Terakreditasi SINTA Peringkat 3
Surat Keputusan Direktur Jenderal Pendidikan Tinggi, Riset, dan Teknologi Nomor 225/E/KPT/2022
masa berlaku mulai Vol.7 No. 1 tahun 2022 s.d Vol. 11 No. 2 tahun 2026

Terbit online pada laman web jurnal:
http://publishing-widyagama.ac.id/ejournal-v2/index.php/jointecs

JOINTECS

JOINTECS
(Journal of Information Technology and Computer Science)

Vol. 8 No. 3 (2023) 115 - 126
e-ISSN:2541-6448 p-ISSN:2541-3619

Klasifikasi Penyakit Pada Daun dan Buah Jambu Menggunakan *Convolutional Neural Network*

Fadlan Sayyidul Anam[1*], Muhammad Rafi Muttaqin[2], Yudhi Raymond Ramadhan[3]
Program Studi Teknik Informatika, Sekolah Tinggi Teknologi Wastukancana
[1]fadlansayyidul44@wastukancana.ac.id, [2]rafi@wastukancana.ac.id, [3]yudhi.raymond@wastukancana.ac.id

Abstract

Guava is a crop commodity in West Java with total production in 2021 reaching 692,488 quintals. This production decreased by -12.82% compared to 2020 which amounted to 794,345 quintals. The research uses deep learning technology with the Convolutional Neural Network (CNN) algorithm and MobileNetV2 architecture to classify digital images of guava leaves and fruits that have been labeled or called supervised learning. The development method used in this research is Cross Industry Standard Process for Data Mining (CRISP-DM). Based on the results of this study, the guava leaf model has excellent evaluation results, training accuracy of 99.6%, validation accuracy of 100%, training loss of 3.2%, and validation loss of 3.1%. The confusion matrix of this model has 100% accuracy from 63 validation data. Meanwhile, the guava fruit model requires a dropout of 0.2 and L2 kernel regularizers of 0.01 to reduce overfitting. This model has a training accuracy of 98.8%, validation accuracy of 91.6%, training loss of 19.1%, and validation loss of 38.6%. The confusion matrix results show that the accuracy of this model reaches 91.6% of 84 validation data. Then the model was successfully implemented into a mobile-based application using the Kotlin programming language.

Keywords: guava; deep learning; supervised learning; convolutional neural network; mobilenetv2

Abstrak

Jambu biji merupakan komoditas tanaman di Jawa Barat dengan jumlah produksi tahun 2021 mencapai 692.488 kuintal. Produksi ini mengalami penurunan sebesar -12,82% dibandingkan dengan tahun 2020 yang sebesar 794.345 kuintal. Penelitian menggunakan teknologi *deep learning* dengan algoritma *Convolutional Neural Network* (CNN) dan berarsitektur *MobileNetV2* untuk melakukan klasifikasi citra digital daun dan buah jambu biji yang telah diberi label atau disebut *supervised learning*. Metode pengembangan yang digunakan dalam penelitian ini adalah *Cross Industry Standard Process for Data Mining* (CRISP-DM). Berdasarkan hasil penelitian ini, *model* daun jambu biji memiliki hasil evaluasi yang sangat baik, *training accuracy* sebesar 99,6%, *validation accuracy* 100%, *training loss* 3,2%, dan *validation loss* 3,1%. *Confusion matrix model* ini memiliki akurasi 100% dari 63 data validasi. Sementara itu, *model* buah jambu biji memerlukan *dropout* sebesar 0,2 dan *kernel regularizers* L2 sebesar 0,01 untuk mengurangi *overfitting. Model* ini memiliki *training accuracy* sebesar 98,8%, *validation accuracy* 91,6%, *training loss* 19,1%, dan *validation loss* 38,6%. Hasil *confusion matrix* menunjukkan akurasi *model* ini mencapai 91,6% dari 84 data validasi. Kemudian *model* berhasil diimplementasikan menjadi aplikasi berbasis *mobile* menggunakan bahasa pemrograman *Kotlin.*

Kata kunci: jambu biji; *deep learning; supervised learning; convolutional neural network; mobilenetv2*

Diterima Redaksi : 18-07-2023 | Selesai Revisi : 06-08-2023 | Diterbitkan Online : 30-09-2023

1. Pendahuluan

Pertanian adalah kegiatan manusia mengelola tanaman, hewan untuk memenuhi kebutuhan pangan manusia. Salah satu sektor pertanian adalah hortikultura yang meliputi tanaman buah-buahan, tanaman sayuran, tanaman hias, dan tanaman obat-obatan. Salah satu jenis tanaman buah-buahan adalah jambu biji. Jambu biji (Psidium guajava L.) telah ditanam dan dimanfaatkan sebagai buah yang penting di daerah tropis seperti India, Indonesia, Pakistan, Bangladesh, dan Amerika Selatan [1]. Jambu biji dapat beradaptasi pada kisaran suhu antara 15 dan 30°C. Di luar kisaran ini, efek suhu yang lebih rendah atau lebih tinggi mengurangi pembentukan buah, sementara suhu malam hari 5 hingga 7°C menghentikan pertumbuhan. Selain itu, suhu rendah menghambat produksi, menyebabkan bunga rontok [2] dan juga dapat menimbulkan beberapa penyakit yang menyerang daun dan buah jambu biji.

Jawa Barat merupakan salah satu penghasil buah jambu biji dengan jumlah produksi pada tahun 2021 yang mencapai 692.488 kuintal, jumlah produksinya mengalami penurunan sebanyak -12.82% dibandingkan dengan jumlah produksi pada tahun 2020 yang mencapai 794.345 kuintal [3]. Pada tahun 2021 tercatat curah hujan di Jawa Barat berdasarkan 4 stasiun pengamatan pengamatan Citeko [4], stasiun pengamatan Bogor [5], stasiun pengamatan Bandung [6], dan stasiun pengamatan Jatiwangi [7] mengalami peningkatan dibanding dengan tahun 2020, sehingga dapat mengakibatkan penyakit yang dapat menyerang jambu, karena suhu ideal jambu biji berkisar antara 15 dan 30°C.

Perkembangan teknologi yang semakin maju dapat memudahkan dalam melakukan identifikasi penyakit yang terjangkit pada daun dan buah jambu, teknologi tersebut adalah *deep learning. Deep learning* adalah bagian dari *machine learning* yang berfokus pada area algoritme yang terinspirasi oleh pemahaman kita tentang bagaimana otak bekerja untuk mendapatkan pengetahuan [8]. Pada penelitian ini teknologi *deep learning* akan melakukan pembelajaran *dataset* penyakit daun dan buah jambu yang sudah diberikan label atau disebut dengan metode *supervised learning.*

Mengenali citra daun dan buah jambu yang terkena penyakit diperlukan algoritma yang dapat mengolah citra digital, yaitu *Convolutional Neural Network* (CNN) yang merupakan contoh utama *deep learning*. Mereka terinspirasi oleh bagaimana *neuron* diatur dalam korteks visual (area otak yang memproses *input* visual) [8]. Dan akan menggunakan arsitektur *MobileNetV2.*

Model deep learning yang telah dibuat akan diimplementasikan menjadi sebuah aplikasi yang dapat diakses melalui *smartphone*. Berdasarkan (*Newzoo*, 2021) pada tahun 2021 pengguna *smartphone* di Indonesia mencapai 178 juta pengguna. Kemudian yang menggunakan *smartphone* dengan sistem operasi *android* pada bulan oktober 2022 menurut (*Statista*, 2022) mencapai 90%, sehingga pada penelitian ini dibuatlah aplikasi yang berbasis sistem operasi *android.*

