BAB I
PENDAHULUAN

1.1 Latar Belakang

Indonesia merupakan negara kepulauan yang membentang pada koordinat 6° LU - 11° LS dan 94° BT – 141° BT, dengan kekayaan sumber daya alam yang melimpah. Sebagai negara agraris, Indonesia menghasilkan beraneka ragam produk pertanian dan perkebunan yang menjadi tulang punggung perekonomian masyarakat. Sektor pertanian tidak hanya berperan sebagai sumber mata pencaharian penduduk, tetapi juga menjadi penggerak perekonomian nasional, terbukti dengan tingginya daya saing komoditas pertanian Indonesia di pasar Internasional. Tidak hanya itu, sektor pertanian juga berkontribusi signifikan terhadap kebutuhan pokok manusia, khususnya pangan, dengan menyumbang sekitar 17,3% terhadap Produk Domestik Bruto (PDB). Hal ini menunjukkan bahwa sektor ini memegang peranan vital dalam perekonomian negara (Djamalu et al., 2019).

Febrialdi (2020) menjelaskan bahwa sektor pertanian yang menjanjikan ini didukung oleh subsektor holtikultura yang berperan vital dalam penyediaan pangan dan peningkatan kesejahteraan masyarakat. Sebagai negara tropis, Indonesia dikaruniai keragaman flora yang mengagumkan, mencakup kurang lebih 28.000 jenis tumbuhan yang mewakili sepersepuluh dari seluruh spesies tumbuhan di dunia. Di antara ribuan spesies tersebut, terdapat sekitar 6.000 jenis yang telah diketahui memiliki nilai guna bagi kehidupan manusia (Nurjannah et al., 2024: 14), termasuk di dalamnya familia *Solanaceae* dikenal sebagai suku terung-terungan yang mencakup 83 genus dengan 2.925 spesies yang mampu beradaptasi di berbagai zona iklim (Mardhiyah et al., 2024).

Komoditas holtikultura tropis Indonesia tidak hanya memenuhi kebutuhan domestik tetapi juga menjadi sumber devisa negara melalui ekspor. Kentang, tomat, dan paprika sebagai anggota familia ini terus menjadi fokus pengembangan sektor pertanian Indonesia untuk mendukung ketahanan pangan nasional sekaligus berkontribusi signifikan terhadap pendapatan petani dan perekonomian nasional secara keseluruhan (Kasenda et al., 2019).

Sigitta et al., (2023) Berdasarkan data Badan Pusat Statistika (BPS), produktivitas tanaman tomat menunjukkan peningkatan yang menggembirakan dengan kenaikan 24% dalam empat tahun terakhir hingga 2019. Sementara itu, produksi kentang menunjukkan pencapaian tertinggi di tahun 2022 sebesar
n mengalami penurunan sebesar 17% menjadi 1.248.513 ton
l serupa terjadi pada paprika yang mencatatkan hasil tertinggi
an 20.997 ton, dengan produktivitas 28,95 ton per hektar.

tanaman ini menunjukkan tren produktivitas yang positif,
s dalam familia *Solanaceae* menciptakan tantangan tersendiri

bagi para petani dalam mengidentifikasi penyakit yang menyerang tanaman-tanaman dalam kelompok ini. Proses identifikasi penyakit pada daun menjadi langkah kritis untuk mencegah kerugian ekonomi dalam produksi pertanian (Bastian et al., 2023). Pada tanaman kentang, penyakit utama yang sering dijumpai meliputi *late blight* dan *early blight* (Amatullah et al., 2021). Sementara pada paprika, *bacterial spot* menjadi ancaman serius (Ilhamsyah & Enri, 2022). Tanaman tomat menghadapi spektrum penyakit yang lebih luas, mencakup *bacterial spot*, *late blight*, *early blight*, *leaf mold*, *septoria leaf spot*, *spider mites*, *target spot*, *yellow leaf curl virus*, dan *mosaic virus* (Anavyanto et al., 2023).

Serangan penyakit pada daun tidak hanya mengancam kesehatan tanaman secara keseluruhan tetapi juga berdampak signifikan pada hasil panen. Meskipun setiap penyakit memiliki gejala khas, proses identifikasi sering kali menjadi rumit karena banyak penyakit menunjukkan gejala yang serupa (Halim et al., 2023). Tantangan ini dipersulit oleh fakta bahwa beberapa gejala dapat tampak hampir identik bagi mata yang belum terlatih, membuat proses diagnosis manual menjadi tidak akurat (Bastian et al., 2023).

Identifikasi manual penyakit tanaman merupakan tugas yang menantang karena melibatkan analisis data yang kompleks dan memakan waktu. Variasi dalam tingkat pengetahuan dan keterampilan di antara petani juga menambah kompleksitas dalam proses identifikasi visual. Akibatnya diagnosis yang tidak tepat dapat terjadi, yang pada gilirannya dapat mengakibatkan penanganan yang tidak efektif (Bastian et al., 2023). Kondisi ini menegaskan kembali pentingnya pemahaman mendalam tentang keragaman dalam familia *Solanaceae* dan karakteristik penyakit yang menyerangnya, menciptakan siklus pembelajaran yang berkelanjutan bagi para petani (Dewi et al., 2015).

Menghadapi kompleksitas tantangan dalam identifikasi manual tersebut, kemajuan teknologi, khususnya dalam bidang kecerdasan buatan, membuka era baru dalam penanganan masalah pertanian. Perkembangan teknologi *deep learning* telah memungkinkan identifikasi penyakit tanaman menjadi lebih akurat dan efisien melalui pengolahan data visual secara otomatis (Taufiqurrahman et al., 2024). Inovasi ini menjadi jawaban atas tantangan yang dihadapi petani dalam mengidentifikasi penyakit tanaman secara manual.

Dalam konteks ini, *Convolutional Neural Network* (CNN) muncul sebagai metode unggulan yang mampu mengekstrak fitur penting dari gambar tanaman yang terinfeksi. Teknologi ini dapat mendeteksi perubahan warna, bercak, dan tekstur abnormal pada daun dengan tingkat akurasi yang tinggi. CNN bekerja dengan mensegmentasi gambar menjadi bagian-bagian kecil, memungkinkan analisis karakteristik visual yang mungkin tidak terlihat oleh mata Suhendar, 2024).

gi ini ke dalam aplikasi *mobile* membuka peluang baru bagi an diagnosis penyakit tanaman secara *real-time* di lapangan. kasi berbasis CNN memungkinkan petani mengidentifikasi anya dengan mengunggah foto menggunakan *smartphone* solusi yang praktis dan mudah di akses. Kemajuan teknologi

ini tidak hanya mempercepat proses identifikasi penyakit, tetapi juga membuka era baru dalam pengelolaan pertanian yang lebih efisien (Taufiqurrahman et al., 2024).

