SKRIPSI

PENGGUNAAN FITUR WARNA CIELAB DALAM SEGMENTASI KEMATANGAN BUAH KAKAO DENGAN ALGORITMA K-MEANS CLUSTERING

DISUSUN OLEH

ANWAR RUDI SETIAWAN RANGKUTI
2109020142

PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS MUHAMMADIYAH SUMATERA UTARA
MEDAN
2025

PENGGUNAAN FITUR WARNA CIELAB DALAM SEGMENTASI KEMATANGAN BUAH KAKAO DENGAN ALGORIMA K-MEANS CLUSTERING

SKRIPSI

Ditujukan sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer (S.Kom) dalam Program Studi Teknologi Informasi pada Fakultas Ilmu Komputer dan Teknologi Informasi, Universitas Muhammadiyah Sumatra Utara

ANWAR RUDI SETIAWAN RANGKUTI

NPM : 2109020142

PROGRAM STUDI INFORMATIKA

FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS MUHAMMADIYAH SUMATRA UTARA

MEDAN

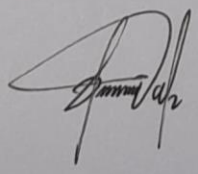
2025

LEMBAR PENGESAHAN

Judul Skripsi	:	PENGGUNAAN FITUR WARNA CIELAB DALAM SEGMENTASI KEMATANGAN BUAH KAKAO DENGAN ALGORITMA K-MEANS CLUSTERING
Nama Mahasiswa	:	Anwar Rudi Setiawan Rangkuti
NPM	:	2109020142
Program Studi	:	Teknologi Informasi

Menyetujui

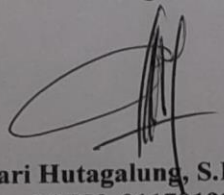
Komisi Pembimbing



(Indah Purnama Sari, S.T., M.Kom)

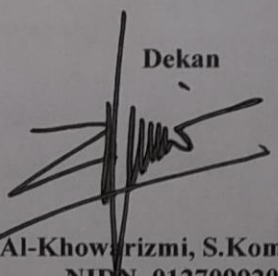
NIDN. 0116049001

Ketua Program Studi



(Fatma Sari Hutagalung, S.Kom., M.Kom.)
NIDN. 0117019301

Dekan



(Dr. Al-Khowarizmi, S.Kom., M.Kom.)
NIDN. 0127099201

PERNYATAAN ORISINALITAS

PENGGUNAAN FITUR WARNA CIELAB DALAM SEGMENTASI KEMATANGAN BUAH KAKAO DENGAN ALGORITMA K-MEANS CLUSTERING

SKRIPSI

Saya menyatakan bahwa karya tulis ini adalah hasil karya sendiri, kecuali beberapa kutipan dan ringkasan yang masing-masing disebutkan sumbernya.

Medan, 19 Agustus 2025

Yang membuat pernyataan

Anwar Rudi Setiawan Rangkuti

NPM. 2109020142

PERNYATAAN PERSETUJUAN PUBLIKASI

KARYA ILMIAH UNTUK KEPENTINGAN

AKADEMIS

Sebagai sivitas akademi Universitas Muhammadiyah Sumatera Utara, saya bertanda tangan di bawah ini:

Nama	: Anwar Rudi Setiawan Rangkuti
NPM	: 2109020142
Program Studi	: Teknologi Informasi
Karya Ilmiah	: Skripsi

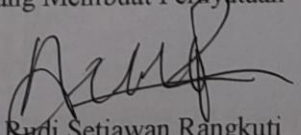
Demi oengembangan ilmu pengetahuan, menyetujui untuk membetikan kepada Universitas Muhammadiyah Sumatera Utara Hak Bebas Royalti Non-Eksekutif ini, *(Non-Exclusive Royalty Free Right)* atas penelitian skripsi yang berjudul:

PENGGUNAAN FITUR WARNA CIELAB DALAM SEGMENTASI KEMATANGAN BUAH KAKAO DENGAN ALGORITMA K-MEANS CLUSTERING

Beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Non-Eksekutif ini, Universitas Muhammadiyah Sumatera Utara berhak menyimpan, mengalih mediah, memformat, mengelola dalam bentuk database, merawat dan mempulikasikan Skripsi saya ini tanpa meminta izin dari saya selama tetap mencantumkan nama saya sebagai penulis dan sebagai pemegang dan atau pemilik hak cipta.

Demikian Pernyataan ini dibuat dengan sebenarnya.

Medan, 19 Agustus 2025
Yang Membuat Pernyataan



Anwar Rudi Setiawan Rangkuti
NPM. 2109020142

RIWAYAT HIDUP

DATA PRIBADI

Nama Lengkap	: Anwar Rudi Setiawan Rangkuti
Tempat dan Tanggal Lahir	: Pematang Sapat, 21 Juni 2002
Alamat Rumah	: Emplasment PTP N VI
Telepon/HP	: 0823 4736 6073
E-mai	: anwarrkt890@gmail.com
Instansi Tempat Kerja	: -
Alamat Kantor	: -

DATA PENDIDIKAN

SD	: SDN 182/VII	TAMAT:2014
SMP	: MTS Musthafawiyah Purbabaru	TAMAT:2017
SMA	: PAKET C PKBM Puspa Kota Medan	TAMAT:2021

KATA PENGANTAR

Assalamualaikum warahmatullahi Wabarakatu

Puji syukur penulis panjatkan kepada Tuhan Yang Maha Esa, atas segala rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan skripsi yang berjudul "PENGGUNAAN FITUR WARNA CIELAB DALAM SEGMENTASI KEMATANG BUAH KAKAO DENGAN ALGORITMA K-MEANS CLUSTERING" ini dengan baik. Skripsi ini disusun sebagai salah satu syarat untuk memperoleh gelar Sarjana pada Program Studi Teknologi Informasi, Fakultas Ilmu Komputer & Teknologi Informasi, Universitas Muhammadiyah Sumatera Utara.

Penulisan skripsi ini tidak lepas dari bantuan, dukungan, serta bimbingan dari berbagai pihak. Oleh karena itu, pada kesempatan ini, penulis ingin menyampaikan rasa terima kasih yang sebesar-besarnya kepada:

1. Bapak Prof. Dr. Agussani, M.AP., Rektor Universitas Muhammadiyah Sumatera Utara (UMSU)
2. Bapak Dr. Al-Khowarizmi, S.Kom., M.Kom. Dekan Fakultas Ilmu Komputer dan Teknologi Informasi (FIKTI) UMSU.
3. Ibu Fatma Sari Hutagalung, S.Kom., M.Kom. Ketua Program Studi Teknologi Informasi .
4. Bapak Mhd. Basri, S.Si, M.Kom Sekretaris Program Studi Teknologi Informasi.

5. Ibu Indah Purnama Sari, S.T., M.Kom selaku pembimbing skripsi yang telah memberikan arahan, saran, dan bimbingan dengan penuh kesabaran dan ketulusan.
6. Almarhum Ayahanda saya dan Ibunda saya yang telah memberikan dukungan dari segi moral maupun materiil serta menjadi motivasi penulis untuk dapat menyelesaikan tugas akhir ini
7. Seluruh Bapak dan Ibu Dosen di Jurusan Teknologi Informasi Fakultas Ilmu Komputer & Teknologi Informasi Universitas Muhammdiyah Sumatera Utara atas ilmu dan didikannya selama perkuliahan.
8. Untuk Putri Rafwani,Terima kasih telah menjadi pasangan yang tak pernah lelah memberi semangat, tawa, dan bahu untuk berbagi lelah. Kehadiranmu membuat setiap langkah terasa lebih ringan dan penuh arti.
9. Sahabat dan teman-teman seperjuangan jurusan teknologi informasi UMSU
10. Semua pihak yang tidak dapat penulis sebutkan satu per satu yang telah membantu dalam penyelesaian skripsi ini.

Penulis menyadari bahwa dalam penulisan skripsi ini masih terdapat kekurangan. Oleh karena itu, kritik dan saran yang membangun sangat penulis harapkan demi perbaikan di masa yang akan datang. Akhir kata, penulis berharap skripsi ini dapat bermanfaat bagi semua pihak yang berkepentingan, khususnya bagi perkembangan ilmu pengetahuan di bidang Nama Bidang Ilmu.

Medan, 19 Agustus 2025

Anwar Rudi Setiawan Rangkuti

NPM. 2009020142

PENGGUNAAN FITUR WARNA CIELAB DALAM SEGMENTASI KEMATANGAN BUAH KAKAO DENGAN ALGORITMA K-MEANS CLUSTERING

ABSTRAK

Penentuan tingkat kematangan buah kakao (Theobroma cacao) merupakan faktor penting dalam memastikan kualitas hasil olahan seperti cokelat. Penelitian ini bertujuan untuk mengeksplorasi efektivitas penggunaan fitur warna CIELAB dalam segmentasi citra buah kakao berdasarkan tingkat kematangannya dengan algoritma K-Means Clustering. CIELAB dipilih karena mampu merepresentasikan persepsi warna manusia secara lebih akurat dibandingkan ruang warna lain seperti RGB atau HSV. Data yang digunakan berupa citra digital buah kakao pada berbagai tingkat kematangan yang diperoleh dari lokasi perkebunan di Kabupaten Deli Serdang. Proses analisis mencakup tahap pra-pemrosesan, konversi ke ruang warna CIELAB, ekstraksi fitur, serta penerapan algoritma K-Means dengan pengujian beberapa nilai *k* untuk menemukan konfigurasi klaster optimal. Hasil segmentasi dievaluasi menggunakan metrik seperti Silhouette Score dan Davies-Bouldin Index, serta visualisasi klaster warna dominan. Hasil penelitian menunjukkan bahwa kombinasi fitur warna CIELAB dan K-Means Clustering mampu mengelompokkan tingkat kematangan buah kakao secara efektif dan objektif.

Kata Kunci : CIELAB, Buah Kakao, Segmentasi Citra, K-Means Clustering, Pengolahan Citra.

ABSTRACT

Determining the ripeness level of cocoa fruit (Theobroma cacao) is an important factor in ensuring the quality of food products such as chocolate. This study aims to explore the effectiveness of using CIELAB color features in segmenting cocoa fruit images based on their ripeness level with the K-Means Clustering algorithm. CIELAB was chosen because it is able to represent human color perception more accurately than other color spaces such as RGB or HSV. The data used are digital images of cocoa fruit at various levels of ripeness obtained from plantation locations in Deli Serdang Regency. The analysis process includes pre-processing stages, conversion to CIELAB color space, feature extraction, and application of the K-Means algorithm by testing several k values to find the optimal cluster configuration. The segmentation results are evaluated using metrics such as Silhouette Score and Davies-Bouldin Index, as well as visualization of dominant color clusters. The results show that the combination of CIELAB color features and K-Means Clustering is able to group the ripeness level of cocoa fruit effectively and objectively.

Keywords : CIELAB, Cocoa Fruit, Image Segmentation, K-Means Clustering, Image Processing

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I

PENDAHULUAN

1.1. Latar Belakang

Kakao (*Theobroma cacao*) merupakan salah satu komoditas pertanian yang memiliki nilai ekonomi tinggi, terutama di negara-negara penghasil seperti Indonesia. Mengingat pentingnya kualitas buah kakao dalam menentukan cita rasa cokelat, pengenalan kematangan buah kakao menjadi aspek krusial dalam proses budidaya dan pengolahan. Kematangan buah kakao yang optimal berpengaruh langsung terhadap kandungan lemak, rasa, dan aroma yang dihasilkan (Ramadhani et al., 2023).

Kematangan buah kakao dapat dinilai melalui berbagai parameter, termasuk warna, ukuran, dan tekstur. Namun, warna buah kakao menjadi indikator yang paling mudah dan cepat untuk diukur. Dalam konteks ini, model warna Cielab menawarkan keunggulan karena dirancang untuk lebih mendekati persepsi manusia terhadap warna. Cielab membagi warna menjadi tiga komponen: L* (lightness), a* (green-red), dan b* (blue-yellow), yang memungkinkan analisis warna yang lebih halus. Penelitian sebelumnya menunjukkan bahwa penggunaan fitur warna Cielab dalam klasifikasi buah dapat meningkatkan akurasi hingga 15% dibandingkan dengan model RGB (Noval Alan Pambudi, Yosep Agus Pranoto, 2021).

Penelitian ini juga akan mempertimbangkan tantangan yang dihadapi dalam penerapan algoritma K-Means Clustering, seperti pemilihan jumlah cluster yang tepat. Jumlah cluster yang tidak sesuai dapat menyebabkan pengelompokan

yang tidak akurat. Oleh karena itu, penelitian ini akan menguji beberapa nilai k (jumlah cluster) untuk menentukan kombinasi yang paling efektif dalam mengidentifikasi kematangan buah kakao. Pengujian yang teliti terhadap parameter algoritma sangat penting untuk mencapai hasil yang optimal dalam klasifikasi (Mahendra et al., 2023).

Kematangan buah kakao merupakan faktor krusial dalam menentukan kualitas biji kakao yang dihasilkan. Salah satu metode yang dapat digunakan untuk mengidentifikasi tingkat kematangan secara akurat adalah dengan memanfaatkan fitur warna Cielab, yang mampu memberikan representasi warna yang lebih objektif dibandingkan metode visual konvensional. Untuk meningkatkan akurasi segmentasi kematangan, algoritma *K-Means Clustering* sering digunakan dalam pengolahan citra. Algoritma ini mampu mengelompokkan buah kakao berdasarkan perbedaan warna, namun efektivitasnya perlu dibandingkan dengan metode lain untuk memastikan keunggulannya dalam klasifikasi. Selain itu, pemilihan jumlah cluster (*nilai k*) dalam algoritma ini memiliki pengaruh signifikan terhadap akurasi segmentasi, sehingga perlu ditentukan secara optimal agar hasil yang diperoleh lebih akurat. Meskipun metode ini menjanjikan, terdapat berbagai tantangan dalam penerapannya, seperti variasi warna alami buah kakao, pencahayaan yang beragam, serta kesulitan dalam menentukan parameter algoritma yang tepat. Oleh karena itu, pengembangan metode yang lebih efektif dalam klasifikasi kematangan buah kakao dapat berkontribusi pada peningkatan efisiensi proses panen dan pengolahan, sehingga berdampak positif terhadap kualitas produksi kakao dalam industri cokelat.

Penelitian ini bertujuan untuk mengeksplorasi penggunaan fitur warna Cielab dalam segmentasi kematangan buah kakao dengan menggunakan algoritma K-Means Clustering. Metode ini diharapkan dapat memberikan solusi yang lebih akurat dan efisien dalam mengidentifikasi tingkat kematangan buah kakao. Pemilihan fitur warna yang tepat merupakan salah satu faktor kunci dalam klasifikasi kematangan buah, di mana Cielab dianggap lebih representatif dibandingkan model warna lainnya seperti RGB atau HSV.

Dengan memanfaatkan fitur warna Cielab dan algoritma K-Means Clustering, diharapkan penelitian ini dapat memberikan kontribusi signifikan terhadap metode identifikasi kematangan buah kakao. Penemuan ini tidak hanya akan bermanfaat bagi petani dan produsen kakao, tetapi juga dapat meningkatkan kualitas produk cokelat yang dihasilkan. Penelitian ini juga akan membuka jalan bagi penelitian lebih lanjut mengenai aplikasi teknologi pengolahan citra dalam industri pertanian.

1.2. Rumusan Masalah

Berdasarkan permasalahan pada latar belakang di atas, penelitian ini akan difokuskan pada beberapa isu utama berikut :

1. Bagaimana tingkat kematangan buah kakao dapat diidentifikasi secara akurat menggunakan fitur warna Cielab?
2. Sejauh mana efektivitas algoritma K-Means Clustering dalam segmentasi kematangan buah kakao dibandingkan dengan metode lain?
3. Bagaimana pemilihan jumlah cluster (nilai k) dalam algoritma K-Means Clustering mempengaruhi akurasi segmentasi kematangan buah kakao?

4. Apa tantangan utama dalam penerapan fitur warna Cielab dan algoritma K-Means Clustering dalam klasifikasi kematangan buah kakao?
5. Bagaimana penerapan metode ini dapat meningkatkan efisiensi dan kualitas hasil produksi kakao dalam industri cokelat?

1.3. Batasan Masalah

Penelitian ini dilakukan dengan batasan-batasan yang ditetapkan secara hati-hati untuk memastikan penelitian tetap fokus dan terarah. Batasan-batasan ini diterapkan agar ruang lingkup penelitian dapat dikelola secara efisien, sehingga menghasilkan keluaran yang sesuai dengan tujuan penelitian. Batasan masalah dalam penelitian ini adalah sebagai berikut:

1. Penelitian ini hanya akan menggunakan fitur warna Cielab dalam analisis kematangan buah kakao dan tidak mempertimbangkan parameter lain seperti ukuran, tekstur, atau kandungan kimia buah.
2. Algoritma yang digunakan dalam segmentasi kematangan buah kakao terbatas pada *K-Means Clustering*, tanpa membandingkan dengan metode lain seperti Deep Learning atau algoritma segmentasi berbasis morfologi.
3. Penelitian ini hanya akan menguji beberapa nilai k *(jumlah cluster*) tertentu yang dipilih berdasarkan penelitian sebelumnya dan eksperimen awal untuk menemukan konfigurasi terbaik.
4. Data yang digunakan dalam penelitian ini hanya berupa gambar buah kakao dalam kondisi pencahayaan tertentu, tanpa mempertimbangkan variasi cahaya alami yang ekstrem atau kondisi lingkungan lainnya.

5. Penelitian ini bersifat eksploratif dan hanya dilakukan dalam skala laboratorium atau simulasi, sehingga tidak mencakup uji coba langsung di lapangan atau penerapan dalam sistem otomasi industri secara real-time.

1.4. Tujuan Penelitian

Tujuan dari penelitian ini diformulasikan untuk memberikan arah yang jelas terhadap pengembangan dan penerapan segmentasi kematangan buah kakao. Tujuan yang ingin dicapai dalam penelitian ini adalah sebagai berikut:

1. Menganalisis penggunaan fitur warna Cielab dalam segmentasi kematangan buah kakao untuk menentukan efektivitasnya dibandingkan dengan model warna lainnya.
2. Menerapkan algoritma *K-Means Clustering* dalam klasifikasi tingkat kematangan buah kakao berdasarkan fitur warna Cielab.
3. Menentukan jumlah cluster (nilai k) yang optimal dalam algoritma *K-Means Clustering* untuk mendapatkan segmentasi kematangan buah kakao yang paling akurat.
4. Mengevaluasi tantangan dalam penerapan algoritma *K-Means Clustering* pada segmentasi kematangan buah kakao, termasuk faktor yang dapat mempengaruhi akurasi segmentasi.
5. Memberikan kontribusi dalam pengembangan metode identifikasi kematangan buah kakao yang lebih akurat dan efisien, sehingga dapat membantu petani dan produsen kakao dalam meningkatkan kualitas hasil panen.

1.5. Manfaat Penelitian

Penelitian ini diharapkan dapat memberikan berbagai manfaat, baik dalam pengembangan teknologi yang lebih praktis untuk pengenalan segmentasi

kematangan buah kakao. Manfaat yang diharapkan dari penelitian ini adalah sebagai berikut:

1. Membantu petani dan produsen kakao dalam mengidentifikasi tingkat kematangan buah secara lebih akurat, sehingga dapat meningkatkan kualitas hasil panen dan produk olahan kakao.
2. Meningkatkan efisiensi dalam proses seleksi buah kakao, dengan mengurangi ketergantungan pada metode manual yang sering kali subjektif dan kurang konsisten.
3. Membuka peluang implementasi teknologi pengolahan citra dalam industri kakao, yang dapat diterapkan dalam sistem otomasi dan pemantauan kualitas buah secara real-time.
4. Mengurangi risiko kesalahan dalam pemilihan buah kakao, sehingga dapat berdampak positif pada produksi cokelat berkualitas tinggi.
5. Menjadi referensi bagi penelitian selanjutnya yang ingin mengembangkan metode klasifikasi kematangan buah menggunakan teknik serupa atau metode yang lebih canggih seperti *Deep Learning*.

BAB II

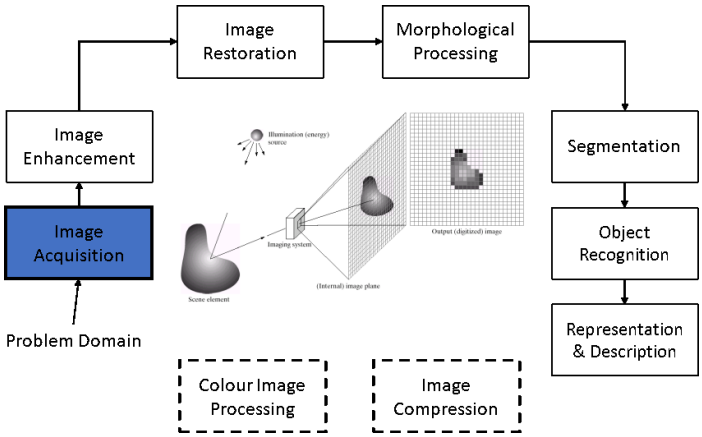
LANDASAN TEORI

2.1. Pengolahan Citra

Pengolahan citra adalah suatu gambar atau kemiripan dengan suatu objek. Citra analog tidak dapat direpresentasikan dalam komputer, sehingga tidak bisa diproses oleh komputer secara langsung. Tentu agar bias diproses di komputer, citra analog harus dikonversi menjadi citra digital. Citra digital adalah citra yang dapat diolah oleh komputer. Sedangkan citra yang dihasilkan dari pralatan digital (citra digital) langsung bisa diolah oleh komputer. Penyebabnya karena di dalam peralatan digital terdapat sistem sampling dan kuantisasi. Sedangkan peralatan analog tidak dilengkapi kedalam sistem tersebut. Sistem sampling adalah sistem yang mengubah kontinu menjadi citra digital dengan cara membagi citra analog menjadi M baris dan N kolom, sehingga menjadi citra diskrit. Semakin besar nilai M dan N, semakin halus citra digital yang dihasilkan. Pertemuan antara baris dan kolom tersebut piksel. Sistem kuantisasi adalah sistem yang melakukan pengubahan intensitas analog ke intensitas diskrit, sehingga dengan proses ini dimungkinkan untuk membuat gradasi warna sesuai dengan kebutuhan. Kedua sistem inilah yang bertugas untuk memotong-motong citra menjadi M baris dan N kolom (proses sampling) sekaligus menentukan besar intensitas yang terdapat di titik tersebut (proses kuantisasi), sehingga menghasilkan resolusi citra yang diinginkan (Jatmika et al., 2020).

Dalam hal lain komputer memiliki cara pandang tersendiri dalam menangani citra yang berbeda dengan manusia. Manusia memiliki kemampuan

mengekstrak informasi yang terkandung dalam citra melalui indra penglihatan dengan mudah. Namun berbeda dengan computer perlu beberapa tahapan khusus yang cukup rumit untuk dapat mengekstrak informasi yang terkandung dalam citra. Tahapantahapan tersebut antaralain seperti proses akuisi data, manipulasi data, visualisasi data, serta proses penyimpanan data (Noval Alan Pambudi, Yosep Agus Pranoto, 2021).



Gambar 2.1 Konsep Pengolahan Citra Digital

(Sumber : Wikipedia. (2025). Fig 1 Kata kunci pengolahan citra digital. Wikipedia. https://commons.wikimedia.org/wiki/File:Fig_1_Kata_kunci_pengolahan_citra_digital.png)

Resizing merupakan tahapan untuk merubah citra menjadi lebih besar ataupun lebih kecil dari ukuran citra aslinya. Hal ini dapat mengakibatkan pergeseran pada nilai warna sehingga mengubah konten digital didalamnya. Pada kasus tertentu resizing juga tertujuan untuk menyamakan ukuran suatu citra input dengan yang lainnya agar mempercepat proses penyelesaian dari suatu kasus

tersebut. Terhadap beberapa algoritma pengubahan ukuran citra yang biasa dipakai, yaitu replikasi pixel, interpolasi bilincar, dan bicubic (Wijaya & Prayudi, 2015).

2.2. Buah Kakao

Buah kakao merupakan salah satu makanan yang populer diseluruh dunia. Pengembangan tanaman kakao di arahkan pada peningkatan lahan yang luas dan peningkatan produksi dan mutu hasil. Buah kakao merupakan salah satu komoditas tanaman perkebunan yang berperan dalam meningkatkan pendapatan negara selai itu tanaman kakao merupakan salah satu komoditas tanaman perkebunan penghasil ekspor yang berperan penting bagi perekonomian (Mayrowani, 2016).

Kakao merupakan tumbuhan berwujud pohon dimana biji dari tumbuhan ini digunakan sebagai produk olahan yang dikenal sebagai cokelat. Habitat asli tanaman kakao adalah hutan tropis dengan naungan pohon pohon yang tinggi, curah hujan tinggi, suhu sepanjang tahun relatif sama, serta kelembaban tinggi yang relatif tetap. Warna buah kakao sangat beragam, tetapi pada dasarnya hanya ada dua macam warna. Buah yang ketika muda berwarna hijau atau hijau agak putih jika sudah matang akan berwarna kuning. Sementara itu, buah yang ketika muda berwarna merah, setelah matang berwarna kuning (Yogiswara et al., 2016).

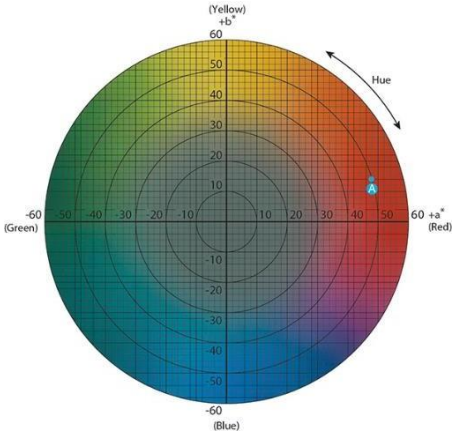
Table 2.1 Tingkat Kematangan Buah Kakao

No.	Gambar	Keterangan
1.		Gambar disamping menunjukkan contoh buah kakao yang termasuk dalam katagori belum matang yang ditandai dengan warna merah
2.		Gambar disamping menunjukkan contoh buah kakao yang termasuk dalam katagori menuju matang dan belum sepenuhnya matang ditandai dengan warna merah kekuningan
3.		Gambar disamping menunjukkan contoh buah kakao yang termasuk dalam katagori tahap awal kematangan atau separuh jalan menuju matang ditandai dengan warna merah yang dilumuri warna kuning
4.		Gambar disamping menunjukkan contoh buah kakao yang termasuk dalam katagori matang sempurna ditandai dengan warna jingga kekuningan

2.3. Ruang Warna CIELAB

Ruang warna CIELAB yang juga dikenal sebagai CIELAB adalah ruang warna yang ditetapkan oleh komisi internasional tentang iluminasi warna (French Commission Internasionale de l'eclairage, dikenal dengan sebutan CIE), dimana mampu menggambarkan semua warna yang dapat dilihat oleh mata manusia (Rulaningtyas et al., 2015). CIELAB merupakan model warna yang dirancang untuk menyerupai luminance (pencahayaan), a dan b sebagai dimensi kromatisitas

(komunikasi Warna Presisi, 2016). L*:0 (hitam); 100 (putih) menyatakan cahaya pantul yang menghasilkan warna akromatik putih, abu-abu dan hitam. Notasi a*: warna kromatik campuran merah hijau dengan nilai +a*(positif) dari 0 sampai +80 untuk warna merah nilai -a* (negatif) dari 0 sampai 80 untuk warna hijau. Notasi b*: warna kromatik campuran birukuning dengan nilai +b* (positif) dari 0 sampai +70 untuk warna kuning dan nilai -b* (negatif) dari 0 sampai -70 untuk warna biru (Noval Alan Pambudi, Yosep Agus Pranoto, 2021).



