PENGARUH AUGMENTASI DATA TERHADAP KINERJA ARSITEKTUR DENSENET-201 DALAM KLASIFIKASI CITRA PENYAKIT TANAMAN PADI

SKRIPSI

Oleh :
AVICENA HALIM
NIM. 210605110118

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2025

PENGARUH AUGMENTASI DATA TERHADAP KINERJA ARSITEKTUR DENSENET-201 DALAM KLASIFIKASI CITRA PENYAKIT TANAMAN PADI

SKRIPSI

Diajukan kepada:
Universitas Islam Negeri Maulana Malik Ibrahim Malang
Untuk memenuhi Salah Satu Persyaratan dalam
Memperoleh Gelar Sarjana Komputer (S.Kom)

Oleh :
AVICENA HALIM
NIM. 210605110118

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2025

HALAMAN PERSETUJUAN

PENGARUH AUGMENTASI DATA TERHADAP KINERJA ARSITEKTUR DENSENET-201 DALAM KLASIFIKASI CITRA PENYAKIT TANAMAN PADI

SKRIPSI

Oleh:
AVICENA HALIM
NIM. 210605110157

Telah Diperiksa dan Disetujui untuk Diuji:
Tanggal: 5 Juni 2025

Pembimbing I,	Pembimbing II,
Tri Mukti Lestari, M.Kom	Roro Inda Melani, M.T, M.Sc
NIP. 19911108 202012 2 005	NIP. 19780925 200501 2 008

Mengetahui,
Ketua Program Studi Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Ir. Fachrul Kurniawan, M.MT., IPU
NIP. 19771020 200912 1 001

HALAMAN PENGESAHAN

PENGARUH AUGMENTASI DATA TERHADAP KINERJA ARSITEKTUR DENSENET-201 DALAM KLASIFIKASI CITRA PENYAKIT TANAMAN PADI

SKRIPSI

Oleh:
AVICENA HALIM
NIM. 210605110118

Telah Dipertahankan di Depan Dewan Penguji Skripsi
dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan
Untuk Memperoleh Gelar Sarjana Komputer (S.Kom)
Tanggal: 20 Juni 2025

Susunan Dewan Penguji

Ketua Penguji	: A'la Syauqi, M.Kom NIP. 19771201 200801 1 007	( )
Anggota Penguji I	: Nurizal Dwi Priandani, M.Kom NIP. 19920830 202203 1 001	( )
Anggota Penguji II	: Tri Mukti Lestari, M.Kom NIP. 19911108 202012 2 005	( )
Anggota Penguji III	: Roro Inda Melani, M.T, M.Sc NIP. 19780925 200501 2 008	( )

Mengetahui dan Mengesahkan,
Ketua Program Studi Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Ir. Fachrul Kurniawan, M.MT., IPU
NIP. 19771020 200912 1 001

PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan di bawah ini:

Nama	: Avicena Halim
NIM	: 210605110118
Fakultas / Program Studi	: Sains dan Teknologi / Teknik Informatika
Judul Skripsi	: Pengaruh Augmentasi Data Terhadap kinerja Arsitektur DenseNet-201 Dalam Klasifikasi Citra Penyakit Tanaman Padi

Menyatakan dengan sebenarnya bahwa Skripsi yang saya tulis ini benar-benar merupakan hasil karya saya sendiri, bukan merupakan pengambil alihan data, tulisan, atau pikiran orang lain yang saya akui sebagai hasil tulisan atau pikiran saya sendiri, kecuali dengan mencantumkan sumber cuplikan pada daftar pustaka.

Apabila dikemudian hari terbukti atau dapat dibuktikan skripsi ini merupakan hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 6 Juni 2025
Yang membuat pernyataan,

Avicena Halim
NIM.210605110118

MOTTO

"Menjadi lebih baik dari yang kemarin."

HALAMAN PERSEMBAHAN

Segala puji syukur tercurah kepada Allah Subhanahu wa Ta'ala atas segala limpahan rahmat, karunia, dan hidayah-Nya sehingga penulis dapat menyelesaikan skripsi ini. Shalawat serta salam penulis haturkan kepada junjungan Nabi Muhammad Shallallahu 'Alaihi Wasallam, yang telah membawa kita dari zaman kegelapan menuju zaman yang penuh ilmu dan cahaya.

Dengan rasa hormat dan terima kasih yang tulus, penulis persembahkan karya ini kepada kedua orang tua, Bapak Wiyono dan Ibu Nurhaidah. Ucapan terima kasih yang sebesar-besarnya atas doa yang tak pernah henti, kasih sayang yang tiada batas, dan dukungan tanpa syarat yang selalu menguatkan penulis dalam setiap langkah perjalanan ini. Semoga skripsi ini menjadi pijakan langkah awal pembuka yang dapat mengantarkan penulis ke masa depan dan cita-cita yang penulis inginkan.

KATA PENGANTAR

Assalamualaikum Wr.Wb.

Alhamdulillahirabbil'alamin, segala puji dan syukur penulis panjatkan ke hadirat Allah SWT atas limpahan rahmat, karunia, dan hidayah-Nya sehingga penulis dapat menyelesaikan tugas akhir skripsi yang berjudul "pengaruh augmentasi data terhadap kinerja arsitektur DenseNet-201 dalam klasifikasi citra penyakit tanaman padi" ini dengan baik dan lancar. Shalawat serta salam senantiasa tercurah kepada junjungan kita, Nabi Muhammad SAW, yang telah membawa umat manusia dari zaman kebodohan menuju zaman yang penuh dengan kebenaran dan ilmu pengetahuan, sebagaimana yang kita rasakan pada saat ini.

Penyusunan skripsi ini dilaksanakan sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer pada Program Studi Teknik Informatika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim Malang. Dalam proses penelitian dan penyusunan tugas akhir ini, penulis menyadari bahwa banyak bantuan, dukungan, serta bimbingan yang diberikan oleh berbagai pihak, baik secara langsung maupun tidak langsung, yang sangat berarti dalam penyelesaian skripsi ini.

Dengan penuh rasa hormat, penulis menyampaikan ucapan terima kasih yang tulus kepada:

1. Prof. Dr. M. Zainuddin, M.A., selaku Rektor Universitas Islam Negeri (UIN) Maulana Malik Ibrahim Malang, yang telah memberikan kesempatan bagi penulis untuk menyelesaikan pendidikan ini.

2. Prof. Dr. Sri Harini, M.Si., selaku Dekan Fakultas Sains dan Teknologi UIN Maulana Malik Ibrahim Malang, atas segala dukungan dan fasilitas yang diberikan kepada penulis selama proses studi.
3. Dr. Ir. Fachrul Kurniawan, M.MT., IPU selaku Ketua Program Studi Teknik Informatika UIN Maulana Malik Ibrahim Malang, yang telah memberikan arahan dan dukungan dalam proses akademik penulis.
4. Tri Mukti Lestari, M.Kom selaku dosen pembimbing I, dan Roro Inda Melani, M.T, M.Sc, selaku dosen pembimbing II, yang telah memberikan bimbingan, masukan, dan motivasi yang sangat berharga selama penyusunan skripsi ini.
5. Fajar Rohman Hariri, M.Kom selaku dosen wali, yang telah memberikan bimbingan, dukungan, serta arahan berharga selama penelitian ini. Atas perhatian dan motivasi yang diberikan, penulis menyampaikan rasa terima kasih yang sebesar-besarnya.
6. Orang tua tercinta, Bapak Wiyono dan Ibu Nurhaidah, serta kakak dan adik penulis, Itho Zainal Muttaqin, M. Rizqi Ramadhan, Latifah Nur Sakinah, dan Athallah Izzuddin yang senantiasa memberikan doa, dukungan, dan semangat tiada henti.
7. Keluarga UMI DOJI yang senantiasa memberikan doa, dukungan, dan kasih sayang yang tiada henti. Semoga kebaikan kalian selalu dibalas dengan keberkahan.
8. Teman-teman Aster dan Pesma Ibadurahman yang selalu memberikan semangat dan kebersamaan selama masa studi.

9. Seluruh dosen dan staf Program Studi Teknik Informatika serta Fakultas Sains dan Teknologi, yang telah memberikan ilmu dan dukungan selama masa studi penulis.

Penulis menyadari bahwa skripsi ini masih memiliki kekurangan dan keterbatasan. Oleh karena itu, penulis sangat mengharapkan kritik dan saran yang membangun demi perbaikan karya ilmiah ini di masa yang akan datang. Semoga skripsi ini dapat memberikan manfaat, baik bagi penulis sendiri maupun bagi perkembangan ilmu pengetahuan, khususnya dalam bidang mitigasi lingkungan dan teknologi informasi.

Malang, 5 Juni 2025

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

ABSTRAK

Halim, Avicena. 2025. **Pengaruh Augmentasi Data Terhadap Kinerja Arsitektur DenseNet-201 Dalam Klasifikasi Citra Penyakit Tanaman Padi.** Skripsi. Program Studi Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang. Pembimbing: (I) Tri Mukti Lestari, M.Kom (II) Roro Inda Melani, M.T, M.Sc.

.

Kata Kunci: Penyakit tanaman padi, klasifikasi citra, DenseNet-201, augmentasi data, *Convolutional Neural Network* (CNN).

Penurunan produktivitas padi di Indonesia, yang disebabkan oleh perubahan iklim dan serangan penyakit tanaman, mendorong pentingnya pengembangan sistem klasifisai untuk penyakit tanaman padi. Penelitian ini bertujuan untuk membandingkan kinerja arsitektur DenseNet-201 dalam klasifikasi penyakit padi seperti *Blast*, *Blight*, dan *Tungro*, dengan penerapan teknik augmentasi data untuk meningkatkan akurasi model. Metode penelitian yang digunakan adalah eksperimen dengan pendekatan *Convolutional Neural Network*s (CNN), di mana dataset penyakit padi digunakan untuk pelatihan model. Augmentasi data diterapkan untuk memperbanyak variasi gambar dalam dataset guna meningkatkan kemampuan model dalam mengatasi variasi kondisi lapangan. Hasil penelitian menunjukkan bahwa penerapan augmentasi data menghasilkan peningkatan signifikan dalam akurasi klasifikasi penyakit padi, dengan model yang menggunakan augmentasi data mencapai akurasi tertinggi sebesar 98%. Model ini juga menunjukkan kemampuan yang lebih baik dalam mengidentifikasi penyakit dengan presisi dan *recall* yang tinggi. Dengan demikian, penggunaan arsitektur DenseNet-201 yang dikombinasikan dengan augmentasi data dapat menjadi solusi efektif dalam mengklasifikasi penyakit tanaman padi dan meningkatkan ketahanan pangan.

ABSTRACT

Halim, Avicena. 2025. **The Impact of Data Augmentation on the Performance of DenseNet-201 Architecture in Rice Plant Disease Image Classification**. Thesis. Department of Informatics Engineering, Faculty of Science and Technology, Maulana Malik Ibrahim State Islamic University Malang. Supervisors: (I) Tri Mukti Lestari, M.Kom (II) Roro Inda Melani, M.T, M.Sc.

.

Keywords: Rice plant disease, image *Class*ification, DenseNet-201, data augmentation, *Convolutional Neural Network* (CNN)

The decline in rice productivity in Indonesia, caused by climate change and plant disease outbreaks, highlights the importance of developing Classification systems for rice plant diseases. This study aims to compare the performance of the DenseNet-201 architecture in Classifying rice plant diseases such as Blast, Blight, and Tungro, with the application of data augmentation techniques to improve model accuracy. The research method employed is experimental using a Convolutional Neural Network (CNN) approach, in which a rice disease dataset is used for model training. Data augmentation is applied to increase the diversity of images in the dataset, enhancing the model's ability to handle variations in field conditions. The results show that the application of data augmentation leads to a significant improvement in the accuracy of rice disease Classification, with the augmented model achieving the highest accuracy of 98%. This model also demonstrates better performance in identifying diseases with high precision and recall. Therefore, the use of DenseNet-201 architecture combined with data augmentation can be an effective solution for Classifying rice plant diseases and improving food security.

مستخلص البحث

حليم، أفيسينا 2025. تأثير زيادة البيانات على أداء بنية DenseNet-201 في تصنيف صور أمراض نبات الأرز. الأطروحة. برنامج دراسة هندسة المعلوماتية، كلية العلوم والتكنولوجيا، الجامعة الإسلامية الحكومية، مولانا مالك إبراهيم مالانج. المشرف: (أولاً) تري موكتي ليستاري، ماجستير في العلوم (ثانيًا) رورو إندا ميلاني، ماجستير في العلوم.

الكلمات المفتاحية: مرض نبات الأرز، تصنيف الصور، DenseNet-201، زيادة البيانات، الشبكة العصبية التلافيفية (CNN).

دفع الانخفاض في إنتاجية الأرز في إندونيسيا، الناجم عن التغير المناخي وهجمات الأمراض النباتية، إلى أهمية تطوير نظام تصنيف لأمراض نبات الأرز. يهدف هذا البحث إلى مقارنة أداء بنية DenseNet-201 في تصنيف أمراض الأرز مثل الآفة واللفحة والتونجرو، مع تطبيق تقنيات زيادة البيانات لتحسين دقة النموذج. طريقة البحث المستخدمة هي تجربة باستخدام نهج الشبكات العصبية التلافيفية (CNN)، حيث يتم استخدام مجموعات بيانات أمراض الأرز لتدريب النموذج. تم تطبيق زيادة البيانات لمضاعفة مجموعة الصور المتنوعة في مجموعة البيانات لتحسين قدرة النموذج على التعامل مع الاختلافات في الظروف الميدانية. أظهرت النتائج أن تطبيق تكثير البيانات أدى إلى تحسن كبير في دقة تصنيف أمراض الأرز، حيث حقق النموذج الذي استخدم تكثير البيانات أعلى دقة بلغت 98%. كما أظهر النموذج أيضًا قدرة أفضل في تحديد الأمراض بدقة واستدعاء عالية. وبالتالي، يمكن أن يكون استخدام بنية DenseNet-201 مع زيادة البيانات حلاً فعالاً في تصنيف أمراض نبات الأرز وتحسين الأمن الغذائي.

BAB I

PENDAHULUAN

1.1 Latar Belakang

Produktivitas padi di Indonesia menurun signifikan akibat perubahan iklim dan serangan penyakit tanaman dalam dua tahun terakhir. Pada 2023, produksi padi turun 2,05% dibandingkan 2022 karena penyusutan lahan panen dan cuaca ekstrem seperti *El Niño*, terutama di sentra produksi seperti Jawa dan Sulawesi (BPS, 2023). Kementerian Pertanian memperkirakan kehilangan produksi mencapai 1,2 juta ton akibat kondisi iklim tersebut (Kontan, 2023). Selain itu, penyakit padi seperti *Blast*, *Blight*, dan *Tungro* juga menjadi penyebab utama (Masnilah et al., 2020). FAO mencatat kerugian global akibat penyakit tanaman bisa mencapai 40% (FAO, 2023).

Serangan penyakit bahkan menyebabkan gagal panen di beberapa wilayah. Situasi ini menekankan pentingnya pengembangan sistem klasifikasi penyakit tanaman yang lebih efektif, agar petani dapat segera mengambil tindakan pencegahan dan menjaga stabilitas produksi padi di masa depan. Di Merauke, serangan *Tungro* pada tahun 2022 telah menyebabkan lahan padi yang luas tidak dapat menghasilkan gabah, memaksa petani mengalami kerugian finansial yang besar. Kondisi ini memengaruhi ketahanan pangan daerah tersebut, karena hasil panen yang menurun drastis. Penyakit ini tidak hanya mengancam ketersediaan pangan lokal, tetapi meningkatkan ketergantungan pada suplai pangan dari luar daerah (Jubi, 2022).

Dalam Islam, manusia diamanahkan sebagai khalifah di bumi, yang memiliki tanggung jawab besar untuk menjaga dan memanfaatkan sumber daya alam secara bijaksana. Firman Allah dalam Surah Yusuf ayat 47:

قَالَ تَزْرَعُونَ سَبْعَ سِنِينَ دَأَبًا فَمَا حَصَدتُّمْ فَذَرُوهُ فِي سُنۢبُلِهِۦٓ إِلَّا قَلِيلًا مِّمَّا تَأْكُلُونَ ﴿٤٧﴾

"(Yusuf) berkata, "Bercocoktanamlah kamu tujuh tahun berturut-turut! Kemudian apa yang kamu tuai, biarkanlah di tangkainya, kecuali sedikit untuk kamu makan." (QS. Yusuf ayat 47).

Ayat ini menjelaskan pentingnya perencanaan dan antisipasi dalam menjaga ketahanan pangan. Nabi Yusuf AS memberikan solusi untuk mengatasi masa sulit dengan menyimpan hasil panen yang melimpah selama masa subur. Ini menunjukkan bahwa ketahanan pangan tidak hanya terkait dengan produksi, tetapi juga dengan penyimpanan dan distribusi yang efisien. Teknologi seperti *Convolutional Neural Networks* (CNN) dan augmentasi data yang diterapkan dalam penelitian untuk klasifikasi penyakit padi adalah salah satu bentuk ikhtiar yang mencerminkan kepatuhan pada perintah Allah untuk memanfaatkan ilmu demi kebaikan bersama.

Oleh karena itu, penerapan teknologi machine learning untuk klasifikasi penyakit tanaman padi dapat menjadi solusi efektif dalam meningkatkan kesadaran serta kualitas penanganan pangan (Bakr et al., 2022). Teknologi ini lebih unggul dibandingkan metode konvensional yang sering memakan waktu, sangat bergantung pada tenaga ahli, dan rentan terhadap kesalahan manusia, yang dapat menurunkan akurasi diagnosis.

Dengan kombinasi augmentasi data pada arsitektur DenseNet-201 menjadikan metode ini sangat efektif untuk mengklasifikasi penyakit tanaman, karena model tidak hanya dapat menangani dataset yang terbatas, tetapi juga belajar dari variasi kondisi lingkungan yang berbeda pada tanaman (Bakr et al., 2022). Hal ini meningkatkan kemampuan model untuk mengenali penyakit dengan akurasi tinggi, yang sangat penting dalam aplikasi dunia nyata, seperti pertanian.

Dalam penelitian sebelumnya, Salim et al. (2023) menggunakan arsitektur CNN seperti Xception dan DenseNet-201 untuk deteksi Buah. Hasil penelitian menunjukkan bahwa DenseNet-201 mencapai akurasi 99,87% pada Fruits-360 dan 99,13% pada *Fruit Recognition Dataset*, sedangkan Xception memperoleh akurasi 98,94% dan 97,73% pada dataset yang sama. Namun, penelitian ini tidak menerapkan teknik augmentasi data yang dapat meningkatkan performa model terhadap variasi kondisi citra di lapangan.

Penelitian lain oleh Ferentinos (2018) menggunakan dataset besar yang mencakup lebih dari 58.000 gambar untuk klasifikasi penyakit tanaman menggunakan CNN. Meskipun akurasinya mencapai lebih dari 97%, penelitian ini hanya menggunakan model ResNet dan AlexNet, tanpa mengeksplorasi efisiensi arsitektur yang lebih modern seperti DenseNet-201 dalam mendeteksi penyakit tanaman padi secara spesifik.

Tedi Setiady (2021) yang mengklasifikasikan penyakit padi dengan menggunakan *Convolutional Neural Network* (CNN) dan arsitektur VGG16 yang dimodifikasi. Dalam penelitian ini, sistem yang dikembangkan berhasil mengklasifikasikan empat kelas penyakit padi, yaitu *Blast*, *Blight*, *Tungro*, dan

healthy, dengan akurasi rata-rata sebesar 97,6%. Selain itu, penelitian ini juga menggunakan teknik augmentasi data untuk meningkatkan variasi data pelatihan, yang terdiri dari transformasi seperti *shear*, *zoom*, rotation, dan flip.

Diharapkan dengan penerapan teknologi *deep learning*, khususnya melalui penggunaan *Convolutional Neural Network*s (CNN) berbasis DenseNet-201, sistem klasifikasi penyakit tanaman padi dapat memberikan kontribusi signifikan dalam mengklasifikasi penyakit. Melalui sistem tersebut, petani dapat mengambil tindakan preventif dengan lebih cepat, sehingga mengurangi potensi kerugian dan dampak negatif terhadap hasil panen.

