Jurnal Dialektika Informatika (Detika)
Vol. 4, No. 2, Mei 2024, hlm. 65-73

DOI: 10.24176/detika.v4i2.12804
p-ISSN: 2746-2811
e-ISSN: 2774-2148

KLASIFIKASI TINGKAT KEMATANGAN BELIMBING MADU BERDASARKAN KARAKTERISTIK WARNA MENGGUNAKAN ALGORITMA *K-NEAREST NEIGHBORS*

Muhammad Saifullah Irfanuddin[1], Mukhamad Nurkamid[2], Tutik Khotimah[3]

[1]Muhammad Saifullah Irfanuddin
[2]Mukhamad Nurkamid
[3]Tutik Khotimah
Email: [1]202051158@std.umk.ac.id, [2]muhammad.nurkamid@umk.ac.id, [3]tutik.khotimah@umk.ac.id

(Naskah masuk: 10 Juni 2024, diterima untuk diterbitkan: 3 Juli 2024)

Abstrak

Penelitian ini menggunakan teknologi pengolahan citra digital untuk mengembangkan sistem klasifikasi tingkat kematangan belimbing madu berdasarkan fitur warna menggunakan model warna *HSV*. Metode *K-Nearest Neighbor (K-NN)* diterapkan untuk klasifikasi tingkat kematangan buah tersebut, dengan *Google Colab* sebagai alat utama dalam analisis citra dan pengolahan data. Tujuan penelitian ini adalah mengklasifikasikan tingkat kematangan belimbing madu berdasarkan fitur warna dengan algoritma *K-NN*, serta merancang sistem yang dapat mengenali dan membedakan tiga tingkat kematangan: mentah, setengah matang, dan matang. Metode pengembangan sistem yang digunakan adalah metode *Waterfall*. Sistem klasifikasi kematangan belimbing madu dibangun menggunakan bahasa *Python*, dengan *K-NN* untuk mengidentifikasi kategori berdasarkan kemiripan data terhadap data pelatihan. Hasil penelitian ini adalah sistem klasifikasi berbasis *Web* dengan *deployment* menggunakan *framework Flask*. Evaluasi model dilakukan dengan *Confusion Matrix* yang menunjukkan akurasi sebesar 98%. Penelitian ini berpotensi menjadi landasan bagi pengembangan sistem serupa untuk mengklasifikasikan kematangan buah lainnya dengan teknologi pengolahan citra canggih.

Kata kunci: *Belimbing Madu, HSV, K-NN, Google Colab, Web*

CLASSIFICATION OF MATURITY LEVEL OF HONEY MARTUMBER BASED ON COLOR CHARACTERISTICS USING THE K-NEAREST NEIGHBOR ALGORITHM

Abstract

This research utilizes digital image processing technology to develop a classification system for the ripeness levels of starfruit based on color features using the HSV color model. The K-Nearest Neighbor (K-NN) method is applied to classify the ripeness levels of the fruit, with Google Colab as the main tool for image analysis and data processing. The aim of this research is to classify the ripeness levels of starfruit based on color features using the K-NN algorithm, and to design a system that can recognize and distinguish three ripeness levels: unripe, half-ripe, and ripe. The system development method used is the Waterfall method. The starfruit ripeness classification system is built using Python, with K-NN to identify categories based on the similarity of data to the training data. The result of this research is a web-based classification system deployed using the Flask framework. Model evaluation is conducted using a Confusion Matrix, showing an accuracy of 98%. This research has the potential to serve as a foundation for developing similar systems to classify the ripeness of other fruits using advanced image processing technology.

Keywords: *Starfruit Honey, HSV, K-NN, Google Colab, Web*

1. PENDAHULUAN

Belimbing madu adalah salah satu buah tropis yang semakin populer dan memiliki nilai komersial yang tinggi dalam industri makanan dan minuman karena rasa manisnya yang unik dan kandungan nutrisinya yang kaya. Bagi produsen dan pedagang belimbing madu, penting untuk memantau tingkat kematangan buah dengan cermat agar dapat memastikan kualitas yang optimal sebelum dikonsumsi atau dijual kepada konsumen. Tingkat kematangan belimbing madu sering kali dapat ditentukan oleh karakteristik warna kulitnya.

Pada saat ini, teknologi pemrosesan citra dan algoritma *Machine Learning*, seperti Algoritma *K-*

Nearest Neighbor (K-NN), telah menjadi alat yang sangat efektif dalam mendukung pengklasifikasian berbagai objek berdasarkan karakteristik visual. Dalam konteks ini, penggunaan teknologi ini untuk mengklasifikasikan tingkat kematangan belimbing madu berdasarkan warna kulitnya menjadi hal yang menarik.

Penelitian ini akan mengeksplorasi kemungkinan penggunaan Algoritma *K-NN* untuk mengklasifikasikan tingkat kematangan belimbing madu berdasarkan karakteristik warna kulitnya. Oleh karena itu, teknologi pengolahan citra digital diusulkan sebagai solusinya. Dalam penelitian ini tingkat kematangan belimbing madu dibagi menjadi tiga kategori, yaitu belimbing madu belum matang, belimbing madu setengah matang dan belimbing madu matang. Untuk melakukan klasifikasi digunakan metode *K-Nearest Neighbor* dengan memeriksa warna belimbing madu berdasarkan fungsi warna *HSV (Hue Saturation Value*. Teknologi ini diharapkan dapat berkontribusi pada klasifikasi belimbing madu yang lebih efisien.