Gambar 2.2 Diagram Warna CIELAB

(Sumber: https://images.app.goo.gl/dF4Tzq8uXxN86DaP7)

Dalam penelitian ini ruang warna yang dipakai sebagai parameter yaitu ruang warna CIELAB. Model warna ini dipilih karena terbukti memberikan hasil yang lebih baik dari pada model warna RGB dalam mengukur nilai kemiripan ciri warna yang lebih akurat dan mengatur kontras pencahayaan dari model warna RBG (Rusman et al., 2023). Dalam penelitian Riries Rulaningtyas segmentasi citra berwarna dengan menggunakan metode clustering berbasis patch untuk identifikasi mycrobacterium tuberculosis. Beberapa metode telah dilakukan dalam penelitian

ini, yaitu adaptive color thresholding pada ruang warna RGB, HSV, CIELAB, yang memberikan hasil segmentasi yang baik pada ruang warna CIELAB.

Berdasarkan buku yang berjudul (Teori dan Aplikasi Pengolahan Citra, Abdul kadir Adhi Susanto, 2013). Berikut merupakan proses transformasi warna RGB menjadi CIELAB, dimulai dengan melakukan perhitungan sebagai berikut:

$$X = 0.412453R + 0.357580G + 0.180423B \quad (2.12)$$

$$Y = 0.212671R + 0.715160G + 0.072169B \quad (2.13)$$

$$Z = 0.019334R + 0.119193G + 0.950227B \quad (2.14)$$

Keterangan:

X = hasil transformasi intensitas warna RGB ke komponen X dalam format XYZ.

Y = hasil transformasi intensitas warna RGB ke komponen Y dalam format XYZ.

Z = hasil transformasi intensitas warna RGB ke komponen Z dalam format XYZ.

Selanjutnya, L*a*b* di definisikan sebagai berikut:

$$L^* = 116f\left(\frac{Y}{Y_n}\right) - 16 \quad (2.15)$$

$$a^* = 500\left[f\left(\frac{X}{X_n}\right) - f\left(\frac{Y}{Y_n}\right)\right] \quad (2.16)$$

$$b^* = 200\left[f\left(\frac{Y}{Y_n}\right) - f\left(\frac{Z}{Z_n}\right)\right] \quad (2.17)$$

Keterangan:

- L* adalah luminance atau kecerahan warna.
- a* adalah komponen warna yang menggambarkan perbedaan antara warna hijau dan merah.
- b* adalah komponen warna yang menggambarkan perbedaan antara warna biru dan kuning.
- Y dan Z adalah nilai-nilai koordinat warna dalam sistem XYZ.
- Y_n adalah nilai luminance normal yang digunakan sebagai referensi.
- f adalah fungsi transformasi yang tergantung pada nilai Y/Y_n.

2.4. Clustering

Cluster merupakan kumpulan objek data yang memiliki kemiripan antara satu dengan yang lain dalam kelompok yang sama dan berbeda objek dengan data kelompok lain. Clustering atau lebih dikenal dengan analisis cluster merupakan proses pengelompokan satu set benda fisik ataupun abstrak ke dalan satu kelas objek yang sama (Nabila et al., 2021). Clustering data di bedakan menjadi dua tujuan, yaitu clustering untuk pemahaman dan clustering untuk penggunaan. Jika tujuan untuk pemahaman maka cluster yang terbentuk harus menangkap struktur alami data, biasanya proses awal untuk kemudian dilanjutkan dengan pekerjaan ini seperti summarization (rata-rata, standar deviasi), pelabelan kelas pada setiap kelompok untuk kemudian digunakan sebagai data latih klarifikasi, dan sebagainya. Sementara jika tujuannya untuk penggunaan, biasanya mencari prototype cluster

yang paling representative terhadap data dan memberikan abstraksi dan setiap objek data dalam cluster dimana sebuah data terletak didalamnya (Lesmana et al., 2019).

Banyak metode clustering yang sudah dikembangkan oleh para ahli, dan memiliki karakter, kelebihan, dan kekurangan pada masing-masing metode. Cluster dapat dibedakan menurut struktur Cluster, keanggotaan data dalam cluster dan kekompakan data dalam cluster. Metode clustering menurut strukturnya di bagi menjadi dua yaitu pengelompokan hirarki dan partitioning. Pengelompokan hirarki memiliki aturan satu data tunggal bisa dianggap sebagai sebuah kelompok, dua atau lebih kelompok kecil dapat bergabung menjadi satu kelompok besar dan begitu seterusnya hingga semua data dapat bergabung menjadi satu kelompok (Noval Alan Pambudi, Yosep Agus Pranoto, 2021).

2.5. K-means Clustering

K-Means merupakan metode penganalisaan data pada Data Mining dimana proses pemodelan tanpa supervisi dan merupakan salah satu metode yang mengelompokan data secara partisi. Pada metode K-Means data dikelompokan menjadi beberapa kelompok dimana setiap kelompok mempunyai karakteristik yang mirip atau sama dengan lainnya namun dengan kelompok lainnya memiliki karakteristik yang berbeda. Metode ini menimalisasi perbedaan antar data di dalam satu cluster serta memaksimalkan perbedaan dengan cluster yang lain (Yunita, 2018). K-means merupakan salah satu metode data clustering non hirarki. Metode ini bertujuan untuk, pengelompokan data yang ada ke dalam kelompok cluster. Data-data yang memiliki karakteristik yang sama dikelompokan dalam satu cluster

yang lain sehingga data yang berada dalam satu cluster kelompok memiliki tingkat variasi yang kecil (Noval Alan Pambudi, Yosep Agus Pranoto, 2021).

Metode K-Means memiliki karakteristik sebagai berikut (Sibuea & Safta, 2017):

1. K-Means merupakan metode pengelompokan yang sederhana dan dapat digunakan dengan mudah.
2. Pada jenis data set tertentu, K-Means tidak dapat melakukan segmentasi data dengan baik di mana hasil segmentasi tidak dapat menentukan pola kelompok yang mewakili karakteristik bentuk alami data.
3. K-Means bias menalami masalah ketika mengelompokkan data yang mengandung outlier.

   Secara umum metode K-Means menggunakan algoritma sebagai berikut:

   a. Tentukan k sebagai jumlah cluster yang di bentuk. Penentuan banyaknya jumlah cluster k dilakukan dengan beberapa faktor seperti pertimbangan teoritis dan konseptual yang diusulkan untuk menentukan berapa banyak cluster.
   b. Bangkitkan k Centroid (titik pusat cluster) awal secara random. Untuk menentukan centroid awal dilakukan secara acak dari beberapa objek yang tersedia sebanyak

k cluster, untuk menghitung centroid cluster ke-i berikutnya, menggunakan rumus sebagai berikut:

$$v = \frac{\sum_{i=1}^{n} x_i}{n} \quad ;i = 1,2,3,...n$$

Dimana; v: centroid pada cluster

Xi: objek ke-i

n: banyaknya objek atau jumlah objek yang menjadi anggota cluster.

c. Hitung jarak setiap objek ke masing-masing centroid dari masing-masing cluster. Kemudian hitung jarak antara objek dengan centroid, dalam penelitian ini menggunakan Euclidian Distance.

$$d(x,y) = \|x - y\| = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \quad ;i = 1,2,3,...,n$$

Dimana; xi: objek x ke-i

y: daya y ke-i

n: banyaknya objek

d. Alokasikan masing-masing objek ke dalam centroid yang paling terdekat.

e. Lakukan iterasi, kemudian tentukan posisi centroid baru dengan menggunakan persamaan.

f. Ulangi langkah 3 jika posisi centroid baru tidak sama.

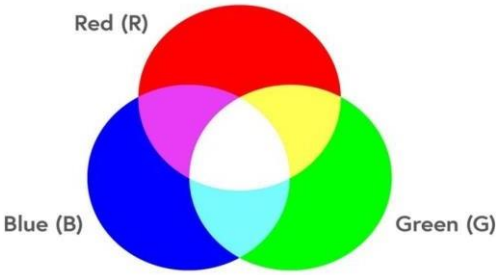
Proses penggabungan titik dilakukan dengan membandingkan matriks kumpulan tugas-tugas pada iterasi

sebelumnya dengan matrik kumpulan tugas-tugas pada iterasi yang sedang berjalan. Jika hasilnya sama maka algoritma k means cluster analysis sudah konvergen, tetapi jika berbeda maka belum konvergen sehingga perlu dilakukan iterasi berikutnya.

2.6. RGB (*Read, Green, And Blue*)

RGB adalah singkatan dari Red-Green-Blue, merupakan tiga warna dasar (primary colors) yang secara umum dijadikan acuan warna lainnya dari basis RGB, dan dapat mengkonversi warna menjadi kode-kode angka yang membuat warna tersebut akan tampil universal. Ruang warna ini menggunakan tiga komponen dasar yaitu merah (Red), hijau (Green), dan biru (Blue) yang diolah menggunakan berbagai teknik untuk mendapatkan berbagai macam warna. Mengolah citra RGB menjadi citra keabuan merupakan salah satu sampel dari pengolahan citra operasi titik. Dalam pemroresan RGB menjadi keabuan dengan cara mengitung rerata intensitas RGB setiap pixel penyusun citra (Ratna, 2020). RGB merupakan ruang warna yang tersusun dari tiga channel warna yakni channel merah (Red), channel hijau (Green), dan channel biru (Blue). Setiap channel mempunyai intensitas warna yang berbeda, nilai intensitas warna minimum adalah nol (0) dan nilai intensitas warna maksimum adalah 255 (8 bit). Setiap piksel citra RGB memiliki variasi warna sejumlah 16.777.216 (256 x 256 x 256). RGB adalah ruang warna aditif yang berarti semua warna dimulai dari hitam dan dibentuk dengan menambahkan warna

dasar R, G dan B. Setiap warna yang tampak merupakan kombinasi dari tiga komponen R, G dan B (Mubarok et al., 2021).



Gambar 2.3 Warna RGB

(Sumber : https://images.app.goo.gl/89j1Ji163e18TjGj6)

2.7. HSV *(Hue, Saturation, And Value*)

Hue yang berarti ukuran panjang gelombang pada warna yang dominan berdasarkan persepsi mata manusia. Saturation menggambarkan banyaknya cahaya putih yang bercampur pada channel hue (Yessy Nabella et al., 2019). Brightness atau value adalah intensitas pemantulan objek yang diterima oleh mata. Intensitas tersebut berarti adanya perubahan warna putih menjadi abu-abu sampai hitam. Berikut merupakan rumus dan langkah-langkah untuk mengkonversi RGB ke HSV (OpenCV, 2017). Segmentasi dengan deteksi HSV dilakukan dengan menganalisis nilai warna tiap piksel citra sesuai fitur yang diinginkan dengan nilai toleransi pada

setiap dimensi warna HSV. Ruang warna HSV terdiri dari 3 komponen, yaitu H menunjukkan jenis warna (seperti merah, biru atau kuning) atau corak warna, yaitu tempat warna tersebut ditemukan dalam spectrum warna, S mewakili tingkat dominasi warna yaitu ukuran seberapa besar kemurnian dari warna tersebut, sedangkan V mewakili tingkat kecerahan yaitu ukuran seberapa besar kecerahan suatu warna seberapa besar cahaya datang dari suatu warna (Areni et al., 2019).

1. Normalisasi citra, dengan membagi nilainya dengan 255 yang ditampilkan pada Persamaan 4 sampai 6.

   $H = H\ /255$ (4)

   $S = S\ /255$ (5)

   $V = V\ /255$ (6)

2. Konversi ke HSV, dengan menggunakan Persamaan 7 sampai 9.

$$V = \max(R, G, B) \quad (7)$$

$$S = \begin{cases} \frac{V - \min(R,G,B)}{V}, if\ v\ != 0 \\ 0\ otherwise \end{cases} \quad (8)$$

$$H = \begin{cases} \frac{60(G-B)}{(V-\min(R,G,B))}\ if\ V = R \\ \frac{120+60(B-R)}{(V-\min(R,G,B))}\ if\ V = G \\ \frac{240+60(R-G)}{(V-\min(R,G,B))}\ if\ V = B \end{cases} \quad (9)$$

3. Mengubah citra ke 8 bit image, dengan menggunakan persamaan 10 sampai 12.

$$V = V\ x\ 255 \quad (10)$$

$$S = S\ x\ 255 \quad (11)$$

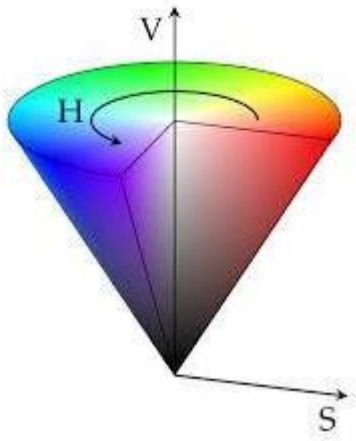
$$H = \frac{H}{2} \quad (12)$$

   Keterangan: H: nilai piksel HSV pada channel H

   S: nilai piksel HSV pada channel S

V: nilai piksel HSV pada channel V

Fitur Statistik Warna oleh Kadir & Susanto (2013) dalam bukunya, dapat dihitung menggunakan Persamaan 13 sampai 16.



Gambar 2.4 Warna HSV

(Sumber : https://images.app.goo.gl/iYSnHJVQg7UVpkks7)

2.8. Google Colab

Google Colaboratory (sering disebut sebagai *Google Colab*) merupakan layanan *cloud computing* yang disediakan oleh Google yang memungkinkan pengguna menulis dan menjalankan kode Python langsung melalui peramban (browser). Google Colab sangat populer dalam dunia pendidikan dan riset karena kemampuannya untuk menyediakan lingkungan pemrograman Python yang bebas instalasi, mudah diakses, serta mendukung kolaborasi daring. Google Colab menyediakan aksesibilitas tinggi bagi pengguna tanpa memerlukan instalasi

perangkat lunak tambahan. Platform ini sangat cocok digunakan dalam pembelajaran pemrograman, khususnya bagi pemula, karena antarmuka yang ramah pengguna dan kemudahan penggunaan (Febby Wilyani et al., 2024).

Google Colab memungkinkan pelaksanaan praktik pemrograman Python secara langsung, termasuk pemrosesan data dan *machine learning*, dengan dukungan sumber daya komputasi yang memadai. Hal ini menjadikan Google Colab sebagai solusi yang efisien untuk pelatihan, pembelajaran interaktif, dan eksperimen berbasis data dalam konteks pendidikan dan penelitian ilmiah. Dengan kemampuannya menjalankan *notebook* berbasis Jupyter, pengguna dapat menggabungkan kode, teks, gambar, dan visualisasi data dalam satu dokumen yang interaktif. Keunggulan ini menjadikan Google Colab tidak hanya sebagai alat belajar pemrograman, tetapi juga sebagai media dokumentasi dan presentasi hasil analisis secara langsung (Febby Wilyani et al., 2024).

Gambar 2.5 Aplikasi Google Colab

(Sumber : https://colab.research.google.com/?hl=id)

Google Colab merupakan layanan berbasis cloud yang memungkinkan pengguna menulis, menjalankan, dan membagikan kode Python secara daring tanpa perlu menginstal perangkat lunak tambahan di komputer pengguna. Hal ini memberikan kemudahan akses dan fleksibilitas dalam pembelajaran dan pengembangan aplikasi. Secara keseluruhan, kolaborasi antara penggunaan Python dan Google Colaboratory memberikan pendekatan yang efektif dalam pengajaran pemrograman. Kombinasi keduanya tidak hanya memudahkan dari sisi teknis dan akses, tetapi juga memberi pengalaman belajar yang interaktif, praktis, dan relevan dengan kebutuhan industri teknologi saat ini (Hidayatullah & Berliana, 2023).

Google Colab digunakan sebagai platform pengembangan dan pengujian kode untuk segmentasi citra buah kakao berdasarkan fitur warna CIELAB menggunakan algoritma K-Means Clustering. Colab memberikan fleksibilitas dalam menjalankan eksperimen dan melakukan visualisasi hasil tanpa memerlukan spesifikasi komputer lokal yang tinggi, sehingga sangat sesuai untuk pengolahan data citra berukuran besar. Colab juga mendukung penggunaan GPU secara gratis, sehingga mempercepat proses pelatihan dan pengolahan data skala besar seperti citra buah kakao.

2.9. Penelitian Terkait

Penelitian terkait Klasifikasi Kematangan Buah Kakao Menggunakan Algoritma *K-Means* berbasis pengolahan citra dapat dilihat pada table 2.2 sebagai berikut:

Tabel 2.2 Penelitian Terkait

No	Nama Tahun dan Judul Penelitian	Metode Penelitian	Hasil Penelitian	Perbedaan penelitian
1.	Laylatul Nadliroh Imana 2019, Klasifikasi Tingkat Kematangan Buah Kakao Menggunakan Ekstraksi Ciri Local Binary Pattern (LBP) dan Neuro Fuzzy.	Metode K NN *(K Nearest Neihgbors*) Neuro Fuzzy, fitur ekstraksi Tekstur, klasifikasi LBP (*Local Binary Pattern*).	Klasifikasi Tingkat kematangan buah kakao *K-Nearest Neibghbor* (K-NN) dan Discrete Cosine Transform, hasil pengujian sistem didapatkan akurasi tertinggi 83,33% dan waktu komputasi tercepat 21,12 detik.	Perbedaannya terdapat ekstraksi dari fitur dimana ekstraksi Discrete Transform menguji ciri *Cosine transform* sedangkan peneliti ini mengandalkan algoritma *K-Means Clustering* yang lebih fokus pada segmentasi berbasis warna dengan memanfaatkan ruang warna Cielab
	Indriyana Naomi Clarita Sinaga 2020,	Metode K NN (*K Nearest*	Hasil Pengujian dan analisis dapat	Terdapat perbedaannya

2.	Klasifikasi Tingkat Kematangan Buah Kakao Menggunakan Metode Discrete Cosine Transform dan *K-Nearst Neighbor*(K-NN).	*Neihgbors*) dan Discrete Cosine Transform fitur ekstraksi ciri, dan klasifikasi RGB	disimpulkan bahwa parameter terbaik untuk sistem ini adalah ukuran resize citra 512×512 pixel, blok DCT 512, mean sebagai ciri statistic, nilai k=1, dan eucledien sebagai jenis distance, dan hasil pengujian sistem didapatkan akurasi tertinggi 83,33% dan waktu komputasi tercepat 21,12 detik.	menggunakan 2 metode yaitu K-NN *(K-Nearest Neihgbors)* dan DCT, fitur ektraksi ciri dan klasifikasi RGB sedangkan penelitian ini lebih menitikberatkan pada segmentasi warna untuk menentukan kematangan buah kakao menggunakan CIELAB dan *k-means clustering*
3.	Izha Mahendra 2019, Klasifikasi Tingkat Kematangan Buah Kakao Berdasarkan Fitur Warna Menggunakan	Metode Euclidean Distance, K NN *(K Nearest Neihgbors)* ekstraksi fitur warna dan klasifikasi HSV.	Metode Euclidean Distance, K NN *(K Nearest Neihgbors*) ekstraksi fitur warna dan klasifikasi HSV.	Perbedaanya adalah lebih berfokus pada pengelompokan warna secara mandir

	Algoritma *K Nearest Neighbor*(K-NN).			menggunakan *k-means* CIELAB.
4.	M Farhan Hussaini Dermawan 2023, Penerapan Image Processing untuk Mengetahui Tingkat Kematangan Kopi Menggunakan Algoritma *K Nearest Neighbor* (K-NN) pada Perkebunan Kopi Malabar Bandung.	Metode K NN (*K Nearest Neihgbors*) fitur ekstraksi warna dan klasifikasi HSV.	Penelitian ini dan hasil akurasi pada K=3 menunjukkan performa yang baik dalam menentukan tingkat kematangan kopi yaitu pada 93,33%.	Terdapat perbedaan pada objeknya yaitu tingkat kematangan kopi sedangkan peneliti berobjek kematangan buah kakao.
5.	Rini Mulyani 2021, Klasifikasi Kematangan Buah Pala Menggunakan Metode *K Nearest Neighbor* (K-NN) Dengan Memanfaatkan Teknologi Citra Digital.	Metode K NN *(K Nearest Neihgbors*) fitur ekstraksi warna dan klasifikasi HSV.	Pengujian data testing memiliki rata-rata akurasi yaitu K=3 akan menghasilkan keakuratan sebesar 68%. Sehingga sistem klasifikasi kematangan buah pala berdasarkan ekstraksi fitur warna Hue, Saturation, Value	Terdapat perbedaan tentang buah kakao menggunakan *K-Means Clustering* untuk segmentasi warna berdasarkan ruang warna Cielab, tanpa memerlukan data referensi atau

			(HSV) dan moment Zernike menggunakan metode K-Nearest Neighbors (K-NN) tersebut layak untuk digunakan sebagaimananya.	proses klasifikasi berbasis jarak.
6.	Heru Pramono Hadi 2022, Analisa Fitur Ekstraksi Ciri dan Warna Dalam Proses Klasifikasi Kematangan Buah Rambutan Berbasis *K Nearest Neighbors* (K NN).	Metode K NN *(K Nearest Neihgbors)* fitur ekstraksi tekstur dan ekstraksi warna, klasifikasi GLCM dan HSV.	Penelitian menggunakan K-NN dengan ekstraksi GLCM menunjukkan akurasi tinggi, mencapai 97,5% pada K=1 dan 62,5% pada K=13. K=3 memberikan hasil terbaik dengan akurasi 93,33%, menetapkan kualitas kopi dengan baik. Pengujian data testing pada K=3 juga menghasilkan akurasi rata-rata 68%,	Perbedaan penelitian ini tidak melibatkan ekstraksi tekstur atau klasifikasi berbasis referensi, melainkan mengelompokkan warna secara mandiri untuk membedakan tingkat kematangan.

			menegaskan kecocokan sistem untuk menentukan kematangan buah pala berdasarkan ekstraksi fitur HSV dan moment Zernike menggunakan K-NN.	
7.	Setya Putra Adenugraha 2022, Klasifikasi Kematangan Buah Pisang Ambon Menggunakan Metode K-NN *(K Nearest Neihgbors)* dan PCA Berdasarkan Citra RGB dan HSV.	Motode K NN *(K Nearest Neihgbors)* dan PCA, ektraksi fitur Warna dan Klasifikasi RGB, HSV.	Hasil akurasi dari klasifikasi tingkat kematangan buah pisang ambon menggunakan metode K-NN sebesar 90,9 % dengan nilai K=5 yang didapat dari 10 data uji dengan klasifikasi akurat, dan 1 data uji dengan klasifikasi tidak akurat.	lebih mengutamakan pengelompokan warna otomatis, sehingga tidak memerlukan dataset pelatihan seperti dalam metode K-NN *(K Nearest Neihgbors)*
8.	A Isatul Masruroh 2023, Klasifikasi Tingkat Kematangan	Metode K NN *(K Nearest Neihgbors)*,	Dari hasil percobaan dan pengujian menggunakan 65 data	Penelitian ini lebih berfokus pada segmentasi warna

	Buah Pepaya California dalam Ruang Warna HSV dengan algoritma K-NN (K-Nearest Neihgbors).	ekstraksi fitur warna dan klasifikasi HSV.	citra buah pepaya California, metode K NN (K-Nearest Neighbors) memberikan akurasi sebesar 86,6667%. Hasil ini dikategorikan sebagai klasifikasi yang baik berdasarkan confusion matrix.	secara otomatis tanpa perlu dataset pelatihan atau klasifikasi berbasis referensi seperti dalam K-NN. Selain itu, penelitian ini menggunakan ruang warna Cielab, yang lebih stabil terhadap perubahan pencahayaan dibandingkan HSV.
9.	Fandy Indra Pratama 2023, Klasifikasi Kematangan Buah Apel Berdasarkan Warna dan Tekstur Menggunakan Algoritma K-NN.	Metode K NN (K Nearest Neihgbors), ekstraksi fitur warna dan tekstur dan klasifikas HSV.	sebanyak 12 buah apel yang terdiri dari matang, sedang, dan mentah menghasilkan tingkat kematangan buah apel sebesar 91,6667% dari total akurasi.	perbedaannya lebih fleksibel untuk analisis visual kematangan buah secara otomatis, tanpa bergantung pada data latih dan uji

Keterbaharuan penelitian ini terletak pada beberapa aspek utama yang membedakannya dari penelitian-penelitian sebelumnya. Penelitian ini menggunakan ruang warna CIELAB untuk ekstraksi fitur warna, yang lebih stabil terhadap perubahan pencahayaan dan lebih mendekati persepsi warna manusia, berbeda dengan penelitian lain yang lebih sering menggunakan ruang warna RGB atau HSV seperti yang dilakukan oleh Izha Mahendra (2019), Fandy Indra Pratama (2023), dan A Isatul Masruroh (2023). Selain itu, penelitian ini memanfaatkan algoritma *K-Means Clustering* untuk segmentasi kematangan buah kakao, yang merupakan teknik unsupervised learning, berbeda dengan penelitian sebelumnya yang lebih fokus pada klasifikasi menggunakan K-NN *(K-Nearest Neighbors*). Teknik segmentasi ini mengelompokkan piksel berdasarkan kemiripan warna dalam ruang CIELAB, memungkinkan segmentasi yang lebih alami dan akurat tanpa bergantung pada label data. Penelitian ini juga memiliki fokus yang lebih spesifik pada buah kakao, sementara banyak penelitian sebelumnya mengkaji buah lainnya seperti apel, pepaya, pisang, atau kopi. Secara keseluruhan, penelitian ini memberikan pendekatan baru dalam segmentasi kematangan buah kakao dengan menggunakan CIELAB dan *K-Means Clustering*, yang membedakannya dari penelitian sebelumnya yang lebih mengandalkan metode klasifikasi berbasis K-NN (*K-Nearest Neighbors).*

BAB III

METODOLOGI PENELITIAN

3.1. Pengumpulan Data

Pengumpulan data merupakan langkah awal yang krusial dalam penelitian ini. Data yang digunakan terdiri dari gambar buah kakao yang diambil dari kebun kakao didaerah. Pengambilan gambar dilakukan dengan menggunakan kamera untuk memastikan bahwa detail warna dan tekstur buah dapat terlihat jelas. Dalam penelitian ini, jumlah gambar yang dikumpulkan mencapai 100 buah, yang mencakup berbagai tahap kematangan, mulai dari buah muda hingga buah yang sudah matang. Menurut data dari Badan Pusat Statistik (BPS) tahun 2022, Indonesia merupakan salah satu produsen kakao terbesar di dunia, sehingga keberagaman varietas dan kondisi pertumbuhan buah kakao di Indonesia memberikan variasi yang signifikan dalam data yang dikumpulkan. Pada penelitian ini menggunakan beberapa teknik pengumpulan data yaitu:

1. Observasi

   Metode pengumpulan data Observasi dilakukan dengan cara pengamatan langsung yaitu dikebun para petani kakao yang terletak di Kabupaten Deli Serdang Kecamatan Sibiru-biru tepatnya di sekitaran Desa Candirejo untuk mengambil buah kakao yang akan di jadikan objek penelitian.