Dalam penelitian ini, kami mengusulkan kombinasi antara arsitektur DenseNet-201 dan augmentasi data untuk meningkatkan akurasi deteksi penyakit padi. Dengan menerapkan augmentasi data yang lebih variatif, model dapat lebih adaptif terhadap berbagai kondisi pencahayaan dan variasi lingkungan yang ada di lapangan. Selain itu, penelitian ini akan membandingkan hasil dengan model sebelumnya yang tidak menerapkan augmentasi, guna menilai efektivitas pendekatan ini dalam meningkatkan performa klasifikasi penyakit padi.

1.2 Pernyataan Masalah

Bagaimana pengaruh augmentasi data terhadap arsitektur DenseNet-201 dalam klasifikasi penyakit tanaman padi *Blast*, *Blight*, dan *Tungro* menggunakan pendekatan *Convolutional Neural Network*s (CNN)?

1.3 Batasan Masalah

a. Penelitian ini hanya mencakup tiga jenis penyakit padi, yaitu *Blast*, *Blight*, dan *Tungro*.

b. Dataset yang digunakan berasal dari sumber terbatas, yaitu 240 gambar yang diambil dari *Kaggle*.
c. Teknik augmentasi data yang digunakan mencakup rotasi, Pembesaran, Pembalikan, dan penyesuaian kecerahan.
d. Model hanya diterapkan pada gambar penyakit padi yang telah melalui proses *preprocessing*, seperti normalisasi dan *resizing*.

1.4 Tujuan Penelitian

Menerapkan arsitektur DenseNet-201 dan teknik augmentasi data dalam klasifikasi penyakit tanaman padi *Blast*, *Blight*, dan *Tungro* dengan menggunakan pendekatan *Convolutional Neural Network*s (CNN).

1.5 Manfaat Penelitian

Berikut adalah manfaat dari dilakukannya penelitian ini, terdapat manfaat, yaitu:

a. Memberikan kontribusi ilmiah dalam pengembangan teknologi berbasis CNN untuk klasifikasi penyakit tanaman.
b. Menambah wawasan tentang penggunaan augmentasi data untuk meningkatkan performa model klasifikasi citra.
c. Membantu petani dalam mengidentifikasi penyakit padi secara cepat dan akurat, sehingga dapat mengambil tindakan lebih awal.
d. Mendukung pemerintah dan institusi pertanian dalam menyediakan sistem klasifikasi penyakit yang efisien dan ekonomis

BAB II

STUDI PUSTAKA

2.1 Penelitian Terkait

Klasifikasi adalah proses pengelompokan suatu objek ke dalam kategori tertentu berdasarkan kesamaan fitur atau karakteristiknya. Dalam bidang *machine learning*, klasifikasi digunakan untuk memprediksi label atau kelas dari suatu data baru berdasarkan pola yang telah dipelajari dari data latih (*training data*) (Schalock et al., 2021). Klasifikasi banyak diterapkan dalam berbagai bidang, termasuk pengolahan teks, pengenalan suara, analisis sinyal, dan pengenalan pola dalam citra. Salah satu aplikasi yang berkembang pesat adalah deteksi penyakit tanaman menggunakan computer vision, yang memungkinkan identifikasi penyakit tanaman padi secara otomatis berdasarkan citra daun yang terinfeksi.

Penelitian yang dilakukan oleh Salim et al. (2023) berfokus pada penerapan model DenseNet-201 dan Xception dalam klasifikasi buah menggunakan dua dataset yang berbeda, yaitu Fruits-360 dan *Fruit Recognition Dataset*. Dataset Fruits-360 yang digunakan dalam penelitian ini merupakan versi terbaru yang mempertahankan ukuran asli gambar nya, berbeda dengan versi lama yang hanya memiliki resolusi 100 × 100 piksel. Dataset ini terdiri dari 24 kelas buah dan sayuran dengan total 6231 gambar untuk *training*, 3114 gambar untuk *validation*, dan 3110 gambar untuk *testing*. Sementara itu, *Fruit Recognition Dataset* terdiri dari 15 kelas buah dengan total 44.406 gambar, yang dirancang untuk menangkap kondisi nyata dalam skenario supermarket, termasuk variasi pencahayaan,

bayangan, sudut kamera, serta variasi pose buah dalam gambar. Untuk memastikan generalisasi model yang lebih baik, dataset ini dibagi menjadi 80% data untuk *training*, 10% untuk *validation*, dan 10% untuk *testing*.

Penelitian lain yang dilakukan oleh Boulent et al. (2019) berfokus pada penerapan CNN dalam deteksi penyakit tanaman dengan dataset yang mencakup gambar tanaman dari berbagai sumber, termasuk citra lapangan dan dataset publik. Dataset dibagi ke dalam *train*, *validation*, dan *test* set dengan rasio 80:10:10, sehingga memastikan bahwa model dapat belajar dari variasi yang cukup luas. Parameter utama dalam penelitian ini meliputi *learning rate decay* untuk meningkatkan stabilitas pelatihan, *batch size* 64, serta fungsi aktivasi ReLU. Model CNN yang dikembangkan juga menggunakan *pretrained weights* dari *ImageNet*, yang memungkinkan model untuk memanfaatkan fitur yang telah dipelajari sebelumnya sebelum diterapkan pada tugas klasifikasi penyakit tanaman.

Selain membandingkan arsitektur CNN, penelitian yang dilakukan oleh Ferentinos (2018) menggunakan dataset besar yang mencakup 58.000 gambar tanaman yang terinfeksi. Dengan dataset sebesar ini, model dapat belajar dari berbagai variasi penyakit tanaman dan kondisi lingkungan yang berbeda. Data dibagi menjadi 80% untuk *training* dan 20% untuk *testing*, memastikan bahwa model mendapatkan cukup informasi selama pelatihan tanpa mengalami *overfitting*. Parameter yang digunakan dalam model mencakup *optimizer Adam*, *batch size* 32, serta fungsi aktivasi ReLU. Selain itu, penelitian ini menerapkan *learning rate* scheduler untuk menyesuaikan kecepatan pembelajaran seiring bertambahnya jumlah *epoch*, guna meningkatkan akurasi model.

Penelitian mengenai deteksi penyakit tanaman menggunakan *deep learning* yang dilakukan Bakr et al. (2022) yang mengembangkan model berbasis DenseNet-201 untuk identifikasi penyakit tanaman melalui transfer learning. Penelitian ini menggunakan dataset yang terdiri dari 28.310 gambar daun dari tiga jenis tanaman (tomat, kentang, dan paprika), yang diklasifikasikan ke dalam 15 kelas berbeda, termasuk beberapa jenis penyakit serta kelas daun sehat. Data dibagi menjadi 75% untuk *training*, 20% untuk *validation*, dan 5% untuk *testing*, dengan *preprocessing* berupa *resizing* gambar menjadi 224 × 224 piksel dan augmentasi seperti rotasi, scaling, dan *flipping* untuk meningkatkan performa model.

Selain itu, Chen et al. (2020) mengeksplorasi penggunaan transfer learning dalam CNN untuk klasifikasi penyakit tanaman. Model CNN yang digunakan sebelumnya telah dilatih menggunakan dataset *ImageNet*, dan kemudian diadaptasi ke dataset tanaman padi dengan menggunakan *fine-tuning*. Pembagian dataset dalam penelitian ini adalah 70% untuk *training*, 15% untuk *validation*, dan 15% untuk *testing*, yang memungkinkan model untuk memanfaatkan transfer learning secara optimal. Parameter pelatihan mencakup *optimizer Adam*, *dropout* 0.5 untuk menghindari *overfitting*, serta fungsi aktivasi ReLU di seluruh lapisan konvolusi.

Penelitian yang dilakukan oleh Ridhovan et al. (2022) berfokus pada deteksi penyakit pada daun pisang menggunakan arsitektur DenseNet-201 dan Inception dengan pendekatan transfer learning serta penerapan augmentasi data untuk meningkatkan performa model. Dataset yang digunakan dalam penelitian ini terdiri dari 936 gambar daun pisang, ke dalam empat kelas, yaitu daun sehat, daun terinfeksi *Cordana*, *Sigatoka*, dan *Pestalotiopsis*. Data dikumpulkan dari sumber

publik dan diolah dengan berbagai teknik *preprocessing*, termasuk *resizing* gambar ke ukuran 224 × 224 piksel serta augmentasi data seperti rotasi, *flipping* vertikal, *shear*, dan *zoom*ing. Untuk meningkatkan akurasi model, dilakukan beberapa teknik penyeimbangan data, termasuk oversampling dan undersampling, guna mengatasi distribusi data yang tidak merata. Data kemudian dibagi menjadi 80% untuk *training* dan 20% untuk *testing*, dan pelatihan model dilakukan dengan parameter seperti *optimizer Adam*, *learning rate* 0.0001, dan *loss* function categorical crossentropy.

Penelitian yang dilakukan oleh Putri et al. (2024) menggunakan DenseNet-201 dan dataset *Mpox Skin Lesion Dataset* v2.0 dari *Mendeley Data* yang terdiri dari 755 gambar mencakup enam kelas: *chickenpox*, *cowpox*, *monkeypox*, *measles*, HFMD, dan kulit normal. Dataset ini dibagi menjadi 80% data pelatihan dan 20% data pengujian, dengan proses validasi menggunakan *stratified k-fold cross-validation* sebanyak 5 *fold*. Parameter yang diatur melalui metode *Random Search* meliputi *learning rate* (0.0001–0.001), *dropout rate* (0.4, 0.5, 0.6, 0.7), *batch size* (32, 64), jumlah filter (256, 512, 1024), *optimizer* (adam, rmsprop), dan jumlah epoch sebanyak 10.

Setiady (2021) dataset yang digunakan untuk klasifikasi penyakit tanaman padi dibagi menjadi tiga bagian: 60% untuk data pelatihan (*training* set), 40% untuk data pengujian (*testing* set), dan 20% untuk data validasi (*validation* set). Proses pelatihan dilakukan menggunakan teknik augmentasi data, yang meliputi transformasi seperti *shear*, *zoom*, *shift*, rotasi, dan *flipping* untuk meningkatkan variasi data. Parameter yang digunakan dalam proses pelatihan mencakup *learning*

rate sebesar 0.0001, *optimizer Adam*, dan *batch size* sebesar 16. Selain itu, penelitian ini menggunakan 30 epoch dalam proses pelatihan untuk mencapai hasil yang optimal.

Kumar et al. (2023) menggunakan DenseNet-201 berbasis *cloud* untuk mendeteksi COVID-19 dari gambar X-ray dada. Data dibagi menjadi 80% untuk pelatihan dan 20% untuk pengujian. Untuk pelatihan, digunakan augmentasi data seperti rotasi, *zoom*, *shift*, dan flip. Parameter yang diterapkan meliputi *batch size* 32, *learning rate* 0.0001, dan *optimizer Adam*. Model ini diterapkan untuk mengklasifikasikan tiga jenis penyakit dada: COVID-19, tuberkulosis, dan dada normal, dengan efisiensi komputasi yang tinggi berkat penggunaan platform *cloud*.

Penelitian oleh Pradipto et al. (2023) membandingkan performa DenseNet201 dan YOLOv8 dalam klasifikasi empat jenis sampah (organik, anorganik, B3, dan residu) menggunakan Raspberry Pi 4 dan kamera malam OV5647. Dataset dikumpulkan dari sumber primer dan sekunder, lalu dibagi menjadi 70% data latih, 15% validasi, dan 15% data uji. Berikut adalah Tabel 2.1 penelitian terkait :

Tabel 2.1 Penelitian Terkait

No	Peneliti	Judul	Objek	Metode	Hasil	Perbedaan Dengan Penelitian Ini
1	Salim et al. (2023)	DenseNet-201 *and Xception Pre-Trained Deep learning Models for Fruit Recognition*	Gambar buah dari berbagai jenis	CNN (DenseNet-201, Xception, MobileNetV3-Small, ResNet-50)	DenseNet-201 mencapai akurasi 99,87% pada Fruits-360 dan 99,13% pada Fruit Recognition Dataset, sedangkan Xception mencapai 98,94% dan 97,73%	Menggunakan dataset penyakit tanaman padi

2	Boulent et al. (2019)	*Convolutional Neural Networks for the automatic identification of plant diseases*	Dataset penyakit tanaman dari berbagai spesies	CNN (AlexNet, ResNet, VGG16)	CNN dapat mencapai akurasi tinggi, tetapi mengalami *overfitting* jika dataset terbatas	Menambahkan augmentasi data
3	Ferentinos (2018)	*Deep learning models for plant disease detection and diagnosis*	Gambar daun dari berbagai tanaman yang mengalami penyakit	CNN (ResNet, AlexNet)	Model berbasis CNN mencapai akurasi lebih dari 97% untuk berbagai penyakit tanaman	Menggunakan DenseNet-201
4	Bakr et al. (2022)	DenseNet-*Based Model for Plant Diseases Diagnosis*	Gambar daun dari tanaman tomat, kentang, dan paprika	CNN (DenseNet-201, VGG16, InceptionV3, ResNet152V2)	DenseNet-201 mencapai akurasi terbaik dengan 99,44% pada *training* dan 98,70% pada validasi	Menggunakan dataset penyakit padi
5	Chen et al. (2020)	*Using deep transfer learning for image-based plant disease identification*	Citra daun tanaman yang mengalami penyakit	CNN (Transfer Learning)	Penggunaan transfer learning dari *ImageNet* meningkatkan akurasi model pada penyakit tanaman padi	Penelitian ini fokus pada augmentasi data dan DenseNet-201, bukan transfer learning
6	Ridhovan et al. (2022)	*Disease Detection in Banana Leaf Plants Using* DenseNet *and Inception Method*	Gambar daun pisang dengan berbagai penyakit	CNN (DenseNet-201, Inception) + Augmentasi Data	DenseNet-201 dengan oversampling mencapai akurasi 84,73%, precision 84,80%, *recall* 84,73%, *F1-score* 84,62%	Menggunakan dataset penyakit padi dan fokus ke DenseNet-201
7	Putri et al. (2024)	*Skin Rash Classification System Using Modified* DenseNet*201 Through Random Search for Hyperparameter Tuning*	Gambar *Mpox Skin Lesion Dataset* v2.0	DenseNet-201 yang dimodifikasi dengan tuning hiperparameter menggunakan *Random Search*	Hasil menunjukkan peningkatan akurasi dari 63% pada model dasar menjadi 80% setelah dilakukan tuning	Menggunakan dataset yang berbeda dan menggunakan augmentasi

8	Setiady (2021)	Klasifikasi penyakit tanaman menggunakan metode *deep learning*	Gambar daun padi yang mengalami *Blast*, *Blight*, *Tungro, healthy*	CNN dan VGG16	Kombinasi CNN dengan *VGG16* akurasi rata-rata sebesar 97,6%	Menggunakan DenseNet-201 dan augmentasi data
9	Kumar et al. (2023)	*Energy-efficient model "*DenseNet*201 based on deep Convolutional Neural Network" using cloud platform for detection of COVID-19 infected patients*	Gambar X-ray dada (COVID-19, tuberkulosis, dada normal)	DenseNet-201 berbasis cloud	Mendeteksi pasien terinfeksi COVID-19 dengan akurasi 99.24% dalam 7.47 menit	Dataset yang berbeda dan menggunakan augmentasi data
10	Pradipto et al. (2023)	Perbandingan Penggunaan DenseNet201 dan YOLOv8 pada Pengembangan Sistem Klasifikasi Sampah pada Raspberry Pi 4 Menggunakan Kamera Penglihatan Malam	Gambar sampah (organik, anorganik, B3, residu)	DenseNet-201 dan YOLOv8 pada Raspberry Pi 4	DenseNet201 unggul di akurasi (97,3%), tapi YOLOv8 lebih efisien dengan akurasi 91,87%, CPU 9,83%, dan daya 6,7W.	Dataset yang berbeda dan Cuma menggunakan DenseNet-201

2.2 *Convolutional Neural Network* (CNN)

Convolutional Neural Network (CNN) adalah salah satu arsitektur *deep learning* yang dirancang khusus untuk pemrosesan data berbasis citra. CNN merupakan pengembangan dari *Artificial Neural Network* (ANN) yang lebih mampu mengenali pola dalam gambar, seperti bentuk, tekstur, dan fitur spasial, melalui proses ekstraksi fitur secara hierarkis. LeCun et al. (1998) memperkenalkan

konsep CNN dalam bentuk LeNet-5, yang pertama kali digunakan untuk mengenali tulisan tangan dalam sistem verifikasi cek. Sejak saat itu, CNN berkembang pesat dan menjadi metode utama dalam berbagai tugas klasifikasi citra, segmentasi, deteksi objek, serta pengenalan pola dalam gambar.

CNN bekerja dengan memanfaatkan lapisan konvolusi yang mengekstraksi fitur dari citra secara bertahap, mulai dari fitur sederhana seperti tepi hingga pola kompleks seperti bentuk penyakit pada daun tanaman. CNN juga memiliki keunggulan utama dalam mengenali pola visual, bahkan jika gambar mengalami rotasi, pencahayaan berbeda, atau memiliki variasi warna dan tekstur. Oleh karena itu, CNN sangat cocok digunakan dalam klasifikasi penyakit tanaman padi, di mana gejala penyakit sering kali tampak dalam bentuk bercak, perubahan warna, atau pola tertentu pada daun.

CNN terdiri dari beberapa lapisan utama yang bekerja secara berurutan untuk mengubah input citra menjadi representasi fitur yang dapat digunakan untuk klasifikasi. Berikut adalah komponen utama dalam arsitektur CNN:

1. *Input Layer* (Lapisan Masukan)

   Lapisan pertama dalam CNN adalah input *Layer*, yang menerima gambar dalam bentuk matriks piksel. Sebuah gambar berwarna (RGB) biasanya direpresentasikan dalam tiga saluran (channels): *Red* (Merah), *Green* (Hijau), dan *Blue* (Biru). Ukuran input biasanya dinormalisasi sebelum diproses agar sesuai dengan arsitektur jaringan yang digunakan.

2. *Convolutional Layer* (Lapisan Konvolusi)

   Lapisan konvolusi merupakan inti dari CNN, di mana proses ekstraksi fitur

dilakukan. Pada lapisan ini, filter (kernel) digunakan untuk melakukan operasi konvolusi terhadap gambar input, menghasilkan *feature maps*. Konvolusi bekerja dengan cara dengan melewatkan filter (kernel) berukuran kecil (misalnya 3×3 atau 5×5) ke seluruh bagian gambar. Setelah itu Menghitung *dot product* antara nilai piksel gambar dan bobot kernel untuk mengekstraksi fitur tertentu, seperti tepi, tekstur, atau pola tertentu yang menghasilkan *feature maps*, yang merupakan representasi dari fitur-fitur dalam gambar. Dengan menggunakan beberapa lapisan konvolusi, CNN dapat mengenali fitur tingkat rendah (garis, sudut) hingga fitur tingkat tinggi (bentuk objek, pola penyakit pada tanaman padi).

3. *Activation Function* (Fungsi Aktivasi)

CNN menggunakan fungsi aktivasi untuk memperkenalkan non-linearitas dalam model. Fungsi aktivasi yang paling umum digunakan adalah ReLU (*Rectified Linear Unit*), yang didefinisikan sebagai:

$$f(x) = \max(0, x) \tag{2.1}$$

Keterangan:
x = nilai input ke fungsi aktivasi.
Jika $x > 0$, maka output = **x**.
Jika $x \leq 0$, maka output = **0**.

ReLU membantu jaringan belajar lebih cepat dan mengurangi risiko *vanishing gradient* yang sering terjadi pada jaringan saraf dalam.

4. *Pooling Layer* (Lapisan Pooling)

Lapisan pooling digunakan untuk mengurangi dimensi *feature maps*, sehingga mengurangi kompleksitas komputasi serta risiko *overfitting*.

Teknik pooling yang umum digunakan adalah: *Max Pooling* yaitu Mengambil nilai maksimum dari setiap area fitur tertentu, *Average pooling* yaitu Mengambil nilai rata-rata dari area fitur tertentu.

Pooling membantu model mempertahankan fitur penting sambil mengurangi ukuran data.

5. *Fully connected Layer* (Lapisan Terhubung Penuh)

   Fully connected Layer menghubungkan hasil ekstraksi fitur ke dalam lapisan output. Di sini, probabilitas klasifikasi dihitung berdasarkan bobot dan fungsi aktivasi *softmax*, sehingga model dapat menentukan kelas penyakit tanaman padi yang sesuai.

