

**IMPLEMENTASI ENSEMBLE DEEP LEARNING
UNTUK KLASIFIKASI PENYAKIT PADA TANAMAN PADI**



Tugas Akhir

**Richard Alvin Pratama
00000034813**

**PROGRAM STUDI TEKNIK KOMPUTER
FAKULTAS TEKNIK & INFORMATIKA
UNIVERSITAS MULTIMEDIA NUSANTARA
TANGERANG
2023**

**IMPLEMENTASI ENSEMBLE DEEP LEARNING
UNTUK KLASIFIKASI PENYAKIT PADA TANAMAN PADI**



Diajukan sebagai Salah Satu Syarat untuk Memperoleh

Gelar Sarjana Teknik Komputer

Richard Alvin Pratama

00000034813

PROGRAM STUDI TEKNIK KOMPUTER

FAKULTAS TEKNIK & INFORMATIKA

UNIVERSITAS MULTIMEDIA NUSANTARA

TANGERANG

2023

HALAMAN PERNYATAAN TIDAK PLAGIAT

Dengan ini saya,

Nama : Richard Alvin Pratama
Nomor Induk Mahasiswa : 00000034813
Program studi : Teknik Komputer

Tugas Akhir dengan judul:

IMPLEMENTASI ENSEMBLE DEEP LEARNING UNTUK KLASIFIKASI PENYAKIT PADA TANAMAN PADI

merupakan hasil karya saya sendiri bukan plagiat dari karya ilmiah yang ditulis oleh orang lain, dan semua sumber, baik yang dikutip maupun dirujuk, telah saya nyatakan dengan benar serta dicantumkan di Daftar Pustaka.

Jika di kemudian hari terbukti ditemukan kecurangan/penyimpangan, baik dalam pelaksanaan skripsi maupun dalam penulisan laporan skripsi, saya bersedia menerima konsekuensi dinyatakan TIDAK LULUS untuk Tugas Akhir yang telah saya tempuh.

Tangerang, 7 Juni 2023



(Richard Alvin Pratama)

MULTIMEDIA
NUSANTARA

HALAMAN PERSETUJUAN

Tugas Akhir dengan judul

Implementasi Ensemble Deep Learning Untuk Klasifikasi Penyakit Pada
Tanaman Padi

Oleh

Nama : Richard Alvin Pratama
NIM : 00000034813
Program Studi : Teknik Komputer
Fakultas : Teknik & Informatika

Telah disetujui untuk diajukan pada

Sidang Ujian Tugas Akhir Universitas Multimedia Nusantara

Tangerang, 7 Juni 2023

Pembimbing



(Nabila Husna Shabrina, S.T., M.T)
(0321099301)

Ketua Program Studi Teknik Komputer



(Samuel, M.T.I)
(0304038902)

HALAMAN PENGESAHAN

Tugas Akhir dengan judul

Implementasi Ensemble Deep Learning Untuk Klasifikasi Penyakit Pada
Tanaman Padi

Oleh

Nama : Richard Alvin Pratama

NIM : 00000034813

Program Studi : Teknik Komputer

Fakultas : Teknologi Informasi dan Komunikasi

Telah diujikan pada hari Rabu, 14 Juni 2023

Pukul 09.00 s.d 11.00 dan dinyatakan

LULUS

Dengan susunan penguji sebagai berikut.

Ketua Sidang

Penguji

Monica Pratiwi, S.ST., M.T
(0325059601)

Samuel Hutagalung, M.T.I
(0304038902)

Pembimbing

Nabila Husna Shabrina, S.T., M.T
(0321099301)

Ketua Program Studi Teknik Komputer

**UNIVERSITAS
MULTIMEDIA
NUSANTARA**

HALAMAN PERSETUJUAN PUBLIKASI KARYA ILMIAH UNTUK KEPENTINGAN AKADEMIS

Sebagai civitas academica Universitas Multimedia Nusantara, saya yang bertanda tangan di bawah ini:

Nama : Richard Alvin Pratama

NIM : 00000034813

Program Studi : Teknik Komputer

Fakultas : Teknik & Informatika

Jenis Karya : ***Tesis/Skripsi/Tugas Akhir** (*coret salah satu)

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Multimedia Nusantara Hak Bebas Royalti Nonekslusif (*Non-exclusive Royalty-Free Right*) atas karya ilmiah saya yang berjudul.

Implementasi Ensemble Deep Learning Untuk Klasifikasi Penyakit Pada Tanaman Padi

Beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Nonekslusif ini, Universitas Multimedia Nusantara berhak menyimpan, mengalihmediakan/mengalihformatkan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan memublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta. Demikian pernyataan ini saya buat dengan sebenarnya.

Tangerang, 7 Juni 2023

Yang menyatakan,



(Richard Alvin Pratama)

**UNIVERSITAS
MULTIMEDIA
NUSANTARA**

KATA PENGANTAR

Puji syukur atas berkat dan Rahmat kepada Tuhan Yang Maha Esa, atas selesainya pembuatan Tugas Akhir dengan judul: “Implementasi Ensemble Deep Learning Untuk Klasifikasi Penyakit Pada Tanaman Padi”. Tugas akhir ini dibuat sebagai syarat untuk lulus dari jurusan teknik Komputer pada Fakultas Teknik & Informatika Universitas Multimedia Nusantara. Saya menyadari bahwa, tanpa bantuan dan bimbingan dari berbagai pihak untuk pembuatan Tugas Akhir ini. Oleh karena itu, saya mengucapkan terima kasih kepada:

1. Dr. Ninok Leksono, selaku Rektor Universitas Multimedia Nusantara.
2. Dr. Eng. Niki Prastomo, selaku Dekan Fakultas Teknik dan Informatika Universitas Multimedia Nusantara.
3. Samuel, M.T.I., selaku Ketua Program Studi Universitas Multimedia Nusantara.
4. Nabila Husna Shabrina, S.T., M.T., sebagai Pembimbing Tugas Akhir yang telah memberikan bimbingan, arahan, dan motivasi atas terselesainya tugas akhir ini.
5. Keluarga dan teman saya yang telah memberikan bantuan dukungan material dan moral, sehingga penulis dapat menyelesaikan tugas akhir ini.
6. Serta pihak lainnya yang telah berkontribusi dalam penelitian yang dilaksanakan.

Semoga karya ilmiah dalam bentuk Tugas Akhir ini dapat bermanfaat, baik sebagai sumber referensi maupun sumber inspirasi.

Tangerang, 7 Juni 2023



(Richard Alvin Pratama)

IMPLEMENTASI ENSEMBLE DEEP LEARNING

UNTUK KLASIFIKASI PENYAKIT PADA TANAMAN PADI

Richard Alvin Pratama

ABSTRAK

Padi adalah salah satu komoditas pangan yang penting untuk dunia terutama negara-negara Asia. Hal tersebut dapat terlihat pada statistik konsumsi beras di dunia, lima negara Asia menempati urutan teratas dengan Indonesia di posisi kelima. Namun, tidak dapat dipungkiri, banyak penyakit yang dapat menyerang tanaman padi melalui berbagai faktor seperti cuaca, kebersihan tanah, dan serangan hama. Sehingga dapat menurunkan produksi tanaman padi, bahkan untuk salah satu penyakit padi yaitu Rice Blast menjadi penyebab turunnya produksi padi sebesar 10-30%. Penyakit tanaman padi sendiri diperkirakan berjumlah 19 penyakit, tetapi pada penelitian ini hanya menggunakan 9 jenis penyakit pada dataset yang diambil dari India bagian selatan. Pengambilan 9 jenis penyakit ini juga didasari kesamaan iklim antara India bagian selatan dan Indonesia dengan penyakit padi yang sama. Dengan banyaknya jenis penyakit pada tanaman padi, menyulitkan petani untuk mengidentifikasi jenis penyakit padi secara tepat. Sehingga diperlukan suatu sistem yang mampu mengidentifikasi penyakit pada tanaman padi secara cepat dan akurat. *Deep learning* dan *ensemble learning* merupakan metode yang mampu untuk mempelajari karakteristik dari penyakit tanaman padi dan mampu melakukan proses klasifikasi yang cepat dan akurat. Pada penelitian ini, akan menggunakan dua *pre-trained* model yaitu EfficientNetV2B0 dan MobileNetV3-Large dan dua metode *ensemble* yaitu *average ensemble* dan *concatenation ensemble*. Untuk mengatasi dataset yang *imbalance*, penelitian ini menggunakan metode *undersampling* dan *oversampling*. Selain itu, untuk mengetahui pengaruh ukuran gambar, penelitian ini juga akan menguji dua ukuran gambar berbeda yaitu 224x224 *pixel* dan 256x256 *pixel*. Supaya dapat langsung diimplementasikan oleh petani, penulis juga akan mendesain sistem *progressive web app*. Hasil yang didapat adalah model *average ensemble* (*oversampling* – 256 px) memiliki akurasi tertinggi sebesar 93.3%. Untuk waktu prediksi tercepat ketika diimplementasi pada sistem *progressive web app*, didapatkan oleh MobileNetV3Large (*oversampling* – 224 px) dengan waktu 208 milidetik. Akurasi yang didapat dipengaruhi oleh penggunaan model, teknik *balancing* dataset, dan ukuran gambar yang berbeda. Sedangkan waktu prediksi dipengaruhi oleh model dan ukuran gambar.

Kata kunci: Klasifikasi, Penyakit Padi, *Deep Learning*, *Ensemble Learning*

IMPLEMENTASI ENSEMBLE DEEP LEARNING

UNTUK KLASIFIKASI PENYAKIT PADA TANAMAN PADI

Richard Alvin Pratama

ABSTRACT

Rice is one of the important food commodities in the world, especially in Asian countries. This can be seen in the statistics of rice consumption worldwide, where five Asian countries rank at the top, with Indonesia in fifth place. However, it cannot be denied that rice plants can be susceptible to various diseases caused by factors such as weather, soil cleanliness, and pest attacks. These factors can reduce rice production, with Rice Blast being one of the diseases that can cause a 10-30% decrease in rice production. There are estimated to be 19 diseases that affect rice plants, but this study focuses on 9 types of diseases in a dataset taken from southern India. The selection of these 9 diseases is based on the similarity of climate conditions between southern India and Indonesia, which experience similar rice diseases. With the existence of multiple diseases affecting rice plants, it becomes challenging for farmers to accurately identify the specific diseases. Therefore, a system that can quickly and accurately identify rice plant diseases is needed. Deep learning and ensemble learning are methods capable of learning the characteristics of rice plant diseases and performing fast and accurate classification. In this study, two pre-trained models, EfficientNetV2B0 and MobileNetV3-Large, as well as two ensemble methods, average ensemble and concatenation ensemble, will be used. To address the imbalanced dataset, undersampling and oversampling techniques are employed. Additionally, to determine the impact of image size, two different image sizes, 224x224 pixels and 256x256 pixels, will be tested. To ensure practical implementation for farmers, the author will design a progressive web app system. The results show that the average ensemble model (oversampling - 256 px) achieves the highest accuracy of 93.3%. The fastest prediction time, when implemented in the progressive web app system, is achieved by MobileNetV3Large (oversampling - 224 px) with a time of 208 milliseconds. The accuracy is influenced by the model used, dataset balancing techniques, and different image sizes. Meanwhile, the prediction time is affected by the model and image size.