Beberapa penelitian terdahulu telah menunjukkan efektivitas CNN dalam deteksi penyakit tanaman *Solanaceae*. Santosa et al. (2023) mengeksplorasi penggunaan model ResNet34 dengan *optimizer* Adam dan *learning rate* 0.0001 untuk klasifikasi penyakit daun kentang dan mencapai akurasi 99% dalam mengidentifikasi penyakit *late blight* dan *early blight*. Dalam konteks yang berbeda, (Nugraha et al., 2023) menerapkan CNN untuk klasifikasi penyakit daun paprika dengan *dataset* 9.892 citra, mencapai tingkat akurasi *training* 97% dalam membedakan daun yang terinfeksi *bacterial spot* dan daun sehat.

Penelitian oleh Rafif et al. (2022) mengkaji efektivitas berbagai model *pre-trained* dan teknik *augmentasi*. Meskipun diterapkan pada objek yang berbeda, hasil penelitian ini membuktikan keunggulan arsitektur *EfficientNetV2-L* yang mencapai akurasi 97,8% pada klasifikasi sekrup 6 kelas dan penggunaan teknik *augmentasi blur* dapat meningkatkan performa hingga 81%.

Perkembangan implementasi CNN dalam aplikasi *mobile* secara khusus didemonstrasikan oleh Taufiqurrahman et al. (2024) yang menggunakan arsitektur *EfficientNet-B0*. Penelitian tersebut berhasil mencapai akurasi 99,55% dalam mendeteksi penyakit daun tomat, dengan waktu deteksi yang efisien diantara 0,150 hingga 0,554 detik. Penggunaan Tensorflow *Lite* memungkinkan model dijalankan pada perangkat *mobile* secara *real-time*, memberikan solusi praktis bagi petani dalam mendiagnosis penyakit tanaman.

Meskipun penelitian-penelitian tersebut telah menunjukkan keberhasilan dalam implementasi *deep learning*, mayoritas penelitian masih terfokus pada satu jenis tanaman spesifik. Hal ini menciptakan keterbatasan dalam aplikasi praktis di lapangan, mengingat petani seringkali menanam berbagai jenis tanaman.

Penelitian ini menghadirkan pendekatan baru dengan mengintegrasikan deteksi penyakit untuk tiga tanaman familia *Solanaceae* (kentang, tomat, dan paprika) dalam satu model *EfficientNetV2-L* yang masih jarang diimplementasikan dalam konteks deteksi penyakit tanaman. Penelitian ini menerapkan dua skenario *augmentasi* yang berbeda yaitu penggunaan teknik *augmentasi blur* saja dan kombinasi beberapa teknik *augmentasi* (*flip*, rotasi, dan *color adjustment*). Penerapan berbagai teknik *augmentasi* ini dikombinasikan dengan variasi *input size* yang berbeda untuk memberikan perspektif baru dalam pengembangan model yang lebih efisien dan akurat. Integrasi model ke dalam aplikasi *mobile* lebih lanjut meningkatkan aksesibilitas dan kepraktisan penggunaan di lapangan. Pendekatan komprehensif ini menawarkan solusi yang lebih lengkap dibandingkan penelitian-penelitian

umnya terfragmentasi pada satu jenis tanaman.

ah

1. Seberapa efektif penerapan *augmentasi blur* dibandingkan dengan *augmentasi* kombinasi (*flip*, rotasi, *color adjustment*) dalam meningkatkan akurasi model?
2. Bagaimana pengaruh variasi *input size* (100x100, 256x256, dan 300x300 piksel) terhadap performa model?
3. Bagaimana mengintegrasikan model *deep learning* dalam aplikasi *mobile* untuk mendeteksi penyakit daun tanaman kentang, tomat, dan paprika, dan memberikan rekomendasi penanganan yang tepat?

1.3 Tujuan dan Manfaat

Tujuan penelitian ini meliputi:

1. Menganalisis efektivitas *augmentasi blur* dibandingkan *augmentasi* kombinasi dalam meningkatkan akurasi model.
2. Menganalisis pengaruh variasi *input size* pada masing-masing jenis *augmentasi* terhadap performa model.
3. Mengintegrasikan model *deep learning* ke dalam platform *mobile* agar dapat digunakan secara praktis di lapangan.

Sementara, manfaat dari penelitian ini adalah sebagai berikut:

1. Menyediakan analisis perbandingan yang komprehensif antara *augmentasi blur* dan *augmentasi* kombinasi untuk penerapan *transfer learning* pada kasus deteksi penyakit daun tanaman *Solanaceae* menggunakan *EfficientNetV2-L*.
2. Memberikan panduan dalam menentukan *input size* yang optimal untuk masing-masing jenis *augmentasi* sehingga dapat mencapai performa yang terbaik.
3. Mempermudah petani dalam mendeteksi penyakit daun pada tanaman kentang, tomat, dan paprika secara cepat dan akurat melalui *mobile*, sehingga dapat mencegah kerugian dan meningkatkan produktivitas pertanian.

1.4 Batasan Masalah

1. Penelitian ini dibatasi pada penggunaan beberapa teknik *augmentasi* tertentu yaitu *blur*, *flip*, rotasi, dan *color adjustment*, tidak mencakup seluruh
ntasi yang ada.
i terbatas pada deteksi penyakit daun dan pembangunan
k tiga jenis tanaman familia *Solanaceae*: tomat, kentang, dan

ng diuji terbatas pada ukuran 100x100, 256x256, dan 300x300
ngujian dilakukan pada satu arsitektur model yang sama.

1.5 Teori

1.5.1 Tanaman Solanaceae

Familia *Solanaceae* atau suku terung-terungan merupakan salah satu kelompok tumbuhan berbunga yang memiliki peran vital dalam memenuhi berbagai kebutuhan manusia. Tanaman dalam familia ini umumnya berbentuk herba atau perdu, dengan beberapa spesies tumbuh sebagai pohon kecil, baik tegak maupun merambat.

Familia *Solanaceae* memiliki nilai ekonomi yang tinggi karena menghasilkan beragam komoditas penting. Dalam bidang pangan, familia ini berkontribusi melalui sayuran dan buah-buahan seperti kentang (*Solanum tuberosum*), tomat (*Lycopersicon lycopersicum Linn*.), terung (*Solanum sp*.), cabai dan paprika (*Capsicum sp*.). Beberapa genus seperti *Physalis* menghasilkan buah-buahan yang dapat dikonsumsi, termasuk di antaranya *cape gooseberry* dan *jamberry*. Selain sebagai sumber pangan, beberapa spesies di familia ini juga dimanfaatkan sebagai tanaman hias seperti *Petunia*, bahan pengobatan, serta menghasilkan alkaloid yang bernilai komersial seperti yang ditemukan pada genus *Nicotiana* dan *Datura* (Mardhiyah et al., 2024).

1.5.2 Penyakit dan Rekomendasi Penanganan pada Daun Tanaman Solanaceae

Penyakit tanaman dapat muncul ketika terjadi interaksi antara tiga faktor utama: patogen (mikroorganisme parasit), tanaman inang, dan kondisi lingkungan yang mendukung (Ankardiansyah et al., 2024). Kerusakan pada tanaman umumnya terjadi ketika patogen menginfeksi bagian-bagian tubuh tanaman seperti daun, batang, dan akar. Patogen yang umum menyerang tanaman termasuk jamur, bakteri, dan virus. Berikut penjelasan mengenai penyakit daun yang menjadi fokus penelitian ini:

1. Bercak bakteri pada paprika (*Pepper bell bacterial spot*)

daun *pepper bell bacterial spot*
9. Kaggle. https://www.kaggle.com/*datasets*/emmarex

oleh bakteri *Xanthomonas campestris pv. Vesicatoria*.