6. *Output Layer* (Lapisan Keluaran)

   Lapisan ini memberikan hasil akhir klasifikasi, misalnya jenis penyakit tanaman padi *(Blast*, *Blight*, atau *Tungro*).

2.3 DenseNet (*Densely Connected Convolutional Networks*)

DenseNet atau *Densely Connected Convolutional Network* adalah salah satu arsitektur *Convolutional Neural Network* (CNN) yang diperkenalkan oleh Huang et al. (2017) untuk mengatasi berbagai keterbatasan pada model CNN konvensional yang sangat dalam, seperti masalah vanishing gradient dan redundansi fitur. Tidak seperti CNN tradisional atau ResNet yang hanya menghubungkan lapisan ke lapisan berikutnya (sekuensial atau dengan *skip-connection*), DenseNet secara unik menghubungkan setiap lapisan dengan semua lapisan sebelumnya dalam bentuk concatenation, bukan penjumlahan. Artinya, setiap lapisan menerima fitur dari semua lapisan sebelumnya sebagai input, dan fitur yang dihasilkannya akan

diteruskan ke semua lapisan berikutnya dalam blok yang sama.

Pendekatan ini memberikan beberapa keunggulan. Pertama, memperkuat aliran informasi dan gradien ke seluruh jaringan, sehingga pelatihan menjadi lebih stabil dan cepat bahkan untuk jaringan yang sangat dalam. Kedua, karena fitur digunakan kembali secara eksplisit antar-lapisan, DenseNet menjadi lebih hemat parameter dibandingkan arsitektur dalam lain seperti ResNet, sambil tetap mempertahankan atau bahkan meningkatkan akurasi. Ketiga, arsitektur ini memiliki efek regularisasi internal, yang mengurangi risiko *overfitting*, terutama saat jumlah data pelatihan terbatas.

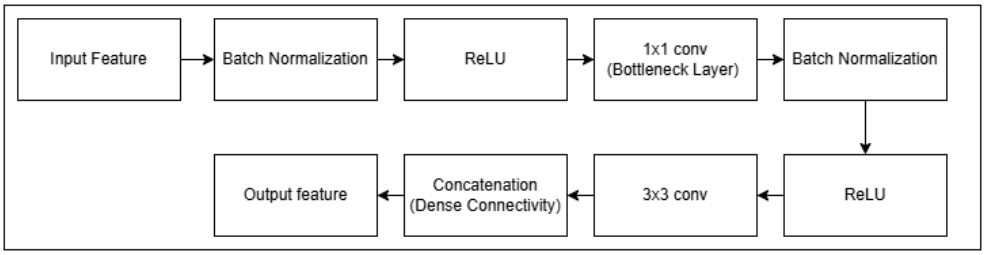
DenseNet terdiri dari beberapa blok padat (*Dense blocks*), yang dipisahkan oleh *Transition Layer*s yang melakukan downsampling menggunakan *convolution* dan pooling. Setiap blok padat mengandung sejumlah *Layer* yang setiap output-nya ditambahkan ke "*state global*" melalui concatenation. Selain itu, pengenalan *Bottleneck Layer*s (dengan 1×1 *convolution*) dan compression factor pada *Transition Layer* membuat varian DenseNet (DenseNet-BC) semakin efisien dari segi memori dan komputasi.

Dalam aplikasi klasifikasi penyakit tanaman padi, DenseNet sangat berguna karena mampu mempelajari pola bercak, perubahan warna, dan gejala penyakit lainnya pada daun tanaman. Dengan menghubungkan setiap lapisan ke lapisan berikutnya, model dapat menggunakan kembali informasi fitur yang telah dipelajari, sehingga meningkatkan akurasi dan efisiensi komputasi. Hal ini membuat DenseNet menjadi pilihan utama dalam berbagai penelitian klasifikasi citra, termasuk dalam deteksi penyakit tanaman.

DenseNet memiliki struktur unik yang membedakannya dari CNN konvensional. Arsitektur ini terdiri dari beberapa komponen utama yang bekerja bersama untuk meningkatkan efisiensi ekstraksi fitur.

1. *Dense Block*

   Setiap *Dense Block* terdiri dari beberapa lapisan utama, yaitu *Batch normalization* (BN), ReLU Activation, 1×1 *Convolution* (*Bottleneck*), 3×3 *Convolution*, dan *Concatenation*. Proses pertama dalam *Dense Block* adalah *Batch normalization*, yang berfungsi untuk menormalkan distribusi data guna mempercepat konvergensi dan mengurangi masalah *vanishing gradient*. Setelah itu, data melewati *ReLU Activation*, yang memberikan non-linearitas agar jaringan dapat belajar pola yang lebih kompleks. Selanjutnya, dilakukan 1×1 *Convolution*, yang bertindak sebagai *Bottleneck Layer* untuk mengurangi jumlah channel sebelum masuk ke lapisan konvolusi utama. Ini membantu mengurangi jumlah parameter dan meningkatkan efisiensi komputasi. Output dari setiap *Layer* dalam *Dense Block* kemudian digabungkan (*concatenation*) dengan semua output dari *Layer* sebelumnya, sehingga setiap *Layer* dapat memanfaatkan semua informasi yang telah dipelajari oleh jaringan. Pendekatan ini memastikan bahwa fitur tidak perlu dipelajari kembali, mengurangi redundansi parameter, dan meningkatkan efisiensi jaringan. Berikut Gambar 2.1 yang menggambarkan arsitektur dari *Dense Block* :



Gambar 2.1 Arsitektur *Dense Block* (Huang et al., 2017)

2. *Transition Layer*

Di antara *Dense Block*, terdapat *Transition Layer*, yang berfungsi untuk mengurangi dimensi data menggunakan teknik 1×1 *convolution* dan *average pooling*. *Transition Layer* membantu mengontrol jumlah fitur yang dikirim ke lapisan berikutnya, sehingga mencegah model menjadi terlalu kompleks dan mengurangi kebutuhan memori.

3. *Growth Rate* (k)

DenseNet memiliki parameter *Growth Rate* (k), yang menentukan jumlah fitur baru yang ditambahkan oleh setiap lapisan konvolusi. Jika k = 32, setiap lapisan akan menambahkan 32 fitur baru ke dalam jaringan, memungkinkan model untuk terus memperkaya informasi tanpa memperbesar jumlah parameter secara signifikan. Jumlah total fitur setelah l lapisan dalam *Dense Block* dapat dihitung dengan rumus:

$$F_l = F_0 + k \times (l - 1) \quad (2,2)$$

Keterangan :
F_l = jumlah total fitur pada lapisan l,
F_0 = jumlah fitur awal sebelum masuk ke *Dense Block*,
k = *Growth Rate* (misalkan 32),
l = indeks lapisan dalam *Dense Block*

4. *Bottleneck Layer*

Untuk meningkatkan efisiensi, DenseNet menggunakan *Bottleneck Layer*, yaitu 1×1 *convolution* sebelum 3×3 *convolution*. Lapisan ini membantu dalam mengurangi jumlah parameter dan meningkatkan efisiensi komputasi. Jumlah fitur yang dikurangi oleh *Bottleneck Layer* dapat dihitung dengan rumus:

$$F_{bottleneck} = 4k \tag{2,3}$$

Keterangan :
$F_{bottleneck}$ = jumlah fitur setelah 1×1 *convolution*,
k = *growth rate*.

5. *Global Average pooling* (GAP) dan *Fully connected Layer*

Sebagai langkah terakhir dalam arsitektur DenseNet, model menerapkan *Global Average pooling* (GAP) untuk merangkum informasi dari seluruh peta fitur sebelum memasuki *fully connected Layer*. Kemudian, *softmax Classifier* digunakan untuk menentukan kelas akhir, seperti klasifikasi penyakit tanaman padi (*Blast*, *Blight*, atau *Tungro*).

2.4 Augmentasi Data

Augmentasi data adalah teknik dalam pembelajaran mesin yang digunakan untuk meningkatkan jumlah dan keragaman data pelatihan dengan cara melakukan transformasi atau modifikasi pada data yang sudah ada (Shorten et al., 2019). Tujuan utama dari augmentasi data adalah untuk memperbaiki generalisasi model, sehingga model tidak hanya bergantung pada data yang terbatas, tetapi juga dapat

mengidentifikasi pola yang lebih luas dan beragam dalam dataset. Dalam konteks pengolahan citra, augmentasi data dilakukan dengan memodifikasi gambar yang ada melalui berbagai teknik, seperti rotasi, translasi, *flipping* (pembalikan horizontal atau vertikal), pemotongan, perubahan skala, dan penyesuaian pencahayaan atau kecerahan.

Teknik ini sangat berguna terutama ketika dataset yang tersedia terbatas atau tidak mencakup variasi yang cukup besar, yang dapat menyebabkan model menjadi *overfitting* yakni model terlalu menyesuaikan diri dengan data pelatihan yang ada dan gagal mengenali pola pada data baru. Augmentasi data membantu memperkaya dataset, memungkinkan model untuk belajar dari variasi yang lebih banyak, dan memperbaiki kemampuan model dalam menghadapi kondisi nyata yang beragam.

2.5 Penyakit Tanaman Padi

Penyakit pada tanaman padi dapat sangat merugikan hasil panen, salah satunya adalah penyakit *Blast* yang disebabkan oleh jamur *Pyricularia oryzae*. Penyakit ini menginfeksi bagian daun, leher malai, dan batang padi, serta dapat menyebabkan penurunan kualitas dan kuantitas hasil padi. Pada fase vegetatif, gejala awal penyakit *Blast* terlihat berupa bercak berbentuk belah ketupat berwarna coklat dengan titik putih di tengahnya. Jika tidak dikendalikan dengan baik, penyakit ini dapat meluas hingga merusak leher malai, yang mengakibatkan tangkai malai patah dan gabah menjadi hampa. Penyebaran jamur ini sangat dipengaruhi oleh kondisi kelembapan yang tinggi, serta suhu yang mendukung pertumbuhannya. Oleh karena itu, penting untuk memilih varietas padi yang tahan

terhadap penyakit *Blast* dan melakukan rotasi tanaman untuk memutuskan siklus hidup jamur penyebabnya (Distan Buleleng, 2023). Berikut ini adalah Gambar 2.2 penyakit *Blast*.

Gambar 2.2 Penyakit *Blast*

Penyakit *Blight* (Kresek) pada tanaman padi disebabkan oleh bakteri *Xanthomonas oryzae*. Bakteri ini menyerang daun tanaman padi, yang menyebabkan terjadinya gejala seperti munculnya garis-garis berwarna kuning hingga coklat pada daun. Pada stadium lanjut, daun akan mengering dan mati. Penyebaran penyakit ini dapat berlangsung sangat cepat, terutama pada kondisi lingkungan yang lembap dan suhu yang tinggi. Penyakit ini sangat merugikan karena dapat mengurangi hasil panen secara signifikan, terlebih pada tahap generatif saat tanaman sudah mulai berbunga. Bakteri penyebab *Blight* dapat menginfeksi tanaman melalui luka pada daun, yang biasanya disebabkan oleh angin atau hujan yang membawa bakteri tersebut (Plantix, n.d.). Berikut ini adalah Gambar 2.3 penyakit *Blight*.

Gambar 2.3 Penyakit *Blight*

Penyakit *Tungro* pada tanaman padi disebabkan oleh infeksi ganda dua jenis virus, yaitu *Rice Tungro Spherical Virus* (RTSV) *dan Rice Tungro Bacilliform Virus* (RTBV). Penyakit ini ditularkan oleh vektor berupa wereng hijau (*Nephotettix virescens*), yang menyebarkan virus saat menghisap cairan tanaman. Gejala penyakit *Tungro* meliputi perubahan warna daun menjadi kuning oranye, yang dimulai dari ujung daun dan kemudian menyebar ke seluruh bagian daun. Selain itu, tanaman yang terinfeksi akan mengalami kerdil dan anakan yang terbatas. Pada stadium lanjut, tanaman padi yang terinfeksi virus ini akan tumbuh sangat lambat dan akhirnya menghasilkan gabah yang hampa (Distan Buleleng, 2023). Berikut ini adalah Gambar 2.4 penyakit *Tungro*.

Gambar 2.4 Penyakit *Tungro*

BAB III

DESAIN DAN IMPLEMENTASI

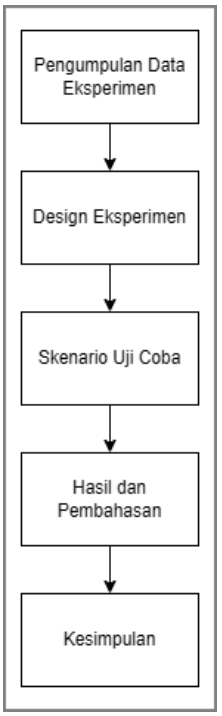
3.1 Prosedur Penelitian

Prosedur penelitian ini dirancang untuk mengembangkan model klasifikasi penyakit tanaman padi dengan memanfaatkan *Convolutional Neural Network* (CNN) menggunakan arsitektur DenseNet-201 serta teknik augmentasi data guna meningkatkan performa model. Langkah-langkah penelitian ini dibagi menjadi beberapa tahapan utama, mulai dari pengumpulan data, *preprocessing* data, implementasi model, pelatihan model, hingga evaluasi performa model.

Pendekatan *deep learning* dipilih dalam penelitian ini karena kemampuannya dalam mengenali pola yang kompleks dalam citra digital. CNN, sebagai salah satu arsitektur *deep learning* yang paling populer dalam image *Class*ification, mampu secara otomatis mengekstraksi fitur dari gambar, sehingga dapat memberikan akurasi yang lebih tinggi dibandingkan metode tradisional. Arsitektur DenseNet-201 dipilih karena memiliki karakteristik unik dalam memanfaatkan kembali fitur dari setiap lapisan sebelumnya, sehingga dapat meningkatkan efisiensi model dan mencegah *vanishing gradient* yang sering terjadi pada jaringan yang sangat dalam.

Penelitian ini dilakukan dengan pendekatan eksperimen, di mana model akan dilatih dengan berbagai kombinasi teknik augmentasi untuk memperoleh hasil terbaik. Setiap tahap penelitian akan dilakukan secara sistematis, dengan tujuan agar model yang dihasilkan dapat diimplementasikan secara praktis dalam

membantu petani dan ahli pertanian dalam mengidentifikasi penyakit pada tanaman padi. Gambar 3.1 memperlihatkan tahapan prosedur penelitian.



Gambar 3.1 Alur Penelitian

Proses penelitian ini dimulai dengan pengumpulan data, yaitu mengumpulkan dataset citra penyakit tanaman padi dari sumber publik dan referensi penelitian, serta memastikan distribusi kelas yang seimbang agar model tidak mengalami bias dalam pelatihan. Selanjutnya, dilakukan *preprocessing* data, yang mencakup augmentasi menggunakan teknik rotasi, *flipping*, *zooming*, dan penyesuaian cahaya untuk meningkatkan variasi dataset, mengubah ukuran gambar menjadi 224×224 piksel agar sesuai dengan format input CNN, serta melakukan normalisasi piksel ke rentang 0 hingga 1 guna meningkatkan stabilitas pelatihan. Setelah itu, dataset dibagi menjadi *training* set dan *testing* set untuk memastikan model dievaluasi dengan baik. Pada tahap desain sisten, alur kerja dari input gambar

hingga klasifikasi dirancang menggunakan CNN berbasis DenseNet-201, dengan menentukan jumlah *Dense blocks*, *Transition Layer*s, serta parameter yang digunakan dalam *transfer learning*. Implementasi model CNN dilakukan dengan menerapkan *transfer learning* dari bobot *pre-trained ImageNet*, diikuti dengan modifikasi *Fully connected Layer* terakhir agar sesuai dengan jumlah kelas dalam dataset penyakit padi. Model kemudian dilatih menggunakan optimasi hyperparameter, termasuk *learning rate*, *batch size*, dan *optimizer*, serta menggunakan *early stopping* untuk mencegah *overfitting*. Evaluasi dilakukan dengan menganalisis metrik akurasi, *Precision*, *Recall*, *F1-score*, serta menggunakan *multiple confusion matrix* untuk memahami distribusi kesalahan prediksi. Terakhir, hasil model dibandingkan dengan dan tanpa augmentasi data serta transfer learning untuk menganalisis dampak setiap metode yang diterapkan dalam meningkatkan performa klasifikasi penyakit tanaman padi.

3.2 Data Eksperimen yang Digunakan

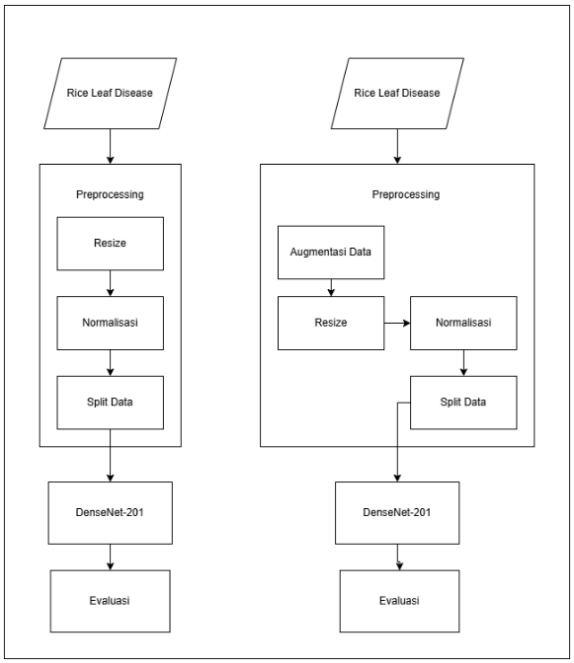
Pada penelitian ini, dataset yang digunakan berasal dari *Kaggle*, dengan judul **"**Leaf Rice Disease Indonesia**"**, yang disediakan oleh tedisetiady (Setiady, 2024). Dataset ini berisi gambar daun padi dari berbagai kondisi dan penyakit yang umum ditemukan di Indonesia, Dataset ini memiliki total 240 gambar, yang dikategorikan ke dalam tiga kelas penyakit. Dataset ini memiliki jumlah data yang cukup untuk digunakan dalam pelatihan model *deep learning*, tetapi tetap memerlukan proses augmentasi data guna meningkatkan variasi gambar dan menghindari *overfitting.* Berikut detail kelasnya pada Tabel 3.1

Tabel 3.1 Detail Kelas Data

Kelas	penyakit	jumlah
A	*Blast*	80
B	*Blight*	80
C	*Tungro*	80
Jumlah		240

3.3 Desain Sistem

Tahap desain sistem merupakan tahap yang digunakan peneliti untuk merencanakan alur penelitian yang terstruktur. Alur penelitian tersebut dapat dilihat pada Gambar 3.2 dibawah ini.



Gambar 3.2 Design Sistem 1 dan Sistem 2

3.4 *Preprocessing*

Preprocessing data merupakan tahap penting dalam penelitian ini untuk memastikan bahwa data citra yang digunakan dalam pelatihan model berada dalam kondisi optimal. Langkah-langkah *preprocessing* ini mencakup augmentasi data,

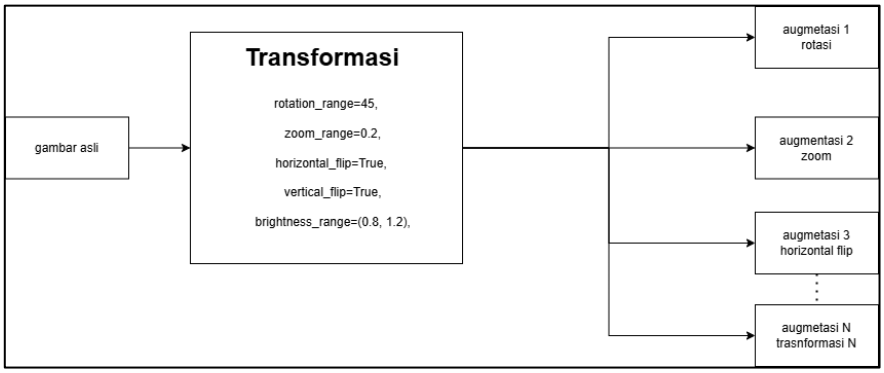
resizing, normalisasi piksel, dan pembagian dataset. Dengan melakukan *preprocessing* yang tepat, model *Convolutional Neural Network* (CNN) berbasis DenseNet-201 dapat dilatih dengan lebih baik dan memiliki generalisasi yang lebih baik terhadap data baru.