2. Study literatur

   Studi Literatur merupakan kegiatan yang berhubungan dengan metode pengumpulan data yaitu membaca referensi dari jurnal, buku,

dan website yang terpercaya serta mencatat dan mengelolah bahan penelitian.

3. Dokumentasi

   Metode pengumpulan data dokumentasi adalah salah satu teknik yang digunakan dalam penelitian untuk mengumpulkan data. Pada intinya, metode dokumentasi adalah metode yang digunakan untuk menelusuri data historis. Dalam penelitian ini dokumentasi digunakan dalam mengumpulkan data citra digital penelitian dan bukti dokumentasi.

Setelah fitur diekstraksi, langkah berikutnya adalah penerapan algoritma *K-Means Clustering*. Algoritma ini dipilih karena kemampuannya untuk mengelompokkan data berdasarkan kesamaan fitur tanpa memerlukan informasi label awal. Dalam penelitian ini, jumlah cluster akan ditentukan dengan menggunakan metode Elbow, yang membantu dalam mengidentifikasi jumlah cluster optimal berdasarkan variasi dalam data. Hasil dari clustering ini akan memberikan gambaran yang jelas mengenai segmentasi kematangan buah kakao. Penelitian menunjukkan bahwa K-Means Clustering dapat digunakan secara efektif dalam pengelompokan data citra, sehingga diharapkan metode ini dapat memberikan hasil yang akurat dalam segmentasi kematangan buah kakao.

3.1.1 Desain Penelitian

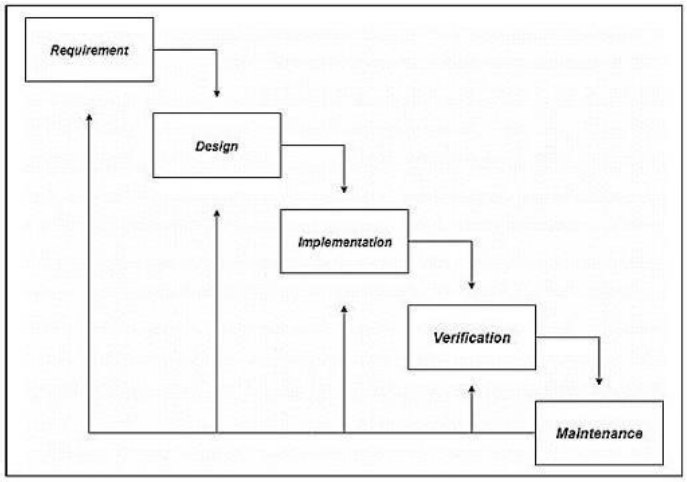
Desain penelitian ini menggunakan pendekatan kuantitatif dengan metode eksperimen. Penelitian ini bertujuan untuk mengevaluasi efektivitas penggunaan fitur warna CIELAB dalam segmentasi kematangan buah

kakao melalui algoritma K-Means Clustering. Penelitian akan dilakukan dengan mengumpulkan data dari lokasi perkebunan kakao yang berada didaerah di Kabupaten Deli Serdang Kecamatan Sibiru-biru. Data yang dikumpulkan akan mencakup sampel buah kakao yang memiliki berbagai tingkat kematangan. Menurut data dari International Cocoa Organization (ICCO), produksi kakao global mencapai sekitar 4,7 juta ton pada tahun 2020, dengan Indonesia sebagai salah satu produsen utama. Oleh karena itu, penting untuk memahami dan mengembangkan metode yang efektif dalam menentukan kematangan buah kakao guna meningkatkan kualitas dan hasil panen.

Penggunaan fitur warna CIELAB diharapkan dapat memberikan representasi yang lebih akurat terhadap warna buah kakao dibandingkan dengan model warna lain seperti RGB. CIELAB dirancang untuk mendekati persepsi manusia terhadap warna, sehingga dapat membantu dalam proses klasifikasi. Penelitian ini akan menggunakan perangkat lunak analisis citra untuk mengonversi gambar buah kakao ke dalam ruang warna CIELAB, yang kemudian akan dianalisis menggunakan algoritma K-Means Clustering. Penelitian sebelumnya menunjukkan bahwa K-Means Clustering efektif digunakan dalam segmentasi citra, dengan akurasi yang dapat mencapai 90%. Dengan memanfaatkan algoritma ini, diharapkan dapat diperoleh pengelompokan yang jelas antara buah kakao yang matang dan yang belum matang.

Pada penelitaian ini memerlukan model pengembangan aplikasi untuk memenuhi kebutuhan model pengembangan aplikasi yang digunakan

yaitu Model waterfall. Tahapan pengembangan model waterfall sebagai berikut.



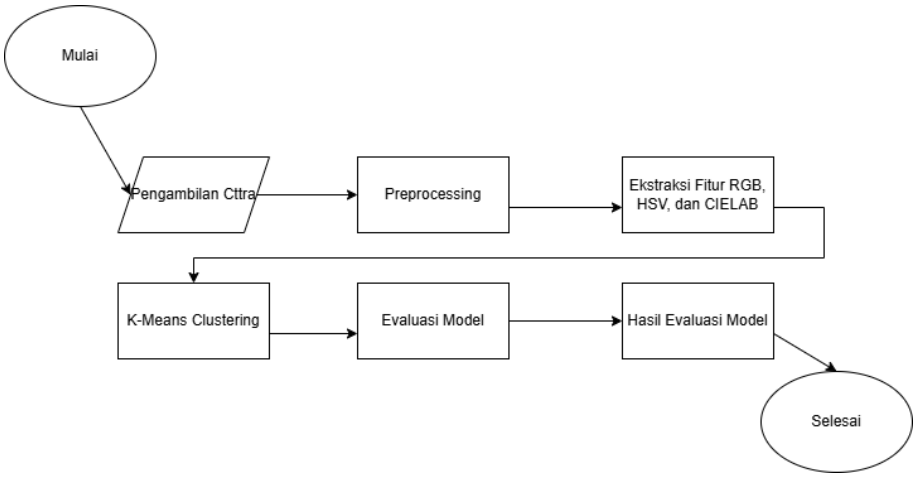
Gambar 3.1 Model Pengembangan Waterfall

Model pengembangan waterfall memiliki 5 tahapan untuk tahapan pembangunan aplikasi. Pada penelitian ini dalam proses pembangunan aplikasi maintenance tidak dilakukan dikarenakan tujuan penelitian yaitu mengetahui kelayakan system aplikasi apabila digunakan pengelompokan tingkat kematangan buah kakao. Berikut penjelasan tahapan pengembangan pada menggunakan metode waterfall.

3.2. Alur Penelitian

Alur penelitian ini mencakup beberapa tahapan penting yang disusun secara sistematis untuk memastikan setiap proses berjalan dengan baik dan menghasilkan data yang valid. Tahapan-tahapan ini akan dijelaskan dalam bentuk diagram alur

untuk memberikan gambaran yang lebih jelas mengenai proses penelitian. Berikut adalah tahapan dalam penelitian ini:



Gambar 3.2 Diagram Alur Penelitian

3.2.1 Pengambilan Citra

Tahap ini mencakup proses mendapatkan dataset citra yang akan digunakan dalam penelitian. Sumber citra dapat berasal dari:

1. Dataset yang sudah tersedia di berbagai platform seperti ImageNet, Kaggle, atau dataset internal yang dikembangkan sendiri.
2. Pengambilan gambar secara manual menggunakan kamera digital atau sensor khusus.
3. Data yang diperoleh harus memenuhi kriteria tertentu seperti resolusi, format file, dan kondisi pencahayaan yang seragam untuk memastikan konsistensi hasil analisis.

Langkah-langkah tambahan yang bisa dilakukan adalah. Labeling data atau Menandai setiap gambar sesuai dengan kategori atau klasifikasinya untuk memudahkan analisis evaluasi hasil klasterisasi. Jika dataset terbatas bisa menggunakan data *augmentation* teknik augmentasi seperti rotasi, flipping, atau peningkatan kontras untuk memperbanyak variasi citra.

3.2.2 Preprocessing

Setelah data citra diperoleh, dilakukan tahap *preprocessing* untuk meningkatkan kualitas gambar sebelum proses analisis lebih lanjut. Teknik yang digunakan meliputi:

1. Konversi ke Grayscale proses mengubah citra berwarna (RGB) menjadi citra skala keabuan (grayscale), di mana setiap piksel hanya memiliki satu nilai intensitas cahaya tanpa informasi warna. Nilai intensitas ini biasanya berada dalam rentang 0 hingga 255, di mana 0 mewakili warna hitam (intensitas terendah), 255 mewakili warna putih (intensitas tertinggi), Nilai di antaranya adalah berbagai tingkat abu-abu.
2. Penghapusan noise Penggunaan filter seperti *Gaussian Blur* atau *Median Filter* untuk mengurangi noise dalam citra.
3. Peningkatan kontras Menggunakan metode *Histogram Equalization* untuk memperjelas perbedaan antara objek dan latar belakang.

4. Resizing dan normalisasi Menyamakan ukuran citra agar memiliki dimensi yang seragam dan menyesuaikan rentang nilai piksel (misalnya 0-1 atau 0-255).

3.2.3 RGB, HSV, & CIELAB

Setelah preprocessing, fitur citra diekstraksi dalam berbagai model warna.

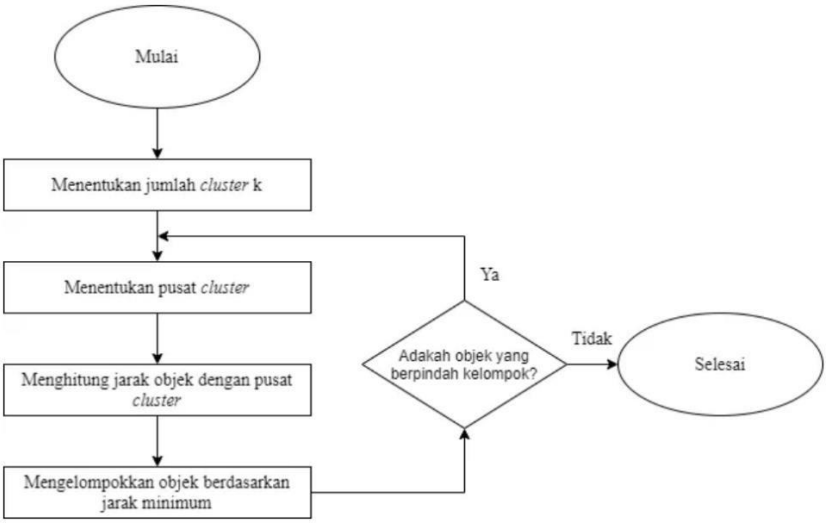
1. RGB (red, green, Blue) Model warna dasar yang umum digunakan dalam tampilan digital. Setiap piksel direpresentasikan sebagai kombinasi tiga nilai warna merah, hijau, dan biru.
2. HSV (Hue, Saturation, Value) Model warna yang lebih dekat dengan cara manusia memahami warna. Hue menentukan rona warna, Saturation menunjukkan seberapa murni warna, dan Value menggambarkan kecerahan.
3. *CIELAB (L, a*, b*)** Model warna yang lebih akurat dalam menggambarkan perbedaan warna dibandingkan RGB. L* mewakili kecerahan, a* menentukan spektrum hijau ke merah, dan b* mewakili spektrum biru ke kuning.

Selain itu, dari setiap model warna, beberapa fitur statistik bisa dihitung, seperti Mean (Rata-rata), Standard Deviation (Simpangan baku), Skewness (Kecondongan distribusi warna), dan Kurtosis (Ketajaman distribusi warna). Fitur-fitur ini kemudian digunakan sebagai input dalam proses klasterisasi.

3.2.4 K-Means Clustering

Pada tahap ini, fitur yang telah diekstraksi digunakan untuk mengelompokkan data menggunakan *K-Means Clustering* Langkah-langkahnya:

1. Menentukan Jumlah Klaster (K) Metode seperti Elbow Method dan Silhouette Score digunakan untuk mencari jumlah klaster optimal.
2. Inisialisasi Centroid, Centroid awal dapat dipilih secara acak atau menggunakan K-Means++ untuk menghindari kesalahan dalam pemilihan centroid awal.
3. Iterasi Klasterisasi, Setiap data dihitung jaraknya ke centroid terdekat menggunakan Euclidean Distance. Data dikelompokkan ke klaster dengan jarak terdekat. Centroid diperbarui berdasarkan rata-rata posisi anggota klaster.
4. Konvergensi, Proses iterasi berhenti jika centroid tidak mengalami perubahan signifikan atau telah mencapai jumlah iterasi maksimum yang ditentukan.



Gambar 3.3 Flowchart K-Means Clustering

3.2.5 Evaluasi Model

Evaluasi model dilakukan untuk menilai kualitas hasil klasterisasi. Beberapa metode evaluasi yang digunakan:

1. Silhouette Score Mengukur seberapa baik setiap sampel dikategorikan dalam klasternya dibandingkan dengan klaster lain.
2. Davies-Bouldin Index (DBI) Mengukur rasio antara rata-rata jarak dalam klaster dan jarak antar klaster.
3. Dunn Index Menghitung rasio antara jarak minimum antar klaster dengan jarak maksimum dalam satu klaster.

4. Perbandingan Visual Jika data memiliki representasi visual, hasil klasterisasi dapat divisualisasikan untuk memastikan bahwa hasilnya masuk akal secara intuitif.
5. Perbandingan dengan Label Data hasil klasterisasi dapat dibandingkan dengan label asli untuk melihat apakah kelompok yang terbentuk sesuai dengan kategori yang diharapkan.

3.2.6 Hasil Evaluasi Model

Hasil evaluasi dianalisis untuk mengetahui apakah model yang digunakan sudah optimal. Jika hasil tidak sesuai, maka bisa dilakukan:

- Optimasi parameter K dalam K-Means.
- Eksperimen dengan metode klasterisasi lain, seperti Fuzzy C-Means atau DBSCAN.
- Meningkatkan kualitas preprocessing data.
- Menambahkan atau mengubah fitur yang diekstraksi.

Jika hasil evaluasi sudah memuaskan, maka penelitian dapat disimpulkan dan diimplementasikan ke dalam aplikasi praktis.

3.3. Rencana Sistem Kerja

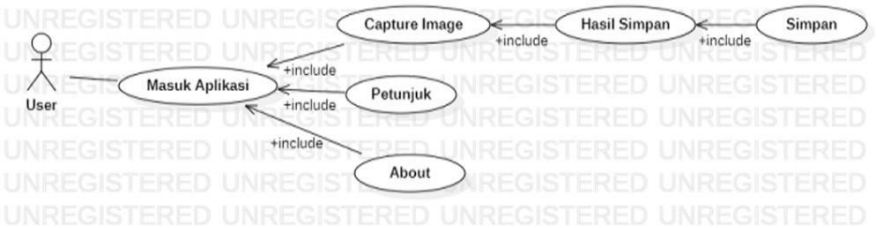
Rencana kerja penelitian ini akan dibagi menjadi beberapa tahap, dimulai dari persiapan alat dan bahan, pengambilan gambar, pengolahan citra, hingga analisis data. Setiap tahap akan dilakukan secara sistematis untuk memastikan

keberhasilan penelitian. Pengambilan gambar akan dilakukan di lokasi perkebunan kakao yang berbeda untuk mendapatkan variasi data. Pengolahan citra akan melibatkan penggunaan google colab untuk memproses citra dan menerapkan algoritma *K-Means Clustering*.

Analisis data akan dilakukan dengan menggunakan perangkat lunak statistik untuk mengevaluasi hasil segmentasi. Selain itu, evaluasi kinerja sistem juga akan dilakukan dengan menghitung nilai akurasi, presisi, dan recall dari hasil segmentasi. Dengan demikian, penelitian ini tidak hanya akan menghasilkan sistem segmentasi buah kakao, tetapi juga memberikan pemahaman yang lebih dalam mengenai efektivitas penggunaan warna CIELAB dalam pengolahan citra.

3.3.1 Use Case Diagram

Use case diagram merupakan sebuah fungsi yang ada pada system untuk pengguna dapat mengetahui aktivitas proses yang ada pada system. Berikut Use Case diagram aplikasi disajikan pada gambar dibawah ini



Gambar 3.4 Use Case Diagram

Setiap skenario pada use case memperlihatkan sebuah proses oleh pengguna (user) memberikan sebuah perintah kepada setiap bagian pada use

case. User dapat melakukan beberapa fungsi yang ada di system diantaranya melakukan capture foto atau ambil gambar yang keluarnya akan menghasilkan pengelompokan dari tingkat kematangan buah kakao, dan melihat petunjuk pemakaian aplikasi.

3.4. Kebutuhan Hardware dan Software

Untuk mendukung penelitian mengenai "Penggunaan Fitur Warna Cielab dalam Segmentasi Kematangan Buah Kakao dengan Algoritma K-Means Clustering," diperlukan perangkat keras dan perangkat lunak yang sesuai agar proses pengolahan citra dan analisis data dapat berjalan dengan optimal.

3.4.1 Kebutuhan Hardware

Tabel 3.1 Kebutuhan Hardware

No	Komponen	Spesifikasi
1.	Prosesor	Intel Core i5
2.	RAM	16 GB
3.	GPU	NVIDIA GTX 1050 Ti
4.	Penyimpanan	SSD 512 GB
5.	Sistem Operasi	Windows 11
6.	Kamera *HP POCO X3GT*	Resolusi minimal 720p, 30 FPS Berfungsi untuk mengambil gambar buah kakao

Dengan spesifikasi perangkat keras yang telah ditetapkan, penggunaan fitur warna CIELAB dalam segmentasi kematangan buah kakao menggunakan algoritma *K-means Clustering* ini diharapkan dapat berjalan secara optimal dan memberikan hasil yang akurat.

3.4.2 Kebutuhan Software

Software yang dibutuhkan untuk penelitian ini harus mampu melakukan segmentasi dan klasifikasi kematangan buah kakao menggunakan fitur warna Cielab dan algoritma K-Means Clustering. Google colab menjadi salah satu perangkat lunak yang ideal karena menyediakan berbagai fungsi bawaan untuk pengolahan citra, seperti konversi model warna ke Cielab, segmentasi berbasis K-Means, serta teknik preprocessing gambar seperti penyesuaian kontras, filtering, dan normalisasi warna. Selain itu, google colab memiliki antarmuka yang mendukung visualisasi hasil segmentasi secara interaktif, memungkinkan analisis lebih mendalam terhadap akurasi klasifikasi. Dengan memanfaatkan google colab, proses identifikasi kematangan buah kakao dapat dilakukan secara efisien dan akurat, memberikan solusi yang lebih objektif bagi petani dan industri kakao.

3.5. Waktu Penelitian

Table 3.2 Waktu Penelitian

No.	Kegiatan Penelitian	Waktu penelitian				
		Januari	Februari	Maret	April	Mei
1.	Observasi dan Analisis					
2.	Pengumpulan Data					
3.	Pembuatan proposal & Bimbingan proposal					
4.	Seminar Proposal					
5.	Riset					
6.	Penyusunan Skripsi					

BAB IV

HASIL DAN PEMBAHASAN

4.1 Hasil dalam Machine Learning

Pada tahap ini dilakukan proses pemodelan *machine learning* menggunakan algoritma K-Means Clustering untuk mengelompokkan tingkat kematangan buah kakao berdasarkan warna menggunakan ruang warna CIELAB (L*, a*, b*). Adapun tahapan dan hasil yang diperoleh adalah sebagai berikut:

1. Import Library Python

Dalam pembuatan model tentunya dibutuhkan library-library python yang tentunya akan membantu dalam proses pembuatannya, adapun library-library tersebut adalah sebagai berikut:

a. cv2 (OpenCV-Python)

Library OpenCV digunakan sebagai alat utama dalam membaca dan memproses gambar digital. Dalam penelitian ini, OpenCV digunakan untuk membaca gambar buah kakao dari direktori dataset, mengubah format warnanya dari BGR ke RGB, dan kemudian mengkonversinya ke ruang warna CIELAB. Proses konversi ini penting karena ruang warna CIELAB lebih sesuai untuk analisis persepsi warna manusia, sehingga membantu dalam membedakan tingkat kematangan buah kakao. Selain itu, OpenCV juga digunakan untuk menampilkan dan menyimpan hasil segmentasi warna hasil clustering.

b. numpy (Numerical Python)

NumPy berperan penting dalam representasi dan manipulasi data gambar dalam bentuk array numerik. Setiap gambar direpresentasikan sebagai

array multidimensi, di mana setiap elemen menyimpan informasi warna piksel. NumPy digunakan untuk melakukan operasi seperti reshaping array piksel, sampling acak piksel untuk efisiensi, serta perhitungan statistik dasar seperti rata-rata dan standar deviasi. Dalam konteks clustering, NumPy mendukung pemrosesan data warna dalam jumlah besar secara cepat dan efisien.

c. matplotlib.pyplot

Library ini digunakan untuk menampilkan hasil visualisasi dalam bentuk gambar dan grafik. Pada penelitian ini, matplotlib digunakan untuk menampilkan gambar asli buah kakao, visualisasi hasil segmentasi warna berdasarkan cluster, serta palet warna dominan hasil clustering. Visualisasi ini sangat membantu dalam memahami bagaimana algoritma mengelompokkan warna serta bagaimana hasil tersebut bisa diinterpretasikan secara visual oleh pengguna.

d. sklearn.cluster.KMeans

Modul ini merupakan inti dari proses clustering yang digunakan dalam penelitian. KMeans dari Scikit-learn digunakan untuk mengelompokkan piksel-piksel warna dari gambar ke dalam beberapa cluster berdasarkan kemiripan warna dalam ruang CIELAB. Setiap cluster mewakili satu kelompok warna dominan, yang kemudian dianalisis untuk menentukan apakah warna tersebut mengindikasikan buah matang, belum matang, atau bukan buah kakao. Pemilihan KMeans memungkinkan pemrosesan unsupervised learning yang efektif tanpa perlu label manual.

e. sklearn.preprocessing.StandardScaler

StandardScaler digunakan untuk menstandarisasi nilai-nilai fitur warna (L*, a*, b*) sebelum dimasukkan ke dalam model KMeans. Proses standarisasi ini memastikan bahwa setiap fitur memiliki skala yang sama, sehingga tidak ada fitur yang mendominasi proses clustering. Hal ini penting untuk menjaga keakuratan dan keadilan dalam pembentukan cluster warna.

f. os

Library os digunakan untuk menangani berbagai operasi sistem file, seperti membaca isi folder dataset, membuat direktori baru untuk menyimpan hasil, dan memverifikasi keberadaan file sebelum diproses. Fungsionalitas ini sangat penting untuk mengotomatisasi proses pengolahan dataset gambar yang jumlahnya banyak dan tersimpan dalam struktur folder tertentu.

g. pathlib.Path

Pathlib digunakan sebagai alternatif modern dari os.path untuk memanipulasi path file dan folder. Dengan pendekatan yang lebih berorientasi objek, Pathlib membuat kode lebih mudah dibaca dan dikelola, terutama dalam hal navigasi ke folder dataset, pencarian gambar, dan pembuatan struktur folder hasil.

h. pandas

Pandas digunakan untuk mengelola dan menganalisis data hasil clustering dalam bentuk tabular. Data seperti nilai rata-rata warna pada tiap cluster dan kategori buah disimpan dalam DataFrame, yang kemudian digunakan

untuk analisis perbandingan. Pandas juga memudahkan penyajian hasil evaluasi dan pengelompokan data secara statistik sebelum divisualisasikan lebih lanjut.

i. collections.Counter

Library ini digunakan untuk menghitung jumlah piksel yang termasuk ke dalam masing-masing cluster. Dengan menggunakan Counter, sistem dapat menentukan cluster mana yang paling dominan dalam sebuah gambar, yang kemudian digunakan untuk mengindikasikan kategori kematangan buah kakao tersebut secara otomatis.

j. seaborn

Seaborn digunakan sebagai alat bantu visualisasi statistik data warna antar kategori buah. Dalam penelitian ini, Seaborn menghasilkan boxplot nilai L, a, dan b untuk setiap kategori (matang, belum matang, bukan kakao), sehingga perbedaan distribusi warna antar kategori dapat divisualisasikan dengan jelas. Visualisasi ini mendukung interpretasi hasil clustering secara lebih intuitif dan estetik.

k. joblib

Joblib digunakan untuk menyimpan model KMeans dan objek lainnya (seperti scaler) ke dalam file dengan format .joblib. Hal ini memungkinkan model yang sudah dilatih tidak perlu dilatih ulang setiap kali digunakan. File model ini juga dapat diintegrasikan ke dalam sistem web dan API, sehingga mempercepat proses prediksi secara real-time.

l. datetime

Modul datetime digunakan untuk memberi cap waktu (timestamp) pada

nama file model yang disimpan. Dengan demikian, setiap model hasil pelatihan dapat diberi identitas unik berdasarkan waktu, yang berguna untuk dokumentasi, pengujian, serta manajemen versi model dalam pengembangan berkelanjutan.