Ada beberapa penelitian terdahulu yang mengklasifikasi penyakit jambu dan yang menggunakan *Convolutional Neural Network* (CNN), diantaranya penelitian yang dilakukan oleh Ari Kurniawan Sudiarto, Khoirida Aelani, dan Fresa Dwi Juniar pada tahun 2020 yang berjudul "Identifikasi Penyakit pada Daun Jambu Kristal Berbasis *Android* dengan Metode *Enterprise Unified Process*". Hasil penelitian ini adalah Sample daun yang berpenyakit dan sehat akan di proses menggunakan aplikasi MATLAB untuk mengetahui nilai matriks RGB *(Red, Green, Blue)* dari sampel daun yang di proses. Kemudian nilai matriks tersebut menjadi acuan untuk aplikasi *android* yang akan menggunakan bahasa pemrograman *JAVA.* Aplikasi *android* akan merubah foto yang diambil menjadi nilai matriks kemudian nilai matriks tersebut akan disandingkan dengan nilai matriks daun berpenyakit dan daun sehat dan akan memberikan output berupa jenis penyakit dan cara penanganannya [11].

Penelitian yang dilakukan oleh Umar Ghoni, Emmie Fatkhunnajah, Helmi Saputra, Anisatun Cahyani pada tahun 2021 yang berjudul "Deteksi Penyakit Daun Pada Citra Daun Jambu Biji Menggunakan Segmentasi Warna". Hasil penelitian ini adalah Sample daun berpenyakit dilakukan proses *preprocessing* untuk mengambil nilai a dan nilai b menggunakan *cielab*, kemudian hasil dari nilai tersebut dimasukan ke algoritma *k-means* untuk mengelompokkan citra daun yang sehat dan berpenyakit [12].

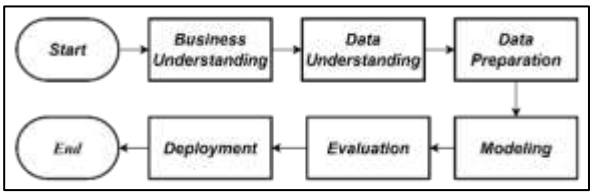
Penelitian yang dilakukan oleh Putri Teresia Ompusunggu pada tahun 2022 yang berjudul "Klasifikasi Penyakit Tanaman Pada Daun Kentang Dengan Metode *Convolutional Neural Network* Arsitektur *MobileNet*". Hasil penelitian ini adalah Dataset dibagi menjadi *train,* validasi, dan *test.* Kemudian akan dilakukan *training model* menggunakan algoritma *Convolutional Neural Network* dan arsitektur *MobileNet*. Akan diukur performanya dengan *confusion matrix* dengan melihat nilai akurasi, *loss, recall*, dan *f1-score* [13].

Penelitian yang dilakukan oleh Mirza Faturrachman, Indra Yustiana, Somantri pada tahun 2022 yang berjudul "Sistem Pendeteksi Penyakit Pada Daun Tanaman Singkong Menggunakan *Deep Learning* Dan *Tensorflow* Berbasis *Android*". Hasil dari penelitian ini

adalah *Dataset* akan dilakukan proses augmentasi terlebih dahulu dan akan dibagi sebanyak 90% untuk data *train* dan 10% untuk *testing*. Kemudian akan dilakukan proses *train model* menggunakan metode *k-fold cross validation*. Selanjutnya *model* akan di konversi menadi file *tensorflow lite*. file *tensorflow lite* akan di *deploy* menjadi aplikasi *android* dengan bahasa pemrograman kotlin, proses klasifikasinya dengan mengambil gambar dari galeri ataupun kamera dan akan memberikan *output* jenis penyakitnya [14].

Tujuan penelitian ini adalah mengembangkan aplikasi *mobile* yang di jalankan pada sistem operasi *android*. Aplikasi tersebut dapat mengenali jenis penyakit yang menyerang daun dan buah jambu, serta akan memberikan informasi untuk menangani penyakit yang menyerang, dengan demikian akan memudahkan pengguna untuk mengambil tindakan pencegahan atau pengobatan yang tepat, sehingga diharapkan akan meningkatkan produktifitas hasil panen dan mengurangi kerugian yang diakibatkan oleh penyakit yang menyerang.

2. Metode Penelitian



Gambar 1. Metode Penelitian

Perancangan *model deep learning* pada penelitian ini menggunakan metode *Cross Industry Standard Process for Data Mining* (CRISP-DM). CRISP-DM terdiri dari 6 tahapan. Tahap pertama, yaitu *business understanding*. *Business understanding* perlu dievaluasi untuk mendapatkan gambaran umum tentang sumber daya yang tersedia dan yang dibutuhkan. [15]. Kedua, *data understanding*. Mengumpulkan data dari sumber data, mengeksplorasi dan mendeskripsikannya, serta memeriksa kualitas data [15]. Ketiga, *data preparation*. Seleksi data harus dilakukan dengan menentukan kriteria inklusi dan eksklusi. Kualitas data yang buruk dapat ditangani dengan membersihkan data [15]. Keempat, *modeling.* Fase ini terdiri dari pemilihan teknik *modeling*, membangun kasus uji dan *model*. Semua teknik data *mining* dapat digunakan. Secara umum, pilihannya tergantung pada masalah bisnis dan data. Yang lebih penting adalah bagaimana menjelaskan pilihan tersebut[15]. Kelima, *evaluation.* Dalam tahap evaluasi, hasilnya diperiksa sesuai dengan *business understanding* yang telah ditetapkan [15]. Pada tahap ini akan evaluasi dengan melihat *training graph* dan menghitung hasil *confusion matrix*. Keenam, *deployment.* Pada fase *deployment* hasilnya bisa berupa laporan akhir ataupun berbentuk *software* (perangkat lunak / aplikasi) [15]. Berikut ini rumus untuk menghitung akurasi pada *confusion matrix.*

$$Accuracy = \frac{Jumlah\ true\ positive}{Jumlah\ data\ validasi} \quad (1)$$

2.1 *Deep learning*

Deep learning adalah bagian dari *machine learning* yang berfokus pada area algoritma yang terinspirasi oleh pemahaman kita tentang bagaimana otak bekerja untuk mendapatkan pengetahuan. *Deep learning* dibangun berdasarkan ide *artificial neural network* dan peningkatannya untuk dapat mengonsumsi data dalam jumlah besar dengan memperdalam *neural network* dengan cara tertentu. [8].

2.2 *Convolutional Neural Network*

Convolutional Neural Network (CNN), merupakan contoh utama *deep learning*. Mereka terinspirasi oleh bagaimana neuron diatur dalam korteks visual (area otak yang memproses input visual) [8]. *Convolutional Neural Network* memiliki beberapa tahapan yaitu *Convolutional Layers, Pooling Layers, Fully Connected Layers.* CNN didesain untuk menangani *input* dengan bentuk 2 dimensi [16].

2.3 *MobileNetV2*

MobileNet mempresentasikan *model* yang efisien untuk aplikasi *mobile* dan *embedded system*. *MobileNet* didasarkan pada arsitektur ramping yang menggunakan konvolusi yang dapat dipisahkan berdasarkan kedalaman untuk membangun jaringan saraf dalam yang ringan [17].

2.4 *Overfitting*

Overfitting terjadi ketika data uji dan data pelatihan berhenti meningkat pada saat yang sama. Hal ini disebabkan karena *model* yang terlalu kompleks mengalami kesulitan untuk menangani potongan-potongan informasi dalam *set* pengujian, yang mungkin berbeda dengan yang ada di *set* pelatihan. Di sisi lain, *model* yang terlalu *fit* cenderung menghafal semua data, termasuk *noise* yang tidak dapat dihindari pada pelatihan [18].