2. LANDASAN TEORI

2.1. Belimbing Madu

Tumbuhan belimbing manis *(Averrhoa carambola, L)*, dikenal dengan beberapa nama seperti; *starfruit* (bahasa inggris), balireng (Bugis), bainang sulapa (Makassar), blimbing legi (Jawa), dan belimbing amis (Sunda). Belimbing sangat terkenal di masyarakat dan banyak ditemukan di daerah tropis. Rasanya segar dan harganya tergolong murah. Umumnya belimbing dikonsumsi dalam bentuk segar, namun belimbing juga dapat dijadikan berbagai bentuk olahan seperti bahan obat alami atau herbal. Seperti anti hipertensi, obat diabetes, pusing dan kelumpuhan. Itulah sebabnya belimbing memiliki nilai ekonomis yang cukup tinggi. (Ibnutama et al., 2023)

2.2. Pengolahan Citra Digital

Gambar berfungsi sebagai informasi visual yang penting dalam multimedia. Berbeda dengan teks, gambar mengandung informasi implisit. Pengolahan citra menggunakan komputer meningkatkan kualitas gambar untuk memudahkan interpretasi oleh manusia atau mesin. Metode ini bertujuan memperoleh informasi melalui konversi gambar menjadi data. Aplikasi pengolah gambar mempermudah manipulasi gambar. (Suryadi et al., 2022)

2.3. Google Colab

Google Colab merupakan *IDE* untuk pemrograman dengan bahasa *Python*. Pada platform ini dapat dilakukan pemrosesan data yang disediakan dari *Google*. *Google Colab* tersedia berbagai pustaka yang diperlukan untuk pengguna dalam mengembangkan aplikasi atau proyek dengan bahasa *Python* pengguna. (Gelar Guntara, 2023)

2.4. Canva

Canva adalah aplikasi desain grafis yang menjembatani penggunanya agar dengan mudah merancang berbagai jenis material kreatif secara online. Mulai dari mendesain kartu ucapan, poster, brosur, infografik, hingga presentasi. *Canva* saat ini tersedia dalam beberapa versi, yaitu pada versi *web*, *iPhone*, dan *Android*. (Hijrah et al., 2021)

2.5. Ngrok

Ngrok merupakan suatu perangkat lunak yang diciptakan oleh Alan Shreve dengan fungsi untuk menghubungkan jaringan publik ke port komputer lokal melalui proses tunneling. *Ngrok* memiliki kemampuan untuk membentuk atau membuka jaringan pribadi melalui *NAT* atau *firewall*, sehingga memungkinkan penghubungan dari *localhost* ke internet dengan aman menggunakan jalur *HTTPS*. (Zenik et al., 2023)

2.6. Flask

Flask ialah suatu kerangka kerja web yang ditulis dalam bahasa pemrograman *Python* dan dikategorikan sebagai *micro-framework*. Fungsinya mencakup peran sebagai landasan untuk aplikasi *web* dan tampilan. Dengan menggunakan *Flask* dan bahasa *Python*, pengembang dapat membuat situs *web* yang terstruktur dan mengelola perilaku suatu situs web dengan lebih simpel. (Budianto et al., 2023)

2.7. Website

World Wide Web (WWW), atau yang sering disebut sebagai *web* atau situs *web*, adalah suatu sistem yang menghadirkan informasi kepada pengguna melalui internet, umumnya dalam bentuk audio, gambar, atau teks. Informasi tersebut disimpan di suatu lokasi yang disebut sebagai *server web*. (Evaryanti et al., 2017)

2.8. Visual Studio Code

Visual Studio Code merupakan sebuah editor kode yang dapat diekstensi dan merupakan perangkat lunak terbuka yang di lisensi di bawah *MIT*. Proyek *VSCode* yang diinisiasi oleh Microsoft, memiliki jumlah kontributor tertinggi di platform *GitHub*. (Ismail Setiawan, 2022)

2.9. HTML

Hypertext Markup Language (HTML) adalah bahasa standar yang digunakan untuk menampilkan konten pada halaman *website*. Fungsi *HTML* antara lain bisa mengelola dan merancang tampilan isi halaman situs *web*,

membuat tabel pada halaman situs *web*, mempublikasikan halaman situs *web* secara *daring*, membuat formulir sebagai *input* serta menangani registrasi dan transaksi melalui situs *web*, dan menampilkan area gambar pada peramban. (Mariko, 2019)

2.10. CSS

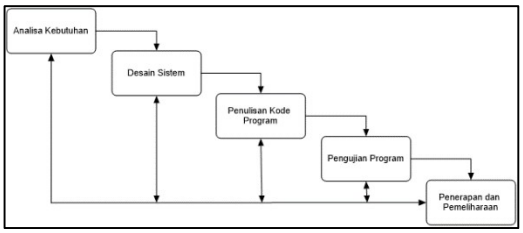
CSS (Cascading Style Sheet) merupakan suatu bahasa pemrograman yang digunakan untuk mengatur desain tampilan pada *web*, seperti warna, jenis huruf, *outline*, latar belakang, penyesuaian tampilan situs *web* dengan ukuran layar, dan sebagainya. Penggunaan *CSS* dalam pembuatan situs *web* ini bertujuan untuk bekerja sama dengan *HTML* agar dapat menciptakan tampilan situs *web* yang menarik. (Sari et al., 2022)