2. Membuat Fungsi dalam Kelas Warna Segmentasi Cacao

Fungsi-fungsi dalam class CacaoColorSegmentation yang merupakan inti dari proses *machine learning* untuk segmentasi tingkat kematangan buah kakao adalah sebagai berikut:

a. Fungsi __init__(self, dataset_path, n_clusters=5)

Fungsi ini merupakan konstruktor yang digunakan untuk menginisialisasi objek dari class CacaoColorSegmentation. Parameter dataset_path menunjukkan lokasi direktori dataset gambar, sedangkan n_clusters menentukan jumlah cluster yang digunakan dalam algoritma KMeans. Selain itu, objek StandardScaler juga diinisialisasi untuk proses normalisasi warna pada ruang warna LAB. Adapun codingannya adalah sebagai berikut:

```
def __init__(self, dataset_path, n_clusters=5):
    """
    Initialize the color segmentation class

    Args:
        dataset_path: Path to the dataset folder containing subfolders
        n_clusters: Number of clusters for K-means
    """

    self.dataset_path = Path(dataset_path)
    self.n_clusters = n_clusters
    self.scaler = StandardScaler()
```

b. Fungsi load_images_from_folder(self, folder_path)

Fungsi ini bertugas untuk memuat seluruh gambar dari suatu direktori. Gambar-gambar dengan ekstensi umum seperti .jpg dan .png akan dimasukkan ke dalam sebuah list bersama path-nya, untuk kemudian diproses lebih lanjut. Adapun codingannya adalah sebagai berikut:

```
def load_images_from_folder(self, folder_path):
    """Load all images from a specific folder"""
    images = []
    image_paths = []

    supported_formats = ('.jpg', '.jpeg', '.png', '.bmp', '.tiff')

    for img_path in Path(folder_path).glob('*'):
        if img_path.suffix.lower() in supported_formats:
            img = cv2.imread(str(img_path))
            if img is not None:
                images.append(img)
                image_paths.append(str(img_path))

    return images, image_paths
```

c. Fungsi rgb_to_lab(self, rgb_image)

Fungsi ini mengonversi gambar dari format BGR (standar OpenCV) ke format CIELAB. Proses ini penting karena ruang warna LAB lebih sesuai untuk membedakan warna secara perseptual dibandingkan dengan RGB atau BGR. Adapun codingannya adalah sebagai berikut:

```
def rgb_to_lab(self, rgb_image):
    """Convert RGB image to CIELAB color space"""
    # Convert BGR to RGB (OpenCV loads as BGR)
    rgb_img = cv2.cvtColor(rgb_image, cv2.COLOR_BGR2RGB)
    # Convert RGB to LAB
    lab_img = cv2.cvtColor(rgb_img, cv2.COLOR_RGB2LAB)
    return lab_img
```

d. Fungsi extract_lab_features(self, images)

Fungsi ini digunakan untuk mengekstrak fitur warna dari setiap gambar. Proses ini melibatkan konversi ke LAB dan sampling acak sebanyak 1.000 piksel per gambar, yang kemudian digabung menjadi satu array besar sebagai data masukan untuk clustering. Adapun codingannya adalah sebagai berikut:

```
def extract_lab_features(self, images):
    """Extract LAB color features from images"""
    all_lab_pixels = []

    for img in images:
        # Convert to LAB

        lab_img = self.rgb_to_lab(img)

        # Reshape to get all pixels

        lab_pixels = lab_img.reshape(-1, 3)

        # Sample pixels to reduce computation (optional)

        # You can adjust sampling rate based on your needs

        sample_size = min(1000, len(lab_pixels))
        indices = np.random.choice(len(lab_pixels), sample_size,
replace=False)
        sampled_pixels = lab_pixels[indices]

        all_lab_pixels.append(sampled_pixels)

    # Combine all pixels

    combined_pixels = np.vstack(all_lab_pixels)
    return combined_pixels
```

e. Fungsi perform_kmeans_clustering(self, lab_features)

Fungsi ini menerapkan algoritma KMeans pada data warna LAB yang telah distandarkan. Sebelum KMeans dijalankan, data diformalisasi menggunakan StandardScaler. Fungsi ini mengembalikan model KMeans dan label setiap data. Adapun codingannya adalah sebagai berikut:

```
def perform_kmeans_clustering(self, lab_features):
    """Perform K-means clustering on LAB features"""
    # Normalize the features
```

```
        lab_features_scaled = self.scaler.fit_transform(lab_features)

        # Apply K-means

        kmeans = KMeans(n_clusters=self.n_clusters, random_state=42,
n_init=10)
        cluster_labels = kmeans.fit_predict(lab_features_scaled)

        return kmeans, cluster_labels
```

f. Fungsi get_dominant_colors(self, kmeans_model, lab_features)

Fungsi ini digunakan untuk mengambil pusat (centroid) dari setiap cluster hasil KMeans dalam ruang warna LAB, lalu dikonversi kembali ke RGB agar bisa divisualisasikan sebagai warna dominan. Adapun codingannya adalah sebagai berikut:

```
def get_dominant_colors(self, kmeans_model, lab_features):
        """Get dominant colors from cluster centers"""
        # Get cluster centers in scaled space

        centers_scaled = kmeans_model.cluster_centers_

        # Inverse transform to get original LAB values

        centers_lab = self.scaler.inverse_transform(centers_scaled)

        # Convert LAB centers to RGB for visualization

        centers_rgb = []
        for center in centers_lab:
            # Create a single pixel image with the LAB color

            lab_pixel = np.uint8([[center]])
            rgb_pixel = cv2.cvtColor(lab_pixel, cv2.COLOR_LAB2RGB)
            centers_rgb.append(rgb_pixel[0][0])

        return centers_lab, centers_rgb
```

g. Fungsi visualize_dominant_colors(self, colors_rgb, title)

Fungsi ini menghasilkan visualisasi palet warna dominan berupa kotak warna horizontal. Setiap kotak mewakili satu cluster warna hasil segmentasi. Adapun codingannya adalah sebagai berikut:

```
def visualize_dominant_colors(self, colors_rgb, title="Dominant
Colors"):
    """Visualize dominant colors"""
    fig, ax = plt.subplots(1, 1, figsize=(12, 2))

    # Create color palette

    palette = np.array(colors_rgb).reshape(1, -1, 3) / 255.0
    ax.imshow(palette, aspect='auto')
    ax.set_xlim(0, len(colors_rgb))
    ax.set_xticks(range(len(colors_rgb)))
    ax.set_xticklabels([f'Cluster {i}' for i in range(len(colors_rgb))])
    ax.set_yticks([])
    ax.set_title(title)

    plt.tight_layout()
    plt.show()
```

h. Fungsi analyze_single_category(self, category_name, category_path) Fungsi ini menjalankan seluruh pipeline untuk satu kategori gambar (misalnya: Kakao Matang), dari memuat gambar, ekstraksi fitur LAB, clustering, hingga visualisasi warna dominan. Adapun codingannya adalah sebagai berikut:

```
def analyze_single_category(self, category_name, category_path):
    """Analyze color segmentation for a single category"""
    print(f"\n=== Analyzing {category_name} ===")

    # Load images

   images, image_paths = self.load_images_from_folder(category_path)
    print(f"Loaded {len(images)} images from {category_name}")

    if len(images) == 0:
       print(f"No images found in {category_path}")
       return None
    # Extract LAB features

    lab_features = self.extract_lab_features(images)
    print(f"Extracted {len(lab_features)} LAB pixel samples")

    # Perform clustering

    kmeans_model, cluster_labels =
self.perform_kmeans_clustering(lab_features)
```

```
        # Get dominant colors

        centers_lab, centers_rgb =
self.get_dominant_colors(kmeans_model, lab_features)

        # Visualize

        self.visualize_dominant_colors(centers_rgb, f"Dominant Colors -
{category_name}")

        # Print LAB values

        print(f"Dominant LAB colors for {category_name}:")
        for i, (lab, rgb) in enumerate(zip(centers_lab, centers_rgb)):
            print(f"  Cluster {i}: LAB({lab[0]:.1f}, {lab[1]:.1f}, {lab[2]:.1f})
- RGB({rgb[0]}, {rgb[1]}, {rgb[2]})")

        return {
            'category': category_name,
            'kmeans_model': kmeans_model,
            'centers_lab': centers_lab,
            'centers_rgb': centers_rgb,
            'cluster_labels': cluster_labels,
            'lab_features': lab_features
        }
```

i. Fungsi analyze_all_categories(self)

Fungsi ini menerapkan proses analisis untuk seluruh kategori gambar yang tersedia, yaitu kakao matang, kakao belum matang, dan bukan kakao, dengan memanggil analyze_single_category() untuk masing-masing kategori. Adapun codingannya adalah sebagai berikut:

```
def analyze_all_categories(self):
        """Analyze all categories in the dataset"""
        categories = {
            'Kakao Matang': self.dataset_path / 'Kakao Matang',
            'Kakao Belum Matang': self.dataset_path / 'Kakao Belum
Matang',
            'Bukan Kakao': self.dataset_path / 'Bukan Kakao'
        }

        results = {}

        for category_name, category_path in categories.items():
            if category_path.exists():
```

```
            result = self.analyze_single_category(category_name,
category_path)
            if result:
                results[category_name] = result
        else:
            print(f"Warning: Path {category_path} does not exist")

    return results
```

j. Fungsi compare_categories(self, results)

Fungsi ini membandingkan hasil segmentasi warna antar kategori. Visualisasi yang digunakan mencakup palet warna dominan dan boxplot distribusi komponen warna LAB untuk setiap kategori. Adapun codingannya adalah sebagai berikut:

```
def compare_categories(self, results):
    """Compare color characteristics between categories"""
    if len(results) < 2:
        print("Need at least 2 categories to compare")
        return

    # Create comparison visualization

    fig, axes = plt.subplots(len(results), 1, figsize=(12, 4 * len(results)))
    if len(results) == 1:
        axes = [axes]

    for idx, (category, result) in enumerate(results.items()):
        palette = np.array(result['centers_rgb']).reshape(1, -1, 3) / 255.0
        axes[idx].imshow(palette, aspect='auto')
        axes[idx].set_xlim(0, len(result['centers_rgb']))
        axes[idx].set_xticks(range(len(result['centers_rgb'])))
        axes[idx].set_xticklabels([f'C{i}' for i in
range(len(result['centers_rgb']))])
        axes[idx].set_yticks([])
        axes[idx].set_title(f'{category} - Dominant Colors')

    plt.tight_layout()
    plt.show()

    # Create LAB comparison dataframe

    comparison_data = []
    for category, result in results.items():
        for i, lab_color in enumerate(result['centers_lab']):
```

```
            comparison_data.append({
                'Category': category,
                'Cluster': i,
                'L': lab_color[0],
                'A': lab_color[1],
                'B': lab_color[2]
            })


        df = pd.DataFrame(comparison_data)

        # Visualize LAB distribution

        fig, axes = plt.subplots(1, 3, figsize=(15, 5))

        for idx, channel in enumerate(['L', 'A', 'B']):
            sns.boxplot(data=df, x='Category', y=channel, ax=axes[idx])
            axes[idx].set_title(f'{channel} Channel Distribution')
            axes[idx].tick_params(axis='x', rotation=45)

        plt.tight_layout()
        plt.show()

        return df
```

k. Fungsi segment_image_colors(self, image_path, kmeans_model)

Fungsi ini menghasilkan peta segmentasi warna untuk sebuah gambar berdasarkan model KMeans. Setiap piksel dalam gambar akan diberi label cluster yang sesuai. Adapun codingannya adalah sebagai berikut:

```
def segment_image_colors(self, image_path, kmeans_model):
    """Segment colors in a single image using trained model"""
    # Load image
    img = cv2.imread(image_path)
    if img is None:
        print(f"Could not load image: {image_path}")
        return None

    # Convert to LAB

    lab_img = self.rgb_to_lab(img)
    original_shape = lab_img.shape[:2]

    # Reshape for prediction

    lab_pixels = lab_img.reshape(-1, 3)
    lab_pixels_scaled = self.scaler.transform(lab_pixels)
```

```
    # Predict clusters

    labels = kmeans_model.predict(lab_pixels_scaled)

    # Reshape back to image shape

    segmented = labels.reshape(original_shape)
    return segmented, img
```

l. Fungsi visualize cluster_segmentation(self, image_path, kmeans_model) Fungsi ini menampilkan hasil segmentasi klaster warna dalam format vertikal, memperlihatkan gambar asli dan representasi masing-masing cluster dalam warna tertentu. Adapun codingannya adalah sebagai berikut:

```
def visualize_cluster_segmentation(self, image_path, kmeans_model,
save_path=None):
    """Create cluster segmentation visualization like the example"""
    # Get segmentation and original imageresult =
    self.segment_image_colors(image_path, kmeans_model) if result is None:

    return None

    segmented, original_img = result
   # Create color map for clusters # Use distinct colors for each cluster

   cluster_colors = [

    [255, 0, 0],        # Red
      [0, 255, 0],      # Green

      [0, 0, 255],      # Blue

      [255, 255, 0],  # Yellow

      [255, 0, 255],    # Magenta

      [0, 255, 255],    # Cyan

      [255, 128, 0],    # Orange

      [128, 0, 255],    # Purple

      [0, 128, 255],    # Light Blue

      [255, 255, 255]   # White

    ]

    # Create individual cluster images

    cluster_images = []
    height, width = segmented.shape
```

```
        for cluster_id in range(self.n_clusters):
            # Create blank image

            cluster_img = np.zeros((height, width, 3), dtype=np.uint8)

            # Get cluster color

            color = cluster_colors[cluster_id % len(cluster_colors)]

            # Fill pixels belonging to this cluster
            mask = segmented == cluster_id
            cluster_img[mask] = color


            cluster_images.append(cluster_img)

        # Create visualization

        fig, axes = plt.subplots(self.n_clusters + 1, 1, figsize=(8, 2 *
(self.n_clusters + 1)))

        # Show original image

        original_rgb = cv2.cvtColor(original_img,
cv2.COLOR_BGR2RGB)
        axes[0].imshow(original_rgb)
        axes[0].set_title('Original Image')
        axes[0].axis('off')

        # Show each cluster

        for i, cluster_img in enumerate(cluster_images):
            cluster_rgb = cv2.cvtColor(cluster_img,
cv2.COLOR_BGR2RGB)
            axes[i + 1].imshow(cluster_rgb)
            axes[i + 1].set_title(f'Cluster {i}')
            axes[i + 1].axis('off')

        plt.tight_layout()

        if save_path:
            plt.savefig(save_path, dpi=150, bbox_inches='tight')

        plt.show()

        return cluster_images
```

m. Fungsi visualize_cluster_segmentation_horizontal(self, image_path, kmeans_model)

Fungsi ini serupa dengan sebelumnya, namun menyajikan hasil segmentasi secara horizontal, yang lebih ringkas untuk visualisasi multi-klaster. Adapun codingannya adalah sebagai berikut:

```
def visualize_cluster_segmentation_horizontal(self, image_path, kmeans_model, save_path=None):
    """Create horizontal cluster segmentation visualization"""
    # Get segmentation and original image

    result = self.segment_image_colors(image_path, kmeans_model)
    if result is None:
        return None
    segmented, original_img = result
    # Create color map for clusters

    cluster_colors = [
        [255, 0, 0],      # Red
        [0, 255, 0],      # Green

        [0, 0, 255],      # Blue

        [255, 255, 0],    # Yellow

        [255, 0, 255],    # Magenta

        [0, 255, 255],    # Cyan

        [255, 128, 0],    # Orange

        [128, 0, 255],    # Purple

        [0, 128, 255],    # Light Blue

        [255, 255, 255]   # White

    ]# Create individual cluster images

    cluster_images = []
    height, width = segmented.shape
    for cluster_id in range(self.n_clusters):
        # Create blank image

        cluster_img = np.zeros((height, width, 3), dtype=np.uint8)
        # Get cluster color

        color = cluster_colors[cluster_id % len(cluster_colors)]
        # Fill pixels belonging to this cluster mask = segmented == cluster_id
        cluster_img[mask] = color

        cluster_images.append(cluster_img)

    # Create HORIZONTAL visualization
```

```
        fig, axes = plt.subplots(1, self.n_clusters + 1, figsize=(3 *
    (self.n_clusters + 1), 4))

        # Show original image

        original_rgb = cv2.cvtColor(original_img,
    cv2.COLOR_BGR2RGB)
        axes[0].imshow(original_rgb)
        axes[0].set_title('Original Image')
        axes[0].axis('off')

        # Show each cluster horizontally

        for i, cluster_img in enumerate(cluster_images):
            cluster_rgb = cv2.cvtColor(cluster_img,
    cv2.COLOR_BGR2RGB)
            axes[i + 1].imshow(cluster_rgb)
            axes[i + 1].set_title(f'Cluster {i}')
            axes[i + 1].axis('off')

        plt.tight_layout()

        if save_path:
            plt.savefig(save_path, dpi=150, bbox_inches='tight')

        plt.show()

        return cluster_images
```

n. Fungsi create_combined_cluster_visualization(self, image_paths, kmeans_model, max_images)

Fungsi ini menyusun hasil segmentasi dari beberapa gambar dalam satu tampilan vertikal, memudahkan perbandingan visual. Adapun codingannya adalah sebagai berikut:

```
def create_combined_cluster_visualization(self, image_paths,
kmeans_model, max_images=5):
    """Create a combined visualization showing multiple images and
their clusters"""
    n_images = min(len(image_paths), max_images)

    fig, axes = plt.subplots(n_images, self.n_clusters + 1,
                             figsize=(2 * (self.n_clusters + 1), 2 * n_images))

    if n_images == 1:
```

```
        axes = axes.reshape(1, -1)

    cluster_colors = [
        [255, 0, 0], [0, 255, 0], [0, 0, 255], [255, 255, 0], [255, 0, 255],
        [0, 255, 255], [255, 128, 0], [128, 0, 255], [0, 128, 255], [255,
255, 255]
    ]


    for img_idx in range(n_images):
        image_path = image_paths[img_idx]

        # Get segmentation

        result = self.segment_image_colors(image_path, kmeans_model)
        if result is None:
            continue

        segmented, original_img = result

        # Show original image

        original_rgb = cv2.cvtColor(original_img,
cv2.COLOR_BGR2RGB)
        axes[img_idx, 0].imshow(original_rgb)
        axes[img_idx, 0].set_title(f'Original {img_idx + 1}' if img_idx
== 0 else '')
        axes[img_idx, 0].axis('off')

        # Show clusters

        height, width = segmented.shape
        for cluster_id in range(self.n_clusters):
            cluster_img = np.zeros((height, width, 3), dtype=np.uint8)
            color = cluster_colors[cluster_id % len(cluster_colors)]
            mask = segmented == cluster_id
            cluster_img[mask] = color

            cluster_rgb = cv2.cvtColor(cluster_img,
cv2.COLOR_BGR2RGB)
            axes[img_idx, cluster_id + 1].imshow(cluster_rgb)

            if img_idx == 0:  # Only add title to first row

                axes[img_idx, cluster_id + 1].set_title(f'Cluster
{cluster_id}')
            axes[img_idx, cluster_id + 1].axis('off')

    plt.tight_layout()
    plt.show()
```

o. Fungsi create_combined_cluster_visualization_horizontal(self, image_paths, kmeans_model, max_images)

Versi horizontal dari fungsi sebelumnya, cocok digunakan untuk membuat grid visualisasi hasil segmentasi per gambar. Adapun codingannya adalah sebagai berikut:

```
def create_combined_cluster_visualization_horizontal(self, image_paths, kmeans_model, max_images=6):
    """Create a horizontal combined visualization showing multiple images and their clusters"""
    n_images = min(len(image_paths), max_images)

    # Create horizontal layout: rows = images, cols = original + clusters

    fig, axes = plt.subplots(n_images, self.n_clusters + 1,
                             figsize=(2.5 * (self.n_clusters + 1), 2.5 * n_images))

    if n_images == 1:
        axes = axes.reshape(1, -1)

    cluster_colors = [
        [255, 0, 0], [0, 255, 0], [0, 0, 255], [255, 255, 0], [255, 0, 255],
        [0, 255, 255], [255, 128, 0], [128, 0, 255], [0, 128, 255], [255, 255, 255]
    ]


    for img_idx in range(n_images):
        image_path = image_paths[img_idx]

        # Get segmentation

        result = self.segment_image_colors(image_path, kmeans_model)
        if result is None:
            continue
        segmented, original_img = result

        # Show original image

        original_rgb = cv2.cvtColor(original_img, cv2.COLOR_BGR2RGB)
        axes[img_idx, 0].imshow(original_rgb)
        axes[img_idx, 0].set_title(f'Original {img_idx + 1}' if img_idx == 0 else '')
        axes[img_idx, 0].axis('off')
```

```
            # Show clusters

            height, width = segmented.shape
            for cluster_id in range(self.n_clusters):
                cluster_img = np.zeros((height, width, 3), dtype=np.uint8)
                color = cluster_colors[cluster_id % len(cluster_colors)]
                mask = segmented == cluster_id
                cluster_img[mask] = color

                cluster_rgb = cv2.cvtColor(cluster_img,
    cv2.COLOR_BGR2RGB)
                axes[img_idx, cluster_id + 1].imshow(cluster_rgb)

                if img_idx == 0:  # Only add title to first row

                    axes[img_idx, cluster_id + 1].set_title(f'Cluster
    {cluster_id}')
                axes[img_idx, cluster_id + 1].axis('off')

        plt.tight_layout()
        plt.show()
```

p. Fungsi analyze_cluster_statistics(self, results)

Fungsi ini digunakan untuk menganalisis statistik warna pada setiap cluster. Statistik yang dihitung mencakup jumlah piksel, persentase, rata-rata, dan simpangan baku untuk setiap komponen warna LAB. Adapun codingannya adalah sebagai berikut:

```
def analyze_cluster_statistics(self, results):
        """Analyze statistics for each cluster across categories"""
        cluster_stats = {}
        for category, result in results.items():
            cluster_stats[category] = {}
            kmeans_model = result['kmeans_model']
            lab_features = result['lab_features']
            # Get cluster assignments

            lab_features_scaled = self.scaler.transform(lab_features)
            cluster_labels = kmeans_model.predict(lab_features_scaled)

            # Calculate statistics for each cluster

            for cluster_id in range(self.n_clusters):
                cluster_mask = cluster_labels == cluster_id
                cluster_pixels = lab_features[cluster_mask]
```

```
            if len(cluster_pixels) > 0:
                cluster_stats[category][f'Cluster_{cluster_id}'] = {
                    'pixel_count': len(cluster_pixels),
                    'percentage': len(cluster_pixels) / len(lab_features) * 100,
                    'mean_L': np.mean(cluster_pixels[:, 0]),
                    'mean_A': np.mean(cluster_pixels[:, 1]),
                    'mean_B': np.mean(cluster_pixels[:, 2]),
                    'std_L': np.std(cluster_pixels[:, 0]),
                    'std_A': np.std(cluster_pixels[:, 1]),
                    'std_B': np.std(cluster_pixels[:, 2])
                }


        return cluster_stats
```

q. Fungsi save_models(self, results, save_directory)

Fungsi ini menyimpan model KMeans yang telah dilatih, termasuk scaler, parameter pelatihan, dan pusat warna dominan ke dalam file.joblib. Adapun codingannya adalah sebagai berikut:

```
def save_models(self, results, save_directory="saved_models"):
    """Save all trained K-means models in one combined file using joblib"""
    import os
    # Create save directory

    os.makedirs(save_directory, exist_ok=True)

    timestamp = datetime.now().strftime("%Y%m%d_%H%M%S")

    # Save only combined model file

    combined_filename = f"all_kmeans_models_{timestamp}.joblib"
    combined_filepath = os.path.join(save_directory, combined_filename)
    combined_data = {
        'all_results': results,
        'scaler': self.scaler,
        'n_clusters': self.n_clusters,
        'timestamp': timestamp,
        'categories': list(results.keys())
    }


    joblib.dump(combined_data, combined_filepath)
```

```
        print(f"All K-means models saved: {combined_filepath}")

        return combined_filepath
```

r. Fungsi load_model(self, model_path)

Fungsi ini digunakan untuk memuat kembali model dan hasil clustering yang telah disimpan, agar dapat digunakan untuk prediksi lanjutan. Adapun codingannya adalah sebagai berikut:

```
def load_model(self, model_path):
    """Load the saved combined K-means models"""
    try:
        model_data = joblib.load(model_path)
        print(f"Combined models loaded successfully from:
{model_path}")

        # Update scaler

        if 'scaler' in model_data:
            self.scaler = model_data['scaler']

        # Update n_clusters

        if 'n_clusters' in model_data:
            self.n_clusters = model_data['n_clusters']

        return model_data['all_results']
    except Exception as e:
        print(f"Error loading models: {e}")
        return None
```

s. Fungsi predict_with_saved_model(self, image_path, loaded_results, category_name)

Fungsi ini menerapkan model clustering yang telah disimpan pada gambar baru, dan menghasilkan segmentasi warna untuk kategori yang ditentukan. Adapun codingannya adalah sebagai berikut:

```
def predict_with_saved_model(self, image_path, loaded_results,
category_name):
    """Use a saved model to predict/segment a new image"""
    if category_name not in loaded_results:
```

```
            print(f"Category '{category_name}' not found in loaded
models")
            print(f"Available categories: {list(loaded_results.keys())}")
            return None

        kmeans_model = loaded_results[category_name]['kmeans_model']
        return self.segment_image_colors(image_path, kmeans_model)
```

t. Fungsi predict_image_category(self, image_path, loaded_results)

Fungsi ini mencoba menebak kategori gambar berdasarkan distribusi warna LAB dan tingkat kemiripan dengan model dari kategori yang telah dianalisis sebelumnya. Hasilnya berupa nama kategori dan skor kepercayaan (confidence score). Adapun codingannya adalah sebagai berikut:

```
def predict_image_category(self, image_path, loaded_results):
        """Predict which category an image belongs to based on color
similarity"""

        # Extract LAB features from the input image

        img = cv2.imread(image_path)
        if img is None:
            print(f"Could not load image: {image_path}")
            return None
        lab_img = self.rgb_to_lab(img)
        lab_pixels = lab_img.reshape(-1, 3)
        # Sample pixels for consistency with training

        sample_size = min(1000, len(lab_pixels))
        indices = np.random.choice(len(lab_pixels), sample_size,
replace=False)
        sampled_pixels = lab_pixels[indices]

        # Scale the features

        lab_features_scaled = self.scaler.transform(sampled_pixels)

        category_scores = {}

        # Test against each category model

        for category_name, result in loaded_results.items():
            kmeans_model = result['kmeans_model']
```

```
            # Get cluster assignments

            cluster_labels = kmeans_model.predict(lab_features_scaled)
            # Calculate similarity score based on cluster distribution

            cluster_counts = np.bincount(cluster_labels,
    minlength=self.n_clusters)
            cluster_percentages = cluster_counts / len(cluster_labels) * 100

            # Calculate distance to cluster centers

            distances = kmeans_model.transform(lab_features_scaled)
            avg_distance = np.mean(np.min(distances, axis=1))
            # Score based on inverse of average distance (lower distance =
    higher score)

            similarity_score = 1 / (1 + avg_distance)

            category_scores[category_name] = {
               'similarity_score': similarity_score,
               'avg_distance': avg_distance,
               'cluster_distribution': cluster_percentages
            }

        # Find best matching category

        best_category = max(category_scores.keys(),
                            key=lambda x: category_scores[x]['similarity_score'])
        return {
            'predicted_category': best_category,
            'confidence_score':
    category_scores[best_category]['similarity_score'],
            'all_scores': category_scores,
            'image_path': image_path
        }
```

u. Fungsi predict_multiple_images(self, image_paths, loaded_results)

Fungsi ini memproses sejumlah gambar sekaligus menggunakan fungsi prediksi kategori, dan menghasilkan prediksi batch dengan confidence masing-masing. Adapun codingannya adalah sebagai berikut:

```
def predict_multiple_images(self, image_paths, loaded_results):
        """Predict categories for multiple images"""

        predictions = []
```

```
        print(f"Predicting categories for {len(image_paths)} images...")

        for i, image_path in enumerate(image_paths):
            print(f"Processing image {i+1}/{len(image_paths)}:
{os.path.basename(image_path)}")

            prediction = self.predict_image_category(image_path,
loaded_results)
            if prediction:
                predictions.append(prediction)
                print(f"  Predicted: {prediction['predicted_category']} "
                      f"(confidence: {prediction['confidence_score']:.3f})")
            else:
                print(f"  Failed to process image")

        return predictions
```

v. Fungsi visualize_prediction_results(self, predictions)