Keywords:Classification, Paddy Disease, Deep Learning, Ensemble Learning

DAFTAR ISI

HALAMAN PERNYATAAN TIDAK PLAGIAT	ii
HALAMAN PERSETUJUAN	iii
HALAMAN PENGESAHAN	iv
HALAMAN PERSETUJUAN PUBLIKASI KARYA ILMIAH UNTUK KEPENTINGAN AKADEMIS	v
KATA PENGANTAR.....	vi
ABSTRAK.....	vii
ABSTRACT.....	viii
DAFTAR ISI	ix
DAFTAR TABEL	xii
DAFTAR GAMBAR.....	xiii
DAFTAR LAMPIRAN	xvi
BAB I PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Identifikasi Masalah	5
1.3 Batasan Penelitian	5
1.4 Tujuan Penelitian.....	6
1.5 Manfaat Penelitian.....	6
1.6 Sistematika Penulisan.....	6
BAB II TINJAUAN PUSTAKA	8
2.1 Penelitian Terdahulu	8
2.2 Tinjauan Teori	17
2.2.1. Deep Learning	17
2.2.2. Convolutional Neural Network (CNN).....	18
2.2.3. MobileNetV3.....	19
2.2.4. EfficientNetV2B0.....	20
2.2.5. Ensemble Learning	21
2.2.6. Balanced dan Imbalanced Dataset	22
2.2.7. Augmentasi Data	24
2.2.8. FastAPI	24

2.2.9.	NuxtJs	24
2.2.10.	Progressive Web App (PWA).....	25
2.2.11.	Karakteristik Penyakit Tanaman Padi.....	25
2.3 Summary	29
BAB III ANALISIS DAN PERANCANGAN SISTEM	31
3.1 Metode Penelitian	31
3.2. Studi Literatur	31
3.3. Perancangan Modul	32
3.3.1	Pengumpulan dan Preprocess Dataset	33
3.3.2	Perancangan Sistem Klasifikasi	36
3.3.3	Evaluasi Metrik pada Model	43
3.4. Perancangan Aplikasi	44
3.4.1	Perancangan Backend Sistem PWA	44
3.4.2	Perancangan Tampilan Sistem PWA.....	45
3.4.3	Proses Deploy Aplikasi.....	47
3.4.4	Analisa Kecepatan Rata-Rata Prediksi	47
BAB IV IMPLEMENTASI DAN PENGUJIAN SISTEM	49
4.1 Spesifikasi Sistem	49
4.1.1	Spesifikasi Hardware Laptop Penulis	49
4.1.2	Spesifikasi Hardware Google Colaboratory Pro	49
4.2 Implementasi Sistem	50
4.2.1	Proses Pengolahan Dataset.....	50
4.2.2	Proses Klasifikasi	53
4.3 Hasil dan Analisis Training Model	59
4.3.1	Performa Training Model Pada Dataset - UnderSampling	59
4.3.2	Performa Training Model Pada Dataset - OverSampling	62
4.4. Hasil Classification Report Testing Model	65
4.4.1	Hasil Classification Report Model Pada Dataset - <i>UnderSampling</i>	65
4.4.2	Hasil Classification Report Model Pada Dataset - <i>OverSampling</i>	66
4.4.3	Rangkuman Model Terbaik Berdasarkan <i>Classification Report</i>	68
4.5. Analisa Confusion Matrix Terhadap Karateristik Kelas	69
4.6. Hasil dan Analisis Kecepatan Prediksi Model pada Sistem PWA	76

4.6.1	Hasil dan Analisa Kecepatan Prediksi Model - UnderSampling	77
4.6.2	Hasil dan Analisa Kecepatan Prediksi Model - OverSampling	78
4.6.3	Analisa Hubungan Waktu Prediksi dan Ukuran File Model.....	79
BAB V SIMPULAN DAN SARAN	83
5.1 Simpulan	83
5.2 Saran	85
DAFTAR PUSTAKA	86
LAMPIRAN	A



DAFTAR TABEL

Table 1. Rangkuman Poin Acuan Penulis pada Penelitian Terdahulu	15
Table 2. Karakteristik Penyakit Tanaman Padi	26
Table 3. Penggunaan Hyperparameter	41
Table 4. Penggunaan Parameter EarlyStopping	43
Table 5. Validasi Akurasi & Loss Terbaik - UnderSampling	61
Table 6. Validasi Akurasi & Loss Terbaik - OverSampling	64
Table 7. Hasil Classification Report Model – Dataset Teknik UnderSampling ...	65
Table 8. Hasil Classification Report Model - Dataset Teknik OverSampling	67
Table 9. Hasil Tiga Kombinasi Model Terbaik.....	68
Table 10. TP Terburuk Pada Semua Percobaan Model	72
Table 11. TP Terbaik Pada Semua Percobaan Model.....	75
Table 12. Kecepatan Prediksi Model UnderSampling.....	77
Table 13. Kecepatan Prediksi Model OverSampling.....	78

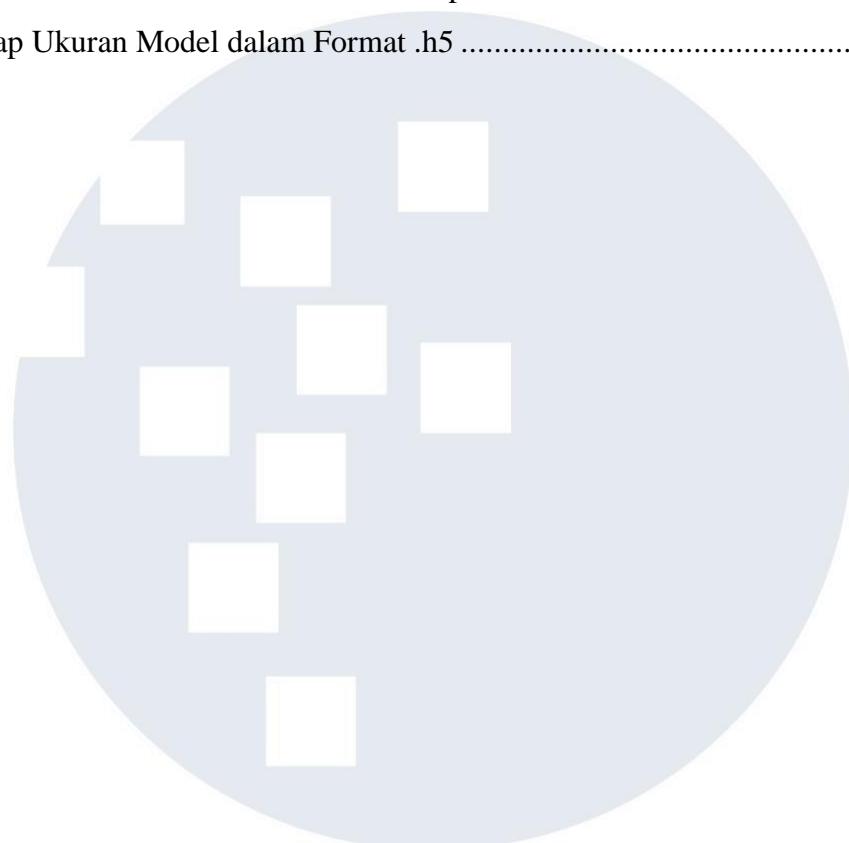


DAFTAR GAMBAR

Gambar 2.1 Ilustrasi Arsitektur CNN [18].....	18
Gambar 2.2 Hasil Modifikasi Model dari MobileNetV2 ke MobileNetV3 [10]... ..	20
Gambar 2.3 Compund Scaling pada EfficientNet [21].....	20
Gambar 2.4 Diagram state of the art.....	30
Gambar 3.1. Metode atau Alur Penelitian	31
Gambar 3.2 Alur Perancangan Modul	32
Gambar 3.3 Pengolahan Dataset; (a) UnderSampling (b) OverSampling	34
Gambar 3.4 Arsitektur MobileNetV3-Large	37
Gambar 3.5 Arsitektur EfficientNetV2B0	37
Gambar 3.6 Skema Penggunaan Average Ensemble.....	38
Gambar 3.7 Skema Penggunaan Concatenation Ensemble.....	39
Gambar 3.8 Penyesuaian Layer	40
Gambar 3.9 Alur Perancangan Sistem Backend.....	45
Gambar 3.10 Alur Perancangan Sistem PWA.....	46
Gambar 3.11 Perancangan Sistem User Interface Sistem PWA	46
Gambar 4.1 Potongan Kode Konfigurasi Augmentasi	50
Gambar 4.2 Contoh Hasil Augmentasi Padi	51
Gambar 4.3 Potongan Kode Augmentasi Kelas Padi	52
Gambar 4.4 Potongan Kode Pemisahan Dataset Latih dan Uji.....	53
Gambar 4.5 Potongan Kode Penyesuaian Layer dan Hyperparameter	53
Gambar 4.6 Potongan Kode Pembagian Data Latih dan Validasi.....	54
Gambar 4.7 Potongan Kode Training Model	54
Gambar 4.8 Potongan Kode Inisialisasi Average Ensemble	55
Gambar 4.9 Potongan Kode Inisialisasi Concatenation Ensemble.....	56
Gambar 4.10 Struktur Folder Backend	57
Gambar 4.11 Tampilan User Interface PWA Paddyist.....	58
Gambar 4.12 Grafik Train & Loss - UnderSampling (224px); (a) EfficientNetV2B0 (b) MobileNetV3-Large (c) Concatenation (d) Average	59

Gambar 4.13 Grafik Train & Loss – UnderSampling (256px); (a)	
EfficientNetV2B0 (b) MobileNetV3Large (c) Concatenation (d) Average	59
Gambar 4.14 Grafik Train & Loss – OverSampling (224px); (a)	
EfficientNetV2B0 (b) MobileNetV3-Large (c) Concatenation (d) Average	62
Gambar 4.15 Grafik Train & Loss – OverSampling (224px); (a)	
EfficientNetV2B0 (b) MobileNetV3-Large (c) Concatenation (d) Average	62
Gambar 4.16 Confusion Matrix EfficientNetV2B0 UnderSampling: (kiri)	224
(kanan) 256.....	69
Gambar 4.17 Confusion Matrix MobileNetV3Large UnderSampling; (kiri)	224
(kanan) 256.....	69
Gambar 4.18 Confusion Matrix Concatenation UnderSampling; (kiri)	224
(kanan) 256	70
Gambar 4.19 Confusion Matrix Average UnderSampling; (kiri)	224
(kanan) 256.....	70
Gambar 4.20 Confusion Matrix EfficientNetV2B0 OverSampling; (kiri)	224
(kanan) 256.....	70
Gambar 4.21 Confusion Matrix MobileNetV3Large OverSampling; (kiri)	224
(kanan) 256.....	71
Gambar 4.22 Confusion Matrix Concatenation OverSampling; (kiri)	224
(kanan) 256	71
Gambar 4.23 Confusion Matrix Average OverSampling; (kiri)	224
(kanan) 256 71	
Gambar 4.24 Analisa Gambar Padi dengan TP Terburuk - UnderSampling	73
Gambar 4.25 Analisa Gambar Padi dengan TP Terburuk - OverSampling	74
Gambar 4.26 Analisa Gambar Padi dengan TP Terbaik	76
Gambar 4.27 Cara Melihat Waktu Prediksi pada PWA – Network Waterfall.....	77
Gambar 4.28 Ukuran File Model-Model Latih Penelitian	79
Gambar 4.29 Grafik Korelasi antara Kecepatan Rata-Rata Prediksi Model Parameter Teknik Balancing terhadap Ukuran Model dalam Format .h5	80
Gambar 4.30 Grafik Korelasi antara Kecepatan Rata-Rata Prediksi Model Parameter Ukuran Gambar terhadap Ukuran Model dalam Format .h5	81

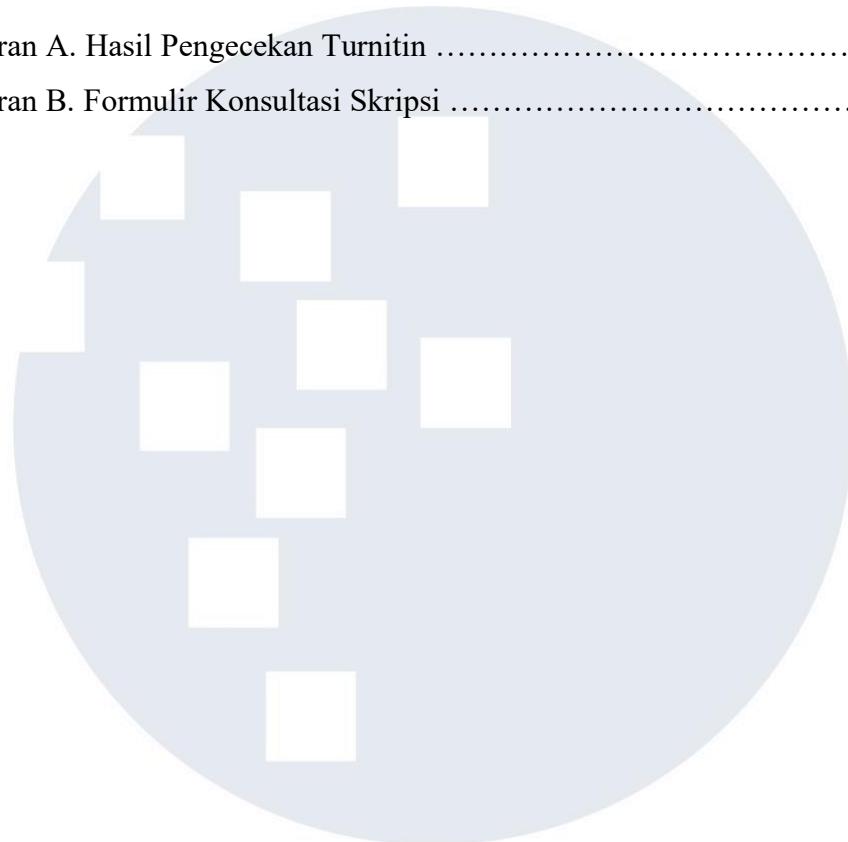
Gambar 4.31 Grafik Korelasi antara Kecepatan Rata-Rata Prediksi Model
terhadap Ukuran Model dalam Format .h5 81



UMN
UNIVERSITAS
MULTIMEDIA
NUSANTARA

DAFTAR LAMPIRAN

Lampiran A. Hasil Pengecekan Turnitin	A
Lampiran B. Formulir Konsultasi Skripsi	C



UMN
UNIVERSITAS
MULTIMEDIA
NUSANTARA

BAB I

PENDAHULUAN

1.1 Latar Belakang

Padi dengan nama latin *Oryza Sativa* merupakan salah satu komoditas pangan yang penting untuk dunia terutama negara-negara Asia. Hal tersebut dapat terlihat pada statistik konsumsi beras di dunia pada tahun 2021/2022 dengan urutan pertama China sekitar 154.890 ton metrik. Disusul oleh India, Vietnam, Bangladesh, dan Indonesia yang menempati posisi kelima [1]. Pada tahun 2021, Indonesia sendiri mengalami penurunan produksi beras dan luas panen sekitar 0.14 juta ton untuk produksi beras dan 0.25 hektar untuk luas panen dengan membandingkan data dari tahun 2020 [2].