3.4.1 Augmentasi Data

Karena jumlah dataset yang terbatas dan kemungkinan ketidakseimbangan jumlah Gambar dalam setiap kelas, teknik augmentasi data diterapkan untuk meningkatkan variasi Gambar yang tersedia. Augmentasi data bertujuan untuk membuat model lebih tahan terhadap perubahan kondisi pencahayaan, rotasi, dan transformasi lain yang bisa terjadi di dunia nyata. Dalam penelitian ini, digunakan *library TensorFlow* dan *Keras* yang menyediakan beragam fungsi augmentasi seperti rotation, *zoom*, *flipping*, dan *brightness adjustment* secara efisien dan mudah diimplementasikan, sehingga mendukung proses pelatihan model yang lebih robust dan generalizable (Abadi et al., 2016).

Selain itu, fungsi *ImageDataGenerator* dari *Keras* melakukan transformasi tersebut secara acak dan langsung (*real-time*) saat pelatihan berlangsung (Paper, D.J., 2021). Artinya, setiap gambar yang diumpankan ke dalam model dapat mengalami satu atau lebih transformasi namun terhitung 1 kali transformasi sekaligus dalam satu *batch* pelatihan. Misalnya, gambar 1 bisa hanya mengalami rotasi saja, gambar 2 mengalami kombinasi rotasi dan flip horizontal, sedangkan gambar 3 mungkin dikenai rotasi, *zoom*, dan penyesuaian cahaya secara bersamaan. Variasi acak ini membuat model belajar dari banyak kemungkinan kondisi citra, sehingga menghasilkan kemampuan generalisasi yang lebih baik terhadap data baru

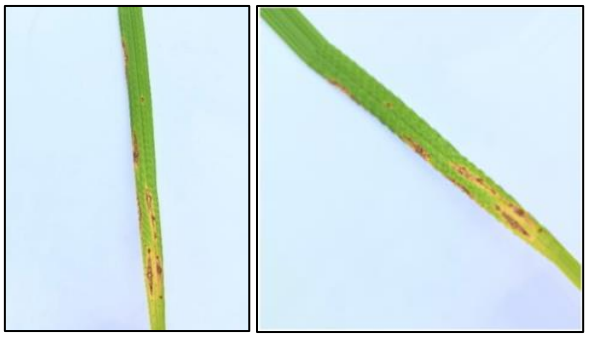
di dunia nyata (Chollet, 2018). Berikut adalah scenario transformasi augmentasi data pada *ImageDataGenerator* pada Gambar 3.3.



Gambar 3.3 Skenario Tranformasi Augmentasi Data

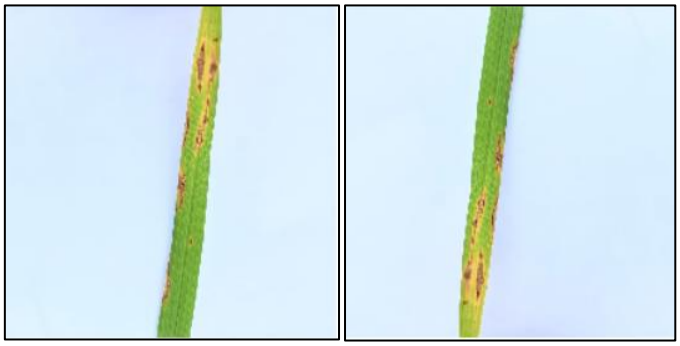
Beberapa jenis Teknik augmentasi yang digunakan dalam penelitian ini adalah:

1. Rotasi (*Rotation*): gambar diputar secara acak dalam hingga 45° untuk meningkatkan variasi posisi penyakit dalam gambar. Berikut hasil transformasinya pada Gambar 3.4:



Gambar 3.4 Contoh Penggunaan Rotasi Sebelum dan Sesudah

2. *Flipping (Horizontal & Vertical Flip)*: digunakan untuk mencerminkan gambar secara horizontal, membantu model mengenali pola dari berbagai sudut. Sedangkan *Flipping vertical* juga diterapkan untuk meningkatkan keragaman data. Berikut hasil transformasinya pada Gambar 3.5:



Gambar 3.5 Contoh Penggunaan *Flip Vertical* dan *Horizontal*

3. *Zooming (Zoom In & Zoom Out)*: *Zoom* acak dalam rentang 80% hingga 120% dari ukuran asli diterapkan untuk memastikan model tetap dapat mengenali fitur penyakit pada berbagai skala. Berikut hasil transformasinya pada Gambar 3.6

Gambar 3.6 Contoh Penggunaan *Zooming*

4. *Brightness adjustment* (Penyesuaian Kecerahan): Kecerahan gambar diubah dalam rentang -20% hingga +20% untuk mensimulasikan variasi pencahayaan alami yang dapat terjadi di lapangan. Berikut hasil transformasinya pada Gambar 3.7:

Gambar 3.7 Contoh Penggunaan Brigtness Adjustment

3.4.2 *Resize* Gambar

Semua gambar dalam dataset diubah ukurannya menjadi 224×224 piksel untuk menyesuaikan dengan input *Layer* dari arsitektur DenseNet-201. Ukuran ini dipilih karena merupakan ukuran standar untuk model yang telah dilatih sebelumnya pada dataset *ImageNet*, yang akan digunakan dalam transfer learning.

3.4.3 Normalisasi Piksel

Setiap gambar yang digunakan dalam penelitian ini memiliki nilai piksel dalam rentang 0 hingga 255, yang merupakan skala intensitas warna dalam format RGB (*Red, Green, Blue*). Jika nilai piksel tidak dinormalisasi, perbedaan skala yang besar antara nilai piksel dapat menghambat kinerja model *Convolutional Neural Network* (CNN). Oleh karena itu, dilakukan normalisasi agar nilai piksel berada dalam rentang [0,1] atau [-1,1] sesuai dengan arsitektur model yang digunakan. Berikut adalah rumus normalisasi *Min-Max* :

$$X_{norm} = \frac{X_i - X_{min}}{X_{max} - X_{min}} \quad (3.1)$$

Keterangan :
X_i = nilai asli dari fitur
X_{min} = nilai minimum dari fitur tersebut
X_{max} = nilai maksimum dari fitur tersebut
X_{norm} = nilai fitur yang sudah dinormalisasi

3.4.4 Split Data

Setelah dilakukan *preprocessing* seperti augmentasi data**,** *resizing* gambar dan normalisasi piksel, langkah selanjutnya adalah pembagian dataset (*data split*). Pembagian dataset bertujuan untuk memastikan bahwa model *Convolutional Neural Network* (CNN) dengan arsitektur DenseNet-201 dapat dievaluasi secara obyektif dan memiliki generalisasi yang baik terhadap data baru. Pembagian dataset yang tepat sangat penting untuk menghindari bias dalam pelatihan dan memastikan bahwa model tidak mengalami *overfitting* atau *underfitting*.

Dalam penelitian ini, dataset dibagi menjadi dua bagian utama, yaitu *training set* dan *testing set*. *Training set* digunakan untuk melatih model, sehingga model dapat belajar mengenali pola penyakit dalam gambar. *Testing set* digunakan untuk menguji performa akhir model setelah proses pelatihan selesai, menggunakan data yang belum pernah dilihat oleh model sebelumnya.

Dalam penelitian ini, dilakukan uji skenario pembagian dataset untuk memastikan performa optimal model *Convolutional Neural Network* (CNN) dengan arsitektur DenseNet-201 dalam klasifikasi penyakit padi. Dataset dibagi menjadi *training set* (60-80%), dan *testing set* (20-40%), yang merupakan pendekatan umum dalam pembelajaran mesin untuk mencegah *overfitting* dan meningkatkan generalisasi model. Pada skenario pertama, digunakan 60% data

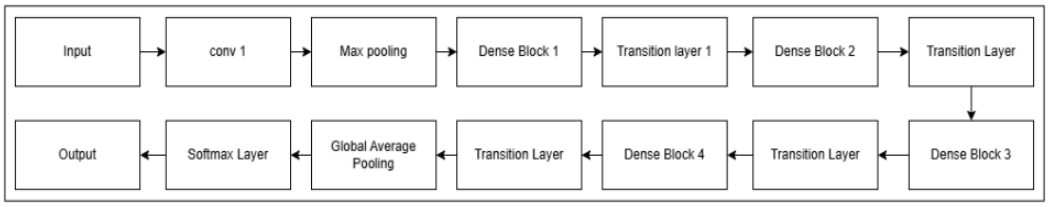
untuk *training* dan 40% untuk *testing*, yang bertujuan untuk memperbanyak jumlah data *testing* guna mendapatkan evaluasi yang lebih representatif dari performa model (Setiady, 2021). Skenario kedua menggunakan 70% data untuk *training* dan 30% untuk *testing*, yang bertujuan untuk menyeimbangkan antara pelatihan dan evaluasi guna menghindari bias dalam pengujian (Chen et al., 2020). Skenario ketiga menggunakan 80% data untuk *training* dan 20% untuk *testing*, yang bertujuan untuk memberikan lebih banyak data latih bagi model sehingga pola penyakit lebih mudah dikenali (Putri et al., 2024). Setelah model selesai dilatih, evaluasi akhir dilakukan menggunakan *testing* set, yang berisi data baru yang tidak pernah digunakan dalam pelatihan sebelumnya, untuk mengukur akurasi, presisi, *Recall*, dan *F1-score* dari model. Dengan pengujian pada skenario yang berbeda, model dapat dipastikan memiliki performa yang stabil serta mampu menggeneralisasi data baru dengan baik, sehingga dapat digunakan untuk klasifikasi penyakit padi secara otomatis.

3.5 Implementasi DenseNet-201

Implementasi model *Convolutional Neural Network* (CNN) berbasis DenseNet-201 merupakan tahap utama dalam penelitian ini untuk melakukan klasifikasi penyakit tanaman padi, yaitu *Blast*, *Blight*, dan *Tungro*. DenseNet-201 dipilih karena memiliki keunggulan dalam meningkatkan propagasi informasi antar *Layer*, mengurangi jumlah parameter yang diperlukan, serta meningkatkan efisiensi pelatihan model dibandingkan arsitektur CNN konvensional.

Selain itu, teknik transfer learning diterapkan dalam penelitian ini, di mana bobot model DenseNet-201 yang telah dilatih pada dataset *ImageNet* digunakan

sebagai bobot awal, sementara lapisan klasifikasi akhir dimodifikasi agar sesuai dengan jumlah kelas dalam dataset penyakit padi. Proses ini difasilitasi oleh *library TensorFlow* dan *Keras* yang menyediakan akses mudah terhadap arsitektur pre-*train*ed models seperti DenseNet-201 melalui modul *TensorFlow.keras.applications*. *Library* ini memungkinkan pengguna untuk melakukan *fine-tuning*, pembekuan *Layer*, serta penyesuaian arsitektur dengan efisien dan terstruktur, sehingga mempercepat proses eksperimen dan pengembangan model (Huang et al., 2017). Detailnya ada pada Gambar 3.8 arsitekturnya sebagai berikut :



Gambar 3.8 Arsitektur DenseNet-201 (Huang et al., 2017)

Arsitektur DenseNet-201 yang diterapkan dalam penelitian ini terdiri dari beberapa komponen utama, yaitu:

1. *Input Layer*: Menerima citra dengan ukuran 224×224 piksel dan 3 kernel warna (RGB).
2. *Convolutional Layer* (7×7, *stride*=2): Digunakan untuk melakukan ekstraksi fitur awal dari citra dengan filter konvolusi ukuran 7×7.
3. *Pooling Layer* (3×3 *Max Pooling*, *stride*=2) :Digunakan untuk mengurangi dimensi citra dengan tetap mempertahankan fitur penting.
4. *Dense Block* dengan formulasi matematis untuk konektivitas dalam *Dense Block* sebagai berikut:

$$x_l = H_l([x_0, x_1, \dots, x_{l-1}]) \quad (3.2)$$

Keterangan :
x_l = output dari lapisan ke-*l,*
H_l = transformasi pada *Layer* (*l*) (kombinasi *Batch normalization*, ReLU, *Convolution*),
$[x_0, x_1, \dots, x_{l-1}]$ = concatenation dari semua fitur sebelum *Layer l*

Untuk *Dense Block* secara detail, antara lain *Dense Block* 1 (6 *Layer*s)*,* *Dense Block* 2 (12 *Layer*s)*, Dense Block* 3 (48 *Layer*s)*,* dan *Dense Block* 4 (32 *Layer*s). Setiap *Dense Block* terdiri dari beberapa *Layer convolutiona*l yang terhubung satu sama lain, memungkinkan informasi dari setiap lapisan digunakan kembali di seluruh jaringan.

5. *Transition Layers*: Setelah setiap *Dense Block*, terdapat *Transition Layer*s yang terdiri dari 1×1 *convolution* dan 2×2 *average pooling* untuk mengurangi dimensi fitur sebelum masuk ke *Dense Block* berikutnya. Jumlah fitur setelah *Transition Layer* dihitung menggunakan *Compression Factor* (θ), dengan rumus:

$$F_{transition} = \theta \times F_{input} \quad (2.3)$$

Keterangan :
$F_{transition}$ = jumlah fitur setelah *Transition Layer*,
θ = *Compression Factor* (0,5),
F_{input} = jumlah fitur sebelum *Transition Layer*

6. *Global Average pooling Layer*: Berfungsi untuk mereduksi ukuran fitur menjadi satu vektor sebelum masuk ke *fully connected Layer*. Perhitungan GAP dilakukan dengan mengambil rata-rata dari semua nilai dalam feature map:

$$GAP_c = \frac{1}{N}\sum_{i=1}^{N} x_{x,c} \quad (2.6)$$

Keterangan :
GAP_c = nilai rata-rata untuk channel *c*,
$x_{x,c}$ = nilai dari feature map pada posisi *i* untuk channel *c*,
N = jumlah total elemen dalam feature map (misalnya 7×7 = 49).

Setelah GAP, outputnya adalah 1×1×F, di mana F adalah jumlah fitur yang tersisa sebelum masuk ke *fully connected Layer*.

7. *Fully connected Layer* (*Softmax Activation Function*): *Softmax Classifier* digunakan untuk melakukan klasifikasi ke dalam tiga kategori penyakit tanaman padi dengan rumus:

$$P(y = c|x) = \frac{e^{z_c}}{\sum_j e^{z_j}} \quad (2,7)$$

Keterangan :
$P(y = c|x)$ = probabilitas bahwa gambar *x* termasuk dalam kelas *c*,
z_c = output dari *Fully connected Layer* untuk kelas *c*,
$\sum_j e^{z_j}$ = jumlah dari semua eksponensial nilai output untuk semua kelas.

3.6 Skenario Uji Coba

Uji skenario dilakukan untuk mengevaluasi kinerja model klasifikasi penyakit tanaman padi menggunakan arsitektur DenseNet-201 berdasarkan berbagai kombinasi *learning rate* dan jumlah epoch. Pengujian ini bertujuan untuk menemukan konfigurasi terbaik yang menghasilkan akurasi tertinggi tanpa menyebabkan *overfitting* atau *underfitting*.

Dalam penelitian ini, dilakukan tiga model split data seperti pada Tabel 3.1. Total data yang digunakan adalah 240 gambar, dengan setiap kelas penyakit (*Blast*, *Blight*, dan *Tungro*) memiliki 80 gambar, serta menggunakan parameter yang tercantum pada Tabel 3.2. Untuk meningkatkan keandalan evaluasi, proses pelatihan menggunakan teknik *Stratified K-Fold Cross Validation* dengan jumlah 5 *fold*, yang memastikan proporsi kelas seimbang pada setiap lipatan data, sehingga

hasil evaluasi model menjadi lebih adil dan representatif. *Stratified K-Fold Cross Validation* adalah teknik validasi silang yang membagi dataset ke dalam K bagian (*fold*) sambil menjaga proporsi jumlah data di tiap kelas tetap seimbang pada setiap *fold*. Ini sangat berguna ketika data terbatas atau tidak seimbang, karena memastikan bahwa tiap subset data (*train* dan validasi) merepresentasikan distribusi label yang serupa dengan dataset asli. Berikut adalah contoh pembagian data ke dalam *fold* pada Gambar 3.9

Gambar 3.9 *Stratified K-Fold Cross Validation* (www.researchgate.net)

Evaluasi model dilakukan dengan melihat akurasi serta analisis menggunakan *multiple confusion matrix* yang memberikan gambaran lebih jelas mengenai distribusi prediksi model.

Tabel 3.2 Split Data

Skenario	*Training*	*Testing*
A	60	40
B	70	30
C	80	20

Tabel 3.3 Parameter

Parameter	
Augmentasi Data	Rotasi, *flipping*, *zoom*ing, *brightness adjustment*
Learning rate	0.0001 - 0.001
Optimizer	*Adam*
Batch size	16
Dropout rate	0.2

Dalam penelitian ini, dilakukan 6 skenario pengujian dengan sebelum dan sesudah menggunakan augmentasi data, sementara parameter lainnya tetap konstan seperti pada Tabel 3.3. jadi pada setiap model akan diuji coba sebelum dan sesudah menggunakan augmentasi data.

1. Uji coba skenario A1, tanpa menggunakan augmentasi data.
2. Uji coba skenario A2, menggunakan augmentasi data.
3. Uji coba skenario B1, tanpa menggunakan augmentasi data.
4. Uji coba skenario B2, menggunakan augmentasi data.
5. Uji coba skenario C1, tanpa menggunakan augmentasi data.
6. Uji coba skenario C2, menggunakan augmentasi data.

Confusion matrix adalah sebuah matriks yang digunakan untuk menilai kinerja model klasifikasi dalam machine learning. Matriks ini memberikan gambaran tentang bagaimana model membandingkan label aktual dengan prediksi yang dihasilkan. Dalam kasus klasifikasi biner, *confusion matrix* biasanya direpresentasikan dalam bentuk tabel 2×2 seperti pada Tabel 3.4, yang terdiri dari empat komponen utama (Luque et al., 2019).

Tabel 3.4 *Confusion matrix*

Actual-Class/ Ground Truth	***Predicted Class***	
	Positive (P)	***Negative (N)***
Positive (P)	*True Positive (TP)*	*False Negative (FN)*
Negative (N)	*False Positive (FP)*	*True Negative (TN)*

a. *True Positive (TP)*: Data yang diklasifikasikan dengan benar sebagai kategori positif.

b. *True Negative (TN)*: Data yang diklasifikasikan dengan benar sebagai kategori negatif.

c. *False Positive (FP)*: Data yang seharusnya negatif tetapi diklasifikasikan secara salah sebagai positif (dikenal sebagai kesalahan *Type I Error*).

d. *False Negative (FN)*: Data yang seharusnya positif tetapi diklasifikasikan secara salah sebagai negatif (dikenal sebagai kesalahan *Type I Error*).

Dengan menggunakan *confusion matrix*, beberapa metrik evaluasi dapat dihitung untuk menilai seberapa baik model dalam melakukan klasifikasi. Metrik-metrik ini mencakup akurasi, *Precision*, *Recall*, dan *F1-score*, yang memberikan informasi lebih mendalam tentang kinerja model, terutama dalam menangani kesalahan klasifikasi.