2.11. Python

Python, sebagai bahasa pemrograman tingkat tinggi, menonjol karena sintaksisnya yang jelas dan elegan, membuatnya mudah dipelajari. Kelebihan *Python* juga terletak pada penggunaan modul-modul yang memiliki struktur data tingkat tinggi, efisien, dan siap digunakan secara langsung. Dalam praktiknya, *source code* aplikasi *Python* umumnya dikompilasi menjadi *bytecode*, suatu format perantara, sebelum dieksekusi. (Ratna & Al Banjari, 2020)

2.12. Metode Waterfall

Metode pengembangan waterfall terdiri dari tahapan requirement, design, implementation, verification, dan maintenance. Pada tahap requirement, dilakukan analisis kebutuhan sistem yang dipahami oleh klien dan staf pengembang. Tahap design mencakup perancangan arsitektur sistem secara keseluruhan hingga algoritma yang rinci. Di tahap implementation, desain diubah menjadi kode program dan diintegrasikan menjadi sistem lengkap. Tahap verification melibatkan pengujian untuk memastikan persyaratan sistem terpenuhi, termasuk verifikasi oleh klien. Tahap akhir, maintenance, mencakup perbaikan sistem sesuai kontrak yang telah disepakati. (Rahayu & Fadillah Rezky, 2023)

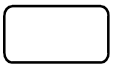
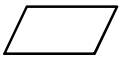
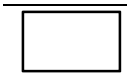
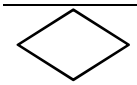


Gambar 1. Metode *Waterfall*

2.13. Flowchart

Diagram alir *(flowchart)* adalah lambang-lambang yang digunakan untuk mengilustrasikan urutan proses yang terjadi dalam suatu program komputer secara sistematis dan logis (Rosaly & Prasetyo, 2019). Simbol *flowchart* dapat dilihat pada tabel 1.

Tabel 1. Simbol *Flowchart*

Simbol	Nama	Fungsi
	Terminal	Lambang ini digunakan untuk memulai atau mengakhiri suatu proses atau kegiatan.
	Input/ Output	Lambang ini dipergunakan untuk mengilustrasikan proses input (baca) dan output (cetak) dalam suatu program.
	Proses	Lambang ini dipakai untuk mengilustrasikan suatu proses yang sedang dalam tahap eksekusi.
	Decision	Lambang ini digunakan untuk melaku an pengujian terhadap suatu kondisi yang sedang dalam proses.
	Flow Line	Lambang ini dipakai untuk mengilustrasikan aliran proses dari satu kegiatan ke kegiatan lainnya.
	Database	Lambang ini dipakai untuk mengilustrasi penyimpanan data.

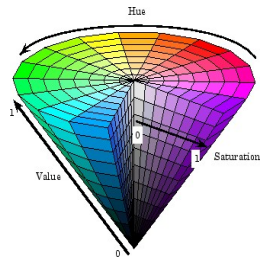
2.14. Ekstrasi Fitur

Fitur adalah tanda unik yang terdapat pada suatu objek dan membantu membedakan suatu objek dengan benda lainnya berdasarkan ciri-ciri tertentu. Fitur-fitur yang biasa diamati adalah fitur warna, fitur bentuk, dan fitur tekstur. Fitur-fitur tersebut dapat dikenali karena beberapa objek mempunyai pola-pola tertentu, sehingga mudah dibedakan oleh manusia. Pada fitur warna, biasanya objek benda dikenali dengan adanya perbedaan warna di ruang warna tertentu. (Kosasih et al., 2021)

2.15. HSV

Ruang warna *HSV* merupakan model warna yang umum digunakan karena ruang warna ini mirip dengan kemampuan persepsi warna mata manusia. Ruang warna *HSV* terdiri dari kata *hue, saturation dan value*. *Hue* mengidentifikasi warna asli, seperti merah, ungu, dan kuning, serta digunakan. Untuk membedakan variasi warna seperti tingkat merah atau hijau dalam cahaya. *Hue* ini berhubungan dengan panjang gelombang cahaya. *Saturation* menunjukkan sejauh mana warna tersebut murni, mengindikasikan sejauh mana warna tersebut mencampur dengan warna putih. *Value*, di sisi lain, mencerminkan seberapa terang atau gelap suatu warna tanpa memperhitungkan jenis

warnanya (Sularida et al., 2018). Pada gambar 2 merupakan ruang warna *HSV*.



Gambar 2. Ruang Warna *HSV*
Sumber: (Syarifah et al., 2022)

Sebelum melakukan ekstraksi ciri *HSV* harus dilakukan konversi terlebih dahulu dari *RGB* ke *HSV* (Syarifah et al., 2022)

$$r = \frac{R}{255} \quad (1)$$

$$g = \frac{G}{255} \quad (2)$$

$$b = \frac{B}{255} \quad (3)$$

Cara konversi normalisasi *RGB* ke *HSV* sebagai rumus berikut:

$$V = \max(r, g, b) \quad (4)$$

$$S = \begin{cases} 0 & \rightarrow otherwise \\ V - \frac{\min(r,g,b)}{v} & \rightarrow jika\ V\ != 0 \end{cases} \quad (5)$$

$$H = \begin{cases} \frac{60° \times (g-b)}{v-\text{mi}\ (r,g,b)} & \rightarrow jika\ V = r \\ \frac{120° + 60°(b-r)}{v-\min(r,g,b)} & \rightarrow jika\ V = g \\ \frac{240° + 60°(r-g)}{v-\min(r,g,b)} & \rightarrow jika\ V = b \end{cases} \quad (6)$$

$$H = H + 360° \rightarrow jika\ H < 0$$

Keterangan:
R = nilai *red*
G = nilai *green*
B = nilai *blue*
r = normalisasi *red*
g = normalisasi *green*
b = normalisasi *blue*
H = nilai *hue*
S = nilai *saturation*
V = nilai *value*