- Gejala pada daun: Bercak kecil berwarna cokelat gelap atau hitam dengan tepi kekuningan yang dapat membesar seperti terlihat pada Gambar 1 dan menyebabkan kematian jaringan (nekrosis), hingga akhirnya daun rontok.
- Penanganan: Pemangkasan bagian daun yang terinfeksi dan aplikasikan Streptocycline 0,01% + Blitox 50 WP 0,25% (Prashad et al., 2011).

2. Paprika sehat (*Pepper bell healthy*)

Gambar 2. Penyakit daun *pepper bell healthy*
Emmanuel, T. O. 2019. Kaggle. https://www.kaggle.com/*datasets*/emmarex/plantdisease

- Seperti ditunjukkan pada Gambar 2, kondisi fisik daun berwarna hijau cerah tanpa bercak, lubang, atau warna kekuningan. Tidak terdapat tanda-tanda layu, keriting, atau kering di tepi.

3. Layu awal pada kentang (*Potato early blight*)

Gambar 3. Penyakit daun *potato early blight*
Emmanuel, T. O. 2019. Kaggle. https://www.kaggle.com/*datasets*/emmarex/plantdisease

disebabkan oleh jamur *Alternaria solani* yang ditampilkan pada
nenunjukkan pola infeksi dari patogen ini.
a daun: Bercak cokelat berbentuk bulat atau oval dengan tepi
cak dapat berkembang menjadi lesi lebih besar dengan bagian
gering dan menghitam, menyebabkan daun menguning hingga

- Penanganan: Pemantauan rutin saat tanaman mencapai tinggi 30 cm dan aplikasikan fungisida berbahan fluopyram (Luna Tranquility, Luna Pro, Velum Rise) (Crop, 2024).

4. Layu akhir pada kentang (*Potato late blight*)

Gambar 4. Penyakit daun *potato late blight*
Emmanuel, T. O. 2019. Kaggle. https://www.kaggle.com/*datasets*/emmarex /plantdisease

- Gambar 4 menampilkan dampak dari jamur *Phytopthora infestans* pada tanaman.
- Gejala pada daun: Bercak hijau kekuningan atau abu-abu dengan tepi berair yang berkembang menjadi lesi cokelat besar. Terdapat lapisan putih seperti jamur di bagian bawah daun yang merupakan spora, menyebabkan daun menguning, mengering, dan rontok dengan cepat.
- Penanganan: Pemangkasan bagian terinfeksi dan pencegahan dengan fungisida klorotalonil (Fung-Onil) (Sarah, n.d.).

5. Kentang sehat (*Potato healthy*)

Gambar 5. Penyakit daun *potato healthy*
2019. Kaggle. https://www.kaggle.com/*datasets*/emmarex

ın sehat pada Gambar 5 dapat diidentifikasi dari permukaan arna hijau cerah, tebal, dan tidak menunjukkan tanda atau kerusakan.

ada tomat (*Tomato bacterial spot*)

Gambar 6. Penyakit daun *tomato bacterial spot*
Emmanuel, T. O. 2019. Kaggle. https://www.kaggle.com/*dataset*s/emmarex/plantdisease

- Disebabkan oleh bakteri *Xanthomonas campestris pv. vesicatoria*. Berkembang pada suhu 23,9–30°C dengan curah hujan tinggi.
- Gejala pada daun: Bercak cokelat kehitaman kecil dikelilingi area kuning pada daun, dapat diamati dengan jelas pada Gambar 6.
- Penanganan: Penyiraman pagi hari, irigasi tetes, dan aplikasikan fungisida tembaga (Matt Gibson, n.d.).

7. Layu awal pada tomat (*Tomato early blight*)

Gambar 7. Penyakit daun *tomato early blight*
Emmanuel, T. O. 2019. Kaggle. https://www.kaggle.com/*dataset*s/emmarex/plantdisease

- Kerusakan akibat jamur *Alternaria tomatophila* yang ditampilkan pada Gambar 7.
- Gejala pada daun: Bercak cokelat melingkar dengan pola konsentris (lingkaran target) pada daun tua.

n: Penggunaan mulsa dan rotasi tanaman, penanaman dalam untuk kontrol kelembapan lebih dianjurkan (Jenna, 2023a).

tomat (*Tomato late blight*)

Gambar 8. Penyakit daun *tomato late blight*
Emmanuel, T. O. 2019. Kaggle. https://www.kaggle.com/*dataset*s/emmarex /plantdisease

- Disebabkan oleh jamur *Oomycete* p*hytopthora infens* (jamur air). Biasanya muncul selama periode basah yang panjang dan menyebar melalui air.
- Gambar 8 memperlihatkan kondisi daun dengan gejala lesi hijau keabu-abuan yang meluas menjadi bercak cokelat kehitaman, dengan lapisan spora putih di bawah daun.
- Penanganan: Hindari penyiraman dari atas, pemangkasan rutin, dan aplikasikan fungisida (Jenna, 2023b).

9. Jamur daun tomat (*Tomato leaf mold*)

Gambar 9. Penyakit daun *tomato leaf mold*
Emmanuel, T. O. 2019. Kaggle. https://www.kaggle.com/*dataset*s/emmarex /plantdisease

- Seperti yang diilustrasikan pada Gambar 9, daun ini terkena jamur *Passalora fulva*.
- Gejala pada daun: Bercak kuning di permukaan atas daun dengan lapisan
  ru cokelat keabu-abuan di bawahnya.
  n: Kurangi kelembapan dengan penyiraman pagi hari dan
  gisida kalsium klorida (Matt Gibson, n.d.).
  otoria pada tomat (*Tomato septoria leaf spot*)

Gambar 10. Penyakit daun *tomato septoria leaf spot*
Emmanuel, T. O. 2019. Kaggle. https://www.kaggle.com/*dataset*s/emmarex /plantdisease

- Disebabkan oleh jamur *Septoria lycopersici*. Umumnya, menyerang daun lebih tua dan menyebar melalui percikan air dalam kondisi lembap.
- Gejala pada daun: Bercak abu-abu bertepi hitam pada daun tua yang dapat dilihat pada Gambar 10.
- Penanganan: Aplikasikan fungisida organik seperti *Bacillus subtilis* dan menjaga lingkungan tetap kering (Glenn, 2024).