Fungsi ini memvisualisasikan hasil prediksi dalam bentuk grafik, seperti boxplot skor kepercayaan dan pie chart distribusi kategori yang terprediksi. Adapun codingannya adalah sebagai berikut:

```
def visualize_prediction_results(self, predictions):
        """Visualize prediction results with confidence scores"""

        if not predictions:
            print("No predictions to visualize")
            return

        # Create summary dataframe

        pred_data = []
        for pred in predictions:
            pred_data.append({
                'Image': os.path.basename(pred['image_path']),
                'Predicted_Category': pred['predicted_category'],
                'Confidence': pred['confidence_score']
            })

        df = pd.DataFrame(pred_data)

        # Plot confidence distribution

        fig, axes = plt.subplots(1, 2, figsize=(15, 5))
```

```
        # Confidence by category

        sns.boxplot(data=df, x='Predicted_Category', y='Confidence',
    ax=axes[0])
        axes[0].set_title('Prediction Confidence by Category')
        axes[0].tick_params(axis='x', rotation=45)

        # Category distribution

        category_counts = df['Predicted_Category'].value_counts()
        axes[1].pie(category_counts.values, labels=category_counts.index,
    autopct='%1.1f%%')
        axes[1].set_title('Predicted Category Distribution')

        plt.tight_layout()
        plt.show()

        # Print summary statistics

        print("\n=== Prediction Summary ===")
        print(f"Total images processed: {len(predictions)}")
        print("\nCategory distribution:")
        for category, count in category_counts.items():
            print(f"  {category}: {count} images
    ({count/len(predictions)*100:.1f}%)")
        print(f"\nAverage confidence: {df['Confidence'].mean():.3f}")
        print(f"Confidence range: {df['Confidence'].min():.3f} -
    {df['Confidence'].max():.3f}")

        return df
```

w. Fungsi predict_and_visualize(self, image_path, loaded_results, show_clusters=True)

Fungsi ini menggabungkan proses prediksi dan visualisasi hasil segmentasi untuk satu gambar, ideal untuk pengujian cepat. Adapun codingannya adalah sebagai berikut:

```
def predict_and_visualize(self, image_path, loaded_results,
show_clusters=True):
    """Predict category and show cluster segmentation"""

    # Get prediction

    prediction = self.predict_image_category(image_path,
loaded_results)
```

```
        if not prediction:
            return None

        print(f"\n=== Prediction Results for
{os.path.basename(image_path)} ===")
        print(f"Predicted Category: {prediction['predicted_category']}")
        print(f"Confidence Score: {prediction['confidence_score']:.3f}")

        print("\nAll category scores:")
        for category, scores in prediction['all_scores'].items():
            print(f"  {category}: {scores['similarity_score']:.3f}")

        # Show cluster segmentation if requested

        if show_clusters:
            best_category = prediction['predicted_category']
            kmeans_model = loaded_results[best_category]['kmeans_model']

            print(f"\nShowing cluster segmentation using {best_category}
model:")
            self.visualize_cluster_segmentation_horizontal(image_path,
kmeans_model)

        return prediction
```

3. Analisis dan Segmentasi Warna berbasis Klaster menggunakan Algoritma K-Means

   Program melakukan segmentasi warna pada gambar-gambar kakao yang dikelompokkan dalam kategori tertentu (misalnya, kakao matang, mentah, fermentasi, dll). Segmentasi dilakukan menggunakan K-Means clustering, yaitu algoritma yang mengelompokkan warna dalam gambar ke dalam sejumlah klaster (dalam hal ini n_clusters = 5). Adapun tahapan pada analisis dan segmentasi klaster adalah sebagai berikut:

   a. Fungsi main()

      Fungsi ini merupakan fungsi utama yang menjalankan keseluruhan proses segmentasi warna pada dataset gambar kakao. Proses diawali dengan menentukan path dataset dan menginisialisasi objek

CacaoColorSegmentation dengan parameter n_clusters=5, yang berarti akan dilakukan pengelompokan warna menjadi 5 klaster. Fungsi ini kemudian menjalankan metode analyze_all_categories() untuk memproses seluruh kategori gambar dalam dataset, menghasilkan model KMeans dan data statistik warna untuk setiap kategori. Selanjutnya, apabila terdapat lebih dari satu kategori, dilakukan perbandingan nilai warna antar kategori menggunakan compare_categories(), yang akan menampilkan rata-rata nilai LAB dari masing-masing kategori. Setelah itu, fungsi ini membuat visualisasi segmentasi warna secara horizontal untuk gambar-gambar contoh dari tiap kategori menggunakan dua metode, yaitu visualize_cluster_segmentation_horizontal() untuk satu gambar dan create_combined_cluster_visualization_horizontal() untuk beberapa gambar. Fungsi save_models() kemudian digunakan untuk menyimpan model hasil pelatihan. Terakhir, statistik distribusi klaster dalam setiap kategori dianalisis dan ditampilkan dengan analyze_cluster_statistics() sebelum fungsi main() ditutup dengan pesan akhir dan mengembalikan hasil analisis serta path model yang disimpan. Adapun codingannya adalah sebagai berikut:

```
def main():
    dataset_path = "/Users/pac/Documents/COCOA-
KMEANS/dataset"  # Replace with your actual path

    # Initialize the segmentation class

    segmenter = CacaoColorSegmentation(dataset_path, n_clusters=5)

    # Analyze all categories

    print("Starting color segmentation analysis...")
    results = segmenter.analyze_all_categories()

    # Compare categories
```

```
    if len(results) > 1:
        print("\n=== Comparing Categories ===")
        comparison_df = segmenter.compare_categories(results)
        print("\nLAB Color Comparison Summary:")
        print(comparison_df.groupby('Category')[['L', 'A', 'B']].mean())
    # Create HORIZONTAL cluster segmentation visualizations for
sample images

    print("\n=== Creating HORIZONTAL Cluster Segmentation
Visualizations ===")
    for category_name, result in results.items():
        print(f"\nProcessing {category_name}...")

        # Get some sample image paths

        category_path = segmenter.dataset_path / category_name
        images, image_paths =
segmenter.load_images_from_folder(category_path)

        if len(image_paths) > 0:
            # Single image detailed HORIZONTAL view
            sample_image = image_paths[0]  # First image
            print(f"Creating horizontal detailed cluster view for:

{sample_image}")
            segmenter.visualize_cluster_segmentation_horizontal(sample_im
age, result['kmeans_model'])
            # Combined HORIZONTAL view of multiple images (max 6)

            if len(image_paths) >= 3:
                print(f"Creating horizontal combined cluster view for
{category_name}")
                segmenter.create_combined_cluster_visualization_horizontal(
                    image_paths[:6], result['kmeans_model'], max_images=6
                )

    # Save models using joblib

    print("\n=== Saving Models ===")
    saved_model_path = segmenter.save_models(results)
    # Analyze cluster statistics

    print("\n=== Cluster Statistics Analysis ===")
    cluster_stats = segmenter.analyze_cluster_statistics(results)

    # Print cluster statistics summary

    for category, stats in cluster_stats.items():
        print(f"\n{category} - Cluster Distribution:")
        for cluster, data in stats.items():
            print(f"  {cluster}: {data['percentage']:.1f}% - "
```

```
                f"LAB({data['mean_L']:.1f}, {data['mean_A']:.1f},
{data['mean_B']:.1f})")

    print("\nAnalysis complete!")
    print(f"Models saved at: {saved_model_path}")

    return results, segmenter, saved_model_path
```

b. Fungsi create_cluster_visualizations_batch()

Fungsi ini bertugas membuat dan menyimpan visualisasi klaster untuk sejumlah gambar dalam tiap kategori secara otomatis. Fungsi ini pertama-tama membuat direktori output untuk menyimpan hasil visualisasi. Kemudian, untuk setiap kategori, fungsi memuat maksimal sepuluh gambar, melakukan segmentasi menggunakan model KMeans yang telah dilatih, dan menghasilkan gambar visualisasi klaster. Visualisasi tersebut disusun secara vertikal, dimulai dari gambar asli, diikuti oleh masing-masing klaster yang diberi warna berbeda (misalnya merah, hijau, biru, kuning, dll.). Seluruh hasil visualisasi disimpan dalam bentuk file PNG. Fungsi ini sangat bermanfaat untuk mengevaluasi dan mendokumentasikan hasil segmentasi klaster secara massal. Adapun codingannya adalah sebagai berikut:

```
def create_cluster_visualizations_batch(segmenter, results,
output_folder="cluster_outputs"):
    """Create cluster visualizations for multiple images and save them"""
    import os

    # Create output directory

    os.makedirs(output_folder, exist_ok=True)

    for category_name, result in results.items():
        category_output = os.path.join(output_folder, category_name)
        os.makedirs(category_output, exist_ok=True)

        # Get image paths
```

```
        category_path = segmenter.dataset_path / category_name
        images, image_paths =
segmenter.load_images_from_folder(category_path)
        # Process up to 10 images per category

        for i, image_path in enumerate(image_paths[:10]):
            output_path = os.path.join(category_output,
f"clusters_{i+1}.png")
            print(f"Processing {category_name} - Image {i+1}/{min(10,
len(image_paths))}")

            # Get segmentation

            segmentation_result =
segmenter.segment_image_colors(image_path, result['kmeans_model'])
            if segmentation_result is None:
                continue
            segmented, original_img = segmentation_result

            # Create and save visualization

            fig, axes = plt.subplots(segmenter.n_clusters + 1, 1, figsize=(8, 2
* (segmenter.n_clusters + 1)))

            # Original image

            original_rgb = cv2.cvtColor(original_img,
cv2.COLOR_BGR2RGB)
            axes[0].imshow(original_rgb)
            axes[0].set_title(f'Original - {os.path.basename(image_path)}')
            axes[0].axis('off')

            # Cluster images

            cluster_colors = [
                [255, 0, 0], [0, 255, 0], [0, 0, 255], [255, 255, 0], [255, 0, 255],
                [0, 255, 255], [255, 128, 0], [128, 0, 255], [0, 128, 255], [255,
255, 255]
            ]

            height, width = segmented.shape
            for cluster_id in range(segmenter.n_clusters):
                cluster_img = np.zeros((height, width, 3), dtype=np.uint8)
                color = cluster_colors[cluster_id % len(cluster_colors)]
                mask = segmented == cluster_id
                cluster_img[mask] = color

                cluster_rgb = cv2.cvtColor(cluster_img,
cv2.COLOR_BGR2RGB)
                axes[cluster_id + 1].imshow(cluster_rgb)
                axes[cluster_id + 1].set_title(f'Cluster {cluster_id}')
```

```
            axes[cluster_id + 1].axis('off')

        plt.tight_layout()
        plt.savefig(output_path, dpi=150, bbox_inches='tight')
        plt.close()  # Close to save memory

    print(f"Cluster visualizations saved to: {output_folder}")
```

c. Fungsi predict_new_images()

Fungsi ini digunakan untuk memprediksi kategori dari gambar baru menggunakan model segmentasi warna yang telah disimpan sebelumnya. Fungsi ini memuat model KMeans dari file menggunakan load_model() dan kemudian memproses setiap gambar baru yang diberikan. Untuk setiap gambar, dilakukan prediksi kategori berdasarkan distribusi warnanya, dan ditampilkan hasil prediksi serta tingkat kepercayaannya (confidence score). Fungsi ini memungkinkan sistem untuk digunakan dalam konteks nyata, misalnya dalam klasifikasi warna biji kakao baru tanpa harus melatih ulang model. Adapun codingannya adalah sebagai berikut:

```
def predict_new_images(model_path, image_paths):
    """Streamlined function to predict new images using saved model"""

    # Initialize segmenter

    segmenter = CacaoColorSegmentation("dummy_path", n_clusters=5)
    # Load models

    loaded_results = segmenter.load_model(model_path)

    if not loaded_results:
        print("Failed to load models")
        return None

    # Make predictions

    predictions = []

    for image_path in image_paths:
```

```
            prediction = segmenter.predict_image_category(image_path,
loaded_results)
        if prediction:
            predictions.append(prediction)
            print(f"Image: {os.path.basename(image_path)}")
            print(f"  Predicted: {prediction['predicted_category']}")
            print(f"  Confidence: {prediction['confidence_score']:.3f}")

    return predictions
```

d. Blok if __name___== "__main__"

Blok ini berfungsi untuk mengeksekusi fungsi main() ketika skrip dijalankan langsung. Blok ini memastikan bahwa proses pelatihan model, analisis kategori, visualisasi, dan penyimpanan model dijalankan secara otomatis saat file dijalankan. Di bagian akhir blok ini, terdapat dua baris kode tambahan yang dikomentari: satu untuk memanggil fungsi create_cluster_visualizations_batch() guna menyimpan visualisasi secara massal, dan satu lagi untuk menggunakan model dalam memprediksi gambar baru dengan memanggil predict_new_images(). Kedua fungsi tambahan ini dapat diaktifkan kapan saja sesuai kebutuhan pengguna. Adapun codingannya adalah sebagai berikut:

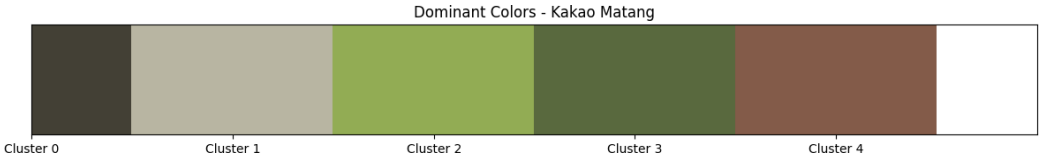
```
if __name___== "__main__":
    # Train models and save them

    results, segmenter, saved_model_path = main()

    # Optionally create batch visualizations

    # create_cluster_visualizations_batch(segmenter, results)

    # Use the model for prediction
    # print("\n" + "="*50)

    # print("USING MODEL FOR PREDICTION")

    # print("="*50)
```

e. Hasil Output

Berikut adalah penjelasan detail dari output program segmentasi warna berbasis K-Means untuk citra kakao:

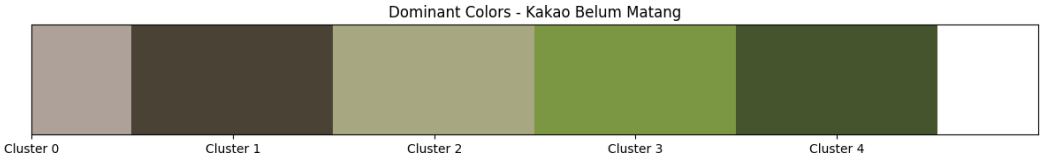
Starting color segmentation analysis...

=== Analyzing Kakao Matang ===
Loaded 257 images from Kakao Matang
Extracted 257000 LAB pixel samples



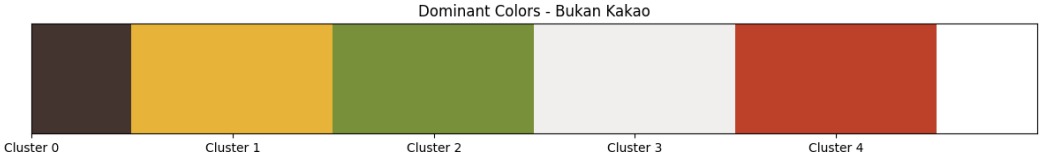
Dominant LAB colors for Kakao Matang:
Cluster 0: LAB(69.4, 127.4, 135.3) - RGB(67, 64, 53)
Cluster 1: LAB(187.3, 127.0, 138.2) - RGB(184, 181, 162)
Cluster 2: LAB(170.8, 105.4, 170.3) - RGB(146, 172, 84)
Cluster 3: LAB(107.5, 114.9, 150.4) - RGB(89, 105, 62)
Cluster 4: LAB(108.7, 142.9, 145.7) - RGB(131, 91, 73)

=== Analyzing Kakao Belum Matang ===
Loaded 307 images from Kakao Belum Matang
Extracted 307000 LAB pixel samples



Dominant LAB colors for Kakao Belum Matang:
Cluster 0: LAB(171.7, 131.6, 135.0) - RGB(173, 161, 153)
Cluster 1: LAB(72.7, 129.5, 137.9) - RGB(73, 66, 53)
Cluster 2: LAB(173.7, 121.2, 149.0) - RGB(167, 168, 129)
Cluster 3: LAB(150.3, 105.0, 168.8) - RGB(124, 151, 68)
Cluster 4: LAB(86.3, 115.8, 149.4) - RGB(70, 84, 45)

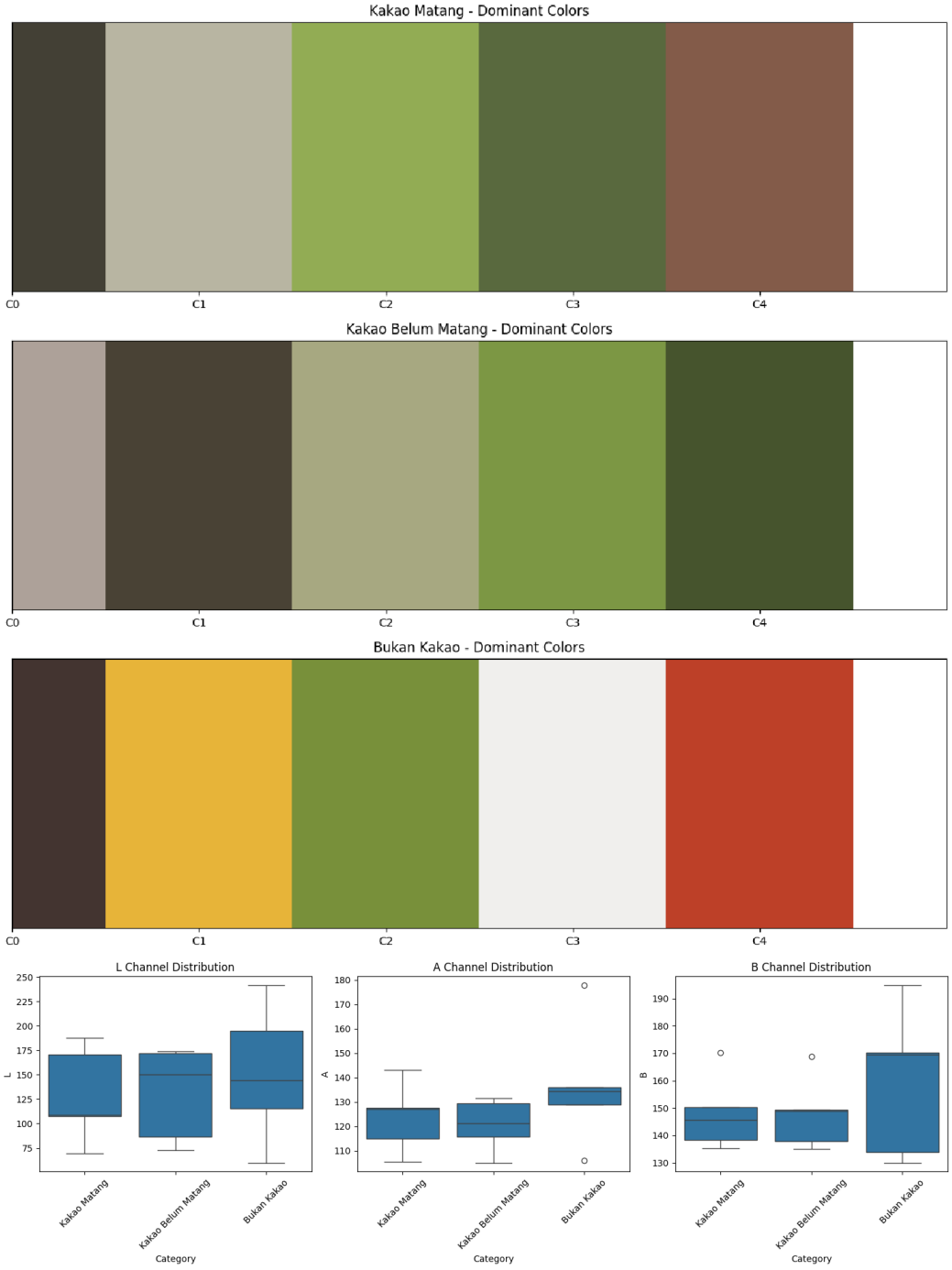
=== Analyzing Bukan Kakao ===
libpng warning: iCCP: known incorrect sRGB profile
Loaded 237 images from Bukan Kakao
Extracted 237000 LAB pixel samples



Dominant LAB colors for Bukan Kakao:
Cluster 0: LAB(59.9, 134.3, 133.9) - RGB(67, 52, 48)
Cluster 1: LAB(194.8, 135.8, 194.8) - RGB(231, 180, 57)

Cluster 2: LAB(144.6, 106.1, 170.3) - RGB(120, 144, 58)
Cluster 3: LAB(241.4, 128.8, 130.0) - RGB(240, 239, 237)
Cluster 4: LAB(115.3, 177.8, 169.5) - RGB(189, 64, 40)

=== Comparing Categories ===



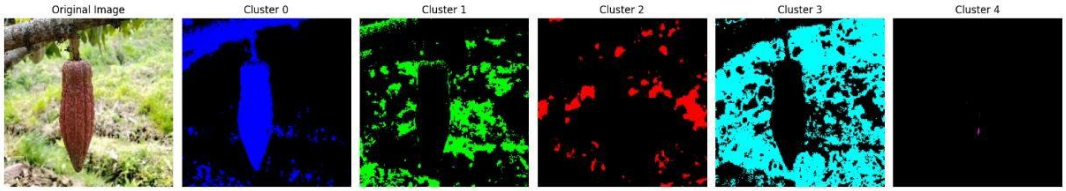
LAB Color Comparison Summary:

Category	L	A	B
Bukan Kakao	151.189536	136.556401	159.710782

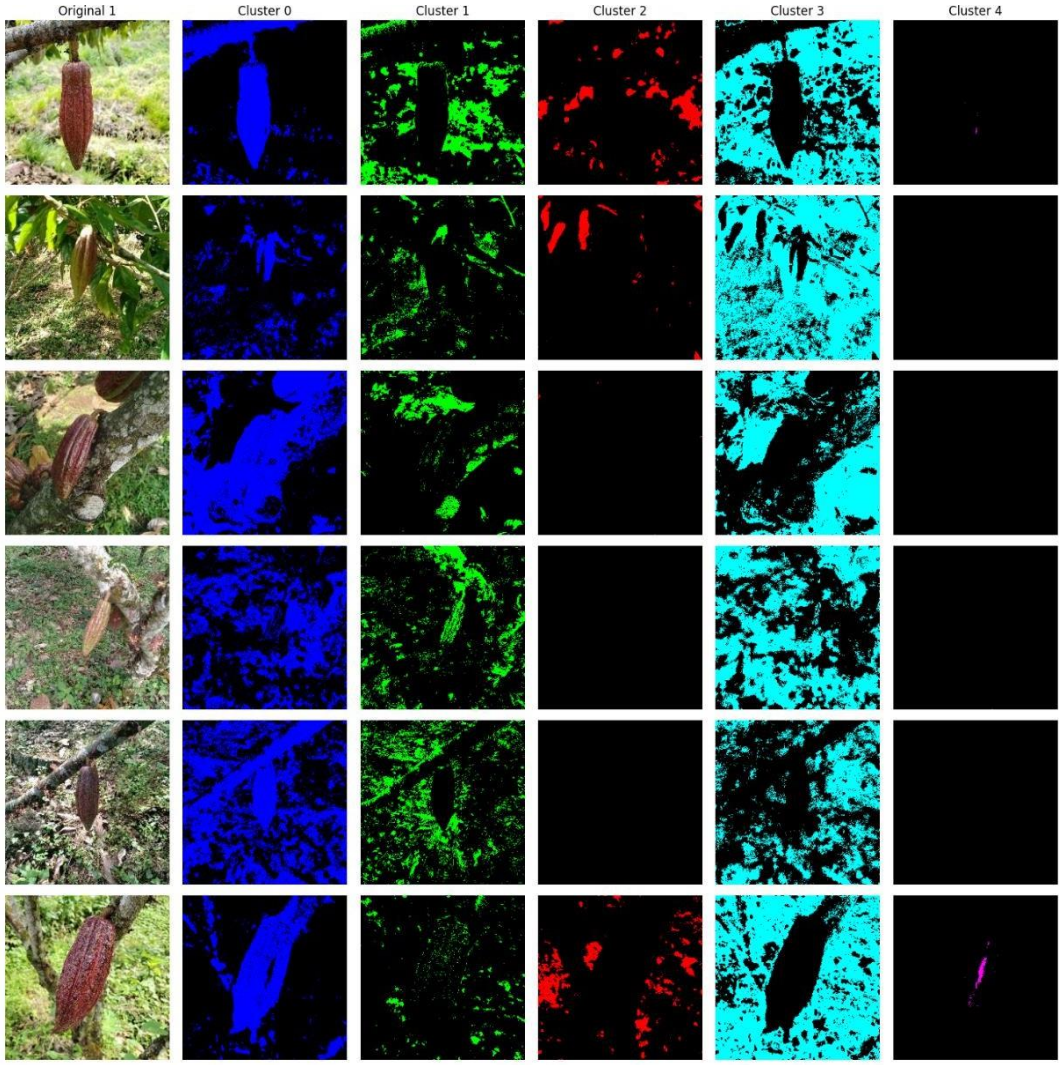
Kakao Belum Matang 130.966880 120.621817 148.015370
Kakao Matang 128.767139 123.541923 147.981503

=== Creating HORIZONTAL Cluster Segmentation Visualizations ===

Processing Kakao Matang...
Creating horizontal detailed cluster view for:
/Users/pac/Documents/COCOA-KMEANS/dataset/Kakao Matang/114_jpg.rf.2fb589e95943087c377d653aedb7760f.jpg

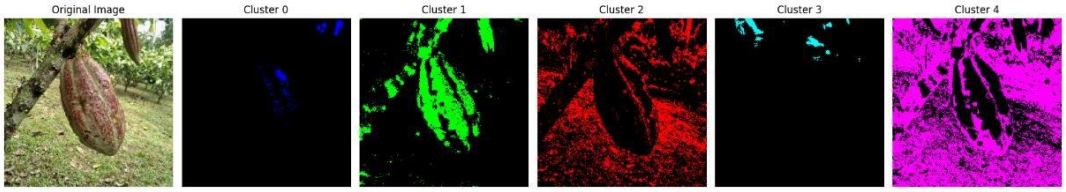


Creating horizontal combined cluster view for Kakao Matang

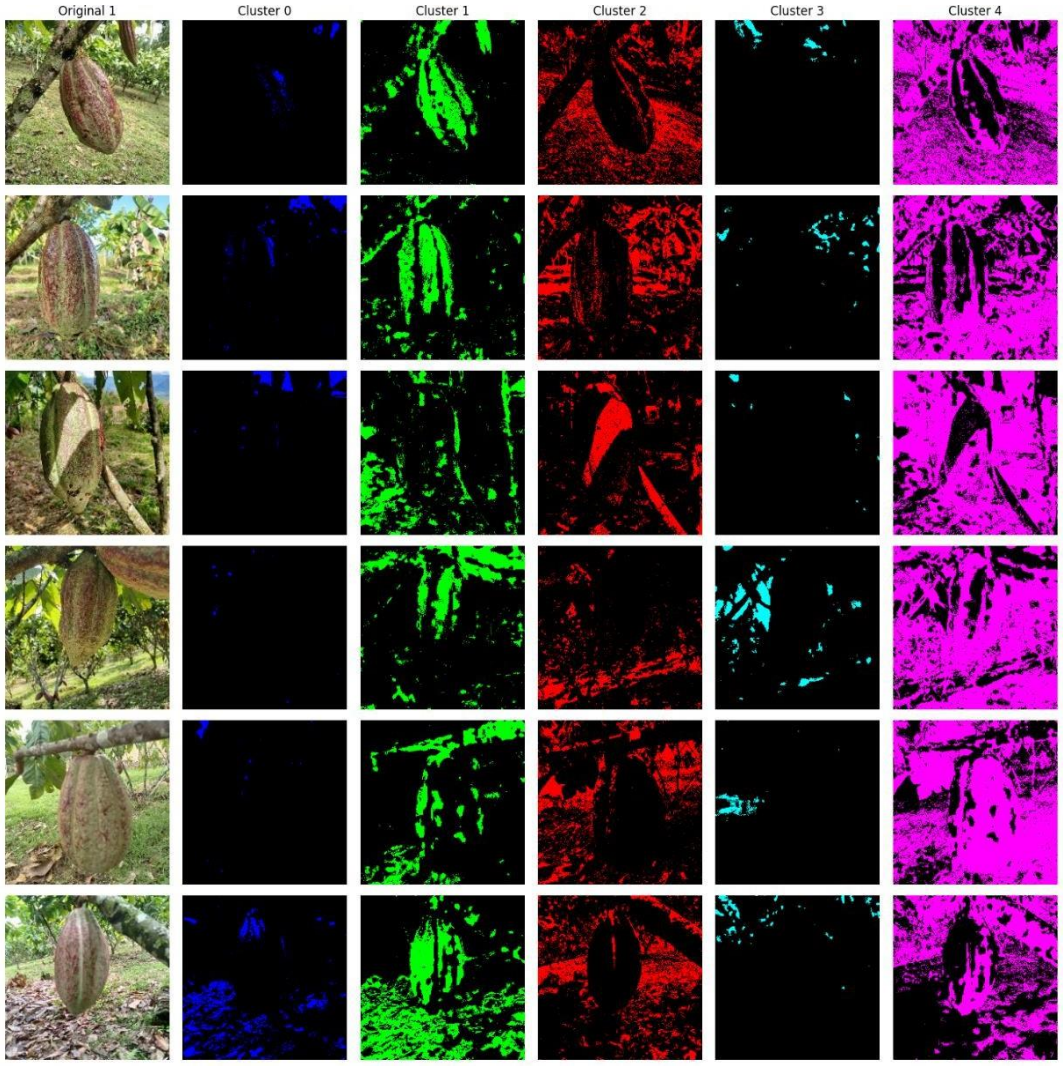


Processing Kakao Belum Matang...