2.16. K-Nearest Neighbor

Metode *K-Nearest Neighbor (K-NN)* adalah suatu teknik di mana mencari nilai terdekat dalam data pelatihan terhadap data uji dengan cara menghitung jarak. Nilai K ini digunakan untuk menentukan seberapa banyak tetangga terdekat yang akan digunakan dalam evaluasi terhadap data yang sedang diproses. (Atmaja & Lusiana, 2023)

Dari berbagai jenis perhitungan yang tersedia, euclidean adalah yang paling umum digunakan dalam klasifikasi *K-NN* dan merupakan perhitungan jarak default pada *python*. Rumus perhitungan jarak *euclidean* digunakan untuk mengukur jarak antara dua titik dalam ruang berdasarkan panjang garis lurus yang menghubungkan titik-titik tersebut. Berikut adalah rumusnya:

$$d_i = \sqrt{\sum_{i=1}^{n}(x_{2i} - x_{1i})^2} \quad (7)$$

Keterangan:
d_i = jarak *euclidean* ke-i
x_{2i} = data *training* ke-i
x_{1i} = data *testing* ke-i
n = banyaknya data *training*
i = baris ke-i

2.17. Confusion Matrix

Confusion matrix adalah metode yang digunakan untuk mengukur kinerja dalam konteks klasifikasi di mana hasilnya dapat terdiri dari beberapa kategori. Dalam aplikasi klasifikasi tingkat kematangan belimbing madu, metode evaluasi akurasi digunakan dengan menggunakan *confusion matrix*. *Confusion matrix* adalah tabel yang digunakan untuk mengilustrasikan performa suatu model atau metode klasifikasi pada *dataset* pengujian, di mana nilai aktual atau sebenarnya telah diketahui. Akurasi adalah seberapa jauh model berhasil mengklasifikasi dengan benar. *Precision* adalah seberapa jauh prediksi positif dari model yang benar-benar relevan dan tepat. *Recall* atau sensitivitas adalah mengukur seberapa jauh model berhasil mendeteksi atau mengenali seluruh data postif sebenarnya. *F1 score* adalah penggabungan dari presisi dan *recall*, memberikan ukuran keseimbangan keduanya. (Febrinamas et al., 2023) Pengujian *confusin matrix* menggunakan rumus sebagai berikut:

$$\text{Akurasi} = \frac{TP+TN}{TP+FN+FP+T}\ x\ 100\% \quad (8)$$

$$\text{Precision} = \frac{TP}{TP+FP} \quad (9)$$

$$\text{Recall} = \frac{TP}{TP+FN} \quad (10)$$

$$\text{F1-Score} = 2\ x\ \frac{\text{Precision x Recall}}{\text{Precision+ Recall}} \quad (11)$$

2.16. Black Box

Uji *black box* memiliki peranan penting dalam pengujian perangkat lunak, yakni untuk memverifikasi apakah keseluruhan fungsi sistem berjalan dengan baik. Ketika seseorang melakukan uji perangkat lunak dengan metode *black box*, tidak diperlukan pengetahuan mengenai pemrograman atau struktur internal dari perangkat lunak tersebut. Pada uji *black box*, penguji tidak memiliki akses untuk melihat

kode sumber dan struktur sistem, hanya berinteraksi melalui antarmuka dengan memberikan *input* dan memeriksa *output* tanpa mengetahui detail bagaimana proses dari *input* menjadi *output* terjadi. (Parlika et al., 2020)

3. METODE PENELITIAN

Penelitian ini menggunakan metode *waterfall* dalam proses pengembangan sistem. Cara membuat tahapan- tahapannya adalah sebagai berikut:

a. Analisa Kebutuhan

Proses analisa kebutuhan dilakukan secara langsung terjun ke lapangan dan wawancara pemilik Oemah Belimbing Madu. Pemilik usaha membutuhkan fungsi aplikasi penyortiran belimbing madu yang cepat dan akurat untuk meningkatkan kualitas belimbing madunya. Studi literatur dilakukan sebelum dilakukan untuk membuka pandangan pada peneliti dan sesudah wawancara utuk mengetahui kebutuhan dalam pembuatan aplikasi klasifikasi kematangan belimbing madu. Dilakukan pengumpulan data belimbing madu dengan tiga kategori mentah, setengah matang, dan matang.

b. Desain

Dalam perancangan aplikasi tingkat kematangan belimbing madu. Fokus dalam perancangan model agar aplikasi bisa cepat dalam komputasi dan akurat dalam menebak kematangan belimbing madu. Desain antarmuka aplikasi dibuat dalam perancangan web agar bisa digunakan untuk semua orang, dan dibuat sederhana untuk mudah dipahami.

c. Penulisan Kode Program

Proses penulisan kode dilakukan sesuai dengan rancangan-rancangan yang telah dibuat di desain. Penulisan kode dilakukan dengan menerjemahkan rancangan sistem menjadi kode program yang dapat di eksekusi. Dengan desain yang jelas dan efektif membantu dalam penulisan kode dan menghasilkan aplikasi yang baik.