11. Tungau laba-laba pada tomat (*Tomato two-spotted spider mite*)

Gambar 11. Penyakit daun *tomato two-spotted spider mite*
Emmanuel, T. O. 2019. Kaggle. https://www.kaggle.com/*dataset*s/emmarex /plantdisease

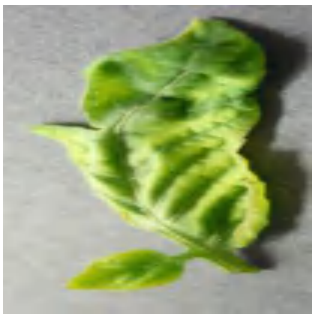
- Gambar 11 menampilkan akibat dari tungau *Tetranychus urticae* yang berkembang pada cuaca panas dan kering.
- Gejala pada daun: Daun menguning/keperakan, terdapat jaring laba-laba di bawah daun, daun mengering dan gugur pada serangan berat.
- Penanganan: Semprotan air bertekanan untuk membersihkan tungau, minyak neem sebagai pestisida alami (*Spider Mites On Tomato ıtify, Kill, & Prevent*, 2021).

pada tomat (*Tomato target spot*)

Gambar 12. Penyakit daun *tomato target spot*
Emmanuel, T. O. 2019. Kaggle. https://www.kaggle.com/*datasets*/emmarex/plantdisease

- Data visual Gambar 12 menunjukkan daun yang terkena jamur *Corynespora cassiicola*.
- Gejala pada daun: Bercak melingkar konsentris seperti target pada daun dan buah, daun menguning dan gugur.
- Penanganan: Pembersihan sisa tanaman, pembuangan tanaman terinfeksi, dan aplikasikan preventif (Mary, 2022).

13. Virus keriting daun kuning tomat (*Tomato yellow leaf curl virus*)



Gambar 13. Penyakit daun *tomato yellow leaf curl virus*
Emmanuel, T. O. 2019. Kaggle. https://www.kaggle.com/*datasets*/emmarex/plantdisease

- Disebabkan virus *Tomato yellow leaf curl virus* yang disebarkan oleh kutu kebul (*Bemisia tabaci*).
- Gejala pada daun: Daun muda menggulung, kerut, dan menguning. Karakteristik akibat virus tersebut terlihat pada Gambar 13.

Menjaga drainase dengan baik, penyiraman konsisten, dan
tin (Huan, 2023).
da tomat (*Tomato mosaic virus*)

Gambar 14. Penyakit daun *tomato mosaic virus*
Emmanuel, T. O. 2019. Kaggle. https://www.kaggle.com/*datasets*/emmarex/plantdisease

- Disebabkan virus *Tobamovirus*.
- Gambar 14 menampilkan infeksi virus tersebut dengan gejala pada daun berupa pola mosaik hijau tua-muda pada daun, daun melintir dan kerut, serta kerdil.
- Penanganan: Sterilisasi peralatan, pemusnahan tanaman terinfeksi, pengendalian serangga vektor (Amy, 2022).

15. Tomat sehat (*Tomato healthy*)

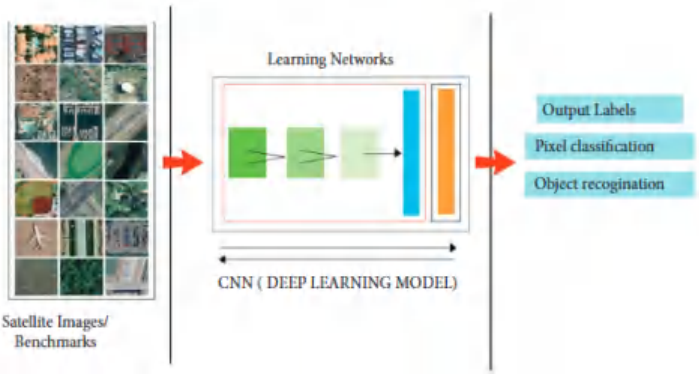
Gambar 15. Penyakit daun *tomato healthy*
Emmanuel, T. O. 2019. Kaggle. https://www.kaggle.com/*datasets*/emmarex/plantdisease

- Kondisi kesehatan daun yang ideal terlihat pada Gambar 15 dengan ciri fisik: Daun hijau segar, tidak ada bercak atau kekuningan, tidak layu/menguning.

1.5.3 Image Classification

dalah teknik *machine learning* untuk mengelompokkan citra ke tentu berdasarkan fitur yang terkandung di dalamnya. Proses ek dalam gambar menjadi nilai numerik yang dapat dikenali nungkinkan pengelompokkan otomatis ke dalam kategori yang rti tumbuhan atau objek lainnya (Suwitono & Kaunang, 2022).

Secara umum, *image classification* terdiri dari tiga tahap utama sebagaimana terlihat pada Gambar 16: *Preprocessing*, *feature extraction*, dan *classification*.

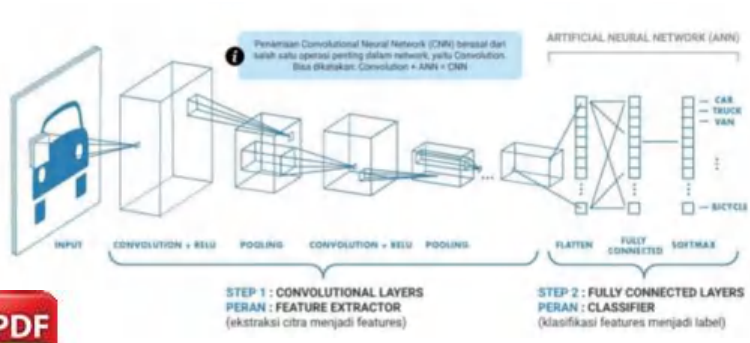


Gambar 16. *Image Classification*
Mehmod et al., 2022. Hindawi. https://onlinelibrary.wiley.com/doi/epdf/10.1155/2022/5880959

1.5.4 Convolutional Neural Network

Convolutional Neural Network adalah jenis jaringan saraf tiruan yang merupakan pengembangan dari *Multi-layer Perceptron* (MLP), dirancang khusus untuk mengolah data berbentuk *grid* seperti citra dua dimensi. CNN meniru cara kerja otak manusia dalam memproses informasi visual, membuatnya efektif dalam mengenali pola dan fitur dalam gambar melalui metode pembelajaran terawasi (*supervised learning*).

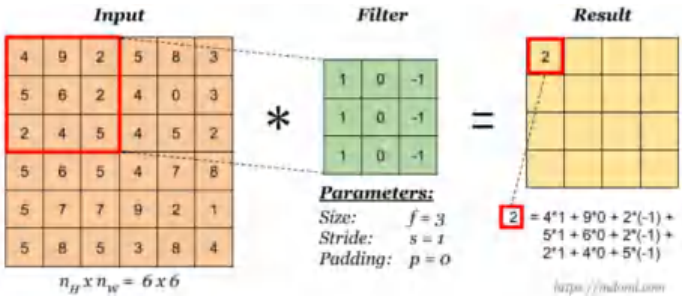
CNN telah banyak diaplikasikan dalam pengenalan wajah, klasifikasi objek, dan analisis citra medis karena kemampuannya dalam mengenali fitur kompleks secara otomatis dari data gambar yang besar (Pangestu & Bunyamin, 2018).



tional Neural Network
//www.mathworks.com/discovery/convolutional-neural-network.