2.5 *Dropout*

Dropout merupakan teknik regularisasi jaringan syaraf dimana beberapa *neuron* akan dipilih secara acak dan tidak akan digunakan selama proses pelatihan *model* jaringan. *Neuron-neuron* tersebut akan diabaikan secara acak. Hal tersebut akan membuat *neuron* yang diabaikan akan dihentikan sementara oleh *model* jaringan. Selain itu, hasil bobot yang baru tidak akan diterapkan pada *neuron* tersebut. *Dropout* akan melakukan proses mencegah terjadinya *overfitting* [19].

2.6 *Kernel regularizers*

Kernel regularizers akan meminimalkan bobot fitur yang memiliki pengaruh kecil pada klasifikasi akhir. Terdapat dua macam *kernel regularizers*, yang pertama L1 *regularizers* akan menghapus beberapa fitur pada *model* sehingga akan mendapatkan *model* yang lebih

sederhana, namun akan kehilangan fitur-fitur pada *model*, yang kedua L2 *regularizers* tidak akan mengurangi fitur pada *model*, namun akan mempelajari fitur-fitur dengan memberikan bobot (*weights*) yang lebih rendah, jadi *model* akan memiliki informasi lebih banyak dibandingkan L1 *regularizers* [18].

2.7 *Android*

Android adalah sistem operasi berbasis Linux bagi telepon seluler seperti telepon pintar dan komputer tablet. Android juga menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri yang akan digunakan untuk berbagai macam piranti gerak [20].

2.8 *Kotlin*

Kotlin adalah bahasa pemrograman yang dikembangkan oleh JetBrains, perusahaan yang juga mengembangkan IDE *Android Studio*. Bahasa Kotlin adalah pengembangan dari bahasa *Java* yang sudah popular sebelumnya[21].

3. Hasil dan Pembahasan

Hasil dan pembahasan pada penelitian ini menggunakan metode *Cross Industry Standard Process for Data Mining* (CRISP-DM). CRISP-DM adalah metode yang pas pada penelitian ini, karena memiliki langkah yang terstruktur, dimulai dari pemahaman masalah yang ada pada tahap *business understanding*, kemudian tahap persiapan data hingga aktivitas pembuatan *model* yang ada pada tahapan *data understanding, data preparation, modeling,* dan *evaluation*, kemudian hingga tahap implementasi aplikasi yang ada pada tahap *deployment*. Berikut ini uraian hasil dan pembahasan pada penelitian ini.

3.1 *Business Understanding*

Penelitian ini bertujuan untuk membuat sebuah aplikasi *mobile* yang dapat melakukan klasifikasi penyakit pada daun dan buah jambu dengan metode *supervised learning* yang menggunakan algoritma *Convolutional Neural Network* dan arsitektur *MobileNetV2*. Aplikasi akan melakukan pengenalan penyakit pada daun dan buah jambu dengan cara mengambil gambar dengan menggunakan kamera *handphone*, kemudian akan menghasilkan output jenis penyakit dari gambar yang telah diambil.

3.2 *Data Understanding*

Data yang digunakan berupa gambar penyakit pada daun dan buah jambu yang didapatkan dari *website Kaggle*, yang merupakan *platform* untuk kolaborasi dalam bidang *data science* dan *machine learning*, *Kaggle* menyediakan berbagai *dataset*, seperti gambar ataupun, file CSV, file JSON, dan jenis *dataset* lainnya yang siap digunakan. Gambar *dataset* ini memiliki ukuran piksel yang beragam, *dataset* memiliki 6 buah *class*/label. Berikut ini *class* tersebut:

Tabel 1. *Class dataset*

Class	Definisi	Sumber
Jambu	Buah jambu sehat	https://www.kaggle.com/datasets/aelchimminut/fruits262
Daun jambu	Daun jambu sehat	
Phytophthora	Penyakit *phytophthora* / busuk pada buah jambu	
Red rust	Penyakit *red rust* / daun karat merah	https://www.kaggle.com/datasets/mhantor/guava-disease
Scab	Penyakit scab / kudis buah jambu	
Stylar end rot	Penyakit stylar end rot / busuk pangkal buah jambu	
Dot	Penyakit *dot* / daun jambu berlubang	https://www.kaggle.com/datasets/omkarmanohardalvi/guava-disease-dataset-4-types

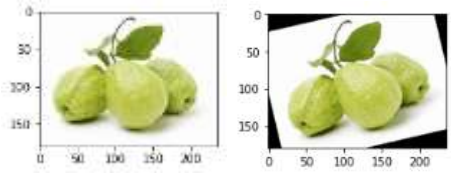
3.3 *Data Preparation*

Sebelum masuk ke proses *modeling*, *dataset* akan melewati proses *data augmentation*. Teknik augmentasi data dapat dikembangkan untuk meningkatkan variasi data menjadi lebih banyak tanpa harus mencari data baru. Pada tahap ini, augmentasi citra dilakukan untuk setiap data. Selain augmentasi, setiap data juga diubah ukurannya agar sesuai dengan bentuk *input* yang dibutuhkan oleh *model* [22]. Berikut ini *data augmentation* yang diterapkan pada penelitian ini:

Pertama, *split dataset*. Dilakukan untuk membagi *dataset* untuk kebutuhan *training* dan validasi. Pada penelitian ini sebesar 80% dari *dataset* untuk kebutuhan *training*, sehingga *model* dapat belajar pola-pola yang ada pada *dataset*. Sementara itu, 20% sisanya dialokasikan untuk kebutuhan validasi, yang akan membantu evaluasi sejauh mana model dapat mengenali data validasi yang tidak ada pada data *training*.

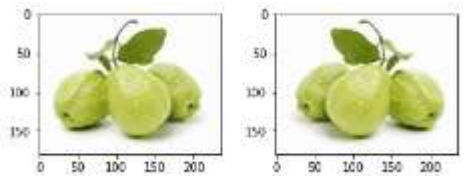
Kedua, *rescale*. Dilakukan untuk merubah skala citra *dataset*, agar semua gambar memiliki skala yang sama. Skala gambar disesuaikan dengan ukuran maksimum *input* yang dibutuhkan oleh arsitektur *MobileNetV2*, yaitu 224 piksel. Pada penelitian ini dilakukan *rescale* sebesar 224 piksel.

Ketiga, *Rotation*. Dilakukan untuk melakukan rotasi gambar. Pada penelitian ini dilakukan rotasi sebesar 20 derajat, sehingga dapat menambah variasi sudut pandang gambar pada *dataset*. Berikut ini gambar 2 yang merupakan gambar sebelum dan sesudah dilakukan *rotation*.

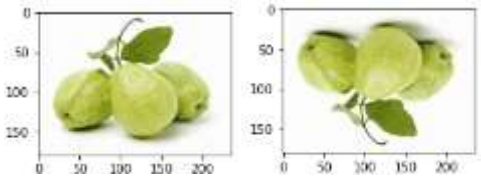


Gambar 2. Hasil *rotation*

Keempat, *horizontal* dan *vertical flip*. Dilakukan untuk membalik citra secara *horizontal* dan *vertical.* Berikut ini gambar 3 dan gambar 4 yang merupakan gambar sebelum dan sesudah dilakukan *horizontal* dan *vertical flip.*



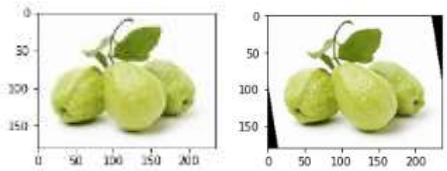
Gambar 3. Hasil *horizontal flip*



Gambar 4. Hasil *vertical flip*

Kelima, *Zoom*. Dilakukan untuk memperbesar gambar. Diharapkan setelah gambar diperbesar maka akan membantu *model* untuk memahami objek dalam skala yang berbeda. Pada penelitian ini dilakukan *zoom* sebesar 20%.