d. Pengujian Program

Pada tahap ini pengujian menggunakan fungsional dengan black box agar menunjukkan aplikasi berjalan dengan baik. Proses pengujian melibatkan pengujian dari model KNN dengan evaluasi confusion matrix, diharapkan aplikasi klasifikasi belimbing madu dapat memiliki tingkat keakuratan yang baik.

e. Penerapan dan Pemeliharaan

Aplikasi yang telah dibuat saat ini sedang digunakan oleh pengguna, dan kegiatan pemeliharaan sedang dilakukan. Pemeliharaan melibatkan tindakan seperti memperbaiki setiap error atau bug yang muncul dalam aplikasi, menambahkan unit-program kecil untuk pengembangan aplikasi, dan melakukan penyesuaian sistem sesuai dengan kebutuhan para pengguna.

4. HASIL DAN PEMBAHASAN

4.1. Hasil

Penelitian ini menghasilkan sistem klasifikasi kematangan buah belimbing madu menggunakan bahasa Python berbasis *web*. Sistem ini dirancang dengan metode KNN untuk mengidentifikasi kategori berdasarkan kemiripan data dengan dataset yang besar. KNN mengklasifikasikan objek baru berdasarkan atribut dan sampel data pelatihan yang sudah ada. Metode ini menggunakan klasifikasi berdasarkan tetangga terdekat untuk memprediksi nilai instance baru yang akan diklasifikasikan. Prinsip kerja K-Nearest Neighbor adalah mencari jarak terdekat antara data yang dievaluasi dari atribut suatu data dengan "k" tetangga terdekat dalam data pelatihan.

Gambar 3. Tampilan Menu Beranda

Pada gambar 4, pengguna dapat memasukkan gambar belimbing madu yang akan diklasifikasi tingkat kematanganya. Gambar harus berformat jpg/jpeg/png agar bisa diproses.

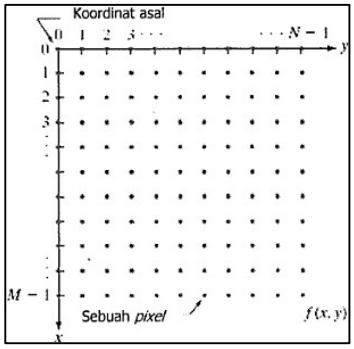
Gambar 4. Tampilan Input Gambar

Pada gambar 5, pengguna dapat melihat hasil klasifikasi belimbing madu yang telah diinputkan. Hasil Tingkat kematangan akan muncul di halaman yang sama.

Gambar 5. Hasil Klasifikasi Belimbing Madu

4.2. Pembahasan

4.2.1. Koordinat Piksel



Gambar 6. Koordinat Piksel
Sumber: www.google.com

4.2.2. Tahap Pengumpulan Data

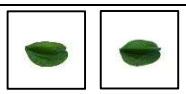
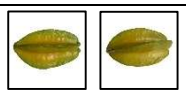
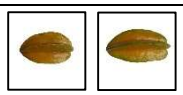
Pada tahap pengumpulan data belimbing madu, peneliti mengambil sampel belimbing madu dengan tiga kategori: mentah, setengah matang, matang Untuk lebih detailnya bisa di lihat pada tabel 2.

Tabel 2. Data Belimbing Madu

No	Kategori	Data	Per Buah	Jumlah
1	Mentah	8 Buah	10 Foto	80
2	Setengah Matang	8 Buah	10 Foto	80
3	Matang	8 Buah	10 Foto	80
Jumlah Keseluruhan Foto				240

Pada tabel 3, ada beberapa contoh data belimbing madu yang diambil di Oemah Belimbing Madu. Gambar tersebut di ambil dengan kamera 12MP dengan latar putih.

Tabel 3. Contoh Pengambilan Data Belimbing Madu

No	Contoh Citra	Label
1		Mentah
2		Setengah Matang
3		Matang

4.2.3. Resize Gambar

Untuk menghasilkan data yang serupa maka diperlukan *resize* gambar. Peneliti me-*resize* gambar dengan ukuran sama yakni 200 x 200 piksel untuk semua gambar. Hal ini dilakukan untuk standarisasi ukuran data agar sesuai dengan model atau algoritma *KNN*.

```
# Ukuran yang diinginkan untuk gambar yang diresize
target_size = (200, 200)
```

Gambar 7. Resize Gambar

4.2.4. Konversi RGB

Pada tabel 4, peneliti menentukan nilai *RGB* per koordinat. Nilai *RGB* menghasilkan nilai dengan rentang 0 – 255. Berikut adalah nilai *RGB* pada salah satu *dataset* per koordinat.

Tabel 4. Sampel *RGB* Per-Gambar

Koordinat		Warna		
X	Y	R	G	B
100	83	47	79	29
100	84	50	80	30
100	85	47	77	27
100	86	44	72	24
100	87	42	70	22

4.2.5. Normalisasi RGB

Normalisasi *RGB* digunakan untuk perbandingan citra agar lebih baik, membandingkan citra yang diambil dengan kondisi Cahaya yang berbeda-beda. Normalisasi citra pada umumnya dilakukan dengan membagi nilai *RGB* dengan 255 agar tercipta angka normalisasi antara 0 sampai 1. Berikut adalah normalisasi per koordinat per gambar pada tabel 5.