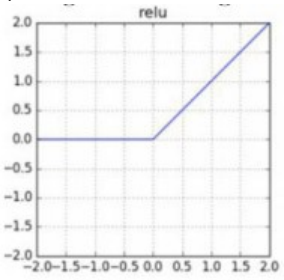
Dalam arsitektur CNN, terdapat dua komponen utama seperti yang ditunjukkan pada Gambar 17. Komponen pertama adalah bagian pembelajaran fitur yang meliputi tiga proses, yaitu:

a. Lapisan konvolusi: Lapisan pertama dalam arsitektur CNN yang bertugas mendeteksi fitur penting dalam gambar. Proses konvolusi dilakukan dengan menggerakkan filter (kernel) yang berisi bobot dari sudut kiri ke kanan bawah gambar. Seperti yang ditunjukkan pada Gambar 18, ketika kernel (kotak hijau) diaplikasikan pada gambar input (kotak jingga), akan menghasilkan *feature map* (kotak kuning) yang menunjukkan kehadiran fitur-fitur tersebut. Output yang dihasilkan berupa transformasi linear dari proses konvolusi ini (Trisiawan & Yuliza, 2022).



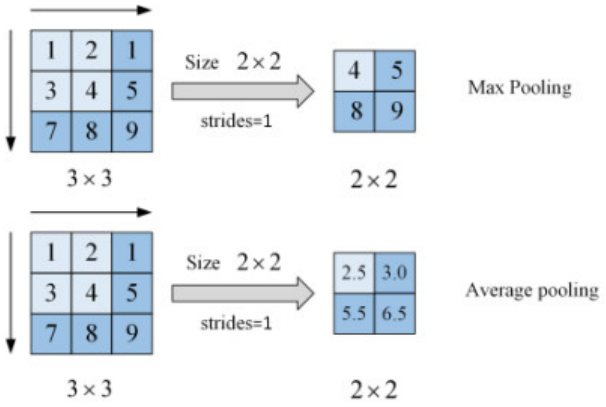
Gambar 18. Lapisan konvolusi
Prijono, B. 2018. IndoML.com. https://indoml.com/2018/03/07/student-notes-convolutional-neural-networks-cnn-introduction/

b. *Rectified Linear Unit* (ReLU): Fungsi aktivasi non-linear yang berperan dalam meningkatkan representasi model. ReLU memiliki karakteristik sederhana: jika input bernilai negatif maka output adalah 0, sedangkan jika input bernilai positif maka output sama dengan nilai inputnya. Dengan formula: jika x ≤ 0 maka x = 0, dan jika x > 0 maka x = x. Gambar 19 mengilustrasikan mekanisme aktivasi ReLU yang mengubah input negatif menjadi 0 dan mempertahankan nilai positif (Mustakim & Hayati, 2021).



l Linear Unit
. Jurnal JTIK. https://journal.lembagakita.org/index.php/jtik/

c. Lapisan *pooling*: Metode untuk mengurangi ukuran matriks melalui operasi *down-sampling*, yang membagi *output layer* konvolusi menjadi beberapa *grid* kecil. Metode ini memastikan fitur yang didapatkan tetap sama meskipun objek gambar mengalami pergeseran, sekaligus mengurangi jumlah parameter pada fitur. Terdapat dua jenis *pooling* yang umum digunakan (Anhar & Putra, 2023):
   - *Max pooling*: Mengambil nilai maksimal dari setiap *grid*.
   - *Average pooling*: Mengambil nilai rata-rata dari setiap *grid* .



Gambar 20. *Pooling*
Chen et al., 2021. Remote Sensing. https://www.mdpi.com/2072-4292/13/22/4712

Seperti ditunjukkan pada Gambar 20, proses *pooling* menggunakan filter (2x2) yang bergeser sesuai nilai *stride* untuk memproses seluruh citra. *Stride* adalah parameter yang menentukan pergeseran filter pada proses *pooling* atau konvolusi. Nilai *stride* menunjukkan berapa piksel filter akan bergerak setiap kali melakukan operasi (Rizal et al., 2020). Selanjutnya, pada komponen klasifikasi terdapat tiga proses, yaitu:

a. Lapisan *flatten*: *Flatten layer* berperan mengubah *output* dari lapisan konvolusi menjadi vektor satu dimensi. Lapisan ini mengkonversi matriks n-dimensi hasil *pooling layer* menjadi format 1-dimensi yang diperlukan sebelum data dapat diproses oleh *fully connected layer*. Proses ini penting karena setelah data melewati lapisan konvolusi dan *pooling* yang mengidentifikasi fitur dan mengurangi dimensi, *output*nya masih berbentuk multidimensi (Anhar & Putra, 2023).

b. Lapisan *fully connected*: Lapisan di mana semua neuron dari lapisan ıubung dengan neuron di lapisan berikutnya. Sebelum masuk *ıtput* dari lapisan konvolusi harus ditransformasi menjadi satu utamanya adalah menyatukan semua *node* dan menentukan berkorelasi dengan kelas tertentu (Kholik, 2021).
ı yang menghitung probabilitas untuk setiap kelas dalam kelas. Biasanya digunakan di akhir *fully connected layer* pada menghasilkan nilai probabilitas antara 0–1 untuk setiap kelas,

dengan total probabilitas seluruh kelas adalah 1. Kelas dengan nilai probabilitas tertinggi akan menjadi hasil prediksi akhir (Mahmud et al., 2019).

1.5.5 Data Augmentasi

Augmentasi data adalah strategi yang memungkinkan praktisi untuk meningkatkan keragaman data yang tersedia untuk model pelatihan tanpa harus mengumpulkan data baru secara langsung (Sanjaya & Ayub, 2020). *Dataset* yang besar sangat mempengaruhi kinerja CNN.

Augmentasi data membantu dengan menambahkan data citra melalui teknik transformasi geometris dan penerapan warna, sehingga mengurangi *overfitting* (Maharana et al., 2022). Teknik ini meningkatkan volume, kualitas, dan keragaman data pelatihan, mempermudah ketahanan dan akurasi model. Data tambahan dihasilkan dari data yang ada atau dengan teknik baru (Mumuni & Mumuni, 2022). Berikut beberapa teknik *augmentasi* data:

a. *Mean Blur*
   Mean blur adalah teknik pengolahan citra yang menggantikan nilai piksel tengah dengan rata-rata nilai-nilai piksel di sekitarnya. *Mean blur* digunakan tidak hanya untuk memperhalus gambar tetapi juga untuk membantu dalam tugas-tugas pengenalan pola (Wang & Jing, 2024). Dalam prosesnya, *mean blur* menggunakan jendela dua dimensi (dikenal sebagai *kernel*) untuk menghitung rata-rata piksel. Ketika kernel bergerak di seluruh gambar, setiap piksel baru ditentukan dari nilai rata-rata piksel-piksel di dalam kernel (Wergifosse, n.d.).
b. Rotasi
   Rotasi adalah salah satu teknik *augmentasi* data geometris yang paling sederhana. Gambar diputar pada sudut tertentu, menggeser nilai piksel dari posisi awal ke posisi akhir yang ditentukan oleh variabel rotasi sebesar θ° terhadap garis horizontal gambar (Kurniawan & Ariatmanto, 2024).
c. *Flip*
   Flip merupakah salah satu transformasi geometris di mana gambar dapat dibalik secara horizontal atau vertikal dengan itu, meningkatkan variasi pengolahan citra berdasarkan simetri gambar asli (Maharana et al., 2022).
d. *Color adjustment*
   Pengaturan warna, seperti saturasi dan kecerahan, teknik-teknik ini meningkatkan kemampuan model dalam memahami dan memproses citra dengan variasi pencahayaan yang berbeda-beda (Maharana et al., 2022).