Penurunan produksi padi pada tahun 2021 di Indonesia, disebabkan oleh berbagai faktor. Dilansir dari CNBC Indonesia [2], penurunan produksi padi disebabkan oleh kemarau tinggi pada bulan Agustus sampai September, bencana banjir pada awal tahun, erupsi gunung semeru, dan serangan hama. Dilansir pada artikel mengenai salah satu penyakit pada padi, dikatakan bahwa Rice Blast bertanggung jawab atas hilangnya hasil produksi padi sekitar 10-30% di dunia [4]. Penurunan produksi padi akibat faktor-faktor tersebut, tentu menimbulkan berbagai jenis penyakit pada tanaman padi. Pada website Rice Knowledge Bank [5], terdapat sekitar 19 penyakit yang dapat menyerang tanaman padi. Penyakit-penyakit tersebut menimbulkan berbagai gejala yang berujung terjadinya penurunan kualitas tanaman padi bahkan sampai kepada gagalnya panen.

Penyakit tanaman padi yang beragam jenisnya, dapat menyulitkan petani dalam melakukan identifikasi terhadap penyakit tersebut. Identifikasi penyakit pada tanaman padi oleh kebanyakan petani, dilakukan secara manual. Petani perlu mengandalkan pengalaman, pengetahuan, serta kemampuan visual untuk mengidentifikasi penyakit tanaman padi. Namun cara ini kurang efisien, memakan waktu, dan memungkinkan kesalahan identifikasi [6]. Dikarenakan jenis penyakit tanaman padi yang banyak, berdasarkan penelitian yang dilakukan oleh R.

Manavalan [7], petani memiliki pengetahuan yang kurang dalam identifikasi penyakit tanaman padi. Kurangnya pengetahuan serta banyaknya jenis penyakit tanaman padi, dapat menimbulkan telatnya penanggulangan dan bahkan kesalahan dalam mengatasi penyakit pada tanaman padi.

Peningkatan teknologi dan pengetahuan yang pesat, memunculkan berbagai cara yang lebih efektif, tepat, dan cepat dalam mengklasifikasi penyakit pada tanaman padi. Teknologi tersebut adalah *deep learning* dan juga *ensemble learning*. *Deep learning* dapat digunakan untuk mengekstrak fitur, analisa serta mempelajari gambar agar dapat mengklasifikasi penyakit pada tanaman padi. Sedangkan untuk *ensemble learning* memiliki fungsi yang sama dengan *deep learning*, tetapi dapat memberikan performa yang lebih baik. *Deep learning* dan *ensemble learning* dapat diimplementasikan pada ponsel para petani sehingga dapat digunakan untuk klasifikasi penyakit pada tanaman padi dengan lebih cepat dan tepat. Berdasarkan penelitian yang dilakukan oleh R. Manavalan [7], padi yang sudah matang sulit untuk disembuhkan. Sehingga dapat dikatakan klasifikasi dengan menggunakan teknologi tersebut kurang berguna. Namun, jika melihat dari sudut pandang yang lain, petani dapat mengklasifikasi penyakit padi yang sudah matang dan dapat digunakan sebagai pembelajaran bagi petani ke depannya untuk mempersiapkan diri dalam menghadapi penyakit serupa, misal perbaikan dalam praktik budidaya, penggunaan pupuk dan pestisida yang tepat. Tidak hanya bagi petani, penggunaan *deep learning* pada penelitian ini juga dapat berguna bagi peneliti di bidang ilmu pertanian. Penelitian ini dapat digunakan menjadi referensi untuk analisis maupun diterapkan pada teknologi lain seperti *Internet of Things* untuk peningkatan kualitas padi.

Penggunaan *deep learning* sudah banyak dilakukan untuk mengklasifikasi berbagai penyakit pada tanaman termasuk tanaman padi. Penelitian yang dilakukan oleh Paddy Doctor (2022) [6] menggunakan metode *deep learning* dengan memakai *pre-trained* model yaitu VGG16 dan MobileNet untuk mengklasifikasi penyakit tanaman padi menggunakan dataset yang mereka ambil sendiri dengan akurasi terbaik yaitu 93,83%. Namun, penelitian ini tidak menjelaskan secara detail analisa model yang digunakan dan hanya menggunakan *pre-trained* model dan

model CNN buatan sendiri. Terdapat juga penelitian klasifikasi penyakit padi lainnya yang dilakukan oleh Fakhri Habib Hawari, et al [8]. Penelitian ini menggunakan model CNN buatan sendiri untuk mengklasifikasi penyakit pada tanaman padi dengan akurasi *testing* 86%. Namun, akurasi yang didapat masih bisa ditingkatkan lagi dengan mencoba metode *ensemble*. Dari kedua penelitian terebut membuktikan bahwa penggunaan *deep learning* untuk klasifikasi penyakit tanaman padi dapat dilakukan baik menggunakan CNN buatan sendiri maupun *pre-trained* model dengan akurasi yang baik. Sedangkan untuk penggunaan *ensemble learning* dapat terlihat pada penelitian yang dilakukan oleh Odd Virgantara Pura, et al [14], yang menggabungkan *pre-trained* model yaitu DenseNet dan MobileNet dengan teknik *ensemble*. Performa yang didapat lebih baik daripada penggunaan *pre-trained* model itu sendiri. Sehingga dari penelitian tersebut, penggunaan *ensemble learning* dapat meningkatkan performa dari penggunaan *deep learning*. Namun dikarenakan dataset yang digunakan hanya terdiri dari 4 kelas, bisa menjadi faktor utama yang menyebabkan performa meningkat.

Dari latar belakang, permasalahan, serta beberapa penelitian yang telah dipaparkan, penulis akan meneliti performa setiap model yang digunakan pada penelitian ini dan membandingkannya, baik antara model yang diteliti dan penelitian terdahulu. Model yang digunakan adalah dua *pre-trained* model yaitu EfficientNetV2B0 dan MobileNetV3-Large serta dua model yang dihasilkan dari penggunaan dua metode *ensemble* yaitu *average ensemble* dan *concatenation ensemble*. Alasan penggunaan kedua *pre-trained* model tersebut adalah karena memiliki parameter yang sama-sama kecil dengan performa yang bagus [10] [11] dan tidak memerlukan *resource* yang besar atau model ringan ketika diimplementasikan pada sistem *progressive web app*. Selain itu penggunaan kedua model ini dilatarbelakangi pada penelitian terdahulu yang memberikan saran untuk mencoba menggunakan model terbaru untuk meningkatkan performa. Selanjutnya, alasan penggunaan kedua metode *ensemble* tersebut adalah karena merupakan metode yang sederhana, tidak perlu melakukan konfigurasi parameter, kompatibilitas dengan berbagai model, serta adanya *library* keras yang menyediakan *function* untuk memudahkan penulis dalam menggunakan metode

ensemble. Selain itu, penggunaan dua metode *ensemble* ini juga didasari oleh penelitian terdahulu yang berhasil membuktikan peningkatan performa yang dibandingkan dengan *pre-trained* model.

Dataset yang digunakan pada penelitian ini, menggunakan dataset sekunder yang diambil pada penelitian terdahulu yaitu dataset Paddy Doctor [9]. Latar belakang penggunaan dataset ini adalah tempat pengambilan gambar padi berada di India bagian selatan yang masuk ke iklim tropis mirip seperti iklim di Indonesia. Sehingga memiliki penyakit tanaman padi yang mirip dan dimungkinkan hasil penelitian dapat diimplementasikan di Indonesia. Dataset terdiri dari sembilan kelas penyakit tanaman padi, yaitu: Bacterial Leaf Blight, Bacterial Leaf Streak, Bacterial Panicle Blight, Blast, Brown Spot, Dead Heart, Downy Mildew, Hispa, dan Tungro. Penulis juga akan menambah satu kelas tanaman padi yang tidak berpenyakit atau normal. Dataset yang didapat bersifat *imbalanced* sehingga akan melewati proses *balancing* yang akan menghasilkan dataset dengan teknik *undersampling* dan dataset dengan teknik *oversampling*. Penelitian ini juga menggunakan dua ukuran gambar yang berbeda yaitu 224x224 *pixel* dan 256x256 *pixel*. Alasan penggunaan dua ukuran gambar tersebut karena pada penelitian Paddy Doctor, menggunakan ukuran gambar 256x256 *pixel* dan untuk ukuran *default* pada model *pre-trained* yang digunakan adalah 224x224 *pixel*. Sehingga pada penelitian ini akan melihat pengaruh ukuran mana yang lebih baik, yang akan diuji pada dua *pre-trained* model dan dua teknik *ensemble* serta juga akan melihat pengaruh dari penggunaan dua teknik *balancing* dataset yang berbeda.

Penelitian tidak hanya meneliti performa setiap percobaan model, tetapi juga meneliti kecepatan rata-rata waktu prediksi ketika model diimplementasi ke sistem Progressive Web App (PWA). Penggunaan PWA telah banyak diimplementasi pada website misalnya Youtube, Tokopedia, dan lainnya. Dikarenakan keuntungan utama penggunaan PWA adalah aksesibilitas lintas *platform* yang membuat PWA dapat diakses pada Android, iOS, Windows, dan website. Dengan adanya sistem PWA yang dibuat, penulis dapat meneliti kecepatan rata-rata waktu prediksi untuk melihat model yang memiliki kecepatan prediksi terbaik. Sehingga dapat dijadikan referensi untuk penelitian lainnya dalam menentukan model yang ingin dipakai.

Alasan lain penggunaan PWA adalah penulis tidak perlu berurusan dengan App Store yang memerlukan biaya agar dapat diinstal pada ponsel secara publik.

1.2 Identifikasi Masalah

Berdasarkan latar belakang yang sudah penulis paparkan, identifikasi masalah yang didapat terdiri dari beberapa poin yaitu:

- 1.2.1. Apakah penggunaan metode *pre-trained deep learning* dan *ensemble learning* dapat meningkatkan performa pada dataset klasifikasi penyakit tanaman padi?
- 1.2.2. Apakah metode *balancing* dataset yaitu *undersampling* dan *oversampling* mampu meningkatkan performa model dan pengaruh terhadap biaya komputasi serta waktu pelatihan?
- 1.2.3. Apakah ukuran gambar yaitu 224x224 *pixel* dan 256x256 *pixel* berpengaruh terhadap model yang akan diuji?
- 1.2.4. Apakah perbedaan model, dataset, dan ukuran gambar mempengaruhi waktu pemrosesan dalam klasifikasi penyakit tanaman padi pada sistem *progressive web app*?

1.3 Batasan Penelitian

Batasan dari penelitian ini adalah sebagai berikut:

- 1.3.1. Penelitian ini menggunakan dataset publik Paddy Doctor [9]
- 1.3.2. Penelitian ini memakai model yang telah dilatih atau *pre-trained* model yang disediakan oleh TensorFlow dan Keras
- 1.3.3. Sistem tidak dapat mengklasifikasi foto padi yang diambil pada malam hari.
- 1.3.4. Penelitian ini tidak menggunakan kelas non-padi untuk membedakan antara padi atau bukan. Sehingga penelitian hanya mampu membedakan 9 penyakit tanaman padi dan tanaman padi normal.
- 1.3.5. Sistem progressive web app yang dibuat hanya sebatas *prototype* dalam artian sistem tidak seluruhnya *dideploy* di *Internet*.
- 1.3.6. Penelitian ini tidak melalukan validasi ke petani atau *experts* pada bidang pertanian.