Tabel 5. Sampel Normalisasi *RGB* Per-Gambar

Koordinat		Warna		
X	Y	R	G	B
100	83	0.4	0.35	0
100	84	0.38	0.35	0
100	85	0.42	0.41	0.05
100	86	0.45	0.45	0.08
100	87	0.4	0.4	0.05

Pada tabel 6 adalah salah satu contoh rata-rata normalisasi RGB per gambar. Nantinya akan digunakan sebagai dataset dalam melakukan penetuan nilai Euclidian.

Tabel 6. Sampel Rata-Rata *RGB* Per-Gambar

Rata-rata Normalisasi RGB		
MeanR	MeanG	MeanB
0.86	0.86	0.76

4.2.6. Ekstrasi HSV

Pada tabel 7 adalah ekstrasi fitur warna *HSV*. Penentuan nilai *HSV* menggunakan rumus pada gambar 1 penentuan *HSV*. Perhitungan pada tabel dihitung per koordinat.

Tabel 7. Sampel *HSV* Per-Gambar

Koordinat		Warna		
X	Y	H	S	V
100	83	43.53	0.97	0.46
100	84	42.05	0.97	0.47
100	85	41.16	0.98	0.49
100	86	40.8	0.98	0.5
100	87	38.91	0.98	0.51

Pada tabel 8 adalah sampel rata-rata nilai *HSV* per gambar. Hal ini dilakukan untuk menentukan variabel dalam menghitung nilai Euclidian.

Tabel 8. Sampel *HSV* Per-Gambar

Rata-rata Keseluruhan per-Gambar		
MeanH	MeanS	MeanV
23.94	0.26	0.86

4.2.7. Dataset

Dataset yang dihasilkan totalnya adalah 240, yaitu (mentah 80 dengan label 1), (setengah matang 80 dengan label 2) dan (matang 80 dengan label 3). *Dataset* ini digunakan untuk mencari data independen (x) yakni *MeanR, MeanG, MeanB, MeanH, MeanS, MeanV* dan variabel yang dituju/target (y) yakni kematangan. Berikut adalah sampel *dataset* pada tabel 9.

Tabel 9. Sampel *Dataset* Belimbing Madu

MeanR	MeanG	MeanB	MeanH	MeanS	MeanV	Label
0,900674412	0,916505392	0,895515294	105,2527102	0,022902318	0,916505392	1
0,846690196	0,830749951	0,694823824	53,70208717	0,179364747	0,846690196	2
0,878270196	0,838240902	0,766423529	38,53069153	0,127348813	0,878270196	3

4.2.8. Proses Klasifikasi KNN

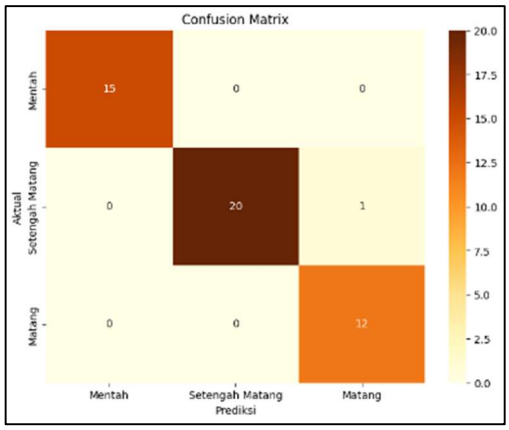
Pada proses *KNN* peneliti menggunakan data (x) sebagai data *independent* dan data (y) sebagai target atau data *dependen*. Peneliti membagi dataset menjadi 20% untuk data *testing* dan membaginya sama rata. Peneliti juga membuat nilai k=7 dengan pendekatan jarak *Euclidian*. Dapat dilihat pada gambar 8.

```
data = pd.read_csv(data_file)  # Membaca file CSV ke dalam DataFrame
X = data[['MeanR', 'MeanG', 'MeanB', 'MeanH', 'MeanS', 'MeanV']]  # Mengambil fitur dari data
y = data['Kematangan']  # Mengambil label dari data

# Membagi data menjadi data latih dan uji
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)

# Inisialisasi model KNN dengan k=7 dan menggunakan Euclidean distance
knn = KNeighborsClassifier(n_neighbors=7, metric='euclidean')

# Melatih model KNN
knn.fit(X_train, y_train)

# Melakukan prediksi pada data uji
y_pred = knn.predict(X_test)
```

Gambar 8. Proses *KNN*

4.2.9. Confusion Matrix

Pada gambar 9, dapat dilihat hasil *consfuion matrix* dari kategori kelas mentah (1), setengah matang (2) dan matang (3). Setiap kelas berisi nilai *confusion matrix* dan dapat dianalisa pada setiap kategori kelas. Berikut adalah *confusion matrix* untuk seluruh kategori.



Gambar 9. Evaluasi *Confusion Matrix*

Berikut adalah perhitungan *confusion matrix* per kelas dari mentah, setengah matang dan matang yang peneliti bagi menjadi 3 tabel.