Akurasi (*Accuracy*) adalah metrik yang mengukur seberapa sering model membuat prediksi yang benar dibandingkan dengan total jumlah prediksi. Akurasi dihitung dengan rumus:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{3.2}$$

Selain akurasi, metrik lain yang penting adalah *Precision*, yang mengukur seberapa tepat model dalam memprediksi kategori positif. recision memberikan gambaran tentang proporsi prediksi positif yang benar-benar termasuk dalam kategori positif. Jika *Precision* tinggi, berarti model jarang membuat kesalahan dalam mengklasifikasikan data negatif sebagai positif (*False Positive* rendah). *Precision* sangat penting dalam situasi di mana kesalahan klasifikasi positif dapat memiliki konsekuensi yang besar, seperti dalam diagnosis penyakit. *Precision* dihitung dengan rumus:

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

Sementara itu, *Recall* atau yang dikenal juga sebagai Sensitivity atau *True Positive* Rate, mengukur kemampuan model dalam menangkap seluruh data yang benar-benar positif. *Recall* berguna untuk mengetahui seberapa banyak kasus positif yang benar-benar teridentifikasi oleh model. Jika *Recall* tinggi, berarti model jarang mengklasifikasikan data positif sebagai negatif (*False Negative* rendah). *Recall* sangat penting dalam skenario seperti klasifikasi penyakit atau penklasifikasian anomali, di mana kehilangan satu kasus positif dapat berdampak besar. *Recall* dihitung dengan rumus:

$$Recall = \frac{TP}{TP+FN} \quad (3,4)$$

Untuk mendapatkan keseimbangan antara *Precision* dan *Recall*, digunakan metrik *F1-score*, yang merupakan rata-rata harmonis dari kedua metrik tersebut. *F1-score* menjadi metrik yang sangat berguna ketika dataset memiliki ketidakseimbangan kelas, di mana akurasi saja tidak cukup untuk menilai kinerja model. Jika nilai *F1-score* tinggi, berarti model memiliki keseimbangan yang baik antara menghindari *False Positives* dan *False Negative*s. *F1-score* dihitung dengan rumus:

$$F1 - Score = \frac{2 \times Precision \times Recall}{Precision+Recall} \quad (3,5)$$

BAB IV

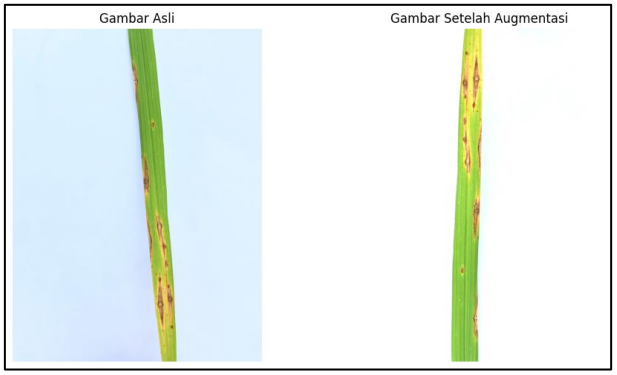
HASIL DAN PEMBAHASAN

4.1 Augmentasi Data

Berdasarkan pada subbab 3.4.1, augmentasi ini sangat penting karena membantu model untuk belajar dari berbagai kondisi pencahayaan, posisi objek, serta ukuran penyakit pada gambar daun padi yang terinfeksi. Semua gambar yang digunakan dalam penelitian ini mengalami proses augmentasi untuk memastikan bahwa model dapat mengenali penyakit padi dari berbagai sudut pandang dan variasi kondisi. Untuk total gambar sebelum dan sesudah gambar di augmentasi terdapat pada Tabel 4.1. dan Gambar 4.1 sebelum dan sesudah augmentasi data.

Tabel 4.1 Tabel Jumlah Data Sebelum dan Sesudah di Augmentasi

Skenario	Jumlah Gambar	
	Non-augmentasi data	Augmentasi data
A	144	1440
B	168	1680
C	172	1720



Gambar 4.1 Foto Daun Padi yang Di-Augmentasi

4.2 Implemetasi Code Arsitektur DenseNet-201

Pada proses pembuatan dan kompilasi model klasifikasi citra berbasis arsitektur DenseNet201 dengan transfer learning menggunakan pustaka *Keras*.

Model awal (*base_model*) memuat DenseNet201 dengan bobot pralatih dari *ImageNet*, tanpa bagian klasifikasi akhirnya (include_top=False) dan menerima input gambar berukuran 224x224 piksel dengan 3 channel warna (RGB). Output dari *base model* kemudian diproses melalui beberapa *Layer GlobalAveragePooling2D* untuk meratakan output spasial, *Dense Layer* dengan 1024 *neuron* dan aktivasi ReLU untuk menambah kapasitas klasifikasi, dan Dropout dengan rasio 0.2. *Layer* akhir menggunakan *Dense* dengan jumlah *neuron* sesuai jumlah kelas dan aktivasi *softmax*, yang menghasilkan *probabilitas* untuk tiap kelas. berikut kode program pada Gambar 4.2:

```
# Model dan kompilasi
base_model = DenseNet201(weights='imagenet', include_top=False, input_shape=(224, 224, 3))
x = base_model.output
x = GlobalAveragePooling2D()(x)
x = Dense(1024, activation='relu')(x)
x = Dropout(0.2)(x)
predictions = Dense(train_generator.num_classes, activation='softmax')(x)

model = Model(inputs=base_model.input, outputs=predictions)

for layer in base_model.layers:
    layer.trainable = False

model.compile(optimizer=Adam(learning_rate=0.0001), loss='categorical_crossentropy', metrics=['accuracy'])
```

Gambar 4.2 Implementasi DenseNet-201 dalam Bentuk Kode Program

Model akhir dibentuk dengan fungsi Model dari *Keras*, yang menggabungkan input dari *base model* dengan *output Layer* klasifikasi baru. Selanjutnya, seluruh *Layer* dari DenseNet201 dibekukan (*Layer.trainable = False*), sehingga bobotnya tidak diperbarui saat pelatihan. Ini merupakan strategi transfer learning yang umum digunakan untuk memanfaatkan fitur yang telah dipelajari dari dataset besar seperti *ImageNet*. Terakhir, model dikompilasi menggunakan *optimizer Adam* dengan *learning_rate*=0.0001, fungsi *loss categorical_crossentropy* karena klasifikasi multi-kelas, dan metrik accuracy untuk evaluasi performa selama pelatihan.

4.3 Pengujian Hasil

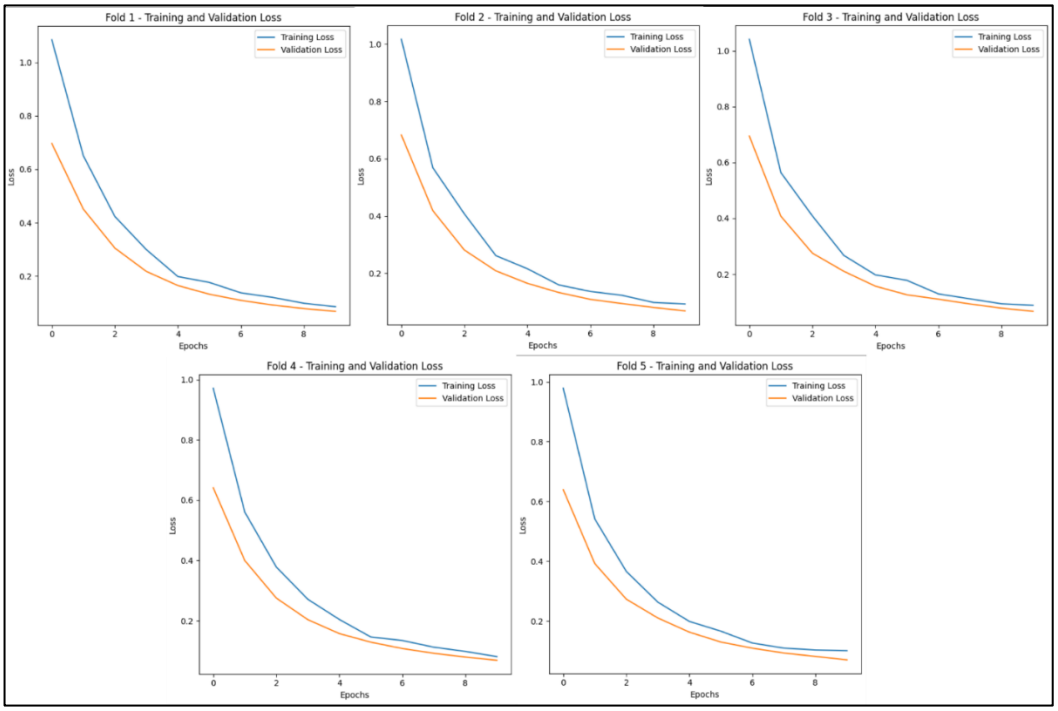
Pengujian hasil dilakukan dengan membandingkan kinerja model dari skenario tanpa dan dengan augmentasi data menggunakan DenseNet-201. Berdasarkan Bab 3.6, Proses pelatihan menggunakan teknik *Stratified K-Fold Cross Validation* dengan jumlah 5 *fold*, setelah dilakukan pengujian menggunakan beberapa skenario model yang berbeda dengan konfigurasi yang sama, untuk melihat perbedaan kinerja model. Skenario model yang tidak menggunakan augmentasi yaitu A1, B1, dan C1 sedangkan skenario model yang menggunakan augmentasi data yaitu A2, B2 dan C2.

4.3.1 Pengujian Tanpa Augmentasi Data

Dalam pengujian ini, dilakukan pengujian terhadap tiga skenario model yang tidak menggunakan augmentasi data, untuk membandingkan hasil klasifikasi penyakit padi dalam kondisi asli tanpa variasi tambahan pada dataset.

4.3.1.1 Skenario A1

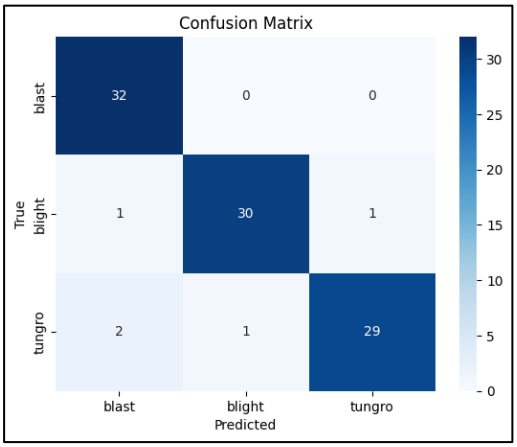
Pada pengujian skenario model A1, dilakukan dengan pembagian data *training* dan data *testing* sebesar 60% : 40%, yang mana sebesar 144 gambar untuk dan 96 gambar . Berikut hasil dari *training* model A1 pada Gambar 4.3



Gambar 4.3 Setiap *Fold Training loss* Model A1

Model ini menunjukkan perilaku konvergen karena *training loss* dan *validation loss* mengalami penurunan yang stabil sepanjang epoch di setiap *fold*. Pada awal pelatihan, *training loss* dan *validation loss* menurun dengan cepat, kemudian mengalami penurunan yang lebih lambat seiring berjalannya waktu, yang menunjukkan bahwa model semakin baik dalam mempelajari pola dari data. Tidak ada lonjakan atau fluktuasi signifikan pada *loss*, yang mengindikasikan bahwa model tidak mengalami divergence. Selain itu, *validation loss* yang terus menurun mendekati angka yang rendah di setiap *fold* menunjukkan bahwa model mampu menggeneralisasi dengan baik pada data yang tidak terlihat, yang merupakan tanda konvergensi yang baik. Proses pelatihan yang stabil ini menunjukkan bahwa model telah mencapai titik stabil dan tidak *overfitting*, sehingga dapat disimpulkan bahwa

model ini konvergen. *Confusion matrix* dari pengujian ini dapat dilihat pada Gambar 4.4 dan Tabel 4.2.



Gambar 4.4 *Confusion matrix* Model A1 Tanpa Augmentasi Data

Tabel 4.2 Hasil *Confision Matrix* Model A1 Tanpa Augmentasi Data

Class	TP	TN	FN	FP
Blast	32	61	0	3
Blight	30	63	2	1
Tungro	29	63	3	1

Dari hasil *confusion matrix* diperoleh perhitungan akurasi, presisi, *recall* dan *f1-score* yang dapat dilihat pada Tabel 4.3.

Tabel 4.3 *Confision Matrix* Model A1 Tanpa Augmentasi Data

Class	presisi	*Recall*	*f1-score*	akurasi
Blast	91%	100%	96%	97%
Blight	97%	94%	95%	97%
Tungro	97%	91%	94%	96%
Average				95%

Berdasarkan Tabel 4.2, Untuk kelas *Blast* model menghasilkan *True Positives* (TP) sebanyak 32, yang berarti ada 32 data yang benar-benar termasuk dalam kelas *Blast* dan diprediksi dengan benar oleh model sebagai *Blast*. *True Negatives* (TN) untuk kelas ini adalah 61, yaitu data dari kelas *Blight* dan *Tungro*

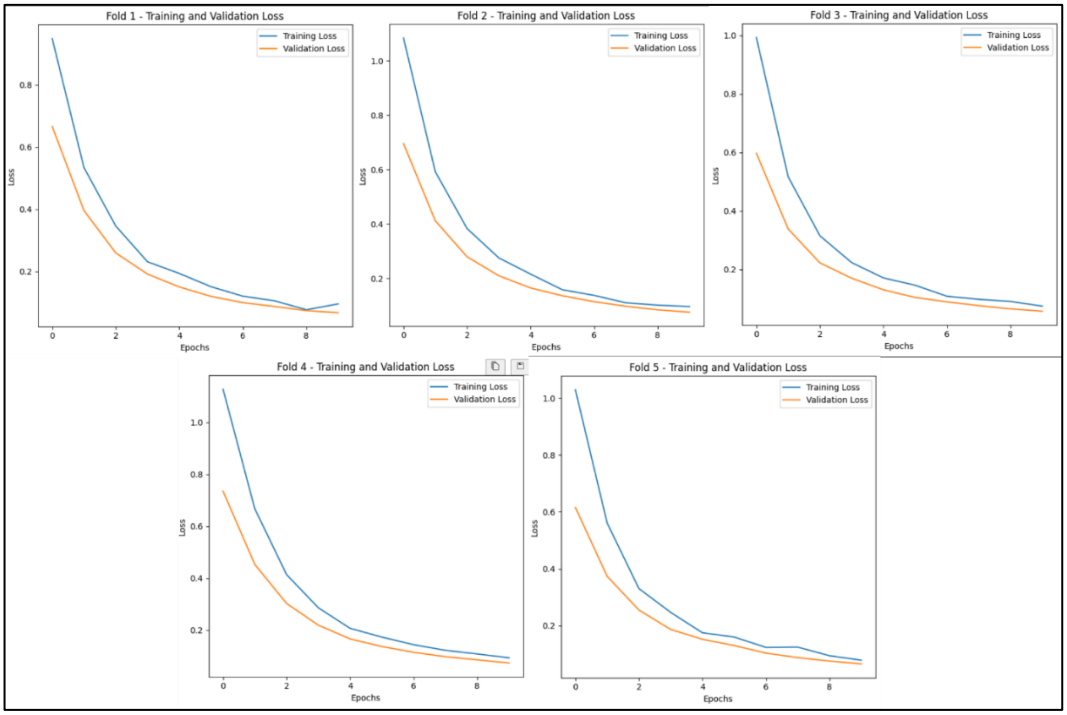
yang diprediksi dengan benar sebagai bukan *Blast*. Sementara itu, *False Negative* (FN) untuk kelas *Blast* adalah 0, yang berarti tidak ada data yang sebenarnya kelas *Blast* yang diprediksi sebagai *Blight* dan *Tungro*. *False Positives* (FP) berjumlah 3, yaitu data yang diprediksi sebagai kelas *Blast* tetapi salah prediksi masing-masing dari 1 kelas *Blight* dan 2 *Tungro* yang benar.

Untuk kelas *Blight*, model menghasilkan *True Positives* (TP) sebanyak 30, yang berarti ada 30 data yang benar-benar termasuk dalam kelas *Blight* dan diprediksi dengan benar oleh model sebagai *Blight*. *True Negatives* (TN) untuk kelas ini adalah 63, yang mencakup data dari kelas *Blast* dan *Tungro* yang diprediksi dengan benar sebagai bukan *Blight*. *False Negative* (FN) berjumlah 2, yang menunjukkan ada 2 data dari kelas *Tungro* yang salah diprediksi sebagai kelas *Blast* dan *Tungro*. Sementara itu, *False Positive* (FP) berjumlah 1, yaitu data yang sebenarnya termasuk dalam kelas *Tungro* tetapi salah diprediksi sebagai *Blight*.

Untuk kelas *Tungro*, model menghasilkan *True Positives* (TP) sebanyak 29, yang berarti ada 29 data yang benar-benar termasuk dalam kelas *Tungro* dan diprediksi dengan benar oleh model sebagai *Tungro*. *True Negatives* (TN) untuk kelas ini adalah 63, yang mencakup data dari kelas *Blast* dan *Blight* yang diprediksi dengan benar sebagai bukan *Tungro*. Sedangkan *False Negative*s (FN) berjumlah 3, yang berarti ada data yang sebenarnya termasuk dalam kelas *Tungro*, tetapi salah diprediksi sebagai 2 *Blast* dan 1 *Blight*. *False Positives* (FP) berjumlah 1, yang menunjukkan ada 1 data yang tidak termasuk dalam kelas *Tungro*, namun diprediksi sebagai *Tungro* yaitu dari kelas *Blight*.

4.3.1.2 Skenario B1

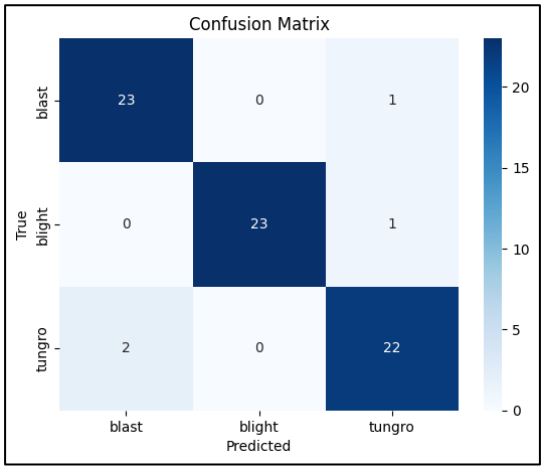
Pada pengujian skenario model B1, dilakukan dengan pembagian data *training* dan data *testing* sebesar 70% : 30%, yang mana sebesar 168 gambar untuk dan 72 gambar. Berikut adalah hasil dari *training* pada Gambar 4.5.



Gambar 4.5 Setiap *Fold Training loss* Model B1

Berdasarkan grafik dan hasil yang ditampilkan, terlihat bahwa model mengalami konvergensi yang stabil selama proses pelatihan. Pada setiap *fold*, *loss* untuk pelatihan dan validasi menurun secara konsisten seiring berjalannya waktu. Kurva *loss training* dan *validation* menunjukkan penurunan yang signifikan, dan gap di antara keduanya semakin mengecil, yang menandakan bahwa model belajar dengan baik tanpa terjebak pada *overfitting* atau *underfitting*. Tidak ada fluktuasi besar pada *loss*, yang biasanya menjadi indikasi divergensi, di mana model gagal mencapai kestabilan. Grafik menunjukkan bahwa model secara bertahap

menyesuaikan parameter-parameter internal dengan data pelatihan dan validasi, menandakan konvergensi yang sehat. *Confusion matrix* dari pengujian ini dapat dilihat pada Gambar 4.6 dan Tabel 4.4.



Gambar 4.6 *Confusion matrix* Model B1 Tanpa Augmentasi Data

Tabel 4.4 Hasil *Confision Matrix* Model B1 Tanpa Augmentasi Data

Class	TP	TN	FN	FP
Blast	23	46	1	2
Blight	23	46	1	0
Tungro	23	63	2	2

Dari hasil *confusion matrix* diperoleh perhitungan akurasi, presisi, *recall* dan *f1-score* yang dapat dilihat pada Tabel 4.5.