Keenam, *Shear*. Dilakukan untuk menggeser gambar sepanjang sumbu agar menciptakan sudut pandang yang berbeda. Pada penelitian ini dilakukan *shear* sebesar 20 derajat. Berikut ini gambar 5 yang merupakan gambar sebelum dan setelah dilakukan *shear*.



Gambar 5. Hasil *shear*

Ketujuh, *fill_mode nearest*. *Fill mode* dibutuhkan untuk mengisi piksel yang kosong setelah sebelumnya dilakukan proses *rotation* dan *shear* yang menyebabkan piksel ada yang tidak terisi, terlihat jelas pada gambar 2 dan gambar 5. *Fill mode nearest* akan mengisi nilai piksel yang kosong dengan cara mengambil nilai piksel terdekat.

Setelah melakukan augmentasi data, selanjutnya akan dilakukan split *dataset* untuk memisahkan antara *train* dan *validation*. Pada penelitian ini sebanyak 80% data akan dijadikan data *training*, sementara 20% sisanya akan dijadikan data *validation.* Berikut ini Gambar 3.6 menyatakan hasil split data dari dataset *model* buah jambu, dan gambar 3.7 menyatakan hasil *split dataset model* daun jambu.

```
Found 349 images belonging to 4 classes.
Found 84 images belonging to 4 classes.
```

Gambar 6. Hasil *split dataset* buah jambu

```
Found 256 images belonging to 3 classes.
Found 63 images belonging to 3 classes.
```

Gambar 7. Hasil *split dataset* daun jambu

3.4 *Modeling*

Tahap *modeling* terbagi menjadi dua bagian, yaitu proses *modeling* yang merupakan tahapan membuat *model*, dan penentuan parameter *modeling* yang merupakan tahap menentukan parameter *model* dan akan membandingkan parameter mana yang memiliki performa paling baik. Berikut ini dua bagian pada *modeling.* Tahap membuat *Model*. Yang dilakukan pertama pada proses ini adalah membuat *base model* yang menggunakan arsitektur *MobileNetV2* dan menambahkan beberapa parameter pada *base model*. Berikut ini uraian penjelasan parameter pada *base model:*

Tabel 2. Penjelasan parameter *base model*

Parameter	Penjelasan
Input_shape	*B*ernilai 224,224,3 yang berarti ukuran piksel 224×224 dengan 3 *channel* warna RGB (*Red, Green, Blue*).
Include_top	Bernilai *false*, berfungsi untuk tidak perlu membawa *prediction node* bawaan dari *MobileNetV2* karena memiliki nilai 1000 kelas, sementara pada penelitian ini tidak mencapai 1000 kelas.
Weights	Bernilai '*imagenet'*, berfungsi untuk bobot dari *layer* yang sudah dilatih pada *library imagenet.*

Selanjutnya membuat *sequential model.* Fungsi dari *sequential model* adalah memungkinkan untuk menambahkan beberapa *layer* yang dibutuhkan. Pada penelitian ini menambahkan *dropout layer*, *GlobalAveragePooling2D layer*, dan *dense layer*. Berikut ini penjelasan parameter *sequential model* pada tabel 3.

Tabel 3. Penjelasan parameter *sequential model*

Parameter	Penjelasan
Base_model	Variabel *base model* yang berisi arsitektur *MobileNetV2*
Dropout	*Dropout* akan menghilangkan *neuron* secara acak untuk memperkecil *overfitting.*
GlobalAveragePooling2D	Akan mengambil nilai rata-rata dari dimensi data yang telah di*input.*
Dense	*Dense* berfungsi untuk menambahkan *fully-connected layer*, yang memiliki kelas 4 buah sesuai kelas dari penyakit pada buah jambu biji, dan 3 kelas sesuai dengan jumlah kelas dari penyakit pada dun jambu biji. Selanjutnya aktivasi *softmax* yang berfungsi untuk mengambil hasil *output* dengan probabilitas paling tinggi yang akan dijadikan sebagai prediksi.

Selanjutnya melakukan *compile model* menggunakan *adam optimizer*, untuk *loss function* menggunakan *categorical_crossentropy* karena memiliki lebih dari dua kelas, dan *metrics* evaluasi menggunakan *accuracy.* Langkah selanjutnya memulai proses *training* dengan menggunakan *model fit.* Disini akan

menambahkan dataset *train* dan *validation* yang telah disimpan pada variabel *train_generator* yang telah dibuat pada tahap *split dataset*, kemudian menentukan jumlah iterasi *training* atau *epochs*, dan berapa banyak *dataset* yang akan diambil setiap sekali *epochs* atau disebut *batch size.*

Tahap menentukan parameter *modeling*. Pada saat melakukan *training model* harus menentukan parameter dalam *model* yang akan menentukan seberapa baiknya *model* tersebut. Parameter yang dimaksud adalah penentuan nilai *dropout*, jumlah *epoch*, penentuan nilai *kernel regularizers* yang ada pada saat membuat *sequential model*. Penentuan parameter berfungsi untuk membandingkan parameter mana yang memiliki performa terbaik yang akan digunakan pada *model.*

Parameter *modeling* yang pertama adalah *dropout*. Dari hasil perbandingan pada tabel 4 dan tabel 5 terlihat model daun jambu memiliki performa yang sangat baik walaupun tanpa menggunakan dropout, sementara model buah jambu terlihat nilai training loss dan validation loss memiliki selesih yang jauh diantara keduanya. Berikut ini perbandingan nilai *dropout* yang digunakan pada *model* buah jambu dan daun jambu yang sama-sama menggunakan 20 *epochs* pada tabel 4 dan tabel 5.

Tabel 4. Perbandingan *dropout model* buah jambu

Dropout	*Training accuracy*	*Validation accuracy*	*Training loss*	*Validation loss*
0	0.977	0.880	0.115	0.313
0.1	0.971	0.892	0.121	0.313
0.2	0.974	0.904	0.114	0.246

Tabel 5. Perbandingan *dropout model* daun jambu

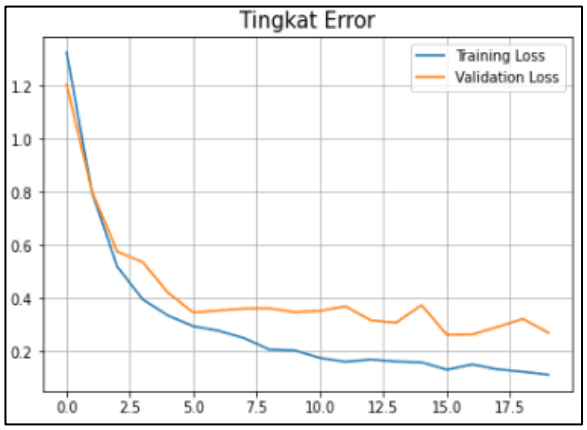
Dropout	*Training accuracy*	*Validation accuracy*	*Training loss*	*Validation loss*
0	0.996	1.0	0.032	0.031
0.1	0.996	1.0	0.037	0.029
0.2	0.996	1.0	0.027	0.027

Parameter *modeling* yang kedua adalah jumlah *epoch*. Untuk *model* daun jambu sudah memiliki nilai *training loss* dan *validation loss* yang sangat baik meskipun hanya menggunakan 20 *epoch*, sehingga pada perbandingan jumlah *epoch* hanya akan membandingkan jumlah *epoch* dari model *buah* jambu, berikut ini tabel 6 hasil perbandingan *epoch model* buah jambu dengan menggunakan *dropout* sebesar 0.2.