/2-L

alah model jaringan saraf *convolutional* yang merupakan *EfficientNet*, dirancang untuk meningkatkan efisiensi pelatihan l ini mengoptimalkan arsitektur *neural network* dengan

menyeimbangkan kedalaman, lebar, dan resolusi jaringan menggunakan teknik *scaling* yang lebih efisien. Huruf 'L' menunjukkan ini adalah versi *Large* dari model, yang menawarkan akurasi lebih tinggi dengan *trade-off* ukuran model dan waktu komputasi yang lebih besar (Sidik et al., 2023). Konsep *EfficientNetV2-L* tervisualisasi pada Gambar 21.

```
model.summary()
```

rescaling (Rescaling)	(None, 256, 256, 3)	0	input_layer[0][0]
stem_conv (Conv2D)	(None, 128, 128, 32)	864	rescaling[0][0]
stem_bn (BatchNormalization)	(None, 128, 128, 32)	128	stem_conv[0][0]
stem_activation (Activation)	(None, 128, 128, 32)	0	stem_bn[0][0]
block1a_project_conv (Conv2D)	(None, 128, 128, 32)	9,216	stem_activation[0][0]
block1a_project_bn (BatchNormalization)	(None, 128, 128, 32)	128	block1a_project_conv[…
block1a_project_activati… (Activation)	(None, 128, 128, 32)	0	block1a_project_bn[0]…
block1a_add (Add)	(None, 128, 128, 32)	0	block1a_project_activ… stem_activation[0][0]
block1b_project_conv (Conv2D)	(None, 128, 128, 32)	9,216	block1a_add[0][0]

Gambar 21. Arsitektur EfficientNetV2-L

1.5.7 Transfer Learning

Transfer learning adalah teknik yang memanfaatkan model CNN yang sudah dilatih sebelumnya (*pre-trained*) untuk menyelesaikan tugas baru yang serupa. Model ini memiliki lapisan konvolusi dan *pooling* yang lebih dalam dibanding CNN standar, sehingga mampu mengekstrak fitur visual lebih baik. *Transfer learning* terdiri dari dua komponen utama: *Feature extractor* (lapisan konvolusi dan *pooling*) dan *classifier* (*fully connected layer*) (Rosadi et al., 2021).

1.5.8 Flutter

Flutter adalah *software development kit* (SDK) dan *framework open-source* yang dikembangkan Google untuk membangun aplikasi *multiplatform* dengan satu basis kode. Ditulis dalam C, C++, dan Dart. Flutter menggunakan *Skia Graphics Engine* dan menawarkan *widget* yang dapat disesuaikan untuk membangun antarmuka (Setiawan et al., 2022).

amework machine learning yang dikembangkan Google pada
en dan pengembangan model *deep learning* dalam skala besar.

Framework ini mendukung *training* menggunakan GPU, memiliki kemampuan diferensiasi otomatis dan dapat digunakan sebagai *backend* untuk Keras. Tensorflow memungkinkan eksperimen dengan jaringan neural tanpa perlu mendefinisikan banyak perhitungan kunci (Pangestu & Bunyamin, 2018).

1.5.10 Hyperparameter

Hyperparameter adalah parameter yang tidak dipelajari oleh model secara langsung, melainkan harus diatur sebelum pelatihan.

- *Learning rate*: Hyperparameter yang sangat berpengaruh terhadap performa model CNN (Mahmud et al., 2019).
- Batch size: Jumlah sampel data yang diproses dalam satu kali pelatihan *neural network*.
- *Epoch*: Satu siklus pengulangan proses pembelajaran CNN pada seluruh data pelatihan. Proses pembelajaran dilakukan berulang kali untuk mencapai nilai eror dan akurasi yang optimal.
- Optimizer: Algoritma optimasi yang bertujuan untuk meminimalkan kesalahan dan menemukan bobot optimal guna memaksimalkan tingkat akurasi (Nuariputri et al., 2023).

1.5.11 Evaluasi Model

Evaluasi model klasifikasi dilakukan menggunakan *classification report*, yang menghasilkan metrik-metrik evaluasi seperti *accuracy*, *precision*, *recall*, dan *F1-score* untuk mengukur performa model pada setiap kelas. Setelah model menyelesaikan proses klasifikasi pada *dataset* uji menggunakan parameter optimal yang diperoleh dari proses *tuning hyperparameter*, nilai metrik dihitung menggunakan rumus yang terlampir pada persamaan 1 hingga 4 (Sidik et al., 2023). Adapun persamaan metrik evaluasi sebagai berikut:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (1)$$

$$Precision = \frac{TP}{TP+TN+FP+FN} \quad (2)$$

$$Recall = \frac{TP}{TP+FP} \quad (3)$$

$$F1\text{-}Score = \frac{TP}{TP+FN} \quad (4)$$

ata positif yang terdeteksi benar.

)ata negatif yang terdeteksi benar.

False Positive (FP): Data negatif yang keliru terdeteksi sebagai positif.

False Negative (FN): Data positif yang keliru terdeteksi sebagai negatif.

BAB II
METODE PENELITIAN

2.1 Waktu dan Tempat Penelitian

Penelitian ini dilaksanakan dari bulan Oktober 2024 hingga Januari 2025, dengan rincian waktu dan tahapan yang dapat dilihat pada Tabel 1 dan bertempat pada Ruang Belajar Sistem Informasi Prodi Sistem Informasi, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Hasanuddin.

Tabel 1. Waktu & Tahapan Penelitian

No	Kegiatan	2024									2025		
		Oktober		November			Desember				Januari		
		3	4	1	2	3	1	2	3	4	1	2	3
1	Analisis Kebutuhan												
2	Desain Sistem												
3	Pengkodean/ Implementasi												
4	Pengujian												

2.2 Instrumen Penelitian

Penelitian ini menggunakan beberapa instrumen yang terdiri dari perangkat lunak (*software*) dan perangkat keras (*hardware*) yang dirincikan dalam Tabel 2 dan Tabel 3 untuk mendukung proses penelitian.