1.4 Tujuan Penelitian

Penelitian ini dilakukan dengan tujuan membandingkan performa beberapa model yang memiliki akurasi terbaik dalam mengklasifikasi penyakit pada tanaman padi baik dengan dataset *undersampling* maupun *oversampling* atau dengan ukuran gambar 224x224 *pixel* dan 256x256 *pixel*. Selain itu menentukan model yang memiliki performa kecepatan pemrosesan terbaik jika diimplementasikan pada sistem progressive web app.

1.5 Manfaat Penelitian

Penelitian ini memiliki beberapa manfaat sebagai berikut:

- 1.5.1. Dapat digunakan sebagai sumber untuk penelitian lainnya sebagai tolak ukur model yang memiliki akurasi terbaik serta pemrosesan yang tercepat jika diimplementasikan pada *progressive web app*.
- 1.5.2. Dapat digunakan sebagai sumber acuan untuk jurnal lainnya dalam pengaruh akurasi baik dengan dataset *undersampling* maupun *oversampling*.
- 1.5.3. Dapat digunakan sebagai sumber acuan dalam penggunaan metode *ensemble learning* yang dapat meningkatkan akurasi model.
- 1.5.4. Meningkatkan penggunaan *progressive web app* sebagai salah satu alternatif dalam membangun aplikasi mobile dan website pada satu kode program.
- 1.5.5. Membantu petani dalam mengklasifikasi penyakit pada tanaman padi mereka untuk meningkatkan produksi padi serta dapat digunakan pada penelitian sejenis dan dapat diimplementasikan pada sistem lainnya seperti IoT.

1.6 Sistematika Penulisan

Sistematika penulisan laporan penelitian disusun dan dibagi atas lima bab sebagai berikut.

1.6.1. BAB 1 PENDAHULUAN

Pada Bab ini berisi latar belakang masalah yang ingin diteliti, identifikasi permasalahan, batasan masalah yang akan diteliti, tujuan penelitian, manfaat penelitian, dan sistematika dalam penulisan laporan.

1.6.2. BAB 2 LANDASAN TEORI

Pada Bab ini menjelaskan landasan penelitian terdahulu yang mendukung penelitian yang dilakukan penulis serta teori yang akan digunakan peneliti dalam meneliti permasalahan seperti Convolutional Neural Network (CNN), *Deep Learning*, *Ensemble Learning*, MobileNetV3-Large, EfficientNetV2B0, Karakteristik Penyakit Tanaman Padi, dan Implementasi pada sistem PWA.

1.6.3. BAB 3 ANALISIS DAN PERANCANGAN SISTEM

Pada Bab ini berisi metode penelitian, perancangan modul, dan perancangan aplikasi.

1.6.4. BAB 4 HASIL DAN ANALISA PENELITIAN

Pada Bab ini berisi hasil dari penelitian yang telah dilakukan serta menganalisa hasil penelitian.

1.6.5. BAB 5 SIMPULAN DAN SARAN

Pada Bab ini berisi simpulan serta saran dari penelitian yang telah dilakukan yang nantinya dapat menjadi referensi untuk civitas akademika lainnya untuk mengembangkan penelitian mereka.

BAB II

TINJAUAN PUSTAKA

2.1 Penelitian Terdahulu

Penulis menemukan beberapa penelitian terdahulu terkait klasifikasi penyakit tanaman padi dengan metode yang berbeda, seperti *deep learning*, *ensemble learning*, serta mengimplementasikan model yang ada pada sistem progressive web app.

2.1.1. *Paddy Doctor: A Visual Image Dataset for Paddy Disease Classification* [6]

Penelitian dengan judul “*Paddy Doctor: A Visual Image Dataset for Paddy Disease Classification*” yang dilakukan oleh Petchiammal A, Briskline Kiruba S, D. Murugan, dan Pandarasamy A. Penelitian ini dilakukan dengan latar belakang penelitian sebelumnya untuk identifikasi memiliki dataset yang kurang detail. Sehingga mengurangi akurasi deteksi dari klasifikasi penyakit pada tanaman padi. Penelitian terdahulu menggunakan dataset sendiri dengan total gambar 13,876 tanaman padi yang sudah terlabel ke dalam sepuluh kelas. Penelitian terdahulu juga menggunakan metode CNN buatan sendiri dan dua pendekatan *transfer learning* yaitu VGG16 dan MobileNet. Hasil akurasi yang didapat adalah 93.83% untuk MobileNet, 93,66% untuk VGG16, dan 87.97% untuk DCNN.

Beberapa poin penting yang dapat diambil oleh penulis adalah sebagai berikut:

- Dataset yang digunakan bukan hasil potongan ataupun hanya gambar sehelai daun padi, melainkan gambar yang memiliki latar belakang tanaman padi lain.
- Salah satu model dan pendekatan yang digunakan pada penelitian terdahulu yaitu MobileNet menjadi acuan bagi penulis untuk

menggunakan versi terbaru dari MobileNet yaitu MobileNetV3-Large.

- Penggunaan ukuran gambar 256x256 *pixel* pada penelitian terdahulu memberikan akurasi yang sangat baik dan menjadi acuan penulis untuk mencoba menggunakan model yang berbeda.
- Hasil penelitian terdahulu kurang memberikan gambaran secara terperinci perbandingan antar model.

2.1.2. Klasifikasi Penyakit Padi Menggunakan Algoritma CNN (*Convolutional Neural Network*) [8]

Penelitian dengan judul “Klasifikasi Penyakit Padi Menggunakan Algoritma CNN (*Convolutional Neural Network*)” yang dilakukan oleh Fakhri Habib Hawari, et al. Penelitian dengan latar belakang untuk meningkatkan produksi padi, minimal menjaga kestabilan ketahanan pangan nasional, dan membantu petani dalam mengklasifikasi penyakit tanaman padi serta mengurangi resiko gagal panen. Dataset yang digunakan menggunakan 1 daun padi yang terdiri dari 4 kelas yaitu *Brown Spot*, *Hawar*, *Leaf Brown*, dan daun sehat. Setiap gambar diubah menjadi 256x256 piksel dengan *batch size* 32. Model yang digunakan juga menggunakan model yang diajukan sendiri. Beberapa layer digunakan seperti *sequential layer*, *conv2D*, *maxPooling2D*, *flatten*, dan *dense layer* untuk membuat model CNN.

Hasil yang didapat dari grafik pembelajaran memiliki nilai akurasi pembelajaran sebesar 85% dan akurasi validasi sebesar 95%. *Training* dilakukan sebanyak 10 epoch. Walaupun memiliki akurasi validasi yang besar, tetapi akurasi uji yang didapat adalah 86% dengan loss yang cukup tinggi yaitu 49%.

Beberapa poin penting yang dapat diambil penulis adalah sebagai berikut:

- Penggunaan teknologi Deep Learning yaitu CNN mampu mengklasifikasikan penyakit tanaman padi dengan akurasi yang cukup baik.
- Berdasarkan saran pada penelitian terdahulu, dapat menambahkan kategori penyakit padi lain. Hal ini dikarenakan kelas penyakit pada penelitian terdahulu ini, hanya memiliki 4 kelas saja.
- Juga menambahkan arsitektur dari CNN lain seperti penggunaan *pre-trained* model untuk meningkatkan akurasi.

2.1.3. *A Deep Learning-Based Mobile App System for Visual Identification of Tomato Plant Disease* [12]

Penelitian dengan judul “*A Deep Learning-Based Mobile App System for Visual Identification of Tomato Plant Disease*” yang dilakukan oleh Aurelius Ryo Wang dan Nabila Husna Shabrina. Penelitian dengan latar belakang tanaman tomat sebagai salah satu peran penting dalam pemenuhan kebutuhan makanan yang rentan terhadap penyakit. Sehingga penelitian ini membuat sistem berbasis aplikasi *mobile* yang dapat mengidentifikasi penyakit pada tanaman tomat dengan metode EfficientNetB0. Hal penting yang dapat penulis dapatkan pada penelitian ini adalah penggunaan EfficientNetB0 pada aplikasi *mobile* dengan waktu prediksi rata-rata 40.3744 ms dan waktu pemrosesan rata-rata 103.183 ms. Selain itu, penelitian terdahulu ini memberikan saran untuk meningkatkan performa hasil klasifikasi dengan mencoba menggunakan model *deep learning* lainnya. Hal ini dikarenakan akurasi validasi yang didapat adalah 92.22% yang dapat dimungkinkan untuk ditingkatkan kembali.

Beberapa poin penting yang dapat diambil oleh penulis adalah sebagai berikut:

- Penggunaan model EfficientNetB0 mampu memberikan performa akurasi validasi yaitu 92.22% pada klasifikasi penyakit

tanaman tomat dan saran yang diberikan penelitian terdahulu, untuk mencoba menggunakan model *deep learning* lainnya untuk meningkatkan akurasi. Sehingga penulis ingin mencoba menggunakan model terbaru dari EfficientNetB0 yaitu EfficientNetV2B0 pada klasifikasi penyakit pada tanaman padi dengan harapan memberikan performa yang sama atau lebih baik.

- Penelitian terdahulu mencoba menganalisa prediksi waktu rata-rata dengan jenis ekstensi gambar yang berbeda-beda (JPEG, *string base 64*, dan *string base 64 x 5*). Hal ini menjadi sumber inspirasi penulis untuk menganalisa prediksi waktu rata-rata, tetapi dengan parameter model, ukuran gambar, dan teknik *balancing dataset* yang berbeda.

2.1.4. Automatic Identification of Disease in Grains Crops through Computational Approches: A Review [7]

Penelitian dengan judul “Automatic Identification of Disease in Grains Crops throught Computational Approches: A Review” yang dilakukan oleh R. Manavalan. Penelitian ini berfokus pada membandingkan berbagai metode baik dari sisi *preprocessing*, segmentasi, ekstraksi fitur, seleksi fitur, dan klasifikasi pada berbagai variasi tanaman biji-bijian. Misalnya pada proses *preprocessing* terdapat berbagai metode seperti teknik *region filling*, *filter median* dan lain sebagainya dibandingkan tingkat akurasi masing-masing dari metode tersebut. Selain itu, penelitian terdahulu ini juga mengabungkan berbagai tahap dengan metode terbaik dan mencari kembali akurasi yang paling tinggi. Penulis mendapatkan banyak pembelajaran baru pada penelitian terdahulu ini karena memaparkan secara detail akurasi metode pada setiap tahap deteksi penyakit pada tanaman biji-bijian.

Beberapa poin penting yang dapat diambil oleh penulis adalah sebagai berikut:

- Penelitian terdahulu ini memperkuat latar belakang penelitian penulis mengenai petani yang memiliki pengetahuan yang kurang mengenai penyakit pada tanaman biji-bijian.
- Padi yang sudah matang dan terkena penyakit sulit atau bahkan tidak dapat disembuhkan. Sehingga menjadi poin penting, sistem klasifikasi penyakit tanaman padi yang dibuat oleh penulis, dapat dijadikan acuan bagi petani untuk kedepannya dalam menanam kembali padi yang baru dengan cara penanaman dan penggunaan pestisida yang tepat berdasarkan tanaman padi yang diklasifikasi melalui sistem.
- Penggunaan berbagai metode pada tahap *preprocessing* dan segmentasi menjadi acuan bagi penulis untuk menggunakan tahap *preprocessing* untuk augmentasi gambar agar dapat memperbaiki hasil latih.

2.1.5. Klasifikasi Penyakit Tanaman Padi Menggunakan Model *Deep Learning* Efficientnet B3 Dengan *Transfer Learning* [13]

Penelitian dengan judul “Klasifikasi Penyakit Tanaman Padi Menggunakan Model *Deep Learning* Efficientnet B3 Dengan *Transfer Learning*” yang dilakukan oleh Endang Anggiratih, Sri Siswanti, Saly Kurnia Octaviani, dan Arumsari. Penelitian ini berfokus pada membandingkan akurasi dua model yang digunakan yaitu EfficientNetB3 dan MobileNetV3. EfficientNetB3 memiliki akurasi tinggi daripada MobileNetV3 dengan hasil 79.53%. Namun MobileNetV3 memiliki *loss* yang lebih rendah dibandingkan EfficientNetB3 yaitu 0.007. Kekurangan dari penelitian ini adalah hanya menggunakan dua kelas klasifikasi yaitu *brown spot* dan *bacterial leaf*. Sehingga akurasi yang didapat bisa sangat tinggi.