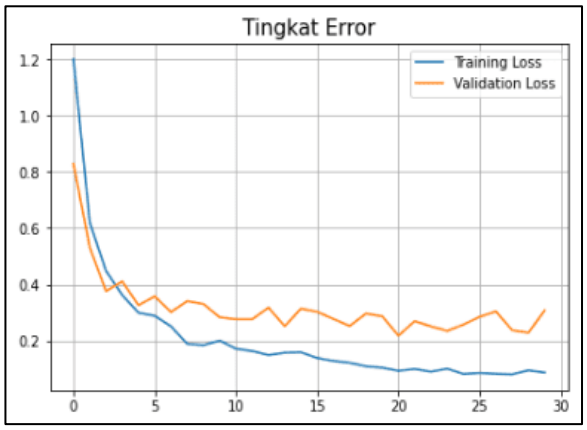
Tabel 6. Perbandingan *epoch model* buah jambu

Epoch	*Training accuracy*	*Validation accuracy*	*Training loss*	*Validation loss*
20	0.974	0.904	0.114	0.246
30	0.988	0.916	0.073	0.257
40	0.982	0.904	0.063	0.232

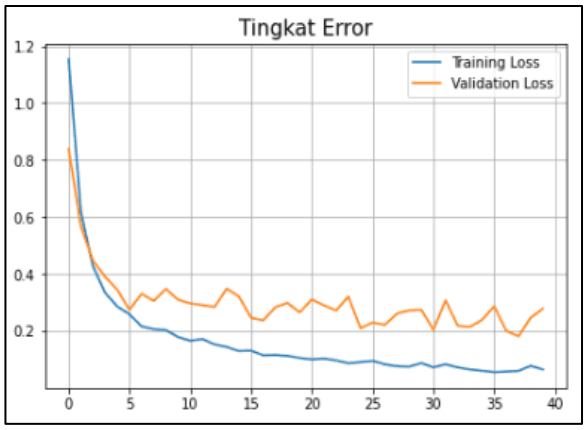
Dapat dilihat pada hasil tabel 6 bahwa masih memiliki selisih yang jauh antara *training loss* dan *validation loss.* Dapat disimpulkan bahwa hasil *model evaluation* masih memiliki nilai *overfitting* yang masih tinggi. Untuk gambaran yang lebih jelas dapat dilihat pada *training graph* tingkat error pada gambar 8, gambar 9, dan gambar 10 berikut ini.



Gambar 8. *Graph loss* 20 *epoch*

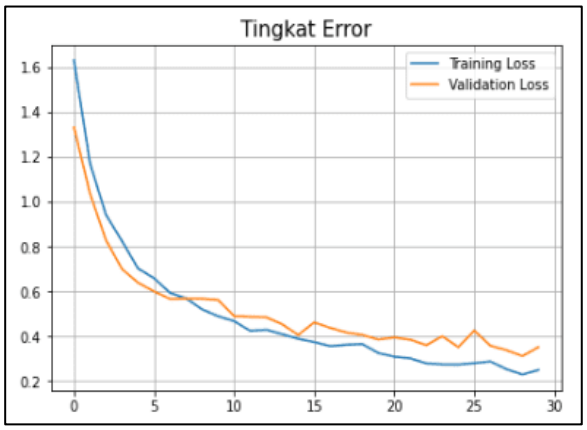


Gambar 9. *Graph loss* 30 *epoch*

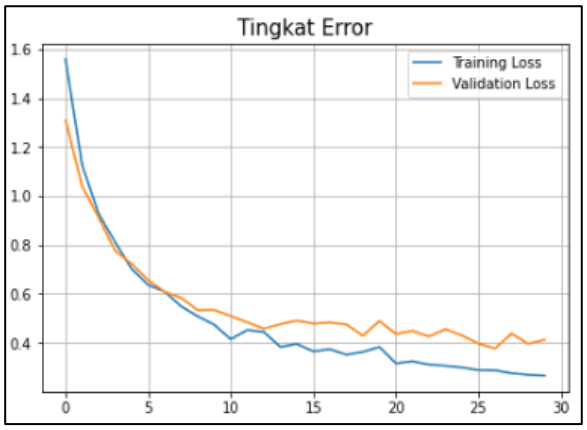


Gambar 10. *Graph loss* 40 *epoch*

Parameter *modeling* yang ketiga adalah *kernel regularizers* yang berfungsi untuk mengurangi *overfitting* pada *mode* buah jambu. Pada penelitian ini menggunakan *kernel regularizers* L2 yang akan memberikan bobot (*weights)* yang rendah. Pada penelitian ini menggunakan *kernel regularizers* L2 sebesar 0.01, *dropout* 0.1 – 0.2, dan *epoch* sebanyak 30. Berikut ini gambar 11 dan gambar 12 perbandingan *graph loss kernel regulariizers* yang menggunakan *dropout* 0.1 dan 0.2.



Gambar 11. *Graph loss kernel regularizers dropout* 0.1



Gambar 12. *Graph loss kernel regularizers, dropout* 0.2

Dari hasil *graph loss* pada gambar 11 dan gambar 12 terlihat *overfitting* berhasil dikurangi. Lebih detailnya akan terlihat dari hasil *model evaluation* pada gambar 13 dan gambar 14 yang menyatakan bahwa *model* yang menggunakan *dropout* sebesar 0.2 memiliki hasil yang lebih baik, dengan nilai *training accuracy* sebesar 98.8%, *validation accuracy* sebesar 91.6%, *training loss* sebesar 19.1%, dan *validation loss* sebesar 38.6%. Berikut ini hasil *model evaluation* setelah menggunakan *kernel regularizers.*

```
11/11 [==============================] - 37s 3s/step - loss: 0.1989 - acc: 0.9857
3/3 [==============================] - 9s 3s/step - loss: 0.4255 - acc: 0.8929
Accuracy (train):  0.9856733679771423
Accuracy (validation): 0.8928571343421936
Loss (train):  0.19891613721847534
Loss (validation): 0.42553257942199707
```

Gambar 13. *Model evaluation kernel regularizers, dropout* 0.1

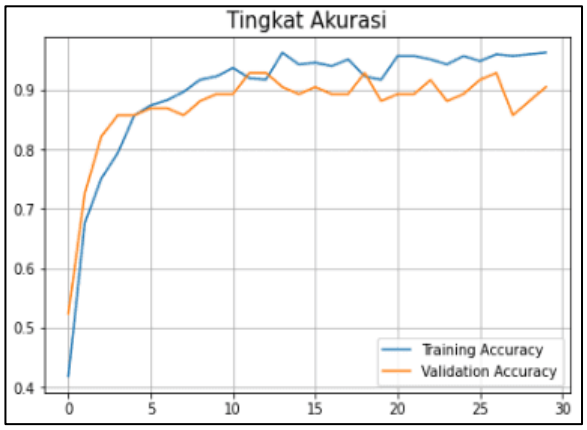
```
11/11 [==============================] - 16s 1s/step - loss: 0.1915 - acc: 0.9885
3/3 [==============================] - 5s 1s/step - loss: 0.3868 - acc: 0.9167
Accuracy (train):  0.9885386824607849
Accuracy (validation): 0.9166666865348816
Loss (train):  0.19153253734111786
Loss (validation): 0.38675835728645325
```