Tabel 4.5 *Confision Matrix* Model B1 Tanpa Augmentasi Data

Class	presisi	*recall*	*f1-score*	akurasi
Blast	92%	96%	94%	96%
Blight	100%	96%	98%	98%
Tungro	92%	92%	92%	94%
average				94%

Berdasarkan Tabel 4.4, pada kelas *Blast* model menghasilkan *True Positives* (TP) sebanyak 23, yang berarti ada 23 data yang benar-benar termasuk dalam kelas *Blast* dan diprediksi dengan benar oleh model sebagai *Blast*. *True Negatives* (TN)

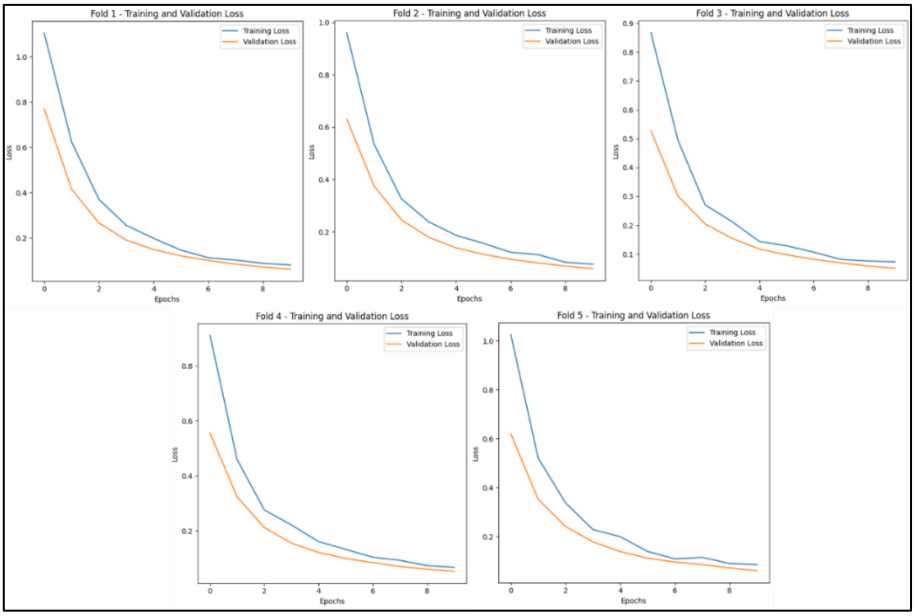
untuk kelas ini adalah 46, yaitu data dari kelas *Blight* dan *Tungro* yang diprediksi dengan benar sebagai bukan *Blast*. Sementara itu, *False Negative*s (FN) berjumlah 1, yaitu data yang sebenarnya termasuk dalam kelas *Blast* tetapi salah diprediksi sebagai *Tungro*. *False Positives* (FP) untuk kelas *Blast* adalah 2, yang berarti ada data yang sebenarnya kelas *Tungro* namun diprediksi sebagai *Blast*.

Untuk kelas *Blight*, model menghasilkan *True Positives* (TP) sebanyak 22, yang berarti ada 23 data yang benar-benar termasuk dalam kelas *Blight* dan diprediksi dengan benar oleh model sebagai *Blight*. *True Negatives* (TN) untuk kelas ini adalah 48, yang mencakup data dari kelas *Blast* dan *Tungro* yang diprediksi dengan benar sebagai bukan *Blight*. *False Negative* (FN) berjumlah 1, yang menunjukkan ada data dari kelas *Blight* yang salah diprediksi sebagai kelas *Tungro*. Sementara itu, *False Positive* (FP) berjumlah 0, yaitu tidak ada data yang dari kelas *Blast* dan blike yang salah diprediksi sebagai *Blight*.

Untuk kelas *Tungro*, model menghasilkan *True Positives* (TP) sebanyak 23, yang berarti ada 23 data yang benar-benar termasuk dalam kelas *Tungro* dan diprediksi dengan benar oleh model sebagai *Tungro*. *True Negatives* (TN) untuk kelas ini adalah 46, yang mencakup data dari kelas *Blast* dan *Blight* yang diprediksi dengan benar sebagai bukan *Tungro*. Sedangkan *False Negative*s (FN) berjumlah 2, yang berarti ada dua data yang yang berasal dari kelas *Blast* dan *Blight*, tetapi salah diprediksi sebagai *Tungro*. *False Positives* (FP) berjumlah 2, yang menunjukkan ada 2 data yang termasuk dalam kelas *Tungro*, namun diprediksi masing masing sebagai *Blast*.

4.3.1.3 Skenario C1

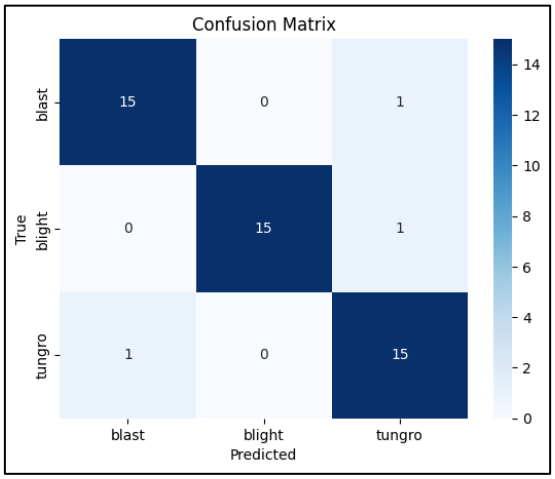
Pada pengujian Skenario model C1, dilakukan dengan pembagian data *training* dan data *testing* sebesar 80% : 20%, yang mana sebesar 192 gambar untuk dan 48 gambar . Berikut adalah hasil dari *training* pada Gambar 4.7.



Gambar 4.7 Setiap *Fold Training loss* Model C1

berdasarkan grafik dari *training loss* dan *validation loss* pada setiap *fold* diatas, model ini menunjukkan tanda-tanda konvergensi yang sangat baik. Sepanjang epoch, *training loss* menurun secara tajam, yang menunjukkan bahwa model belajar dengan cepat dan efektif dari data pelatihan pada awalnya. Di sisi lain, *validation loss* juga mengalami penurunan yang stabil, tanpa fluktuasi besar, yang menunjukkan bahwa model berhasil menggeneralisasi dengan baik pada data validasi. Tidak ada peningkatan *loss* yang tajam atau fluktuasi yang tidak terkendali pada kedua kurva, yang biasanya mengindikasikan *overfitting* atau divergensi. Secara keseluruhan, penurunan *loss* yang konsisten dan mendekati antara data pelatihan dan validasi menandakan bahwa model ini konvergen dengan sehat, tanpa

menunjukkan tanda-tanda masalah seperti divergensi. *Confusion matrix* dari pengujian ini dapat dilihat pada Gambar 4.8 dan Tabel 4.6.



Gambar 4.8 *Confusion matrix* Model C1 Tanpa Augmentasi Data

Tabel 4. 6 Hasil *Confision Matrix* Model C1

Class	TP	TN	FN	FP
Blast	15	31	1	2
Blight	15	32	0	1
Tungro	15	30	1	2

Dari hasil *confusion matrix* diperoleh perhitungan akurasi, presisi, *recall* dan *f1-score* yang dapat dilihat pada Tabel 4.7.

Tabel 4. 7 *Confision Matrix* Model C1 Tanpa Augmentasi Data

Class	presisi	*recall*	*f1-score*	akurasi
Blast	94%	94%	94%	96%
Blight	100%	94%	97%	98%
Tungro	88%	94%	91%	94%
average				94%

Berdasarkan Tabel 4.6, Untuk kelas *Blast* model menghasilkan *True Positives* (TP) sebanyak , yang berarti ada 15 data yang benar-benar termasuk dalam kelas *Blast* dan diprediksi dengan benar oleh model sebagai *Blast*. *True Negatives* (TN) untuk kelas ini adalah 31, yaitu data dari kelas *Blight* dan *Tungro*

yang diprediksi dengan benar sebagai bukan *Blast*. Sementara itu, *False Negative*s (FN) berjumlah 1, yaitu data yang sebenarnya termasuk dalam kelas blasy tetapi salah diprediksi sebagai *Tungro*. *False Positives* (FP) untuk kelas *Blast* adalah 1, yang berarti ada data yang sebenarnya kelas *Tungro* namun diprediksi sebagai *Blast*.

Untuk kelas *Blight*, model menghasilkan *True Positives* (TP) sebanyak 15, yang berarti ada 15 data yang benar-benar termasuk dalam kelas *Blight* dan diprediksi dengan benar oleh model sebagai *Blight*. *True Negatives* (TN) untuk kelas ini adalah 32, yang mencakup data dari kelas *Blast* dan *Tungro* yang diprediksi dengan benar sebagai bukan *Blight*. *False Negative* (FN) berjumlah 1, yang menunjukkan data dari kelas *Blight* yang salah diprediksi sebagai kelas *Tungro*. Sementara itu, *False Positive* (FP) berjumlah 1, yaitu ada data yang sebenarnya termasuk dalam kelas *Tungro* tetapi salah diprediksi sebagai *Blast*.

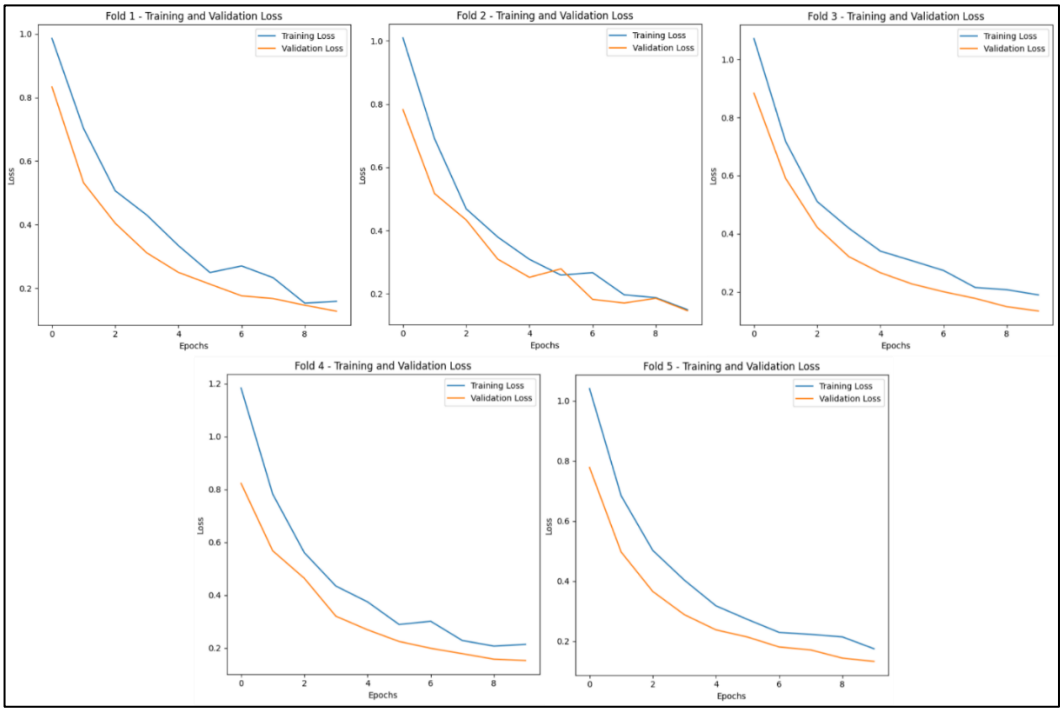
Untuk kelas *Tungro*, model menghasilkan *True Positives* (TP) sebanyak 15, yang berarti ada 15 data yang benar-benar termasuk dalam kelas *Tungro* dan diprediksi dengan benar oleh model sebagai *Tungro*. *True Negatives* (TN) untuk kelas ini adalah 30, yang mencakup data dari kelas *Blast* dan *Blight* yang diprediksi dengan benar sebagai bukan *Tungro*. Sedangkan *False Negative*s (FN) berjumlah 2, yang menunjukkan ada 2 data yang termasuk masing masing sebagai 1 *Blast* dan 1 *Tungro*, namun diprediksi dalam kelas *Tungro*. *False Positives* (FP) berjumlah 1, yang menunjukkan ada 1 data yang berasal dari 1 *Tungro*, namun diprediksi dalam kelas *Blast*.

4.3.2 Pengujian dengan Augmentasi Data

Dalam pengujian ini, dilakukan pengujian terhadap tiga skenario model yang menggunakan augmentasi data, untuk membandingkan hasil klasifikasi penyakit padi dalam kondisi variasi tambahan pada dataset.

4.3.2.1 Skenario A2

Pada pengujian skenario model A2, dilakukan dengan pembagian data *training* dan data *testing* sebesar 60% : 40%, yang mana sebesar 144 Gambar untuk dan 96 Gambar. Berikut hasil dari *training* model A pada Gambar 4.9

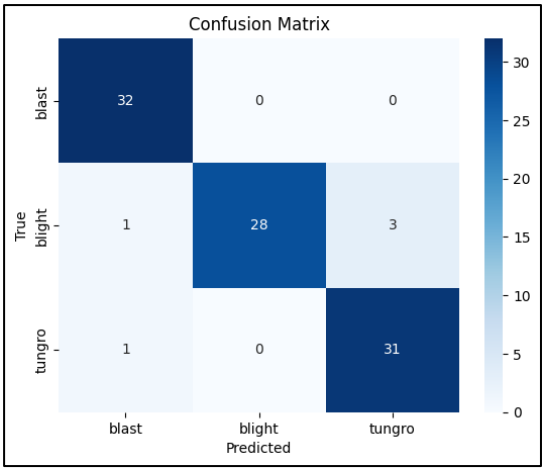


Gambar 4.9 Setiap *Fold training loss* model A2

Melihat grafik *training loss* dan *validation loss* untuk setiap *fold*, model ini menunjukkan tanda-tanda konvergensi yang baik. *Training loss* secara konsisten menurun pada setiap epoch, menunjukkan bahwa model belajar dari data pelatihan dengan stabil. Sementara itu, *validation loss* juga mengikuti tren penurunan yang

serupa dengan sedikit fluktuasi. Penurunan yang stabil pada kedua kurva tanpa adanya lonjakan besar pada *validation loss* menunjukkan bahwa model tidak mengalami *overfitting* dan mampu menggeneralisasi dengan baik pada data validasi.

Pada beberapa *fold*, meskipun ada sedikit fluktuasi, terutama pada *fold* ke-4, tren keseluruhan dari kedua kurva tetap menurun. Ini menandakan bahwa meskipun ada beberapa fluktuasi kecil, model tetap berada pada jalur yang benar untuk mencapai konvergensi. Secara keseluruhan, model ini konvergen dengan baik, karena tidak menunjukkan peningkatan yang tajam pada *validation loss*, yang biasanya merupakan tanda dari divergensi. *Confusion matrix* dari pengujian ini dapat dilihat pada Gambar 4.10 dan Tabel 4.8.



Gambar 4.10 *Confusion Matrix* Model A2 Augmentasi Data

Tabel 4. 8 Hasil *Confusion Matrix* Model A2

Class	TP	TN	FN	FP
Blast	32	62	0	2
Blight	28	64	4	0
Tungro	31	61	1	3

Dari hasil *confusion matrix* diperoleh perhitungan akurasi, presisi, *recall* dan *f1-score* yang dapat dilihat pada Tabel 4.9.

Tabel 4. 9 *Confision Matrix* Model A2 Augmentasi Data

Class	presisi	*recall*	*f1-score*	akurasi
Blast	94%	100%	97%	98%
Blight	100%	88%	93%	96%
Tungro	91%	97%	94%	96%
average				95%

Berdasarkan Tabel 4.8, Untuk kelas *Blast* model menghasilkan *True Positives* (TP) sebanyak , yang berarti ada 32 data yang benar-benar termasuk dalam kelas *Blast* dan diprediksi dengan benar oleh model sebagai *Blast*. *True Negatives* (TN) untuk kelas ini adalah 62, yaitu data dari kelas *Blight* dan *Tungro* yang diprediksi dengan benar sebagai bukan *Blast*. Sementara itu, *False Negative*s (FN) berjumlah 0, yaitu tidak ada data yang sebenarnya termasuk dalam kelas *Blast* yang salah diprediksi sebagai *Blight* dan *Tungro*. *False Positives* (FP) untuk kelas *Blast* adalah 2, yang berarti ada data yang yang masing masing berasal kelas *Blight* dan *Tungro* diprediksi sebagai *Blast*.

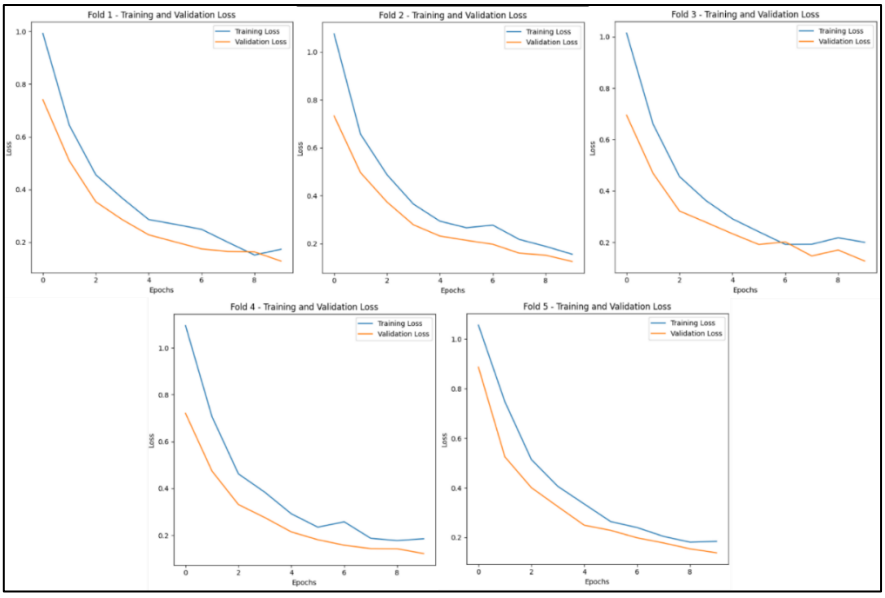
Untuk kelas *Blight*, model menghasilkan *True Positives* (TP) sebanyak 28, yang berarti ada 28 data yang benar-benar termasuk dalam kelas *Blight* dan diprediksi dengan benar oleh model sebagai *Blight*. *True Negatives* (TN) untuk kelas ini adalah 64, yang mencakup data dari kelas *Blast* dan *Tungro* yang diprediksi dengan benar sebagai bukan *Blight*. *False Negative* (FN) berjumlah 4, yang menunjukkan ada data dari kelas *Blight* yang salah diprediksi sebagai kelas 1 *Blast* atau 3 *Tungro*. Sementara itu, *False Positive* (FP) berjumlah 0, yang

menunjukkan tidak ada data dari sebagai kelas *Blast* atau *Tungro* yang salah diprediksi sebagai kelas *Blight*.

Untuk kelas *Tungro*, model menghasilkan *True Positives* (TP) sebanyak 31, yang berarti ada 31 data yang benar-benar termasuk dalam kelas *Tungro* dan diprediksi dengan benar oleh model sebagai *Tungro*. *True Negatives* (TN) untuk kelas ini adalah 61, yang mencakup data dari kelas *Blast* dan *Blight* yang diprediksi dengan benar sebagai bukan *Tungro*. Sedangkan *False Negative*s (FN) berjumlah 3, yang berarti ada data yang sebenarnya termasuk dalam kelas *Blight*, tetapi salah diprediksi sebagai *Tungro*. *False Positives* (FP) berjumlah 1, yang menunjukkan ada 1 data yang diprediksi sebagai kelas *Blast*, namun berasal sebagai kelas *Tungro*.

4.3.2.2 Skenario B2

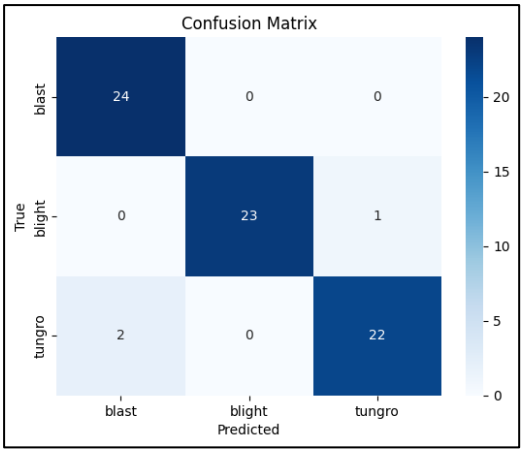
Pada pengujian skenario model B2, dilakukan dengan pembagian data *training* dan data *testing* sebesar 70% : 30%, yang mana sebesar 168 gambar untuk dan 72 gambar. Berikut adalah hasil dari *training* pada Gambar 4.11.



Gambar 4.11 Setiap *Fold Training Loss* Model B2

Melihat grafik *training loss* dan *validation loss* yang diberikan, model ini menunjukkan konvergensi yang sangat baik. Pada setiap *fold*, kita dapat melihat penurunan yang konsisten pada *training loss* dan *validation loss*, yang menunjukkan bahwa model belajar dengan baik dan dapat menggeneralisasi dengan baik pada data validasi. Penurunan pada kedua kurva sangat stabil, tanpa lonjakan besar atau fluktuasi tajam yang bisa menandakan divergensi.