Tabel 2. Instrumen perangkat keras

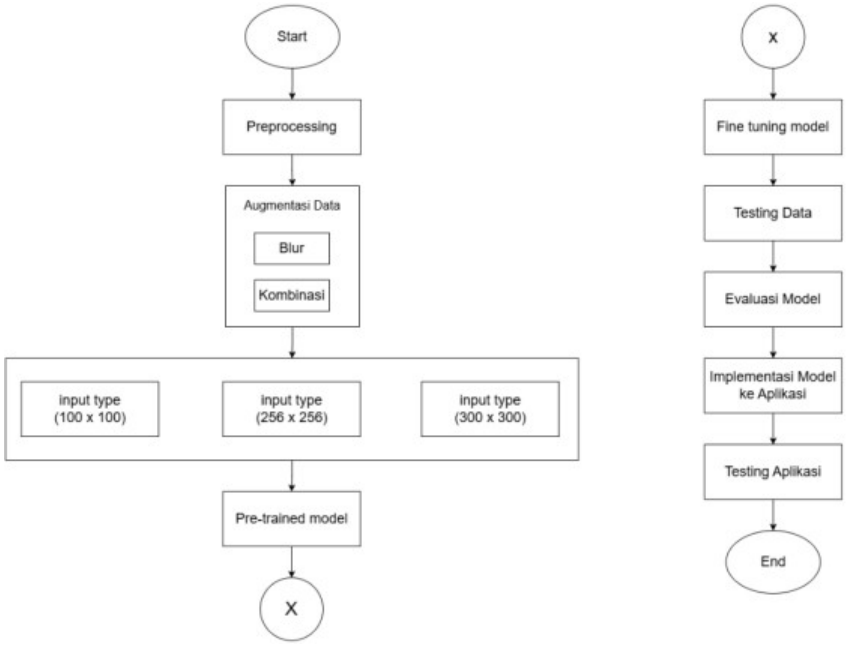
Hardware	**Spesifikasi**
Operating System	Windows 11 Home
Processor	11th Gen Intel(R) Core™ i5
Memory	8 GB
Storage	475gb SSD
GPU	Intel® Iris® Xe Graphics

Tabel 3. Instrumen perangkat lunak

Software	**Keterangan**
Google Colab	Platform berbasis *cloud* untuk menjalankan dan mengembangkan kode python.
Python	Bahasa pemrograman utama yang digunakan untuk pengembangan model dan analisis.
Visual Studio Code	Editor kode sumber yang ringan, dikembangkan oleh Microsoft. Mendukung berbagai bahasa pemrograman melalui ekstensi.
Numpy	Pustaka untuk operasi *array* multidimensi dan komputasi numerik.
Pandas	Pustaka untuk manipulsi dan analisis data berbasis tabel.
Matplotlib	Pustaka untuk membuat grafik dan visualisasi data.
Tensorflow	Pustaka untuk pengembangan dan pelatihan model *deep learning* berbasis CNN.
Scikit-learn	Pustaka untuk *preprocessing* data, evaluasi model, dan pengukuran metrik.
Seaborn	Pustaka untuk visualisasi data yang mendukung pembuatan grafik statistik.
CV2 (OpenCV)	Pustaka untuk pengolahan citra, seperti *resizing* dan *augmentasi* data gambar.
Flutter	*Framework* UI dari Google untuk membuat aplikasi *mobile cross-platform* dengan performa tinggi dan tampilan yang menarik.
Firebase Firestore	*Database cloud* NoSQL yang memungkinkan penyimpanan dan sinkronisasi data *realtime* untuk aplikasi web dan *mobile*.
Niagahoster VPS	Menyediakan server virtual privat dengan performa, keamanan, dan skalabilitas tinggi untuk *hosting website* atau aplikasi.
Termius	Aplikasi untuk mengelola koneksi SSH atau SFTP dengan antarmuka yang lebih ramah pengguna.
Ubuntu	Sistem operasi berbasis Linux yang umum digunakan untuk server dan pengembangan.
Figma	Alat desain berbasis *cloud* untuk membuat UI/UX untuk aplikasi dan situs *website*.

2.3 Tahap Penelitian

Penelitian ini menerapkan model *Systems Development Life Cycle* (SDLC) dengan pendekatan waterfall, yang terdiri dari beberapa tahapan: *preprocessing*, pelatihan data menggunakan *transfer learning*, evaluasi model, implementasi, dan pengujian nelitian ini digambarkan dalam diagram alur pada Gambar 22.



Gambar 22. Diagram alur penelitian

2.4 Metode Pengembangan Sistem

Dalam penelitian ini, pengembangan sistem dilakukan menggunakan metode *waterfall*, salah satu pendekatan dalam *Software Development Life Cycle* (SDLC). Metode ini dikenal sebagai model pengembangan perangkat lunak yang berurutan, dengan proses yang terdiri atas tahapan-tahapan yang dilaksanakan secara bertahap dan sistematis, dimulai dari analisis kebutuhan, perancangan sistem, pengkodean, pengujian, hingga tahap pemeliharaan (Sumantri et al., 2022). Berikut adalah uraian dari masing-masing tahapan dalam metode *waterfall*:

2.4.1. Analisis Kebutuhan

Penelitian ini bertujuan membangun sebuah model klasifikasi penyakit daun tanaman berbasis *deep learning* serta mengintegrasikannya ke dalam aplikasi *mobile* berbasis Android. Dalam tahap analisis kebutuhan, ditentukan beberapa aspek penting, seperti pemilihan arsitektur model, resolusi input citra, serta platform pengembangan

si citra yang digunakan dalam penelitian ini adalah
ah satu varian dari keluarga *EfficientNetV2* yang dikenal
ameter tinggi serta performa pelatihan yang baik (Tan & Le,
hasil evaluasi pada penelitian terdahulu, *EfficientNetV2-L*
na terbaik dibandingkan varian lainnya dalam hal rata-rata
l, dan *F1-score* pada *dataset* klasifikasi multikelas (Rafif et al.,

2022). Model ini kemudian dilatih untuk mengenali 15 kelas citra daun dari tiga jenis tanaman keluarga *Solanaceae*, yaitu kentang, tomat, dan paprika, termasuk kondisi sehat maupun terinfeksi penyakit.

Citra diproses dalam tiga *input size* berbeda, yaitu 100x100, 256x256, dan 300x300 piksel. Setiap resolusi citra digunakan secara terpisah pada model arsitektur yang sama dan diuji dengan dua pendekatan yang berbeda, yaitu *augmentasi blur* dan *augmentasi* kombinasi, sehingga seluruh proses pelatihan mencakup enam skenario berbeda untuk mengevaluasi pengaruh resolusi dan *augmentasi* terhadap performa klasifikasi.

Dataset yang digunakan adalah *PlantVillage Dataset*, yang diakses secara publik dari https://www.kaggle.com/*datasets*/emmarex/plantdisease. *Dataset* ini terdiri dari 20.639 gambar daun tanaman yang diproses terlebih dahulu sebelum dibagi menjadi tiga *subset*, yaitu 55% data pelatihan, 25% data validasi, dan 20% data pengujian.

2.4.2. Desain Sistem

Tahap desain sistem pada penelitian ini mencakup dua aspek utama, yaitu perancangan arsitektur model *deep learning* yang digunakan untuk klasifikasi gambar, serta rancangan antarmuka pengguna (UI/UX) dari aplikasi yang akan menampilkan hasil klasifikasi.