Gambar 14. *Model evaluation kernel regularizers, dropout* 0.2

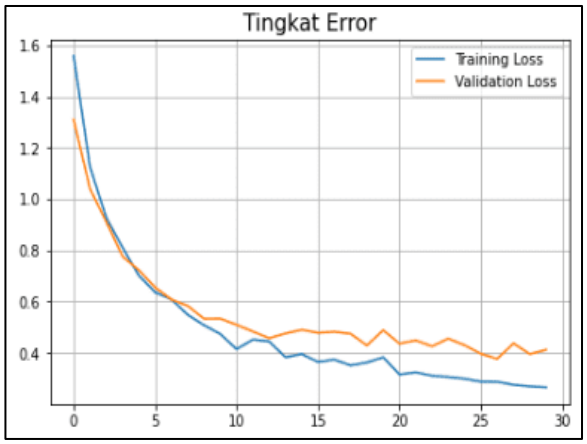
Berdasarkan hasil model evaluation pada gambar 13 dan 14 peneliti memutuskan akan menggunakan *model* yang menggunakan *dropout* 0.2. Model yang menggunakan dropout 0.2 memiliki *validation accuracy* yang lebih baik dan memiliki *validation loss* yang lebih rendah. Selanjutnya *model* yang terpilih akan melewati tahapan *evaluation.*

3.5 *Evaluation*

Pada tahap evaluasi akan dilakukan evaluasi *model* dengan melihat hasil *training graph* dan *confusion matrix.* Berikut ini hasil evaluasi kedua *model.* Pertama, evaluasi *model* buah jambu yang menggunakan 30 *epoch*, 0.2 *dropout*, dan *kernel regularizers* L2 sebesar 0.01.



Gambar 15. *Training graph* tingkat akurasi *model* buah jambu

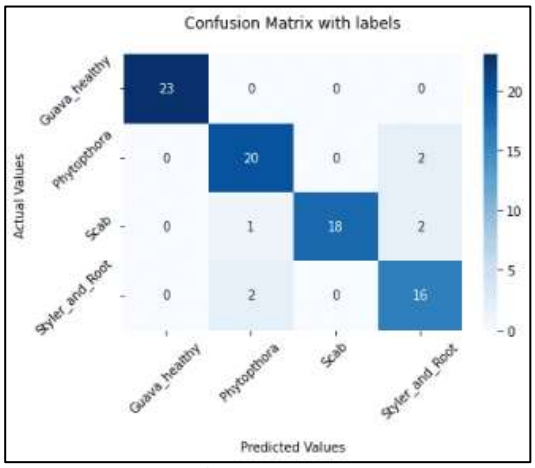


Gambar 16. *Training graph* tingkat *loss model* buah jambu

Dari hasil *training graph* tingkat *loss* pada gambar 16 dapat dilihat bahwa *model* berhasil dikurangi *overfitting* setelah menambahkan *kernel regularizers.* Sehingga bisa dikatan bahwa *kernel regularizers* adalah cara yang ampuh untuk mengurangi *overfitting.* Lebih detailnya dapat dilihat pada gambar 17 hasil *model evaluation* berikut ini.

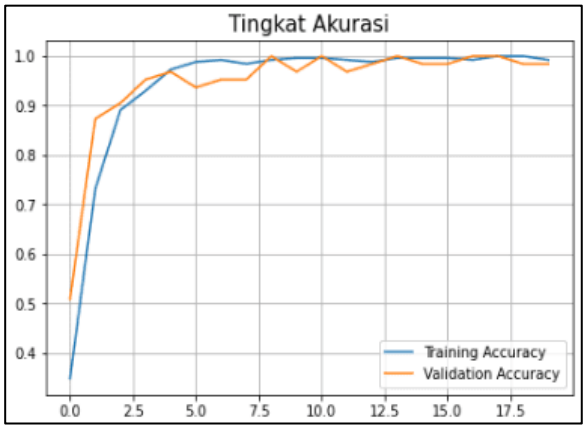
Gambar 17. *Model evaluation model* buah jambu

Dari gambar 17 *training evaluation* diatas menyatakan bahwa *model* memiliki nilai *training accuracy* 98,8% dan *validation accuracy* 91.6%, kemudian nilai *training loss* sebesar 19.1%, dan nilai *validation loss* sebesar 38.6%. Selanjutnya evaluasi *model* buah jambu dengan *confusion matrix*. Berikut ini gambar 18 hasil *confusion matrix* model buah jambu.

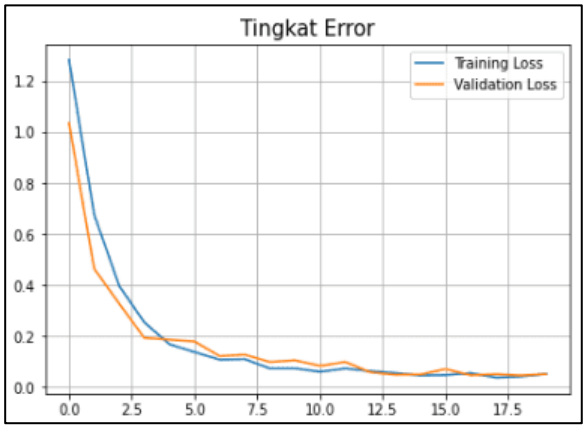


Gambar 18. *Confusion matrix model* buah jambu

Berdasarkan *confusion matrix* pada gambar 18, maka hasil akurasi *model* penyakit pada buah jambu biji dari jumlah data validasi sebanyak 84 data adalah 91,6%. Kedua, evaluasi *model* daun jambu yang hanya menggunakan 20 *epoch* dan tanpa *dropout*.

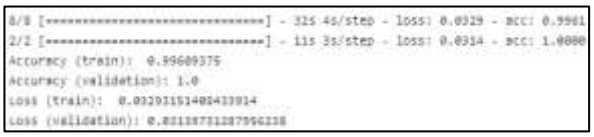


Gambar 19. *Training graph* tingkat akurasi *model* daun jambu



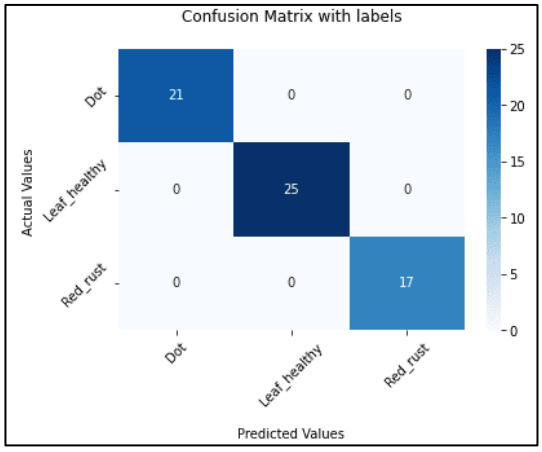
Gambar 20. *Training graph* tingkat *loss model* daun jambu

Dapat dilihat dari hasil *training graph model* daun jambu pada gambar 19 mengindikasikan bahwa *model* memiliki performa yang sangat baik. Selanjutnya pada gambar 20 dapat dilihat bahwa *model* tidak memiliki indikasi mengalami *underfitting* ataupun *overfitting*. Untuk melihat performa *model* lebih detail, berikut ini hasil *model evaluation* untuk model daun jambu.