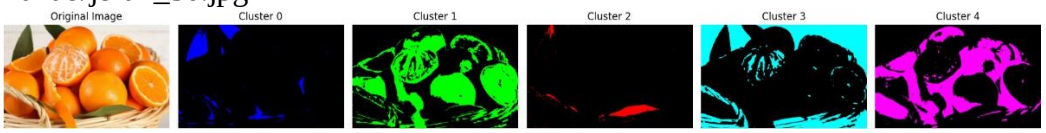
Creating horizontal detailed cluster view for:
/Users/pac/Documents/COCOA-KMEANS/dataset/Kakao Belum Matang/1058_jpg.rf.e2d7bbf783a05ecbd6a5d6a61af9244d.jpg



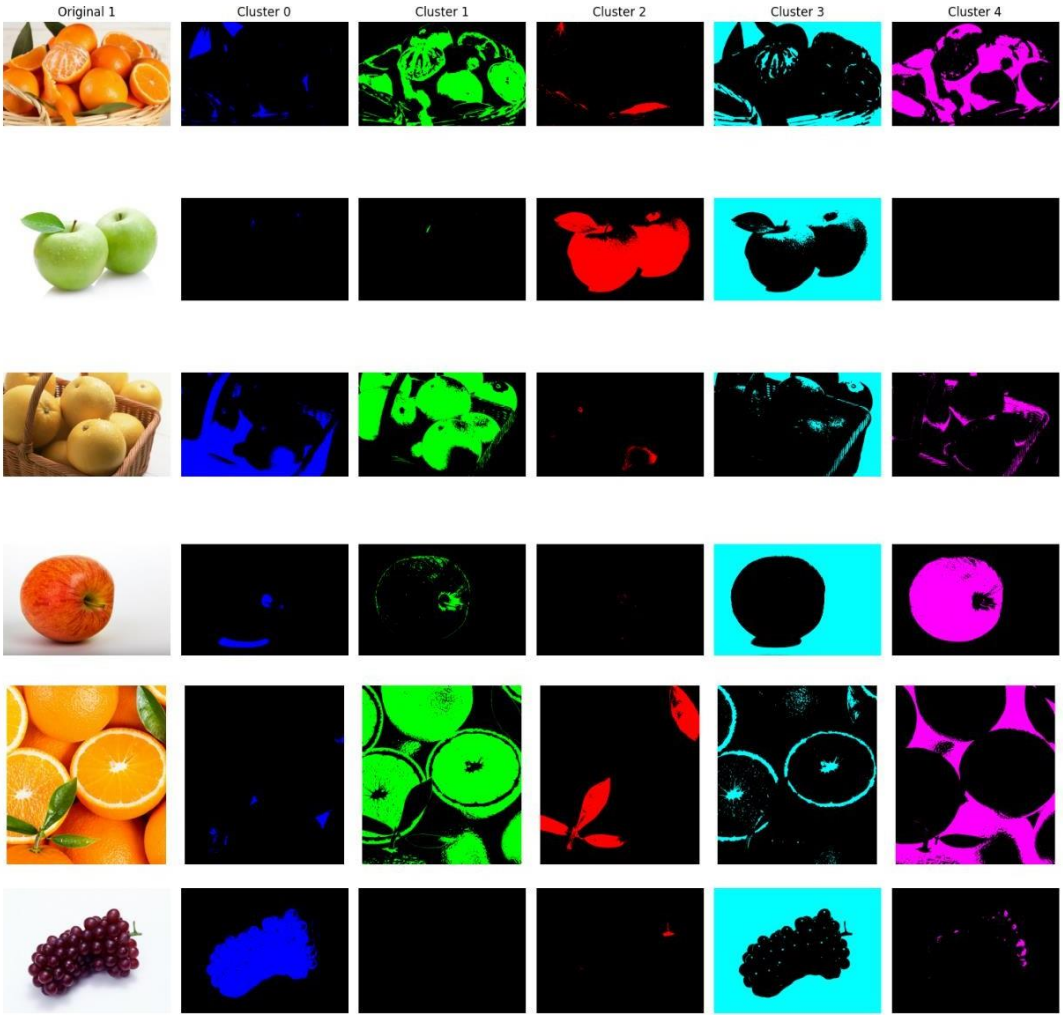
Creating horizontal combined cluster view for Kakao Belum Matang



Processing Bukan Kakao...
libpng warning: iCCP: known incorrect sRGB profile
Creating horizontal detailed cluster view for:
/Users/pac/Documents/COCOA-KMEANS/dataset/Bukan Kakao/jeruk_30.jpg



Creating horizontal combined cluster view for Bukan Kakao



=== Saving Models ===
All K-means models saved:
saved_models/all_kmeans_models_20250525_202920.joblib

=== Cluster Statistics Analysis ===

Kakao Matang - Cluster Distribution:
  Cluster_0: 44.2% - LAB(86.2, 130.1, 136.9)
  Cluster_1: 15.5% - LAB(212.8, 128.6, 137.5)
  Cluster_2: 1.9% - LAB(199.6, 99.3, 188.8)
  Cluster_3: 38.0% - LAB(139.9, 114.6, 153.9)
  Cluster_4: 0.5% - LAB(125.5, 160.5, 157.4)

Kakao Belum Matang - Cluster Distribution:
  Cluster_0: 5.1% - LAB(205.1, 133.4, 130.4)
  Cluster_1: 30.4% - LAB(90.8, 132.1, 135.8)
  Cluster_2: 14.3% - LAB(207.3, 122.8, 146.8)
  Cluster_3: 1.1% - LAB(189.7, 98.2, 188.4)
  Cluster_4: 49.1% - LAB(113.3, 117.6, 150.0)

Bukan Kakao - Cluster Distribution:
  Cluster_0: 16.0% - LAB(59.9, 134.3, 133.9)
  Cluster_1: 17.7% - LAB(194.8, 135.8, 194.8)
  Cluster_2: 13.6% - LAB(144.4, 106.1, 170.3)
  Cluster_3: 35.4% - LAB(241.4, 128.8, 130.0)
  Cluster_4: 17.3% - LAB(115.3, 177.8, 169.6)

Analysis complete!
Models saved at:
saved_models/all_kmeans_models_20250525_202920.joblib

Berikut adalah penjelasan detail dari output program segmentasi warna berbasis K-Means untuk citra kakao:

“Starting color segmentation analysis...”

Ini menandai dimulainya proses analisis segmentasi warna berbasis warna dominan menggunakan algoritma K-Means Clustering dalam warna LAB.

=== Analyzing Kakao Matang ===

Loaded 257 images from Kakao Matang

Program membaca 257 gambar dari folder "Kakao Matang".

Extracted 257000 LAB pixel samples

Dari seluruh gambar, diambil total 257.000 sampel piksel dan dikonversi ke format warna LAB.

Dominant LAB colors for Kakao Matang:

Menampilkan 5 warna dominan hasil clustering:

Cluster 0 s.d. 4: Setiap baris menunjukkan rata-rata nilai LAB dan RGB untuk cluster tersebut.

Contoh:

Cluster 0: LAB(69.4, 127.4, 135.3) - RGB(67, 64, 53)

Ini berarti kelompok warna 0 memiliki karakteristik warna dominan gelap kehijauan, tipikal kakao matang.

=== Analyzing Kakao Belum Matang ===

Loaded 307 images from Kakao Belum Matang

Memproses 307 gambar dari kakao belum matang.

Extracted 307000 LAB pixel samples

Sama seperti sebelumnya, mengambil 307.000 piksel sebagai sampel dari gambar.

Dominant LAB colors for Kakao Belum Matang:

Ditampilkan 5 cluster warna dominan seperti sebelumnya, yang secara umum lebih cerah dan kehijauan dibanding kakao matang, menunjukkan visual buah yang belum matang.

=== Analyzing Bukan Kakao ===

Loaded 237 images from Bukan Kakao

Memuat 237 gambar dari kategori **bukan kakao** (misalnya buah lain).

Extracted 237000 LAB pixel samples

Ekstraksi 237.000 piksel dalam ruang warna LAB.

Dominant LAB colors for Bukan Kakao:

Menampilkan warna-warna dominan yang lebih terang dan lebih beragam (misalnya kuning terang, putih cerah, merah), menunjukkan warna yang berbeda jauh dari kakao.

=== Comparing Categories ===

LAB Color Comparison Summary:

Perbandingan rata-rata warna LAB keseluruhan dari setiap kategori:

Kategori	**L (Lightness)**	**A (Green–Red)**	**B (Blue–Yellow)**
Bukan Kakao	151.19	136.56	159.71
Kakao Belum Matang	130.97	120.62	148.02
Kakao Matang	128.77	123.54	147.98

Terlihat bahwa:

- Bukan Kakao jauh lebih terang (L tinggi) dan kuning (B tinggi).
- Kakao Matang memiliki nilai L paling rendah, artinya lebih gelap.
- Kakao Belum Matang ada di tengah-tengah.

=== Creating HORIZONTAL Cluster Segmentation Visualizations ===

Untuk setiap kategori, program:

- Mengambil satu sampel gambar.
- Menampilkan visualisasi horizontal dari segmentasi warna hasil K-Means.
- Menunjukkan detil tiap cluster warna pada gambar tersebut.

Contoh:

Creating horizontal detailed cluster view for: Kakao Matang/114_jpg

Setelah itu dibuat juga tampilan **gabungan** dari semua cluster dalam satu gambar.

=== Saving Models ===

Model K-Means yang telah dilatih untuk semua kategori disimpan ke file:

saved_models/all_kmeans_models_20250525_202920.joblib

Ini memungkinkan model untuk digunakan kembali tanpa retraining.

=== Cluster Statistics Analysis ===

Analisis distribusi warna untuk tiap kategori:

Kakao Matang:

Cluster	**Persentase**	**LAB (Rata-rata)**
Cluster_0	44.2%	LAB(86.2, 130.1, 136.9) — warna dominan gelap
Cluster_1	15.5%	LAB(212.8, 128.6, 137.5) — warna cerah
Cluster_2	1.9%	LAB(199.6, 99.3, 188.8) — warna keunguan
Cluster_3	38.0%	LAB(139.9, 114.6, 153.9) — hijau kekuningan
Cluster_4	0.5%	LAB(125.5, 160.5, 157.4) — warna minor merah-oranye

Dominasi warna gelap dan hijau tua.

Kakao Belum Matang:

Cluster	**Persentase**	**LAB**
Cluster_0	5.1%	Cerah
Cluster_1	30.4%	Coklat tua

Cluster_2	14.3%	Hijau terang
Cluster_3	1.1%	Ungu tua
Cluster_4	49.1%	Hijau kekuningan (dominan)

Dominan warna hijau kekuningan (lebih muda dibanding matang).

Bukan Kakao:

Cluster	**Persentase**	**LAB**
Cluster_0	16.0%	Gelap kemerahan
Cluster_1	17.7%	Kuning cerah
Cluster_2	13.6%	Hijau terang
Cluster_3	35.4%	Putih terang (sangat cerah)
Cluster_4	17.3%	Merah kekuningan

Warna sangat beragam, tidak khas kakao.

4. Analisis Output Prediksi Model Segmentasi Warna Berbasis K-Means

   Analisis ini bertujuan untuk mengevaluasi performa model segmentasi warna yang menggunakan algoritma K-Means Clustering dalam mengklasifikasikan jenis buah kakao berdasarkan warna dominannya. Model ini dilatih untuk mengenali pola warna dari tiga kategori utama: Kakao Matang, Kakao Belum Matang, dan Bukan Kakao. Dalam proses pengujian, sistem memproses sebuah gambar input dan membandingkan klaster warna dominannya terhadap hasil pelatihan dari setiap kategori. Kategori dengan tingkat kemiripan tertinggi dianggap sebagai hasil prediksi akhir.

Penggunaan K-Means memungkinkan sistem untuk membagi gambar ke dalam sejumlah klaster warna (misalnya 5 klaster) tanpa supervisi langsung, yang kemudian digunakan sebagai ciri utama dalam proses klasifikasi. Setelah model memprediksi kategori gambar, sistem juga memberikan confidence score yang menggambarkan seberapa dekat kesamaan warna gambar uji dengan pola warna kategori tertentu.

Salah satu hal yang penting dari output ini adalah bahwa nilai confidence antar kategori bisa sangat berdekatan, terutama ketika gambar mengandung warna yang ambigu atau berada dalam fase transisi (seperti kakao yang mulai matang). Selain prediksi, sistem juga menampilkan visualisasi segmentasi warna, yang membantu pengguna memahami bagaimana model "melihat" dan mengelompokkan warna dalam citra tersebut.

Secara keseluruhan, pendekatan ini tidak hanya memberikan klasifikasi sederhana, tetapi juga menekankan analisis warna dominan sebagai basis pengambilan keputusan. Pendekatan ini cocok digunakan dalam domain pertanian dan industri pangan untuk membantu otomatisasi pengenalan kondisi buah, dengan potensi dikembangkan lebih lanjut melalui integrasi teknik klasifikasi lanjutan. Adapun codingannya adalah sebagai berikut:

```
def test_single_image():
    """Simple function to test one image"""

    # Initialize segmenter

    segmenter = CacaoColorSegmentation("dummy_path", n_clusters=5)

    # Load the saved models

    model_path = "/Users/pac/Documents/COCOA-KMEANS/saved_models/all_kmeans_models_20250525_202920.joblib"
    loaded_results = segmenter.load_model(model_path)

    if loaded_results:
```

```
        print("Models loaded successfully!")

        # Test image path (update this to your actual image path)
                        test_image = "/Users/pac/Documents/COCOA-
KMEANS/dataset/Kakao Belum
Matang/693_jpg.rf.ab4f6d98a43ed72b2ce5416c60c1f8c3.jpg"

        # Check if file exists

        if os.path.exists(test_image):
            print(f"Testing image: {os.path.basename(test_image)}")

            # Get prediction with visualization

                    prediction = segmenter.predict_and_visualize(test_image,
loaded_results)

            if prediction:
                print(f"\n✅ SUCCESS!")
                print(f"Predicted Category: {prediction['predicted_category']}")
                print(f"Confidence: {prediction['confidence_score']:.3f}")
                return prediction
            else:
                print("❌ Prediction failed")
                return None
        else:
            print(f"❌ Image not found: {test_image}")
            print("Please check the file path")
            return None
    else:
        print("❌ Failed to load models")
        return None

# Run the test

prediction = test_single_image()
```

Penjelasan dari codingan diatas adalah sebagai berikut:

a. Fungsi test_single_image()

Fungsi ini dirancang untuk menguji satu citra kakao menggunakan model segmentasi warna berbasis K-Means. Tujuan utamanya adalah menjalankan keseluruhan alur kerja — mulai dari memuat model, membaca citra, melakukan prediksi, hingga menampilkan hasil prediksi dan visualisasi segmentasi warna. Hal ini sangat berguna sebagai langkah

validasi akhir untuk memastikan bahwa model yang telah dilatih dapat diaplikasikan pada gambar baru secara efektif.

b. Inisialisasi Objek Segmentasi

Baris segmenter = CacaoColorSegmentation("dummy_path", n_clusters=5) menunjukkan pembuatan objek dari class CacaoColorSegmentation, yang secara khusus dibuat untuk menangani segmentasi warna buah kakao. Nilai n_clusters=5 menandakan bahwa metode K-Means akan membagi gambar ke dalam lima kelompok warna dominan. Parameter "dummy_path" tidak digunakan langsung dalam prediksi ini, tetapi kemungkinan disiapkan untuk kompatibilitas class yang memerlukan path saat inisialisasi.

c. Pemanggilan Fungsi load_model()

Langkah berikutnya adalah memuat model yang telah disimpan sebelumnya dengan memanggil fungsi segmenter.load_model(model_path). Path model mengarah ke file .joblib yang menyimpan hasil pelatihan dari beberapa model segmentasi berdasarkan kategori kakao. Output berhasilnya proses ini ditandai dengan pesan: "Combined models loaded successfully..." dan "Models loaded successfully!". Ini menunjukkan bahwa semua model (misalnya: Kakao Matang, Belum Matang, dan Bukan Kakao) tersedia dan dapat digunakan untuk prediksi.

d. Persiapan dan Validasi Gambar Uji

Gambar yang akan diuji berasal dari kategori “Kakao Belum Matang”. Fungsi melakukan validasi terlebih dahulu untuk memastikan bahwa path menuju file gambar memang valid dan file-nya ada. Jika tidak ditemukan, program akan berhenti dan memberikan notifikasi bahwa file tidak ditemukan. Dalam skenario ini, file ditemukan dan proses dilanjutkan ke tahap prediksi.

e. Proses Prediksi dan Visualisasi

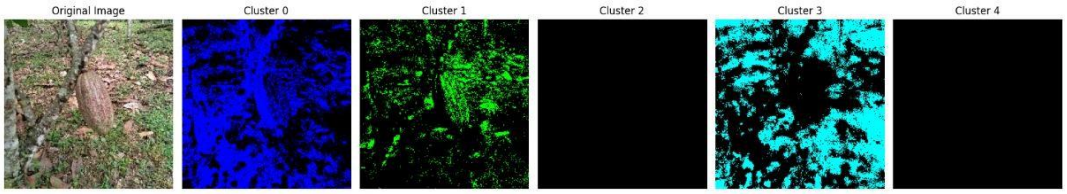
Pemrosesan utama terjadi di segmenter.predict_and_visualize(test_image, loaded_results). Fungsi ini memproses gambar uji dengan membandingkan ciri warna dominan terhadap model yang telah dimuat sebelumnya. Setiap model menghitung seberapa mirip klaster warna gambar terhadap klaster yang dimiliki model. Kemudian, model dengan nilai skor kemiripan tertinggi dianggap sebagai prediksi akhir. Selain itu, fungsi ini juga menampilkan hasil segmentasi visual berdasarkan model terbaik, memberi gambaran bagaimana warna-warna dikelompokkan pada citra.

Setelah mengetahui analisis codingan diatas, maka selanjutnya akan menjelaskan mengenai hasil dari output codingan tersebut. Adapun hasil output dari codingan adalah sebagai berikut:

```
Combined models loaded successfully from:
/Users/pac/Documents/COCOA-
KMEANS/saved_models/all_kmeans_models_20250525_202920.joblib
Models loaded successfully!
Testing image: 693_jpg.rf.ab4f6d98a43ed72b2ce5416c60c1f8c3.jpg

=== Prediction Results for
693_jpg.rf.ab4f6d98a43ed72b2ce5416c60c1f8c3.jpg ===
Predicted Category: Kakao Matang
Confidence Score: 0.537
```

All category scores:
Kakao Matang: 0.537
Kakao Belum Matang: 0.515
Bukan Kakao: 0.536

Showing cluster segmentation using Kakao Matang model:



✅ SUCCESS!
Predicted Category: Kakao Matang
Confidence: 0.537

Output utama dari proses ini menunjukkan bahwa gambar diklasifikasikan sebagai Kakao Matang dengan confidence score 0.537 (atau 53.7%). Menariknya, dua kategori lainnya yaitu Kakao Belum Matang (0.515) dan Bukan Kakao (0.536) juga memiliki skor yang sangat mendekati. Model yang dipilih untuk melakukan visualisasi segmentasi warna adalah Kakao Matang, karena skornya yang sedikit lebih tinggi dibanding yang lain. Visualisasi ini akan membantu pengguna dalam melihat hasil klasterisasi warna dominan pada gambar dan mengevaluasi apakah distribusi warnanya memang sesuai dengan yang diasosiasikan dengan kakao matang. Ini sangat berguna dalam proses analisis lanjutan untuk validasi model.

Dari analisis tersebut, disimpulkan bahwa Fungsi test_single_image() berhasil mengeksekusi seluruh proses dari awal hingga akhir memuat model, membaca gambar, menjalankan prediksi, dan menghasilkan visualisasi segmentasi. Ini membuktikan bahwa pipeline segmentasi warna sudah berjalan dengan baik. Meski begitu, nilai confidence yang hanya sekitar 53% menunjukkan bahwa akurasi klasifikasi masih bisa

ditingkatkan. Beberapa pendekatan untuk meningkatkan kinerja termasuk memperbesar dataset, melakukan preprocessing warna (misalnya color normalization), atau menerapkan metode klasifikasi lain yang lebih kompleks dari K-Means.

4.2 Hasil dalam API Atau Server

API ini merupakan sebuah sistem berbasis web yang dibangun menggunakan Python dan Flask dengan tujuan untuk melakukan segmentasi warna pada gambar buah kakao. Segmentasi warna ini dilakukan menggunakan algoritma K-Means Clustering yang mengelompokkan warna-warna pada gambar menjadi beberapa klaster dominan. Hasil klasterisasi warna ini kemudian dianalisis untuk memprediksi kategori warna buah kakao, seperti apakah buah tersebut matang, mentah, atau rusak.

Selain menghasilkan kategori prediksi dan skor keyakinan (confidence score), API ini juga memberikan visualisasi hasil segmentasi warna dalam bentuk gambar yang dikembalikan dalam format base64, agar mudah ditampilkan di sisi frontend seperti aplikasi web atau mobile. Untuk menjaga kepraktisan, model K-Means disimpan dalam bentuk file .joblib dan dimuat saat API dijalankan.

API ini mendukung metode POST untuk menerima gambar dan mengembalikan hasil analisisnya, serta metode GET untuk keperluan pengecekan koneksi. Secara keseluruhan, API ini menyederhanakan proses otomatisasi pengenalan warna kakao secara visual, yang penting dalam proses klasifikasi buah untuk kebutuhan pertanian atau industri pengolahan kakao. Adapun codingan dari API ini dalam file main.py yang mana codingannya adalah sebagai berikut:

```
import os; os.environ['TF_CPP_MIN_LOG_LEVEL'] = '3'
import io
import base64
from http import HTTPStatus
from dotenv import load_dotenv
from PIL import Image
from werkzeug.utils import secure_filename
from flask import Flask, jsonify, request
from flask_cors import CORS
from cacao import CacaoColorSegmentation

load_dotenv()

app = Flask(__name__)
CORS(app)
cacao = CacaoColorSegmentation("dummy_path", n_clusters=5)

app.config['ALLOWED_EXTENSIONS'] = set(['png', 'jpg', 'jpeg'])
app.config['UPLOAD_FOLDER'] = 'storages/'
app.config['MODEL_KMEANS'] = './models/kmeans.joblib'

model_segmenter = cacao.load_model(app.config['MODEL_KMEANS'])

def allowed_file(filename):
    return '.' in filename and \
           filename.rsplit('.', 1)[1] in app.config['ALLOWED_EXTENSIONS']

@app.route('/', methods=['GET'])
def index():
    return jsonify({
        'code': HTTPStatus.OK,
        'message': 'Success',
        'data': None,
    }), HTTPStatus.OK

@app.route('/predict', methods=['POST'])
def predictSegmenter():
    if model_segmenter:
        request_image = request.files.get('image')
        if request_image and allowed_file(request_image.filename):
            filename = secure_filename(request_image.filename)
            request_image.save(os.path.join(app.config['UPLOAD_FOLDER'],
filename))
            image_path = os.path.join(app.config['UPLOAD_FOLDER'], filename)
            if os.path.exists(image_path):
                prediction,          cluster_images,          save_path          =
cacao.predict_and_visualize(image_path, model_segmenter)
                if prediction:
```

```
                base64_image = None
                with open(save_path, 'rb') as img_file:
                    base64_image                                                    =
f'{base64.b64encode(img_file.read()).decode('utf-8')}'
                return jsonify({
                    'status': {
                        'code': HTTPStatus.OK,
                        'message': 'OK',
                        'predicted': prediction['predicted_category'],
                        'confidence': prediction['confidence_score'],
                        'image_data': base64_image
                    },

                }), HTTPStatus.OK
            else:
                return jsonify({
                    'status': {
                        'code': HTTPStatus.INTERNAL_SERVER_ERROR,
                        'message': 'Prediction failed'
                    }

        else: }), HTTPStatus.INTERNAL_SERVER_ERROR
            return jsonify({
                'status': {
                    'code': HTTPStatus.BAD_REQUEST,
                    'message': 'Image not found'
                }

      else: }), HTTPStatus.BAD_REQUEST
            return jsonify({
                'status': {
                    'code': HTTPStatus.BAD_REQUEST,
                    'message': 'Invalid or missing image file'
                }

   else: }), HTTPStatus.BAD_REQUEST

            return jsonify({
                'status': {
                    'code': HTTPStatus.INTERNAL_SERVER_ERROR,
                    'message': 'Model not loaded'
                }

            }), HTTPStatus.INTERNAL_SERVER_ERROR

if __name__== '__main__':
    app.run(debug=False, host='0.0.0.0', port=int(os.environ.get('PORT', 9000)))
```

Penjelasan analisis API K-Means Segmentasi Warna Kakao terdiri dari beberapa

poin sebagai berikut:

1. **Inisialisasi dan Setup Flask**

   API ini diawali dengan inisialisasi Flask sebagai framework utama untuk membangun server backend. Selain itu, library seperti dotenv digunakan untuk memuat variabel lingkungan seperti PORT, dan CORS diaktifkan untuk memperbolehkan permintaan lintas domain, yang sangat berguna ketika API ini diakses dari frontend berbasis web. Objek CacaoColorSegmentation juga diinstansiasi di awal agar dapat digunakan dalam proses prediksi dan segmentasi gambar.