Desain arsitektur model. Pada tahap desain, model *EfficientNetV2-L* disusun dalam dua tahap, yaitu pelatihan awal dan *fine tuning*, yang masing-masing memiliki konfigurasi arsitektur dan *hyperparameter* tersendiri.

a. Struktur *Layer* Model

Model memiliki struktur *layer* seperti yang ditunjukkan pada Tabel 4 berikut:

Tabel 4. Model *EfficientNetV2-L*

Layer (type)	*Output Shape*	Param #
input_*layer*	(*None*, 100, 100, 3) / (*None*, 256, 256, 3) / (*None*, 300, 300, 3)	0
blok EfficientNetV2-L	-	-
global_average_pooling2d	(*None*, 1280)	0
batchnormalization	(*None*, 1280)	5,120
dense	(*None*, 128)	163,968
dropout	(*None*, 128)	0
dense_1	(*None*, 256)	33,024
n_2	(*None*, 256)	1,024
	(*None*, 256)	0
	(*None*, 15)	3,855
	117,953,839	
:	117,438,191	
ams:	515,648	

Model

Model dilatih dalam dua tahap, seperti yang dijelaskan pada Tabel 5 berikut:

Tabel 5. Struktur *layer*

Tahap pelatihan	Susunan lapisan
Pelatihan awal	• Semua lapisan model dasar dibekukan • *Layer global average pooling* 2D • *Bacth Normalization* • *Layer fully connected* • *Dropout* • *Layer fully connected* • *Bacth Normalization* • *Dropout* • *Output layer*
Fine tuning	• 15 lapisan akhir model dasar diaktifkan • *Layer global average pooling* 2D • *Bacth Normalization* • *Layer fully connected* • *Dropout* • *Layer fully connected* • *Bacth Normalization* • *Dropout* • *Output layer*

c. Konfigurasi *Hyperparameter*

Hyperparameter yang digunakan pada masing-masing tahap penelitian ditunjukkan pada Tabel 6.

Tabel 6. *Hyperparameter* Model

Parameter	Pelatihan awal	Fine tuning
Optimizer	Adam	Adam
Learning rate	1e-3	1e-4
Loss function	*Sparse categorical crossentropy*	*Sparse categorical crossentropy*
Metrics	*Accuracy*	*Accuracy*
Batch size	32	32
Epochs	10	50

d. Teknik pelatihan

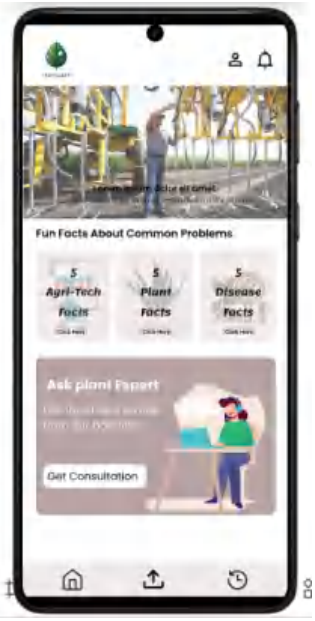
Model juga dilatih dengan parameter pelatihan tambahan sebagaimana ditampilkan

Tabel 7. Teknik Pelatihan

Parameter	Nilai
Early Stopping	• *monitor='val_loss'* • *patience*=5 • *restore_best_weights=True*
Checkpoint Callback	• *'PlantVillageModel_{blur/noblur}_{inputSize}.keras'* • *monitor='val_accuracy'* • *save_best_only=True*
ReduceLROnPlateau	• *monitor='val_loss'* • *factor*=0.1 • *patience*=4 • *min_lr*=1e-6

Desain Antarmuka Aplikasi. Perancangan antarmuka aplikasi deteksi penyakit daun meliputi berbagai komponen yang didesain untuk memudahkan pengguna dalam mengidentifikasi penyakit pada daun. Tampilan dimulai dari *splash screen* yang menampilkan nama aplikasi selama beberapa detik, dapat dilihat di Gambar 23. Gambar 24 menampilkan halaman utama aplikasi yang berisi fitur *fun fact*, deteksi penyakit daun melalui *widget upload*, riwayat deteksi, dan kumpulan artikel informatif. Setelah melakukan deteksi penyakit melalui kamera atau galeri foto akan muncul tampilan seperti pada Gambar 26. Gambar 25 ialah laman yang menampilkan visual dari fakta seputar tumbuhan, *agri-tech*, dan penyakit tanaman.

screen



Gambar 24. Menu utama aplikasi

Gambar 25. Menu *fun fact*

Gambar 26. Tampilan hasil deteksi

2.4.3. Pengkodean/Implementasi

Tahap pengkodean pada penelitian ini menggunakan beberapa perangkat dan teknologi, antara lain: tensorflow, keras, *openCV*, flutter, Firebase *firestore*, niagahoster, dan termius.

2.4.4. Pengujian

a. Model

Tahap pengujian dilakukan pada data uji yang melalui proses ekstraksi fitur. Model dievaluasi menggunakan *confusion matrix* untuk mengukur performa klasifikasi multi-kelas, dengan menghitung nilai akurasi, presisi, *recall*, dan *F1-score*.

b. Aplikasi

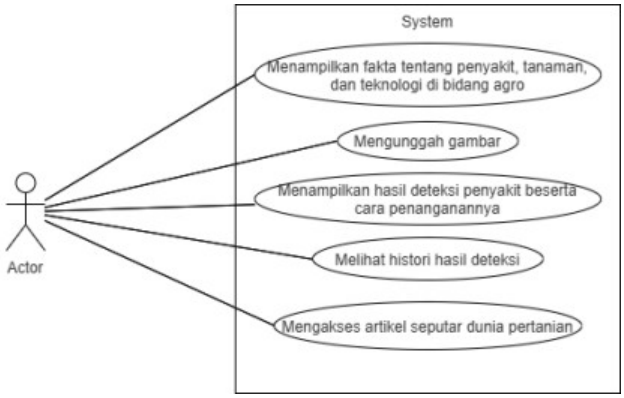
Pengujian aplikasi dilakukan menggunakan metode *black box*, yaitu metode pengujian yang tidak melihat kode program secara langsung, melainkan hanya menguji masukan (*input*) dan keluaran (*output*) dari aplikasi. Fokus pengujian
ma seperti unggah gambar, proses deteksi penyakit tanaman
nyajian hasil deteksi kepada pengguna. Tujuan dari pengujian
astikan bahwa integrasi model ke dalam aplikasi berjalan lancar
sedia dapat digunakan dengan baik oleh pengguna akhir.

an

Tahap pemeliharaan pada penelitian ini belum dapat dilakukan karena sistem masih bergantung pada server berbayar yang harus dilanggan setiap bulan, sehingga jika layanan dihentikan, model dan API tidak dapat berjalan. Selain itu, *output* rekomendasi penanganan penyakit masih bersifat manual dan belum otomatis. Tahap ini direncanakan untuk pengembangan selanjutnya, termasuk integrasi sistem otomatis, pembaruan model, dan penggunaan *cloud* yang lebih stabil.

2.5 Perancangan Sistem

Untuk memperjelas alur penggunaan sistem, penelitian ini juga dilengkapi dengan diagram *use case* yang menggambarkan interaksi antara pengguna dan sistem. Diagram *use case* dapat dilihat pada Gambar 27.



Gambar 27. *Use case*