Gambar 21. *Model evaluation model* daun jambu

Dari gambar 21 *model evaluation model* daun jambu diatas menyatakan bahwa *model* memiliki nilai *training accuracy* 99.66% dan *validation accuracy* yang mencapai 100%, *nilai training loss* yang hanya 3,29%, dan nilai *validation loss* yang hanya 3.13%. Selanjutnya evaluasi *model* daun jambu dengan *confusion matrix*. Berikut ini gambar 22 *confusion matrix model* daun jambu yang menyatakan hasil akurasi dari jumlah data validasi sebanyak 63 data adalah sebesar 100%.



Gambar 22. *Confusion matrix model* daun jambu

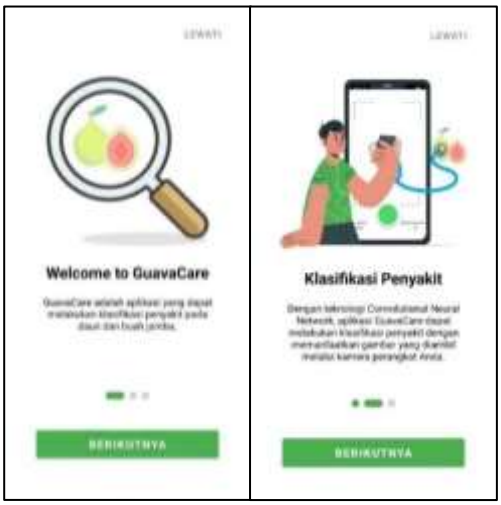
3.6 *Deployment*

Pada tahap *deployment* akan menampilkan tampilan aplikasi yang telah selesai dibuat dengan bahasa pemrograman *Kotlin*. Aplikasi ini memiliki beberapa halaman, yaitu halaman *splash*, halaman *onboarding*, halaman utama, halaman klasifikasi penyakit, dan halaman detail penyakit. Berikut ini tampilan aplikasi yang bernama "GuavaCare".

Pertama, halaman *splash*. Halaman ini merupakan tampilan awal yang akan muncul ketika aplikasi dijalankan. Pada halaman ini menampilkan logo dan nama aplikasinya. Berikut ini gambar 23 halaman *splash* pada aplikasi ini

Gambar 23. *Splashscreen*

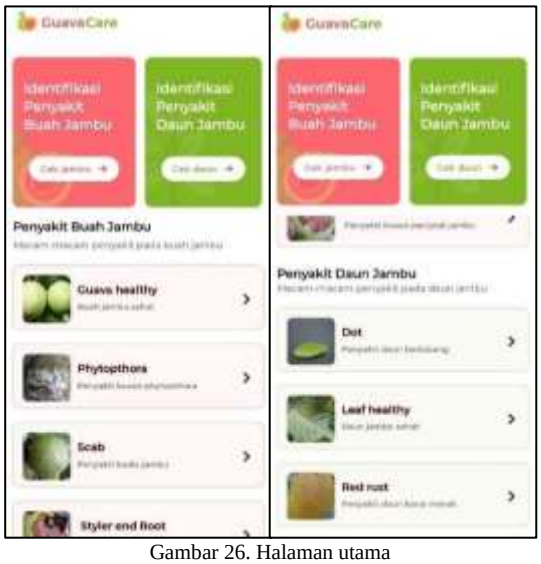
Kedua, halaman *onboarding*. Pada halaman ini akan menjelaskan fitur-fitur utama yang ada pada aplikasi. Pada aplikasi ini memiliki 3 halaman *onboarding*. Berikut ini gambar 24 dan gambar 25 halaman *onboarding* pada aplikasi ini.



Gambar 24. Halaman *onboarding* 1 dan 2

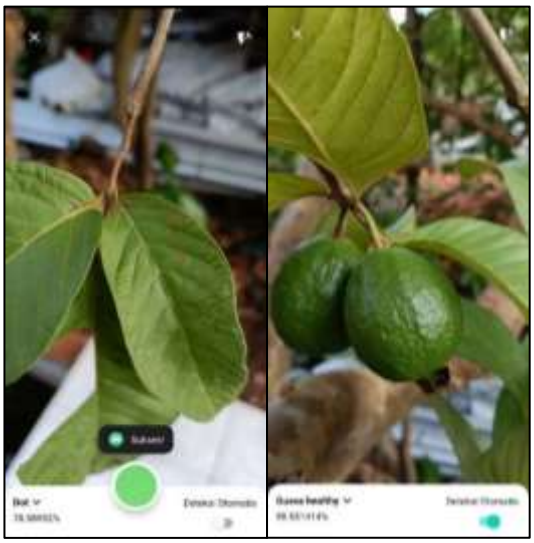
Gambar 25. Halaman *onboarding* 3

Ketiga, halaman utama. Pada halaman ini menampilkan dua buah tombol yang akan ke halaman klasifikasi penyakit pada daun dan penyakit pada buah jambu. Selanjutnya ada daftar penyakit daun dan buah jambu. Berikut ini gambar 26 halaman utama pada aplikasi ini.



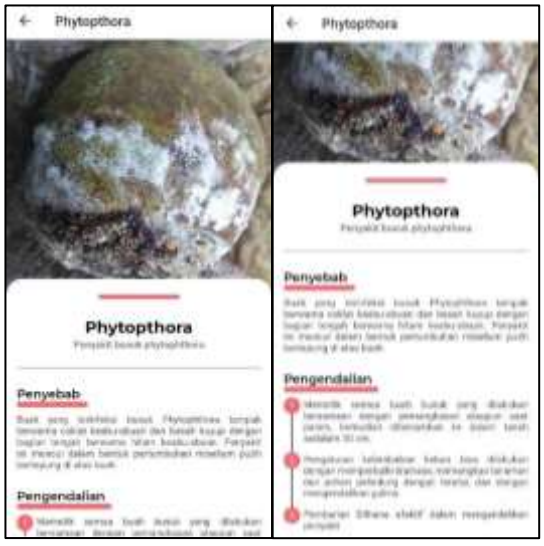
Gambar 26. Halaman utama

Keempat, halaman klasifikasi. Merupakan halaman fitur utama dari aplikasi yang dapat melakukan klasifikasi jenis penyakit pada daun dan buah jambu. Terdapat fitur *flash light* secara otomatis, manual, ataupun tidak menggunakan *flash light*. Untuk melihat hasil klasifikasi bisa secara manual dengan menekan tombol *shutter* atau bisa dengan deteksi otomatis, kemudian hasil klasifikasi dapat ditekan agar berpindah ke halaman detail penyakit.



Gambar 27. Halaman klasifikasi penyakit

Kelima, halaman detail penyakit. Pada halaman detail penyakit akan menampilkan gambar penyakit kemudian penyebab penyakit tersebut menyerang daun atau buah jambu, dan akan menjelaskan pengendalian yang bisa dilakukan dalam membasmi penyakit tersebut. Berikut ini gambar 28 halaman detail penyakit.



Gambar 28. Halaman detail penyakit

4. Kesimpulan

Dalam penelitian ini peneliti berhasil menerapkan metode *supervised learning* dengan menggunakan algoritma *Convolutional Neural Network* dan arsitektur *MobileNetV2*. Hasil *Model* daun jambu memiliki

performa yang sangat baik dan tidak *overfitting* ataupun *underfitting* dengan hasil evaluasi *model training accuracy* sebesar 99.6%, *validation accuracy* sebesar 100%, *training loss* sebesar 3.2%, dan *validation loss* sebesar 3.1%. Hasil *confusion matrix* memiliki akurasi 100% dari jumlah data validasi sebanyak 63 data.