Beberapa poin penting yang dapat diambil oleh penulis adalah sebagai berikut:

- Segmentasi atau pembagian dataset pada penelitian terdahulu ini yaitu 60:20:20 untuk data latih, data validasi, dan data uji memberikan hasil yang sangat baik.
- Penggunaan *epoch* yang berlebihan malah menurunkan tingkat akurasi pada model tersebut. Pada penelitian terdahulu ini, akurasi menurun ketika *epoch* lebih dari 300.
- Akurasi dari EfficientNetB3 dapat ditingkatkan berdasarkan tabel aplikasi keras untuk akurasi tertinggi adalah 79.53%.

2.1.6. *HiT-LIDIA: A Framework for Rice Leaf Disease Classification using Ensemble and Hierarchical Transfer Learning* [14]

Penelitian dengan judul “*HiT-LIDIA: A Framework for Rice Leaf Disease Classification using Ensemble and Hierarchical Transfer Learning*” yang dilakukan oleh Oddy Virgantara Putra, Niken Trisnaningrum, Niken Sylvia Puspitasari, Agung Toto Wibowo, dan Ema Rachmawaty. Penelitian ini berfokus membandingkan beberapa *pre-trained* model yang ada dengan model yang dibuat oleh penulis menggunakan teknik *ensemble* pada penyakit tanaman padi. Penelitian ini menggunakan dua *ensemble* model berbeda dengan tujuan yang berbeda. Model pertama gabungan antara MobileNet dan DenseNet (MK-I) digunakan untuk meneliti tanaman padi yang normal dan berpenyakit. Model kedua gabungan antara DenseNet dan XceptionNet (MK-II) digunakan untuk meneliti tiga penyakit tanaman padi.

Hasil yang didapat, kedua model *ensemble* tersebut mendapatkan akurasi yang lebih tinggi daripada model *pre-trained*. Akurasi yang didapat untuk MK-I adalah 0.89% berada diatas akurasi *pre-trained* model lainnya seperti DenseNet201 yang mendapatkan akurasi 0.88%. Akurasi yang didapat untuk MK-II adalah 0.91% berada diatas akurasi *pre-trained* model lainnya seperti Xception 0.8829%.

Beberapa poin penting yang dapat diambil oleh penulis adalah sebagai berikut:

- Penggunaan metode *ensemble* memiliki akurasi yang lebih tinggi dibandingkan penggunaan *pre-trained* model
- Penyajian evaluasi grafik akurasi dan loss yang rapi dan unik menjadi acuan penulis dalam menyusun hasil penelitian.

2.1.7. *Impact of Image Resolution on Deep Learning Performance in Endoscopy Image Classification: An Experimental Study Using a Large Dataset of Endoscopic Images* [15]

Penelitian dengan judul “*Impact of Image Resolution on Deep Learning Performance in Endoscopy Image Classification: An Experimental Study Using a Large Dataset of Endoscopic Images*” yang dilakukan oleh Vajira Thambawita, Inga Strumke, Steven A. Hicks, Pal Halvorsen, Sravanthi Parasa, dan Michael A. Reigler. Penelitian ini berfokus pada pengaruh penggunaan berbagai ukuran gambar dalam performa *training* dan juga rata-rata waktu hasil prediksi.

Penelitian ini menggunakan lima ukuran gambar yang berbeda serta dua model yang berbeda sebagai bahan pembanding. Ukuran gambar yang digunakan adalah *32x32 pixel*, *64x64 pixel*, *128x128 pixel*, *256x256 pixel*, dan *512x512 pixel*. Sedangkan untuk model yang digunakan yaitu DenseNet-161 dan Resnet-152. Dari hasil penelitian yang dilakukan, dapat diambil kesimpulan bahwa semakin tinggi ukuran gambar juga akan meningkatkan performa akurasi dengan peningkatan yang tidak terlalu signifikan baik untuk kedua model. Selain itu hasil kecepatan waktu prediksi rata-rata menunjukkan hasil yang fluktuatif. Sehingga dapat diambil kesimpulan bahwa meningkatkan ukuran gambar tidak memiliki efek yang besar untuk kecepatan prediksi.

Beberapa poin penting yang dapat diambil oleh penulis adalah sebagai berikut:

- Peningkatan ukuran gambar juga meningkatkan performa akurasi hasil model yang dilatih.

- Peningkatan ukuran gambar tidak memiliki efek yang besar untuk kecepatan prediksi. Hal ini terbukti dari hasil yang didapat bersifat fluktuatif.

Berdasarkan berbagai penelitian terdahulu yang diambil penulis, berikut beberapa poin yang menjadi acuan penelitian yang dilakukan oleh penulis.

Table 1. Rangkuman Poin Acuan Penulis pada Penelitian Terdahulu

Judul Penelitian (Tahun)	Poin yang menjadi acuan penulis
<i>Paddy Doctor: A Visual Image Dataset for Paddy Disease Classification</i> (2022)	<ul style="list-style-type: none"> • Penelitian terdahulu ini memberikan <i>dataset</i> dengan 13,876 gambar terdiri dari 10 kelas (9 penyakit dan 1 normal) • Memastikan bahwa <i>dataset</i> dapat dilatih menggunakan <i>deep learning</i> • Penggunaan model MobileNet sebagai acuan penulis untuk menggunakan model MobileNet terbaru yaitu MobileNetV3-Large • Penggunaan gambar dengan ukuran 256x256 <i>pixel</i> memberikan hasil yang sangat baik dan menjadi acuan penulis untuk mencoba menggunakan ukuran gambar tersebut dengan model yang berbeda
Klasifikasi Penyakit Padi Menggunakan Algoritma CNN (<i>Convolutional Neural Network</i>)	<ul style="list-style-type: none"> • Saran yang diberikan pada penelitian terdahulu dalam menggunakan <i>pre-trained</i> CNN, menjadi acuan penulis untuk menggunakan <i>pre-trained</i> model • Saran yang diberikan pada penelitian terdahulu dalam menggunakan kelas dataset yang lebih banyak, membuat

	<p>penulis mencoba mencari dataset yang memiliki kelas lebih banyak.</p>
<i>A Deep Learning-Based Mobile App System For Visual Identification of Tomato Plant Disease</i>	<ul style="list-style-type: none"> Penggunaan model EfficientNetB0 sebagai acuan penulis untuk menggunakan model EfficientNetV2B0 Sebagai acuan penulis mengimplementasikan model latih ke sistem PWA Analisa rata-rata waktu prediksi yang dilakukan menjadi acuan penulis untuk melakukan analisa rata-rata waktu prediksi diimplementasi pada PWA dengan membandingkan model, ukuran gambar, dan teknik <i>balancing dataset</i> yang berbeda
<i>Automatic Identification of Diseases in Grains Crops Through Computational Approaches: A Review</i>	<ul style="list-style-type: none"> Percobaan dalam berbagai metode <i>preprocess</i>, menjadi sumber acuan pada penelitian ini.
Klasifikasi Penyakit Tanaman Padi Menggunakan Model <i>Deep Learning</i> EfficientNetB3 Dengan <i>Transfer Learning</i>	<ul style="list-style-type: none"> Mendukung pembagian data latih dan data validasi. Menunjukkan peningkatan <i>epoch</i> tidak akan selalu linear dengan peningkatan akurasi.
<i>HiT-LIDIA: A Framework for Rice Leaf Disease Classification using Ensemble and Hierarchical Transfer Learning</i>	<ul style="list-style-type: none"> Penggunaan metode <i>ensemble</i> yang memiliki akurasi lebih baik daripada <i>pre-trained</i> model, menjadi acuan penulis untuk mencobanya. Penyajian evaluasi akurasi dan <i>loss</i> yang unik menjadi acuan penulis untuk menyusun penelitian.

<p><i>Impact of Image Resolution on Deep Learning Performance in Endoscopy Image Classification: An Experimental Study Using a Large Dataset of Endoscopic Images</i></p>	<ul style="list-style-type: none"> • Penggunaan ukuran gambar yang berbeda ternyata berpengaruh terhadap performa model. Sehingga menjadi acuan penulis untuk mencoba menggunakan dua ukuran gambar yang berbeda yaitu 224x224 pixel dan 256x256 pixel • Hasil waktu rata-rata prediksi ternyata bersifat fluktuatif. Penulis ingin mencoba dengan menggunakan model yang lain serta ukuran gambar 224x224 pixel untuk melihat apakah memang waktu rata-rata prediksi bersifat fluktuatif
---	---

2.2 Tinjauan Teori

2.2.1. Deep Learning

Deep learning merupakan salah satu bagian dari *machine learning*, yang simpelnya memiliki tiga atau lebih layer jaringan neural. Jaringan neural pada *deep learning* meniru cara kerja otak melalui kombinasi antara data input, *weight*, dan *bias*. *Deep learning* terdiri dari banyak lapisan *node* yang saling terhubung dan memiliki dua proses yaitu *forward propagation* dan *back propagation* yang digunakan untuk memprediksi *error* dan menyesuaikan *weight* dan *bias* dari setiap *node* yang ada. Proses ini akan terus berulang sampai mendapatkan akurasi yang sesuai.

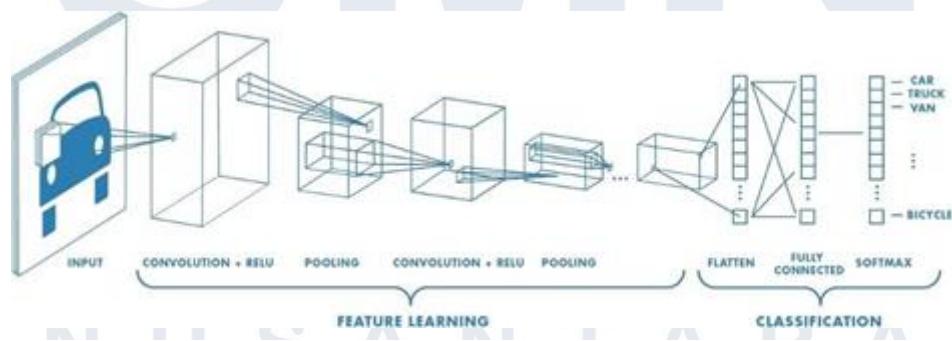
Deep learning sendiri dapat diklasifikasikan ke dalam tiga kategori utama yaitu *unsupervised learning*, *semi-supervised learning*, dan *supervised learning*. *Unsupervised learning* adalah proses pembelajaran yang datanya tidak diberikan label, contohnya adalah *clustering*. *Supervised learning* adalah proses pembelajaran yang datanya diberikan label, contohnya adalah CNN yang akan digunakan pada penelitian ini. *Semi-supervised learning* adalah proses pembelajaran yang datanya beberapa diberi label dan beberapa tidak diberi label, contohnya *Generative Adversarial Networks* (GANs).

2.2.2. Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) merupakan salah satu jenis dari *deep learning* yang dapat digunakan untuk mengklasifikasi, mendeteksi, ataupun mengenali obyek pada suatu gambar. CNN terdiri dari neuron yang memiliki *weight*, *bias*, dan fungsi aktivasi. Strukturnya sendiri meniru struktur neuron pada otak.

CNN bekerja dengan menggerakkan *kernel* (yang dapat diatur ukurannya) pada gambar dan memecah gambar menjadi gambar yang lebih kecil dan saling tindih. Gambar-gambar kecil tersebut akan dimasukkan dalam jaringan *neural* untuk diekstrak fiturnya. Jika ada bagian dari gambar dirasa menarik, maka gambar tersebut akan diambil sebagai *object of interest*. Hasil yang diambil tersebut tersimpan dalam bentuk *array* yang nantinya akan masuk ke proses *downsampling*. *Downsampling* digunakan untuk mengecilkan ukuran dari *array*, tetapi tidak mengurangi informasi penting pada *array* tersebut. *Array* yang berukuran kecil akan masuk ke jaringan saraf terakhir yang bisa disebut dengan *fully connected network* yang bertujuan untuk memprediksi gambar.

CNN sendiri terbagi menjadi 2 bagian, yaitu lapisan ekstraksi fitur, dan lapisan terhubung penuh. Lapisan ekstraksi fitur bertujuan untuk mengubah gambar dalam bentuk angka yang akan menjadi fitur daripada gambar tersebut. Sedangkan untuk lapisan terhubung penuh bertujuan untuk mengubah bentuk *array* fitur gambar menjadi sebuah *vector* yang digunakan untuk mengolah data sehingga dapat diklasifikasikan. Berikut Gambar 2.1 struktur umum dari metode CNN.