Pada beberapa *fold*, meskipun ada sedikit fluktuasi, terutama pada *validation loss*, tren keseluruhan dari kedua kurva tetap menurun, dan setelah beberapa epoch, *validation loss* mencapai stabilitas. Ini menunjukkan bahwa model berhasil menghindari *overfitting* dan tetap dapat belajar dengan baik dari data pelatihan, sekaligus mempertahankan performa yang baik pada data validasi. Secara keseluruhan, model ini konvergen dengan baik, tanpa menunjukkan tanda-tanda divergensi yang biasanya terjadi jika *validation loss* meningkat drastis atau melampaui *training loss*. *Confusion matrix* dari pengujian ini dapat dilihat pada Gambar 4.12 dan Tabel 4.10.



Gambar 4.12 *Confusion Matrix* Model B2 Augmentasi Data

Tabel 4.10 Hasil *Confusion Matrix* Model B2

Class	TP	TN	FN	FP
Blast	24	46	0	2
Blight	23	47	1	0
Tungro	22	47	2	1

Dari hasil *confusion matrix* diperoleh perhitungan akurasi, presisi, *recall* dan *f1-score* yang dapat dilihat pada Tabel 4.11.

Tabel 4.11 *Confusion Matrix* Model B2 Augmentasi Data

Class	presisi	*recall*	*f1-score*	akurasi
Blast	92%	100%	96%	97%
Blight	100%	96%	98%	99%
Tungro	96%	92%	94%	96%
average				96%

Berdasarkan table 4.10, Untuk kelas *Blast True Positive* (TP) adalah jumlah gambar yang benar-benar *Blast* dan diprediksi dengan benar sebagai *Blast*, yang jumlahnya adalah 24. *True Negative* (TN) untuk *Blast* adalah jumlah gambar yang bukan *Blast* dan diprediksi dengan benar sebagai *Blight* atau *Tungro*, yang berjumlah 46. *False Positive* (FP) terjadi ketika gambar yang sebenarnya bukan *Blast* diprediksi sebagai *Blast*, dan dalam hal ini ada 2 gambar yang sebenarnya *Tungro* tetapi diprediksi sebagai *Blast*. *False Negative* (FN) untuk *Blast* adalah gambar yang sebenarnya *Blast*, tetapi diprediksi sebagai *Blight* atau *Tungro*, namun dalam hal ini, tidak ada gambar yang salah diprediksi, sehingga *False Negative* untuk *Blast* adalah 0.

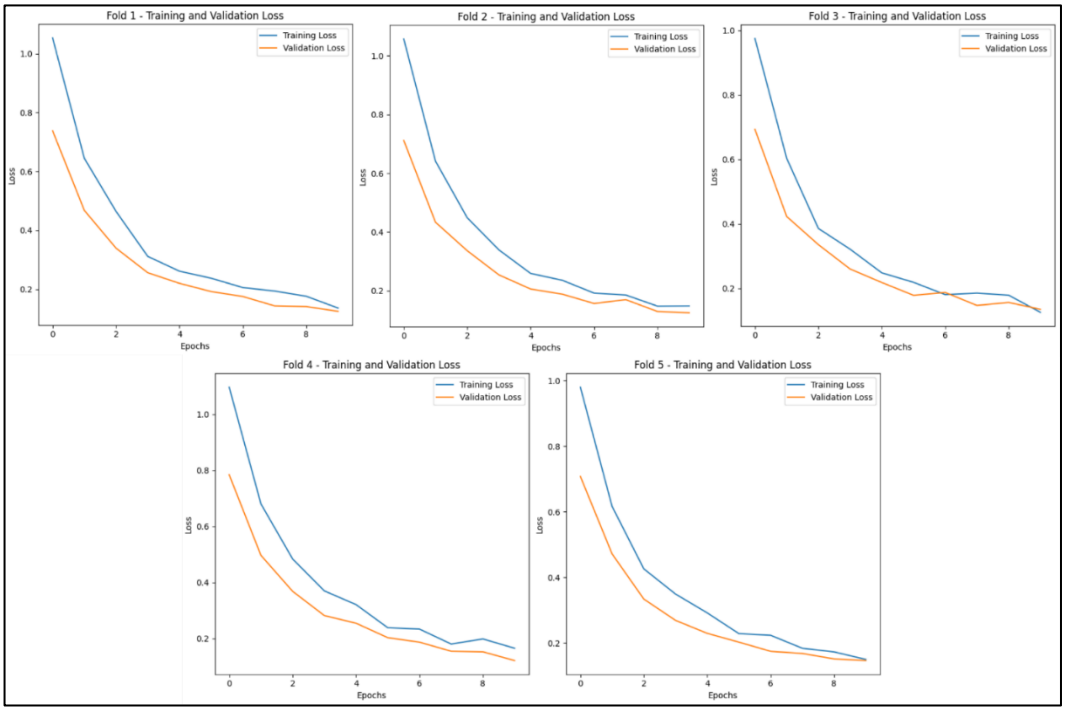
Untuk kelas *Blight*, *True Positive* (TP) adalah jumlah gambar yang benar-benar *Blight* dan diprediksi dengan benar sebagai *Blight*, yang berjumlah 23. *True Negative* (TN) untuk *Blight* adalah gambar yang bukan *Blight* dan diprediksi dengan benar sebagai *Blast* atau *Tungro*, jumlahnya 47. *False Positive* (FP) untuk

Blight adalah gambar yang sebenarnya *Blast* atau *Tungro* tetapi diprediksi sebagai *Blight*, namun tidak ada gambar seperti itu, jadi *False Positive* untuk *Blight* adalah 0. *False Negative* (FN) untuk *Blight* terjadi ketika gambar yang sebenarnya *Blight* diprediksi sebagai *Tungro*, dan ada 1 gambar yang salah diprediksi, sehingga *False Negative* untuk *Blight* adalah 1.

Untuk kelas *Tungro*, *True Positive* (TP) adalah jumlah gambar yang benar-benar *Tungro* dan diprediksi dengan benar sebagai *Tungro*, yang berjumlah 22. *True Negative* (TN) untuk *Tungro* adalah jumlah gambar yang bukan *Tungro* dan diprediksi dengan benar sebagai *Blast* atau *Blight*, jumlahnya 47. *False Positive* (FP) untuk *Tungro* adalah gambar yang sebenarnya bukan *Tungro* tetapi diprediksi sebagai *Tungro*, dan tidak ada gambar seperti itu, sehingga *False Positive* untuk *Tungro* adalah 0. *False Negative* (FN) untuk *Tungro* terjadi ketika gambar yang sebenarnya *Tungro* diprediksi sebagai *Blast*, dan ada 2 gambar yang salah diprediksi, sehingga *False Negative* untuk *Tungro* adalah 2.

4.3.2.3 Skenario C2

Pada pengujian skenario model C2, dilakukan dengan pembagian data *training* dan data *testing* sebesar 80% : 20%, yang mana sebesar 192 gambar untuk dan 48 gambar . Berikut adalah hasil dari *training* pada Gambar 4.13.

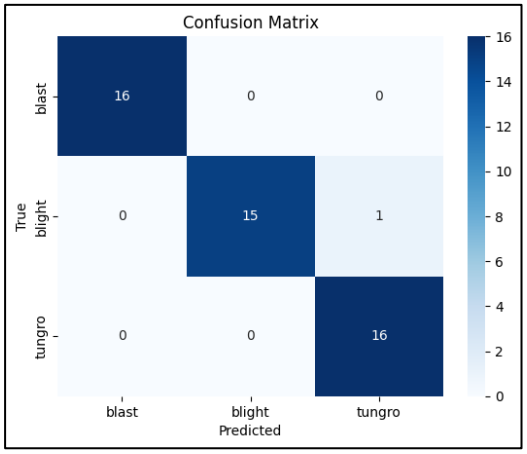


Gambar 4.13 Setiap *Fold Training Loss* Model C2

Melihat grafik *training loss* dan *validation loss* pada setiap *fold*, model ini menunjukkan tanda-tanda konvergensi yang sangat baik. Pada setiap *fold*, baik *training loss* maupun *validation loss* menunjukkan penurunan yang stabil sepanjang proses pelatihan. *Training loss* (kurva biru) secara konsisten menurun, menunjukkan bahwa model belajar dengan efektif dari data pelatihan, sementara *validation loss* (kurva oranye) juga menurun dan berfluktuasi dengan sangat sedikit, menandakan bahwa model dapat menggeneralisasi dengan baik pada data validasi.

Pada beberapa *fold*, meskipun ada sedikit fluktuasi dalam *validation loss*, terutama pada *fold* ke-4 dan ke-5, secara keseluruhan *validation loss* menurun atau stabil setelah beberapa epoch, yang mengindikasikan bahwa model tidak mengalami *overfitting*. *Training loss* juga terus menurun, dengan gap yang relatif kecil antara *training loss* dan *validation loss*, menunjukkan bahwa model berhasil

belajar tanpa tanda-tanda *overfitting* atau divergensi. Secara keseluruhan, model ini dapat dianggap konvergen, dengan hasil yang menunjukkan performa yang stabil dan baik di seluruh *fold* tanpa adanya indikasi divergensi. *Confusion matrix* dari pengujian ini dapat dilihat pada Gambar 4.14 dan Tabel 4.12.



Gambar 4.14 *Confision Matrix* Model C2 Augmentasi Data

Tabel 4.12 Hasil *Confusion Matrix* Model C2 Dengan Augmentasi Data

Class	TP	TN	FN	FP
Blast	16	32	0	0
Blight	15	32	1	0
Tungro	16	31	0	0

Dari hasil *confusion matrix* diperoleh perhitungan akurasi, presisi, *recall* dan *f1-score* yang dapat dilihat pada Tabel 4.13.

Tabel 4.13 *Confision Matrix* Model C2 Augmentasi Data

Class	presisi	*recall*	*f1-score*	akurasi
Blast	100%	100%	100%	100%
Blight	100%	94%	97%	98%
Tungro	94%	100%	97%	98%
average				98%

Berdasarkan table 4.12, Untuk kelas *Blast True Positive* (TP) adalah jumlah gambar yang benar-benar *Blast* dan diprediksi dengan benar sebagai *Blast*, yang jumlahnya adalah 16. *True Negative* (TN) untuk *Blast* adalah jumlah gambar yang

bukan *Blast* dan diprediksi dengan benar sebagai *Blight* atau *Tungro*, yang berjumlah 32. *False Positive* (FP) terjadi ketika gambar yang sebenarnya bukan *Blast* diprediksi sebagai *Blast*, tetapi tidak ada gambar seperti itu dalam hal ini, sehingga *False Positive* untuk *Blast* adalah 0. *False Negative* (FN) untuk *Blast* adalah gambar yang sebenarnya *Blast*, tetapi diprediksi sebagai *Blight* atau *Tungro*, namun dalam hal ini tidak ada gambar yang salah diprediksi sebagai selain *Blast*, sehingga *False Negative* untuk *Blast* adalah 0.

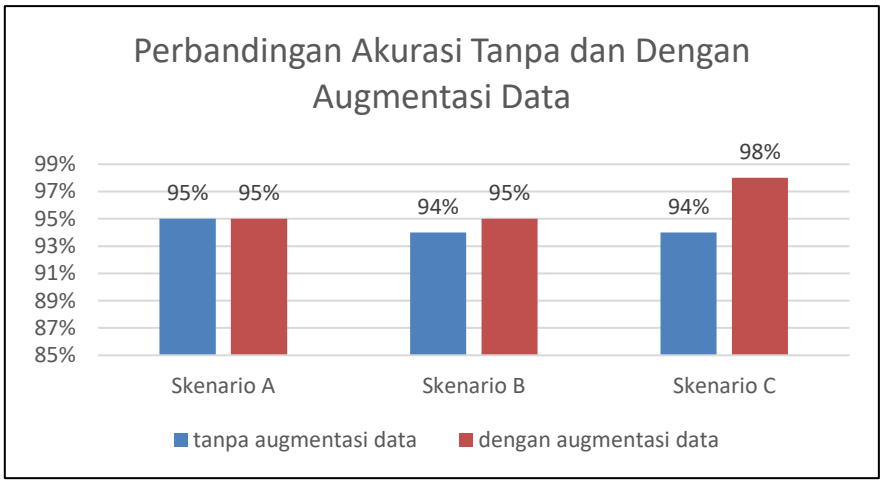
Untuk kelas *Blight*, *True Positive* (TP) adalah jumlah gambar yang benar-benar *Blight* dan diprediksi dengan benar sebagai *Blight*, yang berjumlah 15. *True Negative* (TN) untuk *Blight* adalah gambar yang bukan *Blight* dan diprediksi dengan benar sebagai *Blast* atau *Tungro*, jumlahnya 32. *False Positive* (FP) untuk *Blight* adalah gambar yang sebenarnya *Blast* atau *Tungro* tetapi diprediksi sebagai *Blight*, namun tidak ada gambar seperti itu, jadi *False Positive* untuk *Blight* adalah 0. *False Negative* (FN) untuk *Blight* terjadi ketika gambar yang sebenarnya *Blight* diprediksi sebagai *Tungro*, dan ada 1 gambar yang salah diprediksi, sehingga *False Negative* untuk *Blight* adalah 1.

Untuk kelas *Tungro*, *True Positive* (TP) adalah jumlah gambarg yang benar-benar *Tungro* dan diprediksi dengan benar sebagai *Tungro*, yang berjumlah 16. *True Negative* (TN) untuk *Tungro* adalah jumlah gambar yang bukan *Tungro* dan diprediksi dengan benar sebagai *Blight* atau *Blast*, jumlahnya 31. *False Positive* (FP) untuk *Tungro* adalah gambar yang sebenarnya bukan *Tungro* tetapi diprediksi sebagai *Tungro*, dan tidak ada gambar seperti itu, sehingga *False Positive* untuk *Tungro* adalah 0. *False Negative* (FN) untuk *Tungro* terjadi ketika gambar yang

sebenarnya *Tungro* diprediksi sebagai *Blight*, namun tidak ada gambar yang salah diprediksi, sehingga *False Negative* untuk *Tungro* adalah 0.

4.4 Pembahasan

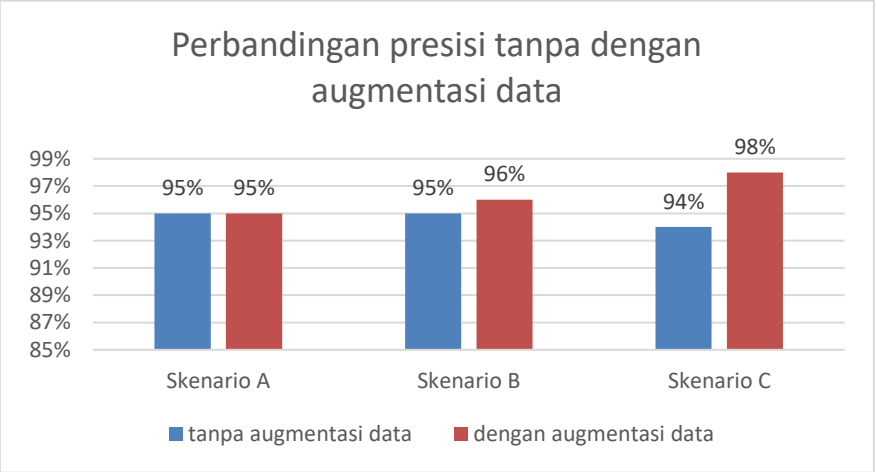
Hasil dari skenario uji coba yang telah dilakukan untuk menentukan skenario mana yang menghasilkan nilai paling optimal dalam melatih model arsitektur DenseNet-201. Dimulai dari perbandingan akurasi Gambar 4.15.



Gambar 4.15 Perbandingan Akurasi Tanpa dan Dengan Augmentasi

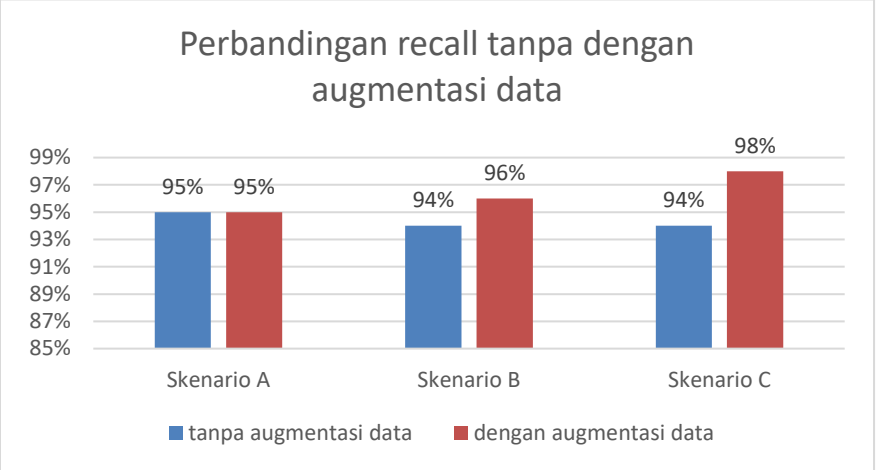
Gambar 4.15 memperlihatkan perbandingan akurasi tiap model dari yang tanpa menggunakan Augmentasi Data dan yang menggunakan. Dapat dilihat bahwa, dimulai skenario dari model A, akurasinya tidak memiliki peningkatan. Sedangkan pada model skenario B2, model yang menggunakan Augmentasi mempunyai akurasi 1% lebih besar yaitu 95%. Dan pada model skenario C2, yang mendapat hasil peningkatan akurasi sebanyak 4% yaitu 98%. Secara keseluruhan, augmentasi data memberi dampak yang berbeda-beda tergantung pada pembagian data yang digunakan oleh masing-masing model. Ini menunjukkan bahwa augmentasi data lebih memberikan manfaat pada model yang memiliki lebih

banyak data pelatihan (seperti Model C), karena model tersebut memiliki lebih banyak kesempatan untuk belajar dan beradaptasi dengan variasi dalam data.



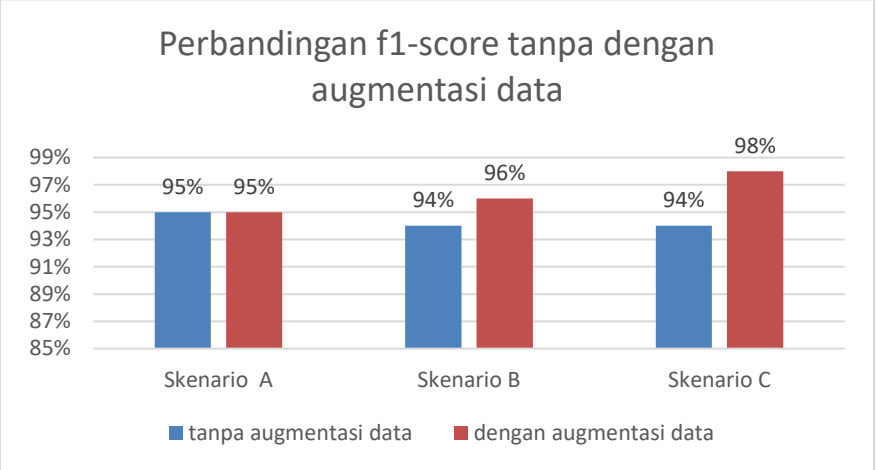
Gambar 4.16 Perbandingan Presisi Tanpa dan Dengan Augmentasi

Gambar 4.16 memperlihatkan perbandingan presisi tiap model dari yang tanpa menggunakan Augmentasi Data dan yang menggunakan. Dapat dilihat bahwa, dimulai skenario dari model A, Augmentasi Data tidak mendapat meningkatkan presisi, namun pada model B dan C, dimana pada model skenario B2, model yang menggunakan Augmentasi mempunyai presisi 1% lebih besar yaitu 96%. Dan pada model skenario C2, yang mendapat hasil presisi tertinggi sebesar 98%. Peningkatan ini mengindikasikan bahwa ketika model memprediksi sebuah kelas (misalnya penyakit *Blast*), ia sangat jarang membuat kesalahan, yaitu memprediksi kelas yang salah (*False Positive*). Model dapat dipercaya untuk meminimalkan kesalahan dalam hal memberikan prediksi positif yang salah. Precision sangat penting dalam situasi di mana kesalahan klasifikasi positif dapat memiliki konsekuensi yang besar, seperti dalam diagnosis penyakit.