2. **Konfigurasi Aplikasi**

   Bagian ini mendefinisikan folder penyimpanan gambar yang diunggah (storages/), ekstensi file gambar yang diperbolehkan (.jpg, .png, .jpeg), dan lokasi file model K-Means yang telah disimpan sebelumnya (kmeans.joblib). Model ini kemudian dimuat saat awal aplikasi dijalankan, dan disimpan ke dalam variabel model_segmenter untuk digunakan dalam proses prediksi.

3. **Endpoint GET sebagai Health Check**

   Fungsi index() dengan metode GET digunakan hanya sebagai endpoint sederhana untuk memverifikasi bahwa API berjalan dengan baik. Saat diakses, endpoint ini akan mengembalikan respons JSON yang menunjukkan status sukses tanpa memproses data apa pun.

4. **Endpoint POST /predict untuk Proses Prediksi**

   Endpoint ini adalah inti dari API. Ia menerima input berupa gambar yang dikirim dalam bentuk *form-data*, memvalidasi ekstensi file, menyimpannya

secara lokal, lalu memanggil fungsi predict_and_visualize() untuk memproses gambar. Hasil dari proses ini berupa prediksi kategori buah kakao, skor kepercayaan, serta gambar visualisasi klaster warna yang disimpan dalam file.

5. **Proses Validasi dan Penyimpanan Gambar**

   Setelah file gambar diterima, sistem akan melakukan validasi apakah ekstensi file diperbolehkan. Jika ya, file akan disimpan ke folder storages/ dengan nama yang diamankan menggunakan secure_filename() untuk mencegah serangan file injection. Setelah disimpan, sistem akan memastikan file tersebut benar-benar ada sebelum melanjutkan proses prediksi.

6. **Pemanggilan Fungsi Segmentasi dan Prediksi**

   Fungsi predict_and_visualize() dipanggil dengan path gambar dan model K-Means sebagai parameter. Fungsi ini bertugas untuk memproses gambar menggunakan segmentasi warna berbasis K-Means, mengidentifikasi klaster dominan, dan memetakan hasilnya ke kategori warna buah kakao. Hasil prediksi dikembalikan dalam bentuk dictionary yang berisi nama kategori dan skor keyakinan.

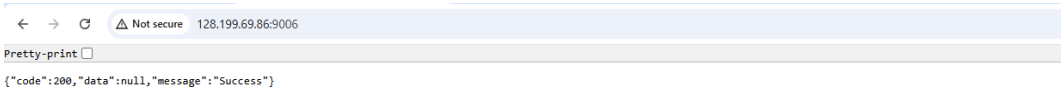
7. **Konversi Gambar Visualisasi ke Format Base64**

   Hasil visualisasi segmentasi warna disimpan dalam file gambar (biasanya PNG). Untuk bisa dikirim melalui API dan ditampilkan di web, gambar ini dibuka dan dikonversi ke format base64. Hasil encoding base64 kemudian disisipkan ke dalam respons JSON sehingga frontend bisa langsung menampilkannya sebagai elemen gambar.

8. **Pengembalian Hasil ke Client**

Setelah seluruh proses selesai, API akan mengembalikan hasil prediksi dalam bentuk JSON. JSON ini mencakup kode status, pesan, nama kategori hasil prediksi, skor kepercayaan, dan gambar hasil segmentasi warna dalam format base64. Jika terjadi kesalahan, seperti gambar tidak ditemukan atau model tidak dimuat, sistem akan mengembalikan status kesalahan (HTTP 400 atau 500) beserta pesan yang sesuai.

9. **Pengujian API**



Gambar 4.1 Server API

Dari Gambar 4.1 dapat disimpulkan bahwa

a. code: 200

   Menunjukkan bahwa permintaan berhasil (HTTP status code 200 = OK). Server berjalan dan menerima permintaan GET.

b. data: null

   Ini menunjukkan bahwa tidak ada data yang dikirim kembali pada permintaan ini, karena memang endpoint / hanya digunakan sebagai health check atau ping test, bukan untuk melakukan prediksi gambar.

c. message: "Success"

   Memberi informasi bahwa permintaan berhasil dan server merespons dengan baik.

Endpoint / biasanya digunakan sebagai health check API. Artinya dapat memastikan bahwa server aktif dan bisa merespons permintaan, tidak melakukan pemrosesan model atau pengolahan gambar, digunakan untuk debugging awal atau memastikan konektivitas sebelum melakukan request POST yang kompleks seperti /predict.

Secara keseluruhan dapat disimpulkan bahwa API ini merupakan sistem cerdas yang menggabungkan teknik pengolahan citra dan pembelajaran mesin sederhana untuk mengklasifikasikan warna buah kakao berdasarkan gambar. Ia dapat dengan mudah diintegrasikan ke dalam berbagai platform untuk membantu pengguna baik petani, peneliti, atau industri dalam mengidentifikasi kondisi buah secara visual. Struktur kodenya modular dan telah menerapkan praktik standar API, seperti validasi input, pemisahan model, dan pengiriman hasil dalam format yang user friendly.

4.3 Hasil dalam Web

Aplikasi web prediksi clustering kakao merupakan hasil implementasi dari penelitian mengenai segmentasi tingkat kematangan buah kakao menggunakan fitur warna CIELAB dan algoritma K-Means Clustering. Aplikasi ini dirancang untuk memudahkan pengguna dalam mengunggah citra buah kakao dan secara otomatis mengelompokkan tingkat kematangannya berdasarkan analisis warna. Pengguna hanya perlu mengunggah gambar dalam format JPG atau PNG, kemudian sistem akan mengirimkan citra tersebut ke backend melalui endpoint prediksi. Backend akan memproses gambar menggunakan algoritma clustering dan mengembalikan hasil berupa cluster kematangan, confidence score, dan visualisasi hasil ekstraksi

warna. Antarmuka aplikasi dibangun menggunakan React dengan komponen utama pada berkas welcome.tsx, yang menangani validasi input, pengiriman data, dan penampilan hasil prediksi. Sistem ini mengedepankan kesederhanaan tampilan namun efektif dalam menyampaikan informasi, serta mengintegrasikan teknologi pengolahan citra digital untuk membantu petani dan produsen kakao dalam meningkatkan kualitas panen secara objektif dan efisien. Penggunaan ruang warna CIELAB dinilai lebih stabil terhadap perubahan pencahayaan dan lebih mendekati persepsi manusia, sehingga cocok digunakan untuk klasifikasi warna buah dalam berbagai tingkat kematangan. Adapun codingan dalam file welcome.tsx adalah sebagai berikut:

```
import React, { useState } from "react";
import { Button } from "~/components/ui/button";
import { Input } from "~/components/ui/input";
import { cn } from "~/lib/utils";

const BACKEND_URL = "http://128.199.69.86:9006";

export function Welcome() {
	const [isSubmitting, setIsSubmitting] = useState<boolean>(false);
	const [image, setImage] = useState<string | null>(null);
	const [selectedFile, setSelectedFile] = useState<File | null>(null);
	const [predicted, setPredicted] = useState<{
		predicted: string | null;
		confidence: string | null;
		image_data: string | null;
	}>({ predicted: null, confidence: null, image_data: null });

	async function handlerPredict(event:
React.MouseEvent<HTMLButtonElement>) {
		event.preventDefault();
		setIsSubmitting(true);

		if (!selectedFile) {
			alert("Pilih gambar terlebih dahulu!");
			return;
		}


		const formData = new FormData();
```

```
                formData.append("image", selectedFile);

                // ✅ Debug isi FormData
                for (const [key, value] of formData.entries()) {
                        console.log(`${key}:`, value);
                }

                try {
                        const response = await
fetch(`${BACKEND_URL}/predict`, {
                                method: "POST",
                                 body: formData,
                        });

                        if (!response.ok) {
                                throw new Error("Gagal melakukan prediksi");
                        }

                        const data = await response.json();
                        console.log("Hasil Prediksi:", data);

                        setPredicted({
                                predicted: data.status.predicted,
                                confidence: data.status.confidence,
                                image_data: data.status.image_data,
                        });

                } catch (error) {
                        console.error("Error:", error);
                        alert("Terjadi kesalahan saat prediksi");
                        setPredicted({
                                predicted: null,
                                confidence: null,
                                image_data: null,
                         });

                } finally {
                        setIsSubmitting(false);
                }
        }

        return (

                <main className="bg-slate-950 w-full min-h-dvh">
                        <div className="bg-slate-900">
                                <div className="max-w-4xl flex p-2 mx-auto">
                                        <h1 className="font-bold text-
white">CLUSTERING CACAO</h1>
                                </div>
                        </div>
```

```
                    <div className={cn("mx-auto p-2 max-w-4xl", image ?
"mt-4" : "mt-64")}>
                        {image && (
                        <img
                                                src={image}
                                                alt="Preview"
                                                className="w-1/2 mx-auto"/>
                                    )}

    <div className="flex gap-2 mt-4">
<Input
                                                id="picture"
                                                type="file"
                                                accept="image/jpeg, image/jpg,
image/png"
                                                className="bg-white"
                                                disabled={isSubmitting}
                                                onChange={(event) => {
                                                        const file =
event.target.files?.[0];

                                                        if (!file) return;
                                                        const allowedTypes =
["image/jpeg", "image/jpg", "image/png"];       if
(!allowedTypes.includes(file.type))     {               alert("Hanya file JPG,
JPEG, dan PNG yang diperbolehkan.");                    event.target.value =
"";                                                     setImage(null);
                                                        setSelectedFile(null);
                                                        return;

            }


        setImage(URL.createObjectURL(file));
                                                }}
                                                        setSelectedFile(file);
                                    />

                                    <Button
                                                className="bg-green-700 cursor-
pointer hover:bg-green-600"             onClick={(event)                =>
handlerPredict(event)}                  disabled={isSubmitting}
                                    >

                                                Prediksi
                                    </Button>

                                </div>
```

```
                    {predicted.predicted && predicted.confidence &&
predicted.image_data && (
                        <div className="mt-12 text-white">
                            <h1 className="text-3xl font-bold
text-center mb-4">HASIL PREDIKSI</h1>
                            <div className="text-lg">
                                <div>
                                    <p className="font-
semibold">Extract Warna CIELAB</p>
                                    <img

        src={`${predicted.image_data}`}

Image"                                          alt="Predicted

                                    />

                                </div>
                                <div className="flex gap-2
mt-4">
                                    <p className="font-
semibold">Cluster:</p>
                                    <p className="font-
semibold">{predicted.predicted}</p>
                                </div>
                                <div className="flex gap-
2">
                                    <p className="font-
semibold">Confidence Score:</p>
                                    <p className="font-
semibold">{(Number(predicted.confidence) * 100).toFixed(2)} %</p>
                                </div>
                            </div>

                    )}      </div>

                </div>
            </main>
        );

}
```

Adapun hasil dari implementasi webnya adalah sebagai berikut:

Gambar 4.2 Tampilan Web Clustering Cacao

Komponen ini terdiri dari beberapa bagian penting yang saling terintegrasi untuk membentuk alur kerja yang efisien, mulai dari pengunggahan gambar, pengiriman ke backend, hingga penampilan hasil prediksi.

Pertama, komponen ini memanfaatkan beberapa hook dari React, seperti useState, untuk mengelola status aplikasi, seperti kondisi pengiriman (isSubmitting), file gambar yang dipilih (selectedFile), tampilan gambar (image), dan hasil prediksi (predicted). Nilai predicted sendiri menyimpan informasi penting yang diterima dari backend, yaitu label cluster kematangan, confidence score, dan gambar hasil ekstraksi warna.

Saat pengguna memilih gambar, komponen akan memvalidasi tipe file untuk memastikan hanya format JPEG, JPG, atau PNG yang diterima. Jika valid, gambar akan ditampilkan secara langsung di halaman menggunakan URL.createObjectURL. Tombol "Prediksi" akan mengirimkan file gambar ke backend melalui permintaan fetch dengan metode POST ke endpoint /predict. Backend kemudian akan memproses citra menggunakan algoritma K-Means dan fitur warna CIELAB.

Setelah respons diterima, data hasil prediksi akan ditampilkan di bawah gambar yang diunggah. Tampilan hasil mencakup gambar hasil ekstraksi warna dan informasi klasifikasi cluster serta confidence score yang dihitung dari model machine learning. Score ini diubah ke dalam bentuk persentase dan ditampilkan secara informatif kepada pengguna.

Desain antarmuka dibuat sederhana dan responsif, dengan warna latar gelap (bg-slate-950) untuk memberikan kontras yang baik terhadap elemen-elemen putih seperti teks dan input gambar. Selain itu, komponen menggunakan Tailwind CSS untuk pengaturan gaya visual, dan mengandalkan komponen UI seperti Button dan Input yang diimpor dari pustaka internal proyek.

Secara keseluruhan, implementasi web ini menyajikan antarmuka yang sederhana dan mudah digunakan, berfokus pada satu fungsi utama yaitu mengunggah gambar dan mendapatkan hasil prediksi kematangan secara otomatis. Setelah gambar diproses, hasil prediksi akan muncul di bawah komponen input, menampilkan informasi seperti label cluster, tingkat kepercayaan (confidence score), dan visualisasi hasil ekstraksi warna, yang semuanya memberikan dukungan analitis dalam proses klasifikasi buah kakao.

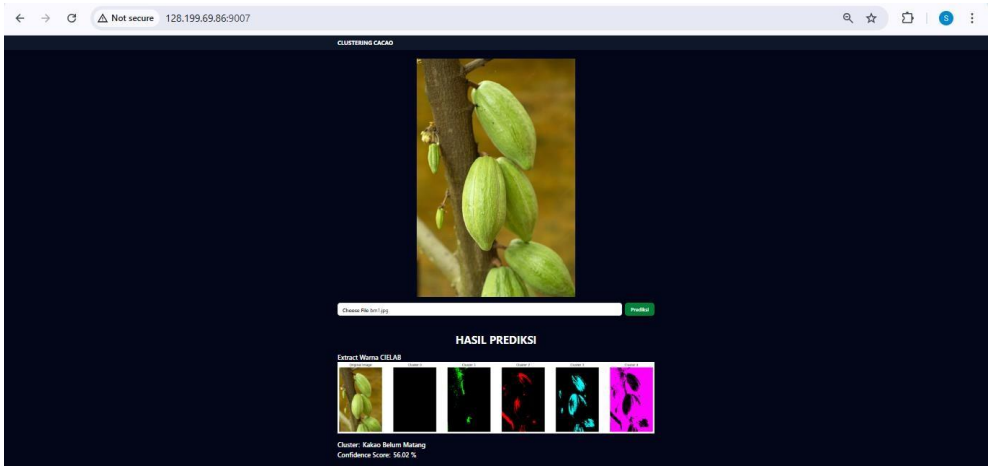
4.4 Pengujian Model Pada Web

Pengujian ini bertujuan untuk mengevaluasi kemampuan aplikasi dalam mengidentifikasi dan mengelompokkan gambar buah kakao ke dalam kategori matang, belum matang, dan bukan kakao. Gambar yang digunakan dalam pengujian ini mewakili kondisi visual yang beragam, baik dari segi warna, pencahayaan, maupun objek. Melalui pengujian ini, diharapkan sistem dapat menunjukkan

akurasi serta ketepatannya dalam mengklasifikasikan gambar berdasarkan fitur warna yang dianalisis oleh algoritma K-Means Clustering dalam ruang warna CIELAB. Hasil dari pengujian ini akan menjadi dasar untuk menilai sejauh mana aplikasi dapat diandalkan dalam proses segmentasi dan identifikasi kematangan buah kakao secara otomatis.

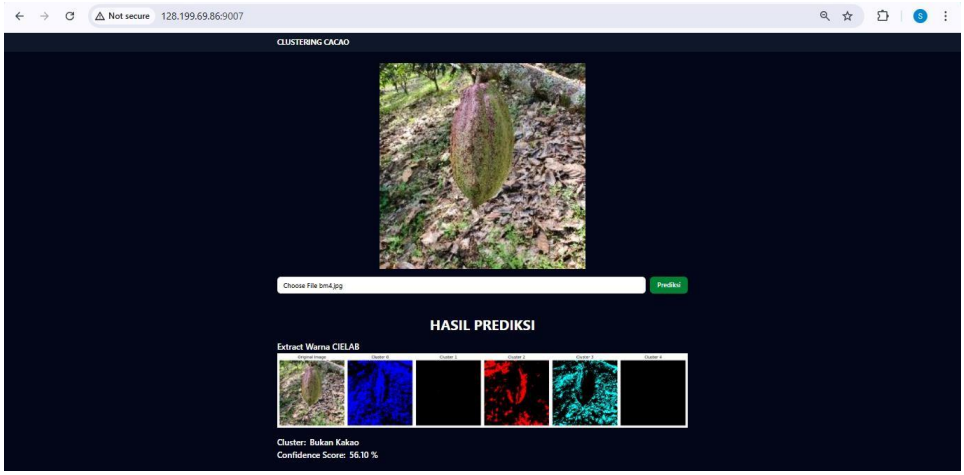
Sebagai langkah lanjutan, dilakukan pengujian sistem terhadap enam gambar uji yang belum pernah dikenali sebelumnya. Gambar-gambar tersebut terdiri dari beberapa buah kakao dalam berbagai tingkat kematangan, serta beberapa gambar yang bukan merupakan buah kakao. Tujuan dari pengujian ini adalah untuk mengevaluasi kemampuan sistem dalam mengklasifikasikan citra berdasarkan fitur warna yang telah dipelajari selama pelatihan, serta untuk mengetahui apakah sistem mampu mengenali gambar yang tidak sesuai dengan kategori kakao (bukan kakao). Hasil dari pengujian ini akan dianalisis berdasarkan output klasifikasi dan nilai kepercayaan (confidence score) yang dihasilkan oleh sistem. Adapun hasil dari pengujian model pada web adalah sebagai berikut

1. Belum Matang

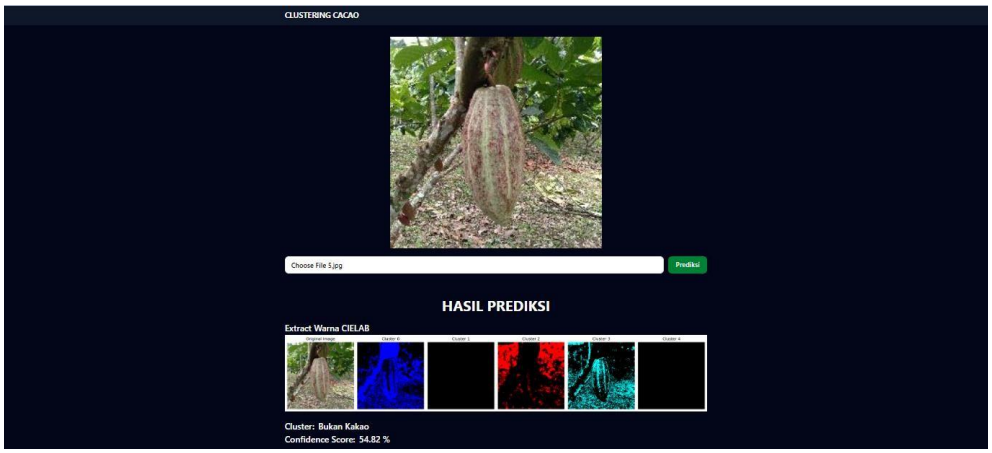


Gambar 4.3 Pengujian Pertama Kakao Belum Matang

Model mampu mengklasifikasikan buah kakao yang belum matang dengan benar pada gambar pertama. Ini menunjukkan bahwa untuk kondisi warna hijau segar dan bentuk khas buah muda, model dapat mengenali kategori "Kakao Belum Matang" dengan tingkat kepercayaan yang cukup tinggi (sekitar 56%).



Gambar 4.4 Pengujian Kedua Kakao Belum Matang



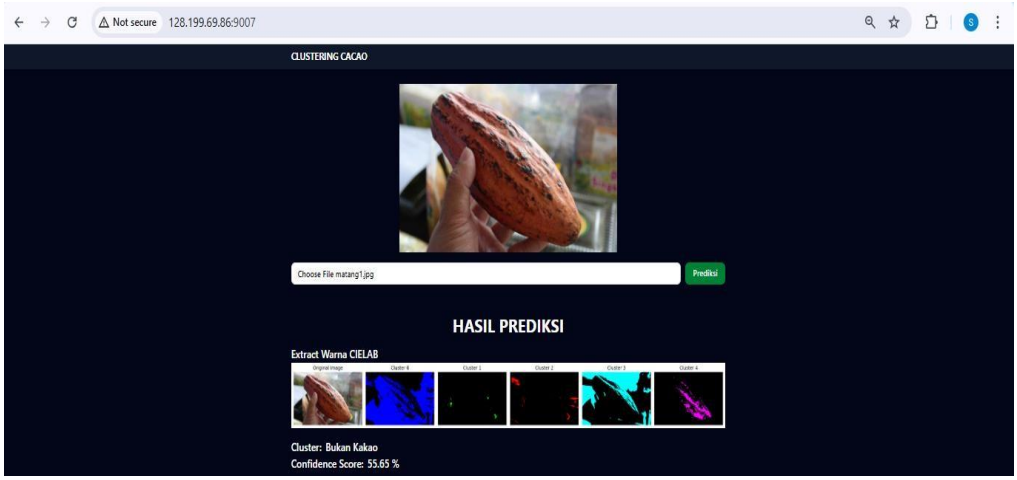
Gambar 4.5 Pengujian Ketiga Kakao Belum Matang

Gambar kedua dan ketiga yang juga menampilkan buah kakao namun dalam kondisi atau sudut yang tidak umum, model justru salah mengklasifikasikannya sebagai "Bukan Kakao". Hal ini mengindikasikan

bahwa model memiliki keterbatasan dalam mengenali variasi visual buah kakao yang sedikit berbeda dari data pelatihan awal.

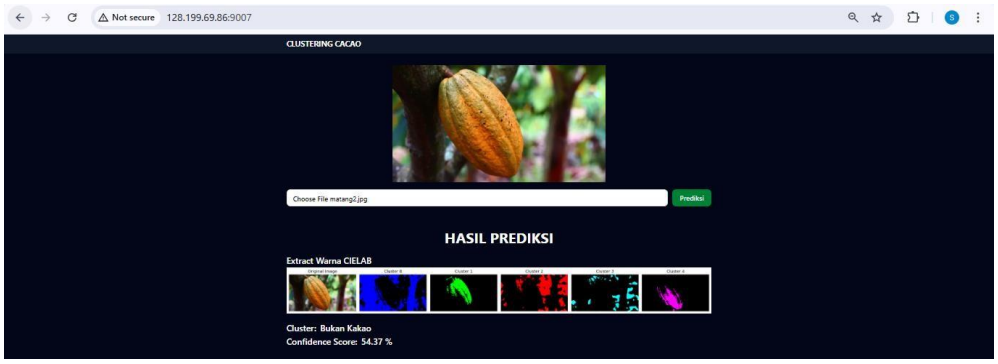
Dari ketiga pengujian dapat disimpulkan bahwa model gagal dalam 2 dari 3 pengujian (akurasi sekitar 33%), yang menunjukkan bahwa performa model saat ini masih rendah dan belum dapat diandalkan untuk klasifikasi yang akurat terhadap buah kakao.

2. Matang

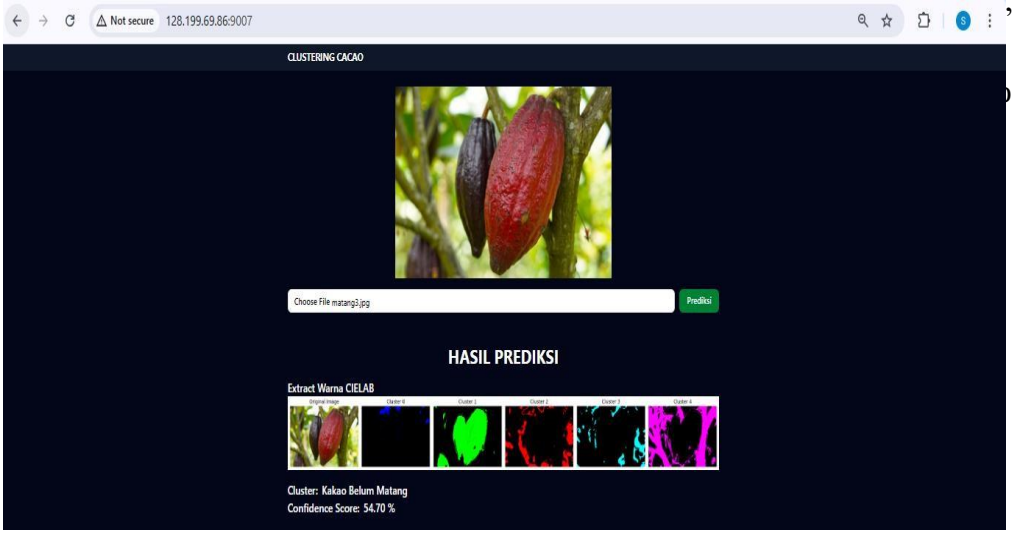


Gambar 4.6 Pengujian Pertama Kakao Matang

Pada gambar keempat, buah kakao yang tampak matang malah diklasifikasikan sebagai "Kakao Belum Matang". Ini menandakan adanya ambiguitas pada model dalam membedakan tingkat kematangan, kemungkinan karena kemiripan warna antara buah matang dan belum matang yang tidak tersegmentasi dengan jelas pada fitur warnanya.