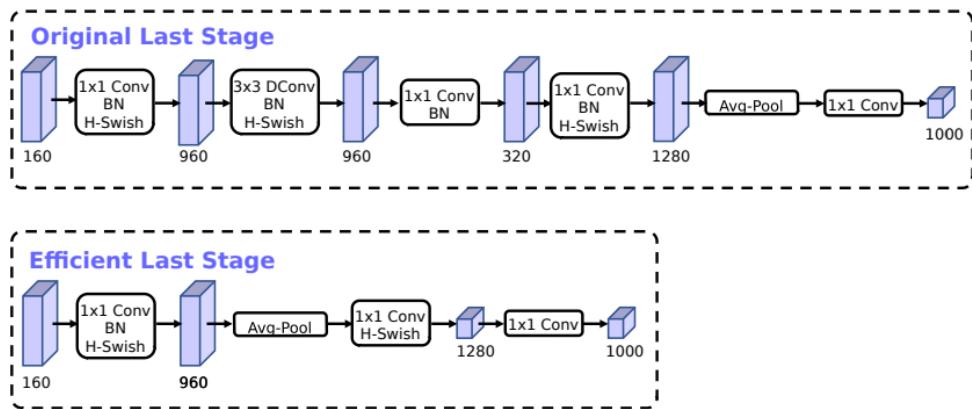
Gambar 2.1 Ilustrasi Arsitektur CNN [18]

2.2.3. MobileNetV3

MobileNetV3 diawali dengan versi pertamanya yaitu MobileNet. MobileNet merupakan model yang dibuat dengan arsitektur jaringan yang efisien dan menggunakan dua *hyperparameter* untuk membangun model yang sangat kecil dan latensi rendah. Sehingga cocok diimplementasi pada perangkat *mobile* maupun aplikasi kamera tertanam. MobileNet dibuat dengan metode *depthwise separable convolution* yang merupakan gabungan dari dua layer yaitu *depthwise convolutions* (menggunakan satu filter pada setiap *input channel*) dan *pointwise convolution* (konvolusi simpel dengan ukuran 1x1). Dengan tambahan *width multiplier* dan *resolution multiplier*. Sehingga dapat memperkecil dan mempercepat model.

Setelah itu muncullah MobileNetV2, dengan urgensi berfokus meningkatkan efisiensi model ke dalam perangkat *mobile* lebih tinggi lagi. Model dasarnya menggunakan pendekatan *bottleneck depth-separable convolution with residuals* dan menggunakan *trade-off hyperparameters*. *Inverted residual bottleneck layer* mengizinkan penggunaan memori yang hemat sangat penting bagi penggunaan aplikasi *mobile*.

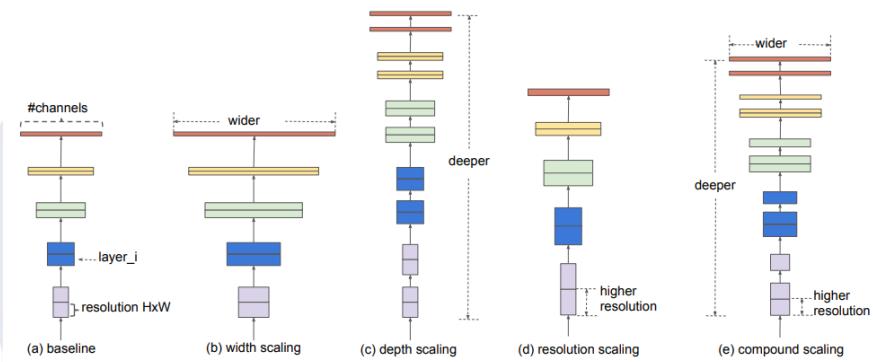
MobileNetV3 versi terbaru dari model MobileNet yang menggunakan dua teknik *network search* yaitu *platform aware NAS* untuk pencarian *block-wise* dan NetAdapt untuk pencarian *layer-wise*. Namun, setelah diteliti ternyata layer awal dan akhir sangat mahal dalam hal komputasi. Sehingga ditawarkan beberapa modifikasi untuk mengurangi latensi pada layer tersebut dengan tetap menjaga akurasi. Modifikasi pertama, pada model MobileNetV2, konvolusi 1x1 digunakan sebagai *final layer* akan dipindahkan ke layer setelah *final average pooling*. Modifikasi kedua, membuang *layer bottleneck* yang ada di belakang layer konvolusi 1x1. Modifikasi ketiga, penggunaan 32 filter pada konvolusi 3x3 tidak terlalu penting dan dapat dikurangi dengan menggunakan *hard swish*. Berikut Gambar 2.2 perubahan layer pada MobileNetV3, baik sebelum dan sesudah dimodifikasi.



Gambar 2.2 Hasil Modifikasi Model dari MobileNetV2 ke MobileNetV3 [10]

2.2.4. EfficientNetV2B0

EfficientNet untuk saat ini memiliki dua versi, EfficientNet dan EfficientNetV2. EfficientNet terbentuk menggunakan pencarian arsitektur neural (NAS) yang menjadi dasar jaringan. Selain itu EfficientNet menggunakan metode *compound scaling* yang merupakan gabungan dari tiga skala yaitu *width*, *depth*, dan *resolution scaling*. Metode *compound scaling* ini yang akan menciptakan EfficientNetB1 sampai EfficientNetB7.



Gambar 2.3 Compund Scaling pada EfficientNet [21]

Selanjutnya EfficientNetV2 datang yang merupakan versi terbaru, lebih cepat, dan memiliki performa lebih baik daripada EfficientNet. Hal yang membuat EfficientNetV2 memiliki performa lebih baik sekaligus perbedaan dari versi sebelumnya yaitu:

- EfficientNetV2 menggunakan MBConv dan Fused-MBConved (pada awal layer). Sedangkan EfficientNet hanya menggunakan

MBConv. Fused-MBConv memberikan performa yang lebih baik pada *mobile* atau akselerator server. Namun tidak semua layer pada EfficientNetV2 diubah dari MBConv menjadi Fused-MBConved. Karena pada percobaan yang dilakukan, agar mendapatkan performa yang lebih baik, harus terdapat kombinasi antara keduanya.

- EfficientNetV2 menggunakan *training aware NAS framework* yang lebih besar daripada *NAS Framework*.
- EfficientNetV2 memiliki layer MBConv yang lebih sedikit untuk mengurangi kelebihan akses memori
- EfficientNetV2 menghapus *stride* terakhir pada *original EfficientNet*, untuk mengurangi besarnya ukuran parameter

2.2.5. Ensemble Learning

Ensemble learning merupakan salah satu metode untuk menggabungkan beberapa model *deep learning*. Hasil penggabungan beberapa model tersebut menghasilkan model baru yang memberikan hasil prediksi lebih baik. Selain itu penggunaan *ensemble learning* juga dapat membantu mengurangi penggunaan dataset yang beragam, mengatasi *overfitting*, dan tidak membutuhkan waktu yang lama dalam membuat model yang baru.

Terdapat beberapa teknik untuk menggabungkan beberapa algoritma deep learning [22], sebagai berikut:

- *Average Method*

Metode *Average* membuat model baru dengan cara mengambil rata-rata sederhana dari dua atau lebih model yang memiliki akurasi yang lebih baik dibandingkan dengan gabungan dua model. Metode *Average* mengambil rata-rata dari nilai weight pada masing-masing model yang digabungkan.

- *Bagging Method*

Metode *Bagging* adalah metode yang menggabungkan versi atau iterasi yang berbeda pada masing-masing model *deep learning* untuk

menghasilkan model yang lebih akurat dan mempunyai performa yang tinggi. Caranya mirip sama dengan metode *Average*, tetapi perbedaannya adalah model baru yang terbuat, tidak memiliki *bias*.

- *Boosting Method*

Metode *Boosting* mirip seperti penggunaan *feedback loop* pada suatu model. Performa model yang sekarang digunakan untuk penyesuaian pada model berikutnya. Ini akan menciptakan *feedback loop* yang bersifat positif dengan mengumpulkan semua faktor yang membantu kesuksesan suatu model.

- *Concatenation Method*

Metode ini menggunakan *input* yang berbeda pada masing-masing model dan menggabungkannya ke satu model yang sama. Sehingga dataset yang akan dilatih akan memiliki dimensi yang lebih banyak dibandingkan *dataset* awal. Namun, jika kita menggabungkan beberapa model secara berulang, akan menyebabkan *overfitting* dan hilangnya informasi penting pada data yang dilatih.

- *Stacking Method*

Metode *Stacking* mengintegrasikan beberapa model *deep learning* layaknya disusun untuk mendapatkan performa yang lebih baik. Simpelnya adalah hasil dari model pertama akan digunakan untuk model kedua.

2.2.6. Balanced dan Imbalanced Dataset

Dalam mengumpulkan data, wajar jika data yang terkumpul tidaklah seimbang, ada yang lebih sedikit dan ada yang lebih banyak. Contohnya ketika mengidentifikasi penyakit pada tanaman padi, penyakit Dead Heart lebih sering ditemukan daripada penyakit Bacterial Panicle Blight. Hal ini tercermin pada *database* yang akan digunakan oleh penulis. Dataset inilah yang disebut *dataset* yang tidak *balanced* atau *imbalanced dataset*. *Imbalanced dataset* adalah ketika terdapat perbedaan yang cukup tinggi antara kelas positif dan kelas negatif. Sedangkan *balanced dataset* adalah ketika jumlah data kelas positif dan kelas negatif memiliki jumlah data yang hampir sama.

Imbalanced data tidak selalu buruk bagi performa model, jika perbedaan antar kelas tidak terlalu signifikan. Namun alangkah baiknya untuk melakukan *balancing data* untuk mendapatkan performa model lebih baik. Terdapat beberapa cara untuk membalancing data yang penulis gunakan untuk penelitian ini, yaitu:

- *Over-sampling*

Over-sampling adalah teknik yang memperbanyak jumlah data pada kelas yang lebih sedikit daripada kelas mayoritas. Untuk memperbanyak jumlah data, dapat menggunakan metode augmentasi data yang akan dijelaskan pada subbab berikutnya. Keuntungan menggunakan *Over-sampling* adalah tidak ada informasi yang hilang saat proses latih. Namun kekurangan yang didapat adalah dapat menyebabkan *Overfitting*.

- *Under-sampling*

Under-sampling adalah teknik mengurangi jumlah data pada kelas yang lebih banyak daripada distribusi kelas yang ada. Keuntungan menggunakan *Under-sampling* adalah mempercepat waktu proses *training* dan menyelesaikan permasalahan memori saat proses *training*. Namun kekurangan yang didapat adalah dapat menghilangkan beberapa informasi penting yang mungkin ada pada gambar yang kita kurangi.

- Teknik *Ensemble Learning* pada Model

Ensemble learning pada model adalah teknik lain yang dapat digunakan untuk menangani kumpulan data yang tidak seimbang. Teknik ini menggabungkan hasil atau kinerja beberapa model untuk mendapatkan peningkatan performa model yang baru. Terdapat berbagai metode yang dapat digunakan untuk mengimplementasi *ensemble learning*, yaitu *Bagging*, *Boosting*, dan lainnya yang telah dibahas pada sub-bab sebelumnya.

2.2.7. Augmentasi Data

Augmentasi data adalah teknik yang dapat dilakukan untuk mengatasi dataset yang jumlah per kelasnya tidak sama. Tujuan penggunaan augmentasi data adalah untuk memperbanyak variasi pada dataset, balancing data, menghindari *overfitting*, serta meningkatkan performa. Banyak teknik yang digunakan untuk agumentasi data seperti meningkatkan kecerahan, memotong gambar, membalikkan gambar secara vertikal maupun *horizontal*, *blur*, *zoom*, *translasi*, dan lain sebagainya.

2.2.8. FastAPI

FastAPI adalah *framework* untuk pengembangan *website* yang dapat digunakan pada Python 3.6 maupun versi terbaru. FastAPI dirilis pada tahun 2018 dan tujuan utamanya adalah membangun aplikasi web dengan cepat dan REST API. FastAPI sekarang digunakan di Uber, Microsoft, Explosion AI dan lainnya [25]. Alasan penulis memilih FastAPI adalah sebagai berikut:

- Performa FastAPI sangat baik bahkan menjadi salah satu *framework* python tercepat.
- Dukungan asinkronus yang membuat penulis dapat menggunakan satu *framework* untuk semua *endpoint*.
- Dokumentasi API yang sangat mudah dibuat karena otomatis dari FastAPI tanpa harus membuat secara manual

2.2.9. NuxtJs

NuxtJs adalah *framework* yang digunakan untuk membuat *Universal VueJS Application* [26]. Selain itu NuxtJs juga merupakan salah satu *framework open source* untuk membangun *website Front-End* yang berbasis VueJs. NuxtJs memudahkan penulis dalam membuat tampilan serta koneksi ke sistem *backend*. NuxtJs memiliki beberapa kelebihan yang menjadi dasar oleh penulis untuk menggunakan *framework* ini untuk penelitian, antara lain:

- Dipercaya memiliki performa lebih baik daripada *framework* SSR lainnya.