Gambar 4.17 Perbandingan *Recall* Tanpa dan Dengan Augmentasi

Gambar 4.17 memperlihatkan perbandingan *recall* tiap model dari yang tanpa menggunakan Augmentasi Data dan yang menggunakan. Dapat dilihat bahwa, dimulai skenario dari model A, Augmentasi Data tidak dapat peningkatan, namun pada model B dan C, dimana pada model skenario B2, model yang menggunakan Augmentasi mempunyai *recall* 2% lebih besar yaitu 96%. Dan pada model skenario C2, yang mendapat hasil akurasi *recall* sebesar 98%, Peningkatan ini menunjukkan yang berarti model dapat mengklasifikasi sebagian besar dari setiap kelas yang benar-benar ada. model sangat baik dalam menangkap semua gambar yang termasuk dalam setiap kategori, dengan sedikit gambar yang terlewatkan (*False Negative*). *Recall* sangat penting dalam skenario seperti klasifikasi penyakit atau pengklasifikasian anomali, di mana kehilangan satu kasus positif dapat berdampak besar.



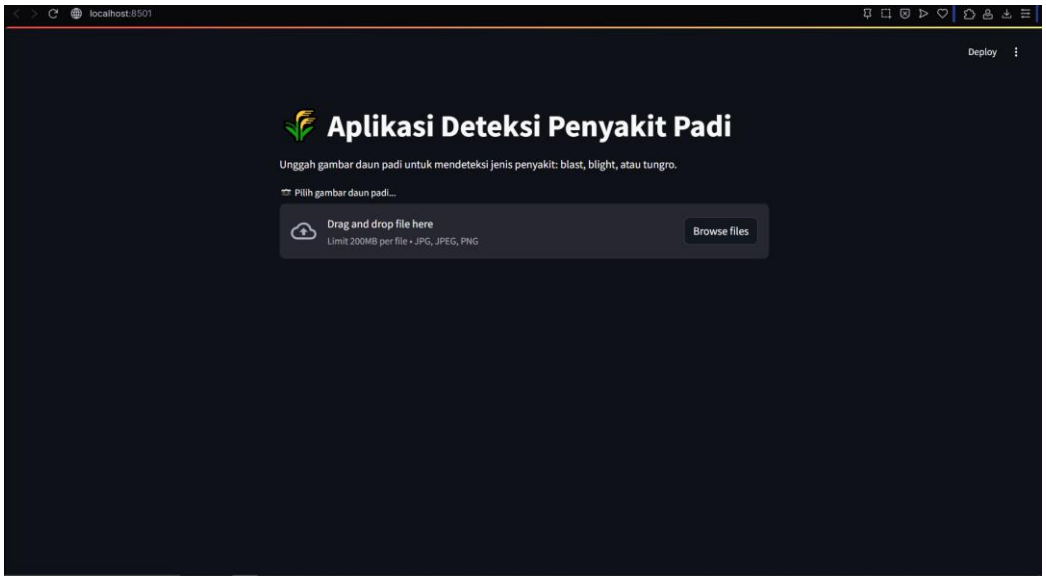
Gambar 4.18 Perbandingan *f1-score* Tanpa dan Dengan Augmentasi

Gambar 4.18 memperlihatkan perbandingan *f1-score* tiap model dari yang tanpa menggunakan Augmentasi Data dan yang menggunakan. Dapat dilihat bahwa, dimulai dari model A, Augmentasi Data tidak memiliki peningkatan. namun pada model B dan C, dimana pada model B2, model yang menggunakan Augmentasi mempunyai *f1-score* 2% lebih besar yaitu 96%. Dan pada model C2, yang mendapat hasil akurasi *recall* sebesar 98%, Peningkatan Ini menunjukkan keseimbangan yang baik antara menghindari *False Positives* dan *False Negative*s, yang berarti model memiliki performa yang solid dan konsisten dalam mengklasifikasikan gambar tanpa bias besar terhadap kelas tertentu.

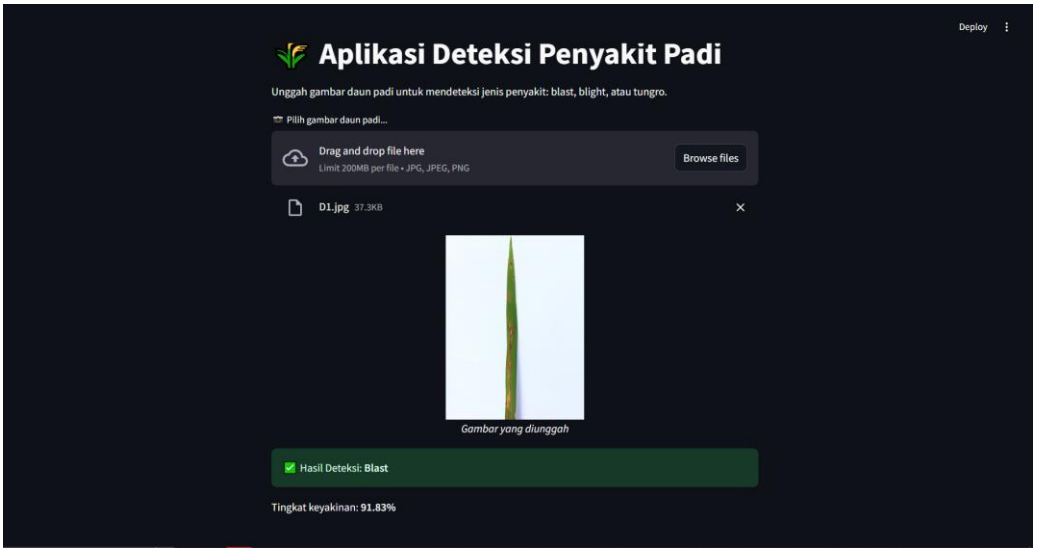
Secara keseluruhan dari dari hasil evaluasi ini menunjukan bahwa model yang menggunakan augmentasi data dalam mengklasifikasi gambar daun tanaman padi dengan dataset yang terbatas memiliki performa yang baik. ada beberapa upaya untuk meningkatkan performa dari model tersebut seperti penambahan data pelatihan dan *testing* sehingga dapat menghindari terjadinya *overfitting* pada model.

4.5 Pengaplikasian Model

Setelah model klasifikasi penyakit tanaman padi selesai dilatih menggunakan arsitektur DenseNet-201 dan telah melalui proses evaluasi, langkah selanjutnya adalah mengimplementasikan model tersebut ke dalam antarmuka berbasis web menggunakan *Streamlit*. *Streamlit* merupakan *framework Python* yang bersifat *open source* dan memungkinkan pembuatan aplikasi web interaktif dengan cepat, khususnya untuk kebutuhan data science dan machine learning. Berikut contoh *User Interface* untuk aplikasi deteksi penyakit padi pada Gambar 4.19.



Gambar 4.19 *User Interface* Aplikasi Deteksi Penyakit Padi



Gambar 4.20 Penggunaan Aplikasi Deteksi Penyakit Padi

Pada tampilan aplikasi deteksi penyakit padi pada Gambar 4.20, pengguna akan melihat antarmuka yang bersih dan terpusat dengan elemen-elemen yang intuitif. Di bagian atas, terdapat tombol unggah gambar dengan label "Pilih gambar daun padi...", yang memungkinkan pengguna untuk mengunggah file gambar daun padi dalam format JPG, JPEG, atau PNG. Setelah pengguna memilih file, nama file akan ditampilkan, menandakan bahwa gambar berhasil diunggah.

Setelah gambar dimuat, aplikasi menampilkan *preview* gambar tersebut tepat di tengah halaman menggunakan elemen HTML yang dirender melalui *Streamlit*. Gambar ini ditampilkan dengan ukuran yang telah diperkecil (lebar 200 piksel), dilengkapi keterangan "Gambar yang diunggah" yang berada tepat di bawahnya dalam bentuk teks miring. Penempatan gambar di tengah ini memberikan kesan visual yang lebih rapi dan fokus kepada pengguna saat menunggu proses klasifikasi dijalankan.

Penggunaan *Streamlit* bertujuan untuk mempermudah pengguna, seperti petani atau penyuluh pertanian, dalam mengunggah gambar daun padi yang

terindikasi terkena penyakit, dan secara otomatis mendapatkan hasil diagnosis berdasarkan model klasifikasi yang telah dilatih. Antarmuka ini memungkinkan pengguna untuk berinteraksi langsung dengan sistem tanpa perlu memahami detail teknis dari proses klasifikasi.

Dalam Islam, ilmu pengetahuan dan teknologi dipandang sebagai karunia dari Allah yang harus digunakan untuk kemaslahatan umat manusia. Prinsip hubungan manusia dengan Allah dan hubungan dengan sesama menjadi landasan utama dalam setiap kegiatan, termasuk dalam melakukan penelitian. Penelitian ini tidak hanya difokuskan pada peningkatan akurasi model, tetapi juga bertujuan untuk memberikan manfaat nyata bagi masyarakat, khususnya dalam upaya meningkatkan produktivitas hasil panen padi secara lebih efisien. Nilai-nilai Islam tercermin dalam setiap tahapan pelaksanaan penelitian ini. Seperti pada ayat berikut pada surah Al-Yusuf ayat 48 :

ثُمَّ يَأْتِيْ مِنْۢ بَعْدِ ذٰلِكَ سَبْعٌ شِدَادٌ يَّأْكُلْنَ مَا قَدَّمْتُمْ لَهُنَّ اِلَّا قَلِيْلًا مِّمَّا تُحْصِنُوْنَ ﴿٤٨﴾

“Yusuf berkata: "Supaya kamu bertanam tujuh tahun (lamanya) sebagaimana biasa; maka apa yang kamu tuai hendaklah kamu biarkan dibulirnya kecuali sedikit untuk kamu makan.” (Q.S Yusuf :48).

Ayat ini mengajarkan pentingnya perencanaan dan pengelolaan sumber daya secara bijak untuk menghadapi masa sulit, serta pentingnya ketahanan pangan dalam menjaga kesejahteraan masyarakat. melalu penggunaan teknologi yang ada sehingga dapat dengan mudah menjaga hasil panen padi secara lebih efisien. Maka dari itu diperlukan strategi yang matang untuk mempertahankan pangan. Sejalan dengan ayat tersebut, terdapat hadits yang mengatakan bahwa:

مَا مِنْ مُسْلِمٍ يَغْرِسُ غَرْسًا, أَوْ يَزْرَعُ زَرْعًا فَيَأْكُلَ مِنْهُ طَيْرٌ أَوْ إِنْسَانٌ أَوْ بَهِيْمَةٌ إِلاَّ كَانَ لَهُ بِهِ صَدَقَةٌ

"Tidaklah seorang muslim menanam pohon, tidak pula menanam tanaman kemudian pohon/ tanaman tersebut dimakan oleh burung, manusia atau binatang melainkan menjadi sedekah baginya" (HR. Imam Bukhari hadits no.2321)

Hadits ini mendorong peningkatan produktivitas pertanian dan kesejahteraan masyarakat melalui teknologi. Dengan memanfaatkan teknologi sehingga dapat meningkatkan hasil panen tanaman, sehingga lebih banyak orang yang dapat memperoleh manfaat dari hasil tanaman tersebut. Hal ini sejalan dengan ajaran sedekah yang dijelaskan dalam hadis, di mana manfaat tanaman yang dinikmati oleh orang lain juga menjadi sedekah bagi si penanam.

BAB V

KESIMPULAN

5.1 Kesimpulan

Berdasarkan hasil pengujian dan analisis yang telah dilakukan, dapat disimpulkan bahwa augmentasi data memiliki pengaruh signifikan terhadap peningkatan performa arsitektur DenseNet-201 dalam klasifikasi citra penyakit tanaman padi. Dengan teknik augmentasi data yang diterapkan pada penelitian ini yang dilakukan secara acak menggunakan *ImageDataGenerator* dari *library Keras* menghasilkan model skenario C2 (dengan 80% data latih dan augmentasi) menunjukkan performa terbaik dengan akurasi mencapai 98%. Oleh karena itu, teknik augmentasi data terbukti mampu meningkatkan kemampuan generalisasi model DenseNet-201 dalam mengklasifikasi penyakit *Blast*, *Blight*, dan *Tungro* secara lebih akurat dan andal.

Dengan demikian, penelitian ini menunjukkan bahwa penggunaan DenseNet-201 yang dikombinasikan dengan augmentasi data dapat menjadi solusi yang efektif untuk menklasifikasi penyakit tanaman padi dan dapat membantu para petani untuk mengambil langkah pencegahan yang tepat guna meningkatkan hasil pertanian dan ketahanan pangan.

5.2 Saran

Berdasarkan hasil penelitian ini, terdapat beberapa saran yang dapat dipertimbangkan untuk pengembangan lebih lanjut dalam penerapan teknologi klasifikasi penyakit tanaman padi menggunakan model *deep learning*:

1. Peningkatan Kualitas Dataset

   Meskipun dataset yang digunakan dalam penelitian ini cukup representatif, kualitas dan kuantitas dataset yang lebih baik dapat memberikan hasil yang lebih optimal. Penambahan variasi gambar dari berbagai kondisi cuaca, sudut pandang, serta berbagai jenis penyakit lainnya yang dapat memperkaya data pelatihan.

2. Pemanfaatan Augmentasi Data

   Disarankan untuk penelitian selanjutnya agar mempertimbangkan penggunaan teknik augmentasi yang dikontrol secara sistematis selain augmentasi acak, guna membandingkan pengaruh jenis augmentasi terhadap performa model secara akurat dan terkontrol.

3. Eksplorasi Arsitektur Lain

   DenseNet-201 terbukti efektif, namun eksplorasi terhadap arsitektur *deep learning* lain, seperti EfficientNet atau ResNet, dapat memberikan perspektif baru dalam meningkatkan kinerja klasifikasi. Model-model ini dapat memberikan trade-off antara akurasi dan efisiensi komputasi yang lebih baik, terutama pada perangkat dengan kapasitas terbatas.

4. Implementasi di Lapangan

   Pengujian lebih lanjut diperlukan untuk menerapkan sistem klasifikasi penyakit padi berbasis model *deep learning* di lapangan. Diperlukan evaluasi dalam kondisi nyata, di mana variasi lingkungan, kualitas gambar, dan faktor-faktor eksternal lainnya dapat memengaruhi kinerja model.

DAFTAR PUSTAKA

Abadi, M., et al. (2016). *TensorFlow*: A System for Large-Scale Machine Learning. *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*.

Badan Pusat Statistik. (2023, 16 Oktober). *Luas Panen dan Produksi Padi di Indonesia 2023 (Angka Sementara)*. Badan Pusat Statistik. Diakses Diakses pada 6 Februari 2025 dari https://www.bps.go.id/id/pressrelease/2023/10/16/2037/luas-panen-dan-produksi-padi-di-indonesia-2023--angka-sementara-.html

Bakr, M., Abdel-Gaber, S., Nasr, M., & Hazman, M. (2022). DenseNet-based model for plant diseases diagnosis. European Journal of Electrical Engineering and Computer Science, 6(5). https://doi.org/10.24018/ejece.2022.6.5.458

Boulent, J., Foucher, S., Theau, J., & St-Charles, P. L. (2019). *Convolutional Neural Network*s for the automatic identification of plant diseases. Frontiers in Plant Science, 10, 941.

Chen, J., Chen, J., Zhang, D., Sun, Y., Nanehkaran, Y. A., & Jiang, D. (2020). Using deep transfer learning for image-based plant disease identification. Computers and Electronics in Agriculture, 173, 105393.

Chollet, F. (2018). *Deep learning* with *Python*. Manning Publications.
Dokumentasi Keras: *ImageDataGenerator*
https://keras.io/api/*preprocessing*/image/

Distan Buleleng. (2023). Penyakit Blas pada Tanaman Padi. Retrieved from https://distan.bulelengkab.go.id/informasi/detail/artikel/35_penyakit-blas-pyricularia-oryzae

Distan Buleleng. (2020). Blas Padi (Pyricularia oryzae) dan Agen Pengendalinya. Retrieved from https://distan.bulelengkab.go.id/informasi/detail/artikel/35_blas-padi-pyricularia-oryzae-dan-agen-pengendalinya

FAO. (2023). *Global Report on Plant Disease Impact on Crop Yields*. FAO.

Ferentinos, K. P. (2018). *Deep learning* models for plant disease detection and diagnosis. Computers and Electronics in Agriculture, 145, 311-318.

Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). *Densely Connected Convolutional Network*s. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2261–2269. https://doi.org/10.1109/CVPR.2017.243

Jubi. (2022). *Puluhan hektar tanaman padi di Merauke diserang hama Tungro, petani terancam gagal panen.* Diakses pada 6 Februari 2025, dari https://jubi.id

Kontan. (2023, 13 Juli). *Akibat El Nino, Indonesia Terancam Kehilangan Panen 1,2 Juta Ton Beras*. Diakses Diakses pada 6 Februari 2025 dari https://nasional.kontan.co.id/news/akibat-el-nino-indonesia-terancam-kehilangan-panen-12-juta-ton-beras

Kumar, S., Singh, V. P., Pal, S., & Jaiswal, P. (2023). Energy-efficient model "DenseNet201 based on deep *Convolutional Neural Network*" using cloud platform for detection of COVID-19 infected patients. Epidemiology Methods, 12(1), 20210047. https://doi.org/10.1515/em-2021-0047

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 2278-2324. https://doi.org/10.1109/5.726791

Luque, A., Carrasco, A., Martín, A., & De las Heras, A. (2019). The impact of *Class* imbalance in *Class*ification performance metrics based on the *confusion matrix*. *Pattern Recognition*, 91, 216-231. https://doi.org/10.1016/j.patcog.2019.02.023

Masnilah, R., Wahyuni, W. S., Dwi, S., Majid, A., Addy, H. S., & Wafa, A. (2020).Insidensi dan Keparahan Penyakit Penting Tanaman Padi di Kabupaten Jember. Agritrop, 18(1), 1–12. http://jurnal.unmuhjember.ac.id/index.php/AGRITROP/article/view/3103

Paper, D. (2021). Increase the Diversity of Your Dataset with Data Augmentation. In: State-of-the-Art *Deep learning* Models in *TensorFlow*. Apress, Berkeley, CA. https://doi.org/10.1007/978-1-4842-7341-8_2

Pertanian Sulteng. (2020). Mengenal Penyakit *Tungro* pada Padi dan Cara Pengendaliannya. Retrieved from https://pertanian.sultengprov.go.id/mengenal-penyakit-*Tungro*-gejala-dan-cara-pengendalian

Plantix. (n.d.). Bacterial *Blight* of Rice. Retrieved from https://plantix.net/id/*library*/plant-diseases/300014/bacterial-*Blight*-of-rice

Pradipto, A., Regasari Mardi Putri, R. ., & Setia Budi, A. (2025). Perbandingan Penggunaan DenseNet201 dan YOLOv8 Pada Pengembangan Sistem Klasifikasi Sampah pada Raspberry Pi 4 Menggunakan Kamera Penglihatan Malam. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *9*(3). Diambil dari https://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/14613

Putri, F. N. R., Isnanto, R. R., & Sugiharto, A. (2024). Skin Rash *Class*ification System Using Modified DenseNet201 Through *Random Search* for Hyperparameter Tuning. Jurnal Ilmiah Kursor, 12(4), 179–190. https://doi.org/10.21107/kursor.v12i4.391

Ridhovan, A., Suharso, A., & Rozikin, C. (2022). *Disease detection in banana leaf plants using* DenseNet *and Inception method*. Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi), 6(5), 710-718. https://doi.org/10.29207/resti.v6i5.4202

Salim, F., Saeed, F., Basurra, S., Qasem, S. N., & Al-Hadhrami, T. (2023). DenseNet-201 *and Xception pre-trained deep learning models for fruit recognition*. **Electronics, 12**(3132). https://doi.org/10.3390/electronics12143132

Schalock, R. L., Luckasson, R., & Tassé, M. J. (2021). An overview of intellectual disability: Definition, diagnosis, *Class*ification, and systems of supports. American Journal on Intellectual and Developmental Disabilities, 126(6), 439-442.

Setiady, T. P. B. (2021). *Klasifikasi Penyakit Tanaman Padi Menggunakan Metode Deep learning* (Skripsi Sarjana, Universitas Hasanuddin). Universitas Hasanuddin.

Setiady, T. (2024). *Leaf rice disease Indonesia [Dataset].* Kaggle. Diakses pada 6 Februari 2025, dari https://www.kaggle.com/datasets/tedisetiady/leaf-rice-disease-indonesia.

Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on image data augmentation for *deep learning*. Journal of Big Data, 6(1), 60.