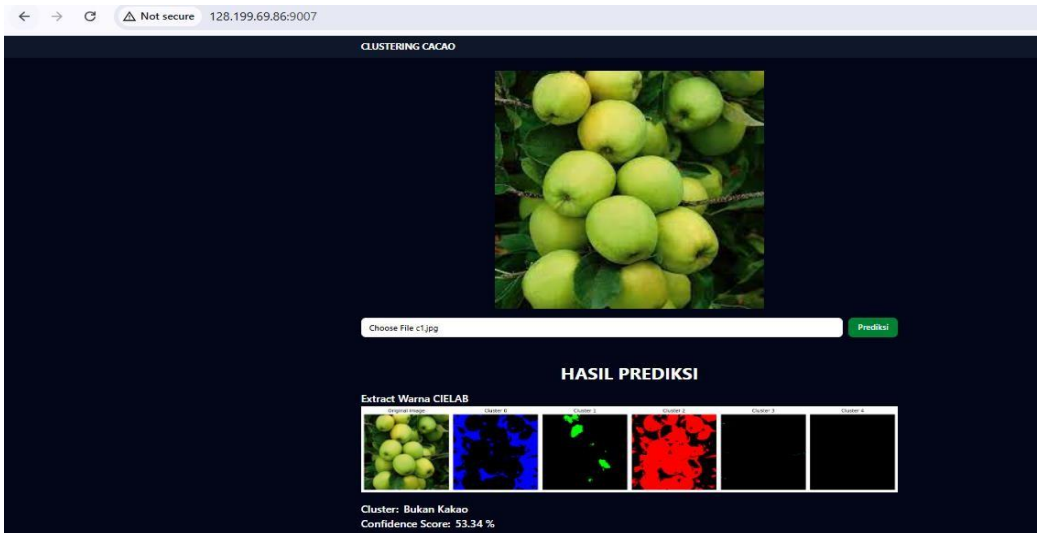
Gambar 4.7 Pengujian Kedua Kakao Matang



Gambar 4.8 Pengujian Ketiga Kakao Matang

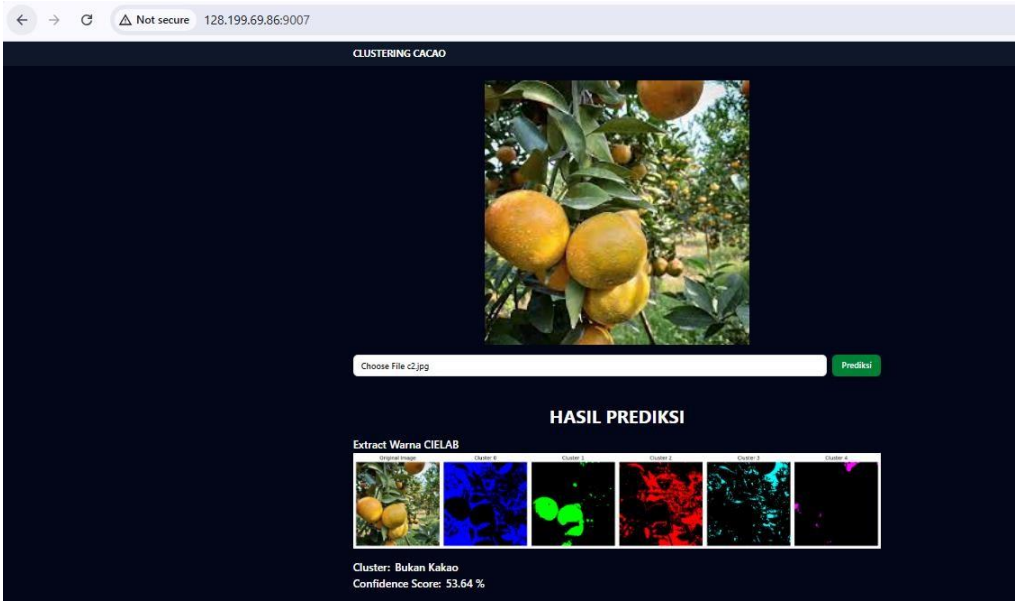
Buah adalah kakao, dengan bentuk khas dan permukaan bergaris. Model mengeluarkan hasil "Kakao Belum Matang" dengan confidence 54.70%, padahal objek adalah kakao matang.

3. Bukan Kakao



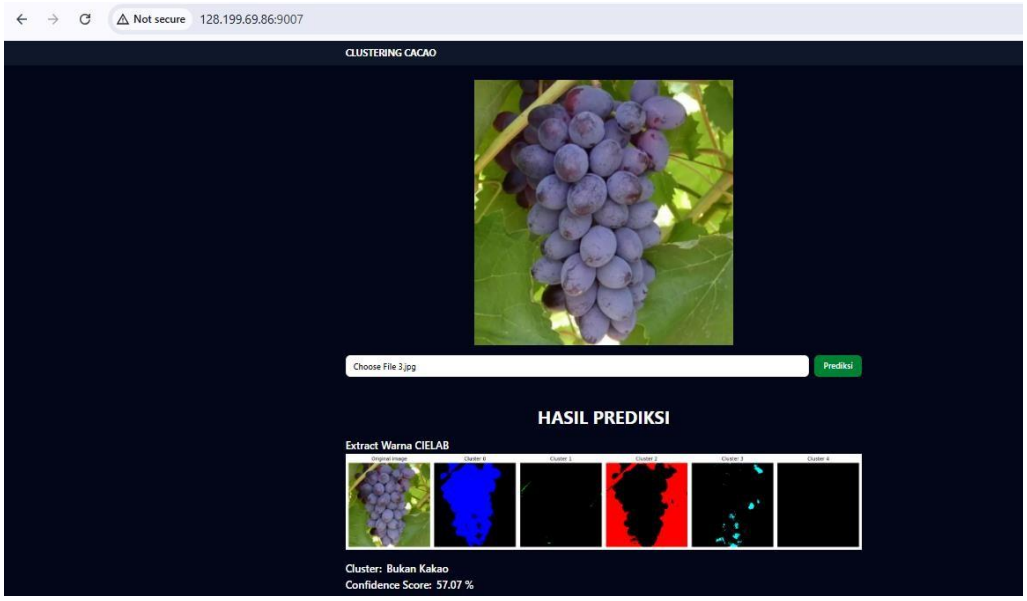
Gambar 4.9 Pengujian Pertama Bukan Kakao

Model berhasil mengklasifikasikan dengan tepat bahwa objek bukan kakao. Warna dominan hijau serta bentuk bulat tidak sesuai dengan karakteristik buah kakao. Confidence score sebesar 53.34% mendukung klasifikasi ini meskipun tidak terlalu tinggi**.**



Gambar 4.10 Pengujian Kedua Bukan Kakao

Objek juga berhasil diklasifikasikan sebagai bukan kakao, dengan confidence score 53.64%. Warna jingga dan struktur permukaan yang lebih halus dibanding kakao kemungkinan menjadi indikator pembeda yang dikenalimodel.



Gambar 4.11 Pengujian Ketiga Bukan Kakao

Dapat dikenali dengan baik sebagai **bukan kakao**. Meskipun bentuk bulat kecil dan warna gelap bisa membingungkan model, namun prediksi tepat dengan confidence score cukup tinggi **57.07%**, menandakan keandalan dalam membedakan struktur buah berbentuk bergerombol.

Tabel 4.1 Ringkasan Prediksi Gambar

No	Hasil Prediksi	Confidence Score	Ground Truth	Keterangan Prediksi
1	Kakao Belum Matang	56.02%	Kakao Belum Matang	Benar
2	Bukan Kakao	56.10%	Kakao Belum Matang	Salah
3	Bukan Kakao	54.37%	Kakao Belum Matang	Salah
4	Bukan Kakao	55.65%	Kakao Matang	Salah
5	Bukan Kakao	54.47%	Kakao Matang	Salah
6	Kakao Belum Matang	54.37%	Kakao Matang	Salah
7	Bukan Kakao	53.34%	Bukan Kakao	Benar
8	Bukan Kakao	53.64%	Bukan Kakao	Benar
9	Bukan Kakao	57.07%	Bukan Kakao	Benar

Dari Tabel 4.1 dapat disimpulkan bahwa:

Total data uji: 9 gambar

Prediksi benar:

1 gambar kakao belum matang

3 gambar bukan kakao

Prediksi salah:

2 gambar kakao belum matang

3 gambar kakao matang

Total benar = 4 dari 9 gambar

Akurasi model = 44,44%

Model klasifikasi kakao ini menunjukkan kinerja yang buruk dalam mengenali kakao matang, dengan 0 dari 3 gambar kakao matang dikenali dengan benar. Sebaliknya, model bekerja cukup baik untuk mengenali buah-buah yang bukan kakao, menunjukkan bahwa pemisahan terhadap kategori "bukan kakao" lebih akurat.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Penelitian ini bertujuan untuk menguji efektivitas fitur warna CIELAB dalam klasifikasi kematangan buah kakao dengan algoritma K-Means Clustering. Berdasarkan hasil pengujian terhadap sembilan gambar uji, model hanya berhasil mengklasifikasikan empat gambar dengan benar, menghasilkan tingkat akurasi sebesar 44,44%. Hasil yang benar terdiri atas satu gambar kakao belum matang dan tiga gambar buah bukan kakao. Namun, model gagal mengidentifikasi semua gambar kakao matang dengan tepat. Bahkan, satu gambar kakao matang diprediksi sebagai “Kakao Belum Matang”, sementara dua lainnya justru diklasifikasikan sebagai “Bukan Kakao”.

Selain itu, dari tiga gambar kakao belum matang, hanya satu yang dikenali dengan benar, sedangkan dua lainnya juga diprediksi sebagai “Bukan Kakao”. Ini menunjukkan bahwa model memiliki kelemahan mendasar dalam membedakan antara kakao dan non-kakao, serta dalam menentukan tingkat kematangan buah kakao itu sendiri. Model justru menunjukkan performa paling stabil saat mengenali buah bukan kakao, di mana seluruh gambar kategori ini diprediksi dengan benar.

Secara keseluruhan, hasil ini menunjukkan bahwa penggunaan fitur warna CIELAB yang diproses dengan K-Means Clustering tidak cukup andal untuk klasifikasi tingkat kematangan buah kakao secara akurat. Ketergantungan pada fitur warna semata tidak mampu mengakomodasi variasi visual buah kakao matang yang cenderung kompleks, baik dari segi warna, pencahayaan, maupun tekstur.

Selain itu, pendekatan unsupervised yang digunakan tidak memberikan jaminan bahwa hasil klaster sesuai dengan label ground truth, terutama dalam konteks pengelompokan tingkat kematangan buah. Karena itu, diperlukan pendekatan yang lebih kompleks dan menyeluruh, baik dari segi fitur maupun metode klasifikasi.

5.2 **Saran**

Berdasarkan hasil penelitian dan evaluasi terhadap kinerja model K-Means Clustering dalam mengklasifikasikan kematangan buah kakao menggunakan fitur warna CIELAB, terdapat beberapa saran yang dapat diberikan untuk pengembangan lebih lanjut:

1. Menambah jumlah dan variasi data citra buah kakao, terutama untuk kategori kakao matang dan belum matang, agar model memiliki representasi yang lebih beragam dalam proses pelatihan.
2. Mengombinasikan fitur warna CIELAB dengan fitur lain seperti tekstur, kontur, dan bentuk untuk meningkatkan akurasi dalam membedakan tingkat kematangan buah.
3. Menggunakan algoritma klasifikasi berbasis supervised learning seperti SVM, Random Forest, atau Convolutional Neural Network (CNN) untuk menggantikan atau melengkapi pendekatan K-Means yang bersifat unsupervised.
4. Menerapkan teknik augmentasi data, seperti rotasi, perubahan pencahayaan, dan zoom, agar model lebih tahan terhadap variasi kondisi pencitraan di lapangan.

5. Mengoptimalkan nilai K dalam K-Means menggunakan metode seperti Elbow Method atau Silhouette Score untuk menentukan jumlah klaster yang paling representatif terhadap data.
6. Melakukan preprocessing lanjutan, seperti normalisasi warna, filtering noise, atau segmentasi awal objek, untuk membantu proses klasterisasi berjalan lebih stabil dan akurat.

DAFTAR PUSTAKA

Areni, I. S., Amirullah, I., & Arifin, N. (2019). Klasifikasi Kematangan Stroberi Berbasis Segmentasi Warna dengan Metode HSV. *Jurnal Penelitian Enjiniring*, *23*(2), 113–116. https://doi.org/10.25042/jpe.112019.03

Febby Wilyani, Qonaah Nuryan Arif, & Fitri Aslimar. (2024). Pengenalan Dasar Pemrograman Python Dengan Google Colaboratory. *Jurnal Pelayanan Dan Pengabdian Masyarakat Indonesia*, *3*(1), 08–14. https://doi.org/10.55606/jppmi.v3i1.1087

Jatmika, S., Aprilianto, T., & Idris, M. (2020). Ekstraksi Fitur Untuk Mengidentifikasi Marga Tanaman Menggunakan Algoritma Backpropagation. *POSITIF : Jurnal Sistem Dan Teknologi Informasi*, *6*(1), 56. https://doi.org/10.31961/positif.v6i1.907

Lesmana, A. A., Purwanto, Y., & Dinimaharawati, A. (2019). Implementasi Algoritma K-Means Untuk Clustering Penyakit HIV/AIDS Di Indonesia Implementation of K-Means Algorithm for Clustering of Hiv / Aids Disease in Indonesia. *E-Proceeding of Engineering*, *6*(2 Agustus 2019), 5564–5580.

Mahendra, I., Rachmat, N., kunci-Buah Kakao, K., & Distance, E. (2023). Klasifikasi Tingkat Kematangan Buah Kakao Berdasarkan Fitur Warna Menggunakan Algoritma K-Nearest Neighbor. *Jurnal Algoritme*, *4*(1), 31–42. https://doi.org/10.35957/algoritme.xxxx

Mayrowani, H. (2016). Pengembangan Pertanian Organik di Indonesia. *Forum Penelitian Agro Ekonomi*, *30*(2), 91. https://doi.org/10.21082/fae.v30n2.2012.91-108

Mubarok, H., Murni, S., & Santoni, M. M. (2021). Penerapan Algoritma K-Nearest Neighbor untuk Klasifikasi Tingkat Kematangan Buah Tomat Berdasarkan Fitur Warna. *Seminar Nasional Mahasiswa Ilmu Komputer Dan Aplikasinya (SENAMIKA) Jakarta-Indonesia*, *1*, 773–782. https://conference.upnvj.ac.id/index.php/senamika/article/view/1438

Nabila, Z., Rahman Isnain, A., & Abidin, Z. (2021). Analisis Data Mining Untuk Clustering Kasus Covid-19 Di Provinsi Lampung Dengan Algoritma K-Means. *Jurnal Teknologi Dan Sistem Informasi (JTSI)*, *2*(2), 100. http://jim.teknokrat.ac.id/index.php/JTSI

Noval Alan Pambudi, Yosep Agus Pranoto, A. P. S. (2021). Pengenalan tingkat kematangan buah kopi berdasarkan fitur warna cielab dengan k-means clustering. *JATI (Jurnal Mahasiswa Teknik Informatika)*, *5*(2), 728–732.

Ramadhani, R. A., Rifin, A., & Novianti, T. (2023). ANALISIS DAYA SAING DAN KEBIJAKAN BEA KELUAR PADA KOMODITAS KAKAO (Theobroma cacao) INDONESIA Analysis of Competitiveness and Export Duty Fees Regulation on Indonesian Cocoa (Theobroma cacao) Commodity. *Analisis Kebijakan Pertanian*, *21*(2), 171–186.

Ratna, S. (2020). Pengolahan Citra Digital Dan Histogram Dengan Phyton Dan Text Editor Phycharm. *Technologia: Jurnal Ilmiah*, *11*(3), 181. https://doi.org/10.31602/tji.v11i3.3294

Rulaningtyas, R., B. Suksmono, A., L. R. Mengko, T., & Putri Saptawati, G. A. (2015). Segmentasi Citra Berwarna dengan Menggunakan Metode Clustering Berbasis Patch untuk Identifikasi Mycobacterium Tuberculosis. *Jurnal Biosains Pascasarjana*, *17*(1), 19. https://doi.org/10.20473/jbp.v17i1.2015.19-25

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

Hidayatullah, A., & Berliana, A. (2023). Workshop Pengenalan Dasar Pemrograman Python Dengan Google Colaboratory. *Prosiding ABDIMAS CORISINDO 2023*.Sibuea, M. L., & Safta, A. (2017). Pemetaan Siswa Berprestasi Menggunakan Metode K-Means Clustring. *Jurteksi*, *4*(1), 85–92. https://doi.org/10.33330/jurteksi.v4i1.28

Wijaya, E. S., & Prayudi, Y. (2015). Integrasi Metode Steganografi DCS Pada Image Dengan Kriptografi Blowfish Sebagai Model Anti Forensik Untuk Keamanan Ganda Konten Digital. *SNATI (Seminar Nasional Aplikasi Teknologi Informasi)*, 11–17.

Yessy Nabella, F., Arum Sari, Y., & Cahya Wihandika, R. (2019). Seleksi Fitur Information Gain Pada Klasifikasi Citra Makanan Menggunakan Hue Saturation Value dan Gray Level Co-Occurrence Matrix. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *3*(2), 1892–1900. http://j-ptiik.ub.ac.id

Yogiswara, G. H., Magdalena, R., P, H. F. T. S., Elektro, F. T., & Telkom, U. (2016). Identifikasi Jenis Penyakit Pada Kakao Dengan Pengolahan Citra Digital Dan K-Nearest Neighbor Cacao Disease Identification Using Digital Image Processing and. *E-Proceding of Engineering*, *3*(1), 371–377.

Yunita, F. (2018). Penerapan Data Mining Menggunkan Algoritma K-Means Clustring Pada Penerimaan Mahasiswa Baru. *Sistemasi*, *7*(3), 238. https://doi.org/10.32520/stmsi.v7i3.388

BAB_15_anwar_rangkuti_beserta_cover1-1756916336629

ORIGINALITY REPORT

21%	18%	7%	14%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	ejournal.itn.ac.id Internet Source	4%
2	skripsi.tunasbangsa.ac.id Internet Source	1%
3	eprints.uty.ac.id Internet Source	1%
4	Submitted to Universitas Muhammadiyah Sumatera Utara Student Paper	1%
5	123 Internet Source dok.com	1%
6	www.researchgate.net Internet Source	1%
7	Submitted to Colorado School of Mines Student Paper	1%
8	csrid.potensi-utama.ac.id Internet Source	<1%
9	cot.unhas.ac.id Internet Source	<1%

10	Submitted to Dublin City University Student Paper	<1%
11	repository.its.ac.id Internet Source	<1%
12	Submitted to Technological Institute of the Philippines Student Paper	<1%
13	conference.upnvj.ac.id Internet Source	<1%
14	docplayer.info Internet Source	<1%
15	neper.sourceforge.net Internet Source	<1%
16	Submitted to University of Hertfordshire Student Paper	<1%
17	text-id.123dok.com Internet Source	<1%
18	Submitted to The Robert Gordon University Student Paper	<1%
19	Submitted to University of Sydney Student Paper	<1%
20	Akhmad Fadjeri, Bayu Aji Saputra, Dicki Kusuma Adri Ariyanto, Lisna Kurniatin. "Karakteristik Morfologi Tanaman Selada Menggunakan Pengolahan Citra Digital", Jurnal Ilmiah SINUS, 2022 Publication	<1%

21	Submitted to University of Witwatersrand Student Paper	<1%
22	Submitted to University of Technology, Sydney Student Paper	<1%
23	eprints3.upgris.ac.id Internet Source	<1%
24	**programtalk.com** Internet Source	<1%
25	Rosa Rosiana, Willy Prihartono, Fathurrohman Fathurrohman. "Implementation of K-Means Algorithm for Sub-district Grouping Based on Rice Plant Productivity in Cirebon Regency", Jurnal Informatika Terpadu, 2025 Publication	<1%
26	opengeoai.org Internet Source	<1%
27	repository.ub.ac.id Internet Source	<1%
28	Submitted to University of Ulster Student Paper	<1%
29	etheses.uin-malang.ac.id Internet Source	<1%
30	gitext.gfz-potsdam.de Internet Source	<1%

31	Submitted to King's College Student Paper	<1%
32	Submitted to Trident University International Student Paper	<1%
33	eprints.iain-surakarta.ac.id Internet Source	<1%
34	Submitted to Houston Community College Student Paper	<1%
35	Submitted to University of Northampton Student Paper	<1%
36	Submitted to Universitas Muhammadiyah Sukabumi Student Paper	<1%
37	Submitted to National Yang Ming Chiao Tung University Student Paper	<1%
38	Submitted to UM Surabaya Student Paper	<1%
39	docs.cohere.com Internet Source	<1%

40	Abi Maulana, Aulia Ullah, Ahmad Faizal, Hilman Zarory. "Dual Sistem Keamanan Pada Pintu Dengan Pengenalan Wajah Local Binary Pattern Histogram (LBPH) Dan Sidik Jari serta Notifikasi Telegram", JURNAL Al-AZHAR INDONESIA SERI SAINS DAN TEKNOLOGI, 2025 Publication	<1%
41	Submitted to University of Huddersfield Student Paper	<1%
42	jurnal.fikom.umi.ac.id Internet Source	<1%
43	Submitted to Melbourne Institute of Technology Student Paper	<1%
44	Submitted to 2U Southern Methodist University Student Paper	<1%
45	Submitted to UCL Student Paper	<1%
46	digilib.unhas.ac.id Internet Source	<1%
47	www.geopostcodes.com Internet Source	<1%
48	Submitted to Queen Mary and Westfield College Student Paper	<1%
49	Submitted to Udayana University Student Paper	<1%

50	Submitted to University College London Student Paper	<1%
51	Zalzabila Rani, Bain Khusnul Khotimah. "ANALISIS SENTIMEN TERHADAP KARAPAN SAPI DI TWITTER MENGGUNAKAN METODE K-MEANS DAN SUPPORT VECTOR MACHINE (SVM)", Jurnal Informatika dan Teknik Elektro Terapan, 2025 Publication	<1%
52	Frencis Matheos Sarimole, Muhamad Aqil Septiansyah. "Analisis Pola Kinerja Anak dalam Tes Membaca untuk Mengidentifikasi Anak yang Membutuhkan Pendampingan Dini Menggunakan Algoritma K-Means Clustering di PAUD Seroja", Jurnal Indonesia : Manajemen Informatika dan Komunikasi, 2024 Publication	<1%
53	Submitted to Kaunas University of Technology Student Paper	<1%
54	Submitted to University of Leicester Student Paper	<1%
55	Submitted to Universitas Dinamika Student Paper	<1%
56	Submitted to University of Queensland Student Paper	<1%
57	repository.uncp.ac.id Internet Source	<1%

58	repository.upi.edu Internet Source	<1%
59	Submitted to Universitas Negeri Jakarta Student Paper	<1%
60	Submitted to University of San Diego Student Paper	<1%
61	Submitted to Aston University Student Paper	<1%
62	Submitted to Deakin University Student Paper	<1%
63	Submitted to EARLY MAKERS Group SA Tii Student Paper	<1%
64	Submitted to Fakultas Teknik Student Paper	<1%
65	Submitted to Konsorsium Perguruan Tinggi Swasta Indonesia II Student Paper	<1%
66	Submitted to STT PLN Student Paper	<1%
67	Submitted to University of Western Australia Student Paper	<1%
68	jurnal.unpad.ac.id Internet Source	<1%

69	Submitted to Monash University Student Paper	<1%
70	johannessimatupang.wordpress.com Internet Source	<1%
71	repository.umsu.ac.id Internet Source	<1%
72	Submitted to Universitas Brawijaya Student Paper	<1%
73	Submitted to University of Keele Student Paper	<1%
74	community.shopify.dev Internet Source	<1%
75	deepnote.com Internet Source	<1%
76	Submitted to City University Student Paper	<1%
77	Submitted to Trinity College Dublin Student Paper	<1%
78	Submitted to University Politehnica of Bucharest Student Paper	<1%

79	gitlab.icenter.tsinghua.edu.cn Internet Source	<1%
80	gitlab.sliit.lk Internet Source	<1%
81	widuri.raharja.info Internet Source	<1%
82	www.belbuk.com Internet Source	<1%
83	Ananda Aufa Alya Putri, Sabrina AuliaRahmah. "IMPLEMENTASI DATA MINING DENGAN ALGORITMA K-MEANS CLUSTERING UNTUK ANALISIS BISNIS PADA PERUSAHAAN ASURANSI", Djtechno: Jurnal Teknologi Informasi, 2024 Publication	<1%
84	Taufik Rahman, Abdul Ahmad Sahroni. "Penerapan K-Means untuk Pengelompokan Hasil Belajar Informatika", The Indonesian Journal of Computer Science, 2025 Publication	<1%
85	codeclimate.com Internet Source	<1%
86	ejournal.pabki.org Internet Source	<1%

87	farnoviarahma.wordpress.com Internet Source	<1%
88	id.123dok.com Internet Source	<1%
89	terrorizer.us Internet Source	<1%
90	Submitted to Ecole Mondiale World School Student Paper	<1%
91	Lohitha Nagalla, Siva Nandini Govapudi, Rekha Desireddy, B. Suvarna. "Chapter 24 An Effective Deep Learning Model forAir-Scripted Alphabet Recognition System", Springer Science and Business Media LLC, 2025 Publication	<1%
92	NAUFAL ARIF HIDAYATULLAH, WillyPrihartono, Fathur Rohman. "CLUSTERING PENERIMA BANTUAN PANGAN BERBASIS ALGORITMA K-MEANS UNTUK MENINGKATKAN EFEKTIVITAS PROGRAM SOSIAL DI KOTA/KABUPATEN CIREBON", Jurnal Informatika dan Teknik Elektro Terapan, 2025 Publication	<1%
93	Submitted to Taylor's Education Group Student Paper	<1%

94	Winar Wahyu Prastanika, Arie Wahyu Wijayanto. "Analisis Hard dan Soft Clustering Untuk Pengelompokan Indikator Ketahanan Pangan Indonesia 2021", Jurnal Sistem dan Teknologi Informasi (JustIN), 2023 Publication	<1%
95	adoc.pub Internet Source	<1%
96	andriaw.blogspot.com Internet Source	<1%
97	debuggercafe.com Internet Source	<1%
98	eprints.ums.ac.id Internet Source	<1%
99	forum.kde.org Internet Source	<1%
100	idoc.pub Internet Source	<1%
101	patents.google.com Internet Source	<1%
102	pdffox.com Internet Source	<1%
103	raw.githubusercontent.com Internet Source	<1%
104	repository.unja.ac.id Internet Source	<1%

105	scholar.unand.ac.id Internet Source	<1%
106	transloadit.com Internet Source	<1%
107	www.ejurnal.stmik-budidarma.ac.id Internet Source	<1%
108	www.jojonomic.com Internet Source	<1%
109	www.npmjs.com Internet Source	<1%
110	Michael Kevin Adinata, Ali Mahmudi, Yosep Agus Pranoto. "Klasterisasi Daerah Rawan Bencana Alam Menggunakan Algoritma KMeans", Infotek: Jurnal Informatika dan Teknologi, 2025 Publication	<1%