- Penggunaan yang mudah, memudahkan penulis untuk mempelajari dan membangun PWA serta dokumentasi yang tertata rapi.
- NuxtJs telah menyediakan *framework* CSS yang membantu dalam pembuatan tampilan.
- Terdapat Nuxt-PWA membantu dalam pengembangan sistem PWA.

2.2.10. Progressive Web App (PWA)

Progressive Web App atau PWA adalah sebuah *website* yang dapat berperilaku layaknya aplikasi *mobile*. PWA mengambil keuntungan dari aplikasi *mobile* tanpa harus mengunduh PWA melalui *app store*. Google menggunakan istilah FIRE (*Fast, Integrated, Reliable, dan Engaging*) pada PWA yang artinya cepat, terintegrasi, dapat diandalkan dan menarik [27]. Google mengatakan seperti tersebut dikarenakan PWA memiliki beberapa kelebihan, yaitu:

- Penghematan biaya produksi karena dengan hanya membuat sistem PWA, dapat berjalan di pada website maupun aplikasi *mobile*.
- Dapat digunakan pada berbagai perangkat, Android, iOS, dan Windows.
- Tidak perlu berurusan dengan *App Store* yang membutuhkan biaya registrasi (25 dollar) dan langganan (99 dollar per tahun).
- Performa yang lebih cepat, bahkan Uber mengatakan PWA adalah alasan sistem mereka dapat berjalan di jaringan 2G.
- *Update* otomatis akan sangat membantu pengguna yang jarang membuka *app store*.
- Ramah storage dan juga bandwidth.

2.2.11. Karakteristik Penyakit Tanaman Padi

Padi (*Oryza Sativa*) merupakan tanaman budidaya yang sangat penting bagi manusia di seluruh dunia terutama di negara Asia. Bagi Indonesia sendiri, padi yang diolah menjadi nasi, merupakan makanan pokok sebagai sumber energi. Tidak hanya itu padi juga menjadi komoditas pangan utama di Indonesia [28] dan juga beberapa negara Asia lainnya. Padi sendiri ditanam di area sawah dan memiliki karakteristik tersendiri dalam cara menanamnya.

Padi memiliki karakteristik anatomi yaitu akar, batang, daun, bunga, dan buah. Akar tanaman padi berfungsi dalam penyerapan nutrisi yang akan disalurkan ke bagian padi lainnya melalui batang padi. Batang padi sendiri dapat tumbuh sekitar 107-115 cm. Untuk daunnya, padi memiliki daun bersisik yang terbagi menjadi beberapa bagian yaitu helaihan daun, pelepas daun, dan lidah daun. Normalnya daun padi berwarna hijau dengan posisi tegak. Padi juga mempunyai bunga dan buah, yang nantinya buah tersebut akan tertutup lemma dan pelea saat mengalami penyerbukan. Saat dewasa, pelea dan lemma akan membuka serta membentuk sudut yang nantinya akan dipanen dan dikonsumsi sebagai makanan pokok [28].

Namun seperti pada tanaman lainnya, padi juga memiliki berbagai jenis penyakit. Berikut penjelasan mengenai penyakit pada tanaman padi berdasarkan *dataset* yang penulis gunakan untuk penelitian. Informasi mengenai penyakit tanaman padi diambil pada beberapa *website* yang tersedia secara publik [5][29][30][31][32].

Table 2. Karakteristik Penyakit Tanaman Padi

Penyakit	Gejala	Penyebab
Bacterial Leaf Blight [5]	 <p>Pada bibit, daun yang terinfeksi berubah menjadi hijau keabuan dan menggulung. Saat penyakit mulai berkembang, daun menguning dan layu</p>	Adanya gulma yang terinfeksi. Lingkungan tropis dan subtropis terutama di sawah yang kurang diairi air. Penyakit ini berkembang di suhu 25-34 C

Bacterial Leaf Streak [5]		Daun memiliki garis-garis coklat tetapi tipis. Jika sudah parah, seluruh daun bisa berwarna coklat	Muncul di area dengan temperatur dan kelembapan tinggi. Biasanya muncul di daerah tropis dan subtropis
Bacterial Panicle Blight [32]		Malai mengalami pembusukan, perubahan warna coklat muda pada sepertiga bagian bawah hingga setengah malai.	Karena panas, cuaca kering yang membantu bakteri berkembang
Blast [5]		Daun memiliki bercak elips dan bagian tengah berwarna keputihan hingga abu-abu dengan batas merah hingga kecoklatan	Terjadi di daerah dengan tanah kurang lembap, curah hujan tinggi, suhu dingin di siang hari. Embun juga mendukung perkembangan penyakit

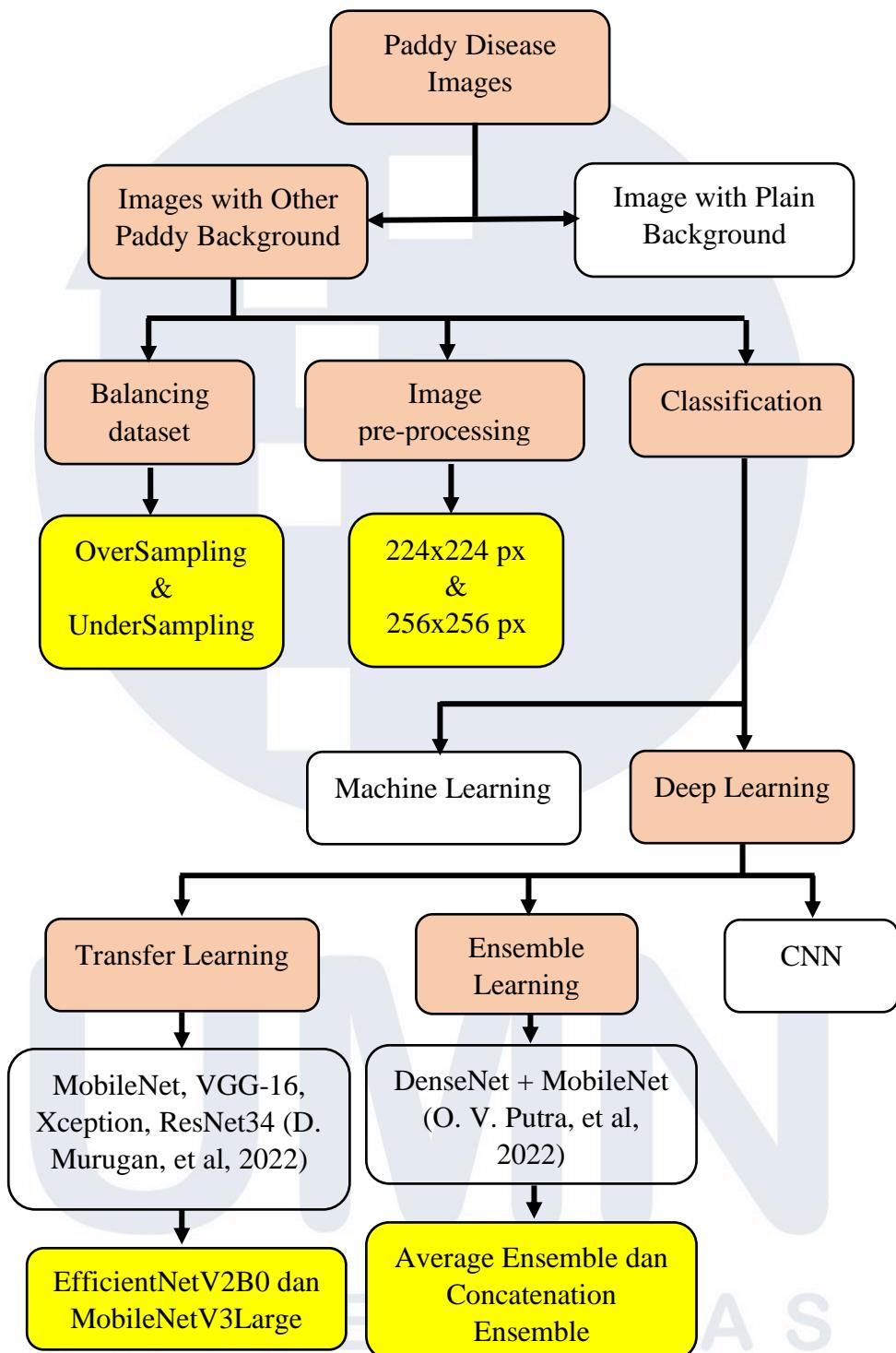
Brown Spot [5] 	Daun memiliki bercak lingkaran hingga lonjong dengan bagian tengah berwarna coklat muda hingga abu-abu, dikelilingi tepi berwarna coklat kemerahan	Muncul di area dengan kelembapan (86% - 100%) dan suhu (16 – 36 C). Tanah yang tidak tergenang air dan kekurangan nutrisi
Dead Heart [31] 	Malai berwarna keputihan dan tidak berisi. Lubang kecil pada batang	Larva hama pelubang batang padi
Downy Mildew [29] 	Bintik kuning pada lapisan daun atas yang menyebar ke beberapa titik pada daun	-
Hispa [30] 	Pengikisan permukaan atas helai daun menghasilkan guratan putih. Daun keputihan dan berselaput	Adanya hama hispa, hujan lebat, curah hujan rendah, perbedaan suhu malam siang kecil, menguntungkan hama

Tungro [5]		Daun berubah warna kuning / oranye dimulai dari ujung daun sampai bawah daun. Juga dapat menunjukkan bintik-bintik berwarna karat. Tanaman dapat menunjukkan gejala kerdiril dan pembungaan tertunda.	Menular dari tanaman lain saat proses pemindahan bibit, persemaian, pembajakan dan proses lainnya yang kurang tepat
------------	---	---	---

2.3 Summary

Berdasarkan penelitian terdahulu dan tinjauan teori yang penulis analisa, maka rancangan penelitian yang akan penulis lakukan adalah sebagai berikut:

- Dataset akan menggunakan dataset publik dari Paddy Doctor [9]. Dataset tersebut bersifat *imbalanced*, sehingga penulis akan menggunakan teknik undersampling dan oversampling sebagai tahap *data balancing* dataset.
- Sistem akan menggunakan teknik *deep learning* yang akan menggunakan *pre-trained model* dan *ensemble learning*. Penggunaan *pre-trained model* terbukti mampu memberikan performa yang baik dan tidak perlu merancang sistem model dari awal. Penggunaan *ensemble learning* juga terbukti mampu memberikan performa lebih baik dari *pre-trained model* dengan cara menggabungkan dua atau lebih model.
- Analisa performa setiap model akan menggunakan bantuan grafik akurasi dan *loss*, *classification report*, dan *confusion matrix*.
- Model yang telah dilatih akan diimplementasi pada FastAPI yang merupakan satu kesatuan sistem PWA yang akan penulis buat dengan pemilihan NuxtJS sebagai *user interface* dan konfigurasi PWA.
- Analisa rata-rata waktu prediksi model pada sistem PWA akan dibantu oleh *network waterfall* serta menggunakan analisa korelasi antara rata-rata waktu prediksi model dengan ukuran *file h5* model.



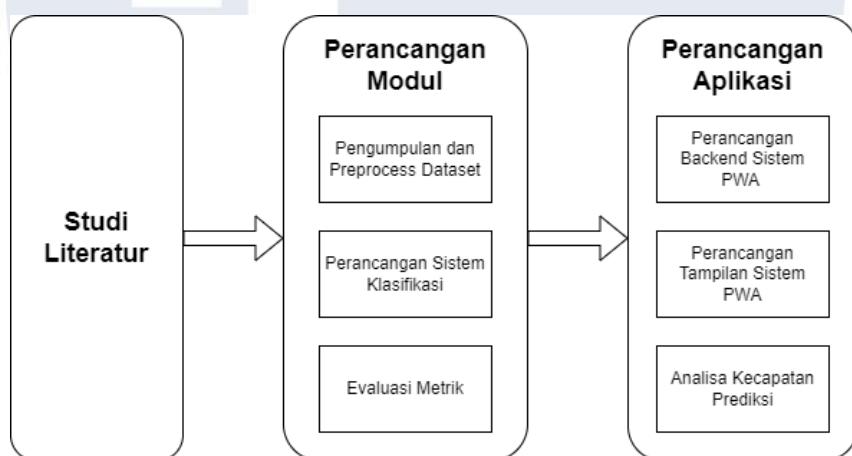
Gambar 2.4 Diagram state of the art

BAB III

ANALISIS DAN PERANCANGAN SISTEM

3.1 Metode Penelitian

Pada penelitian yang dilakukan oleh penulis terdapat beberapa tahapan untuk meneliti serta merancang sistem dengan tujuan menjawab identifikasi masalah yang telah penulis buat. Metode penelitian yang digunakan penulis terdiri dari studi literatur, perancangan modul, dan perancangan aplikasi. Berikut Gambar 3.1, metode atau alur penelitian yang penulis lakukan.