Tabel 10. Hasil *Confusion Matrix* Mentah

	Prediksi	
Aktual	Mentah	Bukan Mentah
Mentah	15 (TP)	0 (FN)
Bukan Mentah	0 (FP)	33 (TN)

$$\text{Akurasi} = \frac{15+33}{15+0+0+33} x\ 100\% = 100\%$$

$$\text{Precision} = \frac{15}{15+10} = 1$$

$$\text{Recall} = \frac{15}{15+10} = 1$$

$$\text{F1 - Score} = 2 \text{ x } \frac{1 \text{ x } 1}{1+1} = 1$$

Tabel 11. Hasil *Confusion Matrix* Setengah Matang

	Prediksi	
Aktual	Setengah Matang	Bukan Setengah Matang
Setengah Matang	20 (TP)	1 (FN)
Bukan Setengah Matang	0 (FP)	27 (TN)

$$\text{Akurasi} = \frac{20+27}{20+1+0+27} x\ 100\% = 98\%$$

$$\text{Precision} = \frac{20}{20+0} = 1$$

$$\text{Recall} = \frac{20}{20+1} = 0.952$$

$$\text{F1 - Score} = 2 \text{ x } \frac{1 \text{ x } 0.952}{1+0.952} = 0.975$$

Tabel 121. Hasil *Confusion Matrix* Matang

	Prediksi	
Aktual	Matang	Bukan Matang
Matang	12 (TP)	1 (FN)
Bukan Mentah	0 (FP)	35 (TN)

$$\text{Akurasi} = \frac{12+35}{12+1+0+35} x\ 100\% = 98\%$$

$$\text{Precision} = \frac{12}{12+0} = 1$$

$$\text{Recall} = \frac{12}{12+1} = 0.923$$

$$\text{F1 - Score} = 2 \text{ x } \frac{1 \text{ x } 0.923}{1+0.923} = 0.96$$

Dari pengujian yang dilakukan dengan 48 data uji, dengan masing masing 15 data uji mentah, 21 data uji setengah matang, dan 12 data uji matang. Didapatkan akurasi 98%. Hasilnya dapat dilihat dalam laporan klasifikasi pada gambar 10.

Laporan Klasifikasi:

	precision	recall	f1-score	support
1	1.00	1.00	1.00	15
2	1.00	0.95	0.98	21
3	0.92	1.00	0.96	12
accuracy			0.98	48
macro avg	0.97	0.98	0.98	48
weighted avg	0.98	0.98	0.98	48

Gambar 10. Laporan Klasifikasi

Perhitungan *Macro Avg* memberikan bobot yang sama untuk setiap kelas sebagai berikut:

$$\text{Precision Macro Avg} = \frac{1+1+0.92}{3} = 0.97$$

$$\text{Recall Macro Avg} = \frac{1+0.95+1}{3} = 0.98$$

$$\text{F1-Score Macro Avg} = \frac{1+0.98+0.96}{3} = 0.98$$

Perhitungan *Weight Avg* memberikan bobot berdasarkan jumlah data pada setiap kelas sebagai berikut:

Precision Weight Avg

$$= \frac{(1x15) + (1x21) + (0.92x12)}{48} = 0.98$$

Recall Weight Avg

$$= \frac{(1x15) + (0.95x21) + (1x12)}{48} = 0.98$$

F1-Score Weight Avg

$$= \frac{(1x15) + (0.98x21) + (0.96x12)}{48} = 0.98$$

4.2.10. Pengujian Blackbox

Pengujian *blackbox* dilakukan guna untuk menguji dari sisi fungsional perangkat lunak. Untuk memastikan fungsi berjalan dan sesuai dengan kebutuhan sistem. Pengujian ada pada tabel 13.

Tabel 13. Pengujian *Blackbox*

No	Aktifitas	Hasil	Pengujian	Kesimpulan
1	Pengguna membuka menu beranda	Menampilkan menu beranda	Tampil menu beranda	Berhasil
2	Pengguna memasukkan file berformat jpg/jpeg/ png	Sistem memproses & tampil hasil klasifikasi	Tampil hasil klasifikasi	Berhasil
3	Pengguna memasukkan file bukan format jpg/jpeg/ png	Sistem tidak dapat diproses kecuali format jpg/jpeg/ png	Tampil keterangan tidak dapat diproses kecuali format jpg/jpeg/ png	Berhasil

5. PENUTUP

5.1. Kesimpulan

Berdasarkan hasil penelitian, peneliti menyimpulkan bahwa pengambilan data gambar belimbing madu harus menggunakan latar putih polos dan pencahayaan yang tepat untuk memaksimalkan ekstraksi fitur warna. Model dikembangkan menggunakan Flask untuk memungkinkan klasifikasi melalui web, dan pengujian Black Box berhasil dilakukan melalui pengujian menu. Evaluasi model menggunakan confusion matrix menunjukkan akurasi sebesar 98%.

5.2. Saran

Berdasarkan hasil penelitian, peneliti menyarankan agar penelitian selanjutnya melakukan perbandingan dengan algoritma lain untuk mengetahui peningkatan akurasi. Selain itu, disarankan untuk membandingkan dengan objek penelitian lain guna memperoleh akurasi yang lebih baik. Peneliti juga perlu melakukan training data dengan jumlah data yang lebih banyak untuk hasil yang lebih maksimal. Augmentasi dan segmentasi gambar juga disarankan untuk meningkatkan akurasi hasil. Terakhir, dalam deployment model ke web, disarankan menggunakan input kamera langsung untuk memungkinkan klasifikasi secara real-time.

DAFTAR PUSTAKA

Atmaja, W. P., & Lusiana, V. (2023). Klasifikasi Jenis Batik Pekalongan Menggunakan Citra HSI Dengan Metode K-Nearest Neighbor. *Jurnal INSTEK: Informatika Sains Dan Teknologi*, *8*(1), 1–8.

