. 1, pp. 138–148, 2025. [Online]. Available: https://jurnal.itg.ac.id/index.php/algoritma/article/view/2182

[11] B. Aribowo and S. Fairuz, *Panduan Praktis Machine Learning Klasifikasi Menggunakan Python*. Yogyakarta, Indonesia: Diandra Kreatif, 2024.

[12] A. L. Nasution and R. N. S. Fatonah, *Klasifikasi Kondisi Peralatan Elektronik Metode Gaussian Naive Bayes*. Penerbit Buku Pedia, 2023.

[13] E. Predianto and B. Sutomo, "Klasifikasi Jenis Bunga dengan Algoritma Convolutional Neural Network (CNN) Menggunakan Metode Region-Based Convolutional Neural Network (R-CNN)," *Cyberspace: Jurnal Pendidikan Teknologi Informasi*, vol. 8, no. 2, pp. 1–15, 2024. [Online]. Available: https://jurnal.ar-raniry.ac.id/index.php/cyberspace/article/view/25441

[14] D. K. F. Wantasen, A. Yusupa, dan V. Taringan, "Klasifikasi Uang Kertas Menggunakan Convolutional Neural Network," *Variable Research Journal*, vol. 2, no. 02, pp. 658–668, 2025. [Online]. Available: https://variablejournal.my.id/index.php/VRJ/article/view/225

[15] R. P. Ray, *Analisis Pengaruh Fungsi Aktivasi CNN terhadap Performa Klasifikasi Hewan*, Doctoral dissertation, Universitas Medan Area, 2025. [Online].