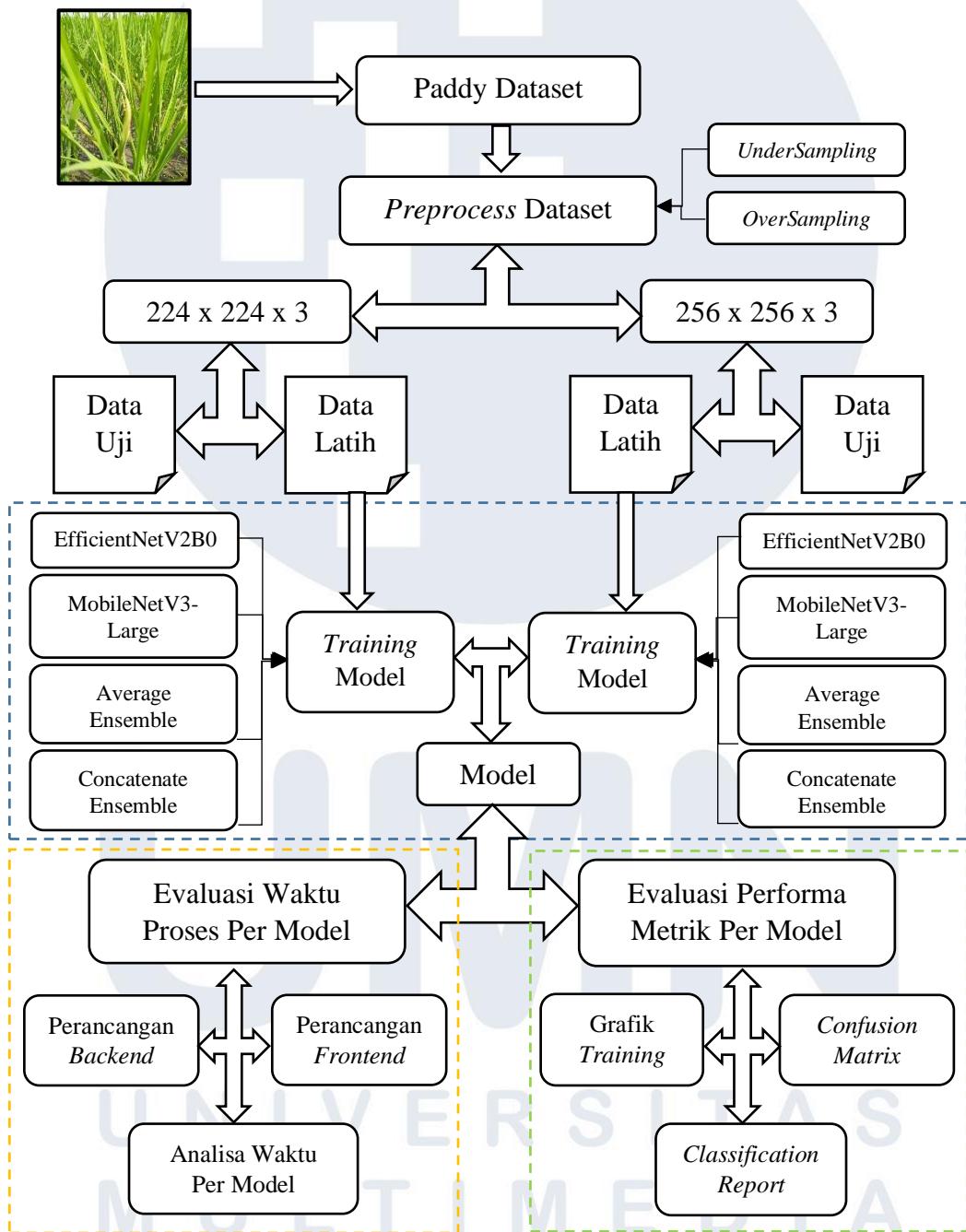
Gambar 3.1. Metode atau Alur Penelitian

3.2. Studi Literatur

Dalam melakukan studi literatur, penulis mempelajari penelitian terdahulu terkait klasifikasi penyakit tanaman padi, *deep learning*, *ensemble learning*, maupun pengimplementasian model ke dalam sistem PWA. Dalam mempelajari berbagai penelitian terdahulu, penulis dapat menentukan metode dan model yang akan digunakan untuk klasifikasi penyakit tanaman padi. Penulis juga berdiskusi bersama salah satu dosen Program Studi Teknik Komputer terkait penelitian yang akan dikerjakan penulis. Penulis juga membaca berbagai *website* mengenai cara penggunaan *deep learning* dan *ensemble learning* pada *dataset* gambar dan juga mengenai cara pengimplementasian model yang telah dilatih ke dalam sistem PWA. Penulis juga mempelajari karakteristik pada penyakit tanaman padi yang nantinya dapat digunakan sebagai fondasi dasar penulis dalam meneliti.

3.3. Perancangan Modul

Perancangan modul meliputi proses penulis memperoleh *dataset*, mengolah dataset tersebut, melakukan perancangan sistem klasifikasi, dan dilanjutkan dengan evaluasi metrik. Berikut alur perancangan modul dapat terlihat pada Gambar 3.2.



Gambar 3.2 Alur Perancangan Modul

3.3.1 Pengumpulan dan Preprocess Dataset

Pengumpulan data pada penelitian ini menggunakan data yang tersedia secara publik yaitu Paddy Doctor Dataset. Dataset diambil pada sawah dekat distrik Tirunelveli, India. Pengumpulan data diambil pada bulan Februari sampai bulan April 2021 dengan umur padi diantara 40 sampai 80 hari. Dataset diambil menggunakan kamera ponsel CAT 562 Pro untuk menangkap gambar dengan resolusi RGB yang tinggi.

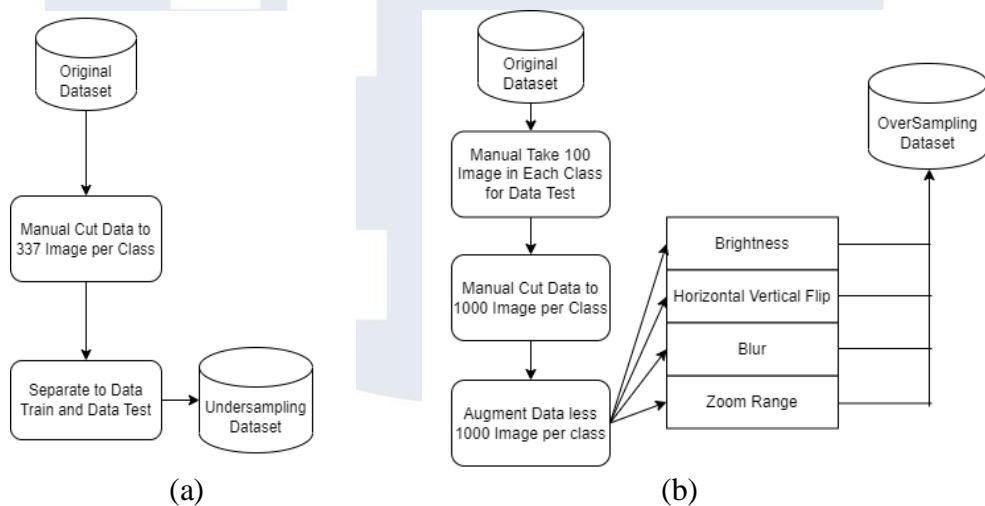
Berdasarkan dataset yang didapat penulis melalui *website* Kaggle, Gambar yang didapat 13,878 gambar yang telah dibagi menjadi dua bagian, 10,407 gambar untuk data latih dan 3,469 untuk data uji. Data latih telah dibagi menjadi 10 kelas yaitu 9 kelas penyakit tanaman padi dan 1 kelas tanaman padi normal. Kelas penyakit tanaman padi terdiri dari Bacterial Leaf Blight, Bacterial Leaf Streak, Bacterial Panicle Blight, Blast, Brown Spot, Dead Heart, Downy Mildew, Hispa, dan Tungro.

Pada penelitian ini, peneliti hanya menggunakan data latih yang telah diperoleh. Data uji tidak digunakan oleh penulis dikarenakan data belum terlabel. Jumlah kelas pada setiap data latih memiliki jumlah yang berbeda-beda. Berikut rincian detail jumlah data per kelas penyakit tanaman padi.

Tabel 3. Dataset Kelas Tanaman Padi - Imbalanced

Dataset Kelas Tanaman Padi	Jumlah
Bacterial Leaf Blight	479
Bacterial Leaf Streak	380
Bacterial Panicle Blight	337
Blast	1738
Brown Spot	965
Dead Heart	1442
Downy Mildew	620
Hispa	1594
Tungro	1088
Normal	1764
Total	10407

Data latih yang didapat dibagi menjadi dua dataset dengan dua teknik *balancing dataset*, yaitu *undersampling dataset* dan *oversampling dataset*. Perbedaan kedua dataset ini adalah jumlah gambar yang akan diuji. Untuk dataset dengan teknik undersampling memiliki total gambar yaitu 3370 gambar. Sedangkan untuk dataset dengan teknik oversampling, memiliki total 11000 gambar. Alasan penulis menggunakan dua teknik *balancing* yang berbeda adalah ingin melihat apakah terdapat pengaruh yang selaras antara biaya komputasi dan waktu pelatihan terhadap performa yang didapat. Berikut Gambar 3.3, pengolahan dataset undersampling dan dataset oversampling.



Gambar 3.3 Pengolahan Dataset; (a) UnderSampling (b) OverSampling

Untuk dataset teknik undersampling didapat dengan cara melakukan pemotongan jumlah data per kelas secara manual untuk data yang lebih dari 337 gambar. Sehingga semua kelas pada dataset teknik undersampling akan memiliki 337 gambar per kelas. Alasan penulis memotong jumlah gambar per kelas menjadi 337 gambar adalah penulis mengambil jumlah gambar terendah pada kelas penyakit yaitu Bacterial Panicle Blight yang hanya memiliki 337 gambar. Dataset ini nantinya akan dibagi menjadi data latih (3030 gambar), data validasi (303 gambar), dan data uji (340 gambar).

Untuk dataset teknik oversampling, penulis mengambil terlebih dahulu 100 gambar untuk setiap kelas dataset *original* dan memasukkannya ke dalam data uji. Sehingga terdapat 1000 gambar data uji yang telah dipisah penulis sebelum

melakukan augmentasi. Selanjutnya penulis akan melakukan pemotongan data secara manual untuk kelas yang memiliki gambar lebih dari 1000 gambar. Untuk data yang kurang dari 1000 gambar akan melalui proses augmentasi. Sehingga akan menghasilkan 1000 gambar per kelas dengan total gambar 10.000 gambar. Dataset ini nantinya akan dibagi menjadi data latih (9000 gambar) dan data validasi (1000 gambar).

Augmentasi data yang dilakukan adalah pengubah kecerahan, membalik gambar (vertikal dan horizontal), pengaburan gambar, dan pengubah rentang zoom. Alasan penggunaan augmentasi tersebut berdasarkan penelitian yang dilakukan Ryukin Aranta Lika [12] dengan kesimpulan pengubah kecerahan dan pengaburan gambar memberikan akurasi yang paling baik. Selain itu, penulis juga mengkondisikan gambar yang akan diambil petani yang dapat berbeda tingkat kecerahan, pengaburan (kualitas kamera ponsel), kedekatan dan kejauhan pengambilan gambar (*zoom range*). Berikut detail implementasi augmentasi pada dataset dengan teknik oversampling.

- Pembalikan gambar baik horizontal maupun vertikal dilakukan dengan besar 180 derajat.
- Pengubahan kecerahan dilakukan secara acak dengan jangkauan minimum yaitu 