Budianto, A. J., Ocsa, P., & Saian, N. (2023). Pengembangan Modul Inventory Management pada Aplikasi Master Distribution Centre System Menggunakan Framework Flask di PT XYZ. *Jurnal Teknologi Informasi Dan Komunikasi)*, *7*(2), 2023. https://doi.org/10.35870/jti

Evaryanti, L. E., Agung Raditya, I. G. L., & Krisna Juliharta, I. G. L. (2017). Rancang Bangun

Sistem Informasi Perpustakaan Berbasis Website Pada SMKN 1 Gianyar. *SNATIKA*, *04*, 1689–1699.

Febrinamas, D. R., Hidayati, R., & Nirmala, I. (2023). Klasifikasi Buah Pinang Berdasarkan Data Sensor Menggunakan Metode K-Nearest Neighbor Berbasis Web. *Journal of Computer System and Informatics (JoSYC)*, *4*(4), 1046–1055. https://doi.org/10.47065/josyc.v4i4.3805

Gelar Guntara, R. (2023). Pemanfaatan Google Colab Untuk Aplikasi Pendeteksian Masker Wajah Menggunakan Algoritma Deep Learning YOLOv7. *Jurnal Teknologi Dan Sistem Informasi Bisnis*, *5*(1), 55–60. https://doi.org/10.47233/jteksis.v5i1.750

Hijrah, L., Fikry Arransyah, M., Putri, K., Arija, N., Putri, R. K., & Bisnis, A. (2021). Pelatihan Penggunaan Canva Bagi Siswa di Samarinda. *Jurnal Pelayanan Kepada Masyarakat*, *3*(1), 98–106.

Ibnutama, K., Gilang Suryanata, M., Putri, R. O., & Al Hafiz, A. (2023). Seleksi Tingkat Kematangan Citra Buah Belimbing Menggunakan Ruang Warna CMYK. *Jurnal SAINTIKOM (Jurnal Sains Manajemen Informatika Dan Komputer)*, *22*(2), 302–310. https://ojs.trigunadharma.ac.id/index.php/jis/index

Ismail Setiawan. (2022). Komparasi Kinerja Integrated Development Environment (IDE) Dalam Mengeksekusi Perintah Python. *SATESI: Jurnal Sains Teknologi Dan Sistem Informasi*, *2*(1), 52–59. https://doi.org/10.54259/satesi.v2i1.784

Kosasih, R., Klasifikasi: Kematangan, T., Tingkat, K., Pisang, K., Ekstraksi, B., Tekstur, F., & Knn, A. (2021). Classification of Banana Ripe Level Based on Texture Features and KNN Algorithms. In *Jurnal Nasional Teknik Elektro dan Teknologi Informasi* | (Vol. 10, Issue 4).

Mariko, S. (2019). Aplikasi Website Berbasis HTML Dan JavaScript untuk Menyelesaikan Fungsi Integral Pada Mata Kuliah Kalkulus. *Jurnal Inovasi Teknologi Pendidikan*, *6*(1), 80–91. https://doi.org/10.21831/jitp.v6.1.22280

Parlika, R., Ardhian Nisaa', T., Ningrum, S. M., & Haque, B. A. (2020). Studi Literatur Kekurangan dan Kelebihan Pengujian Black Box. *TEKNOMATIKA*, *10*(02), 1–5.

Rahayu, D., & Fadillah Rezky, S. (2023). Perancangan Aplikasi Lowongan Kerja Berbasiskan Web dengan Menggunakan Metode Waterfall. In *Jurnal SIKOM (Sistem Informasi Komputer): Vol. XX*.

Ratna, S., & Al Banjari, M. A. (2020). Pengolahan Citra Digital dan Histogram dengan Phyton dan Text Editor Phycharm. In *Technologia* (Vol. 11, Issue 3).

Rosaly, R., & Prasetyo, A. (2019). *Pengertian Flowchart Beserta Fungsi dan Simbol-simbol Flowchart yang Paling Umum Digunakan*.

Sari, I. P., Qathrunada, F., Lubis, N., & Anggraini, T. (2022). *Perancangan Sistem Absensi Pegawai Kantoran Secara Online pada Website Berbasis HTML dan CSS*.

Sularida, N., Sari, J. Y., Purwanti, I., & Purnama, N. (2018). Identifikasi Tingkat Kematangan Buah Pisang Menggunakan Metode Ektraksi Ciri Statistik Pada Warna Kulit Buah. *Jurnal ULTIMATICS*, *X*(2), 98–102.

Suryadi, A., Vivi Putri, M., & Lia Febrianti, E. (2022). Pengolahan Citra Digital Dan Logika Fuzzy Dalam Identifikasi Tingkat Kematangan Buah. In *Journal of Science and Social Research* (Issue 2). http://jurnal.goretanpena.com/index.php/JSSR

Syarifah, A., Riadi, A. A., & Susanto, A. (2022). Klasifikasi Tingkat Kematangan Jambu Bol Berbasis Pengolahan Citra Digital Menggunakan Metode K-Nearest Neighbor. *JIMP: Jurnal Informatika Merdeka Pasuruan*, *7*(1), 27–35.

Zenik, A., Olivia Sereati, C., Indriati, K., Wijayanti, L., & Teknik Elektro, P. (2023). *Analisis Audio Capture untuk Pengumpulan Data pada Smart Speaker* (Vol. 16, Issue 1).