Sedangkan *model* buah jambu berhasil dikurangi *overfitting* dengan menggunakan *dropout* 0.2 dan *kernel regularizers* L2 sebesar 0.01. Hasil evaluasi model memiliki nilai *training accuracy* 98,8% dan *validation accuracy* 91.6%, kemudian nilai *training loss* sebesar 19.1%, dan nilai *validation loss* sebesar 38.6%. Hasil *confusion matrix* model buah jambu memiliki akurasi 91.6% dari jumlah data validasi sebanyak 84 data.

Selanjutnya pada penelitian ini, peneliti berhasil menerapkan *model* yang telah dibuat menjadi aplikasi berbasis *mobile*. Aplikasi ini menggunakan bahasa pemrograman *Kotlin.* Dalam mengintegrasikan *model* ke dalam aplikasi, maka *model* yang telah dibuat dapat dengan mudah diakses oleh pengguna, sehingga aplikasi ini diharapkan dapat memiliki manfaat untuk menambah produktifitas buah jambu yang sempat menurun.

Daftar Pustaka

[1] M. Kumar *et al.*, "Guava (Psidium guajava L .) Leaves : Nutritional Composition," *Foods*, vol. 10, no. 752, pp. 1–20, 2021, doi: https://doi.org/10.3390/foods10040752.

[2] G. Fischer and L. M. Melgarejo, "Ecophysiological aspects of guava (Psidium guajava L.). A review," *Rev. Colomb. Ciencias Hortícolas*, vol. 15, no. 2, pp. 0–3, 2021, doi: 10.17584/rcch.2021v15i2.12355.

[3] Badan Pusat Statistik Provinsi Jawa Barat, "Produksi Hortikultura Buah dan Sayur Tahunan Provinsi Jawa Barat 2021," *BPS Provinsi Jawa Barat*, pp. 1–128, 2022, [Online]. Available: https://jabar.bps.go.id/publication/2022/12/23/1bb94ee2b41974c0e1cb3ab8/produksi-hortikultura-buah-dan-sayur-tahunan-provinsi-jawa-barat-2021.html

[4] BPS Jawa Barat, "Curah Hujan di Stasiun Pengamatan Meteorologi Citeko Menurut Bulan (mm), 2019-2021," Jul. 25, 2022. https://jabar.bps.go.id/indicator/151/433/1/curah-hujan-di-stasiun-pengamatan-meteorologi-citeko-menurut-bulan.html (accessed Mar. 06, 2023).

[5] BPS Jawa Barat, "Curah Hujan di Stasiun Pengamatan Klimatologi Bogor Menurut Bulan (mm), 2019-2022," Jul. 25, 2022. https://jabar.bps.go.id/indicator/151/430/1/-curah-hujan-di-stasiun-pengamatan-klimatologi-bogor-menurut-bulan.html (accessed Mar. 06, 2023).

[6] BPS Jawa Barat, "Pengamatan Curah Hujan di Stasiun Pengamatan Geofisika Bandung Menurut Bulan (mm), 2019-2021," Jul. 25, 2022. https://jabar.bps.go.id/indicator/151/425/1/pengamatan-curah-hujan-di-stasiun-pengamatan-geofisika-bandung-menurut-bulan.html (accessed Mar. 06, 2023).

[7] BPS Jawa Barat, "Jumlah Curah Hujan di Stasiun Pengamatan Meteorologi Jatiwangi Menurut Bulan (mm), 2019-2021," Jul. 25, 2022. https://jabar.bps.go.id/indicator/151/438/1/jumlah-curah-hujan-di-stasiun-pengamatan-meteorologi-jatiwangi-menurut-bulan.html (accessed Mar. 06, 2023).

[8] T. Amaratunga, *Deep Learning on Windows*. 2021. doi: 10.1007/978-1-4842-6431-7.

[9] "Top Countries/Markets by Smartphone Penetration & Users | Newzoo." https://newzoo.com/insights/rankings/top-countries-by-smartphone-penetration-and-users (accessed Nov. 30, 2022).

[10] "Indonesia: mobile OS share 2022 | Statista." https://www.statista.com/statistics/262205/market-share-held-by-mobile-operating-systems-in-indonesia/ (accessed Nov. 30, 2022).

[11] A. K. Sudiarto, K. Aelani, and F. Dwi Juniar, "Identifikasi Penyakit pada Daun Jambu Kristal Berbasis Android dengan Metode Enterprise Unified Process," *Jt. (Journal Inf. Technol.*, vol. 2, no. 1, pp. 1–8, 2020.

[12] Umar, "Deteksi Penyakit Daun Pada Citra Daun Jambu Biji Menggunakan Segmentasi Warna," *J. Tek. Inform. dan Sist. Inf.*, vol. 1, no. 1, pp. 23–30, 2021, [Online]. Available: https://jurtisi.stmikmpb.ac.id/index.php/home/article/view/8

[13] P. T. Ompusunggu, "Klasifikasi Penyakit Tanaman Pada Daun Kentang Dengan Metode Convolutional Neural Network Arsitektur Mobilenet," *J. Syntax Fusion*, Vol. 33, No. 1, Pp. 1–12, 2022, Doi: Https://Doi.Org/10.54543/Fusion.V2i09.217.

[14] M. Fatturachman, I. Yustiana, and Somantri, "Sistem Pendeteksi Penyakit Pada Daun Tanaman Singkong Menggunakan Deep Learning Dan Tensorflow Berbasis Android," *IJIS - Indones. J. Inf. Syst.*, vol. 7, no. 2, pp. 176–184, 2022, doi: 10.36549/ijis.v7i2.225.

[15] C. Schröer, F. Kruse, and J. M. Gómez, "A systematic literature review on applying CRISP-DM process model," *Procedia Comput. Sci.*, vol. 181, no. 2019, pp. 526–534, 2021, doi: 10.1016/j.procs.2021.01.199.

[16] M. F. Naufal *et al.*, "Klasifikasi Citra Game Batu Kertas Gunting Menggunakan Convolutional Neural Network," *Techno.Com*, vol. 20, no. 1, pp. 166–174, 2021, doi: 10.33633/tc.v20i1.4273.

[17] A. G. Howard *et al.*, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," 2017, [Online]. Available: http://arxiv.org/abs/1704.04861

[18] X. Ying, "An Overview of Overfitting and its Solutions," *J. Phys. Conf. Ser.*, vol. 1168, no. 2, 2019, doi: 10.1088/1742-6596/1168/2/022022.

[19] T. Dwi Antoko, M. Azhar Ridani, and A. Eko Minarno, "Klasifikasi Buah Zaitun Menggunakan Convolution Neural Network," *Komputika J. Sist. Komput.*, vol. 10, no. 2, pp. 119–126, 2021, doi: 10.34010/komputika.v10i2.4475.

[20] B. S. Sulastio, H. Anggono, and A. D. Putra, "Sistem Informasi Geografis Untuk Menentukan Lokasi Rawan Macet Di Jam Kerja Pada Kota Bandarlampung Pada Berbasis Android," *J. Teknol. dan Sist. Inf.*, vol. 2, no. 1, pp. 104–111, 2021, [Online]. Available: http://jim.teknokrat.ac.id/index.php/JTSI

[21] N. S. Sibarani, G. Munawar, and B. Wisnuadhi, "Analisis Performa Aplikasi Android Pada Bahasa Pemrograman Java dan Kotlin.," *Ind. Res. Work. Natl. Semin.*, no. July, 2018.

[22] M. Ilhamsyah and U. Enri, "Identification of Bacterial Spot Diseases on Paprika Leaves Using Cnn and Transfer Learning," *J. Pilar Nusa Mandiri*, vol. 18, no. 1, pp. 17–24, 2022, doi: 10.33480/pilar.v18i1.2755.

Halaman ini sengaja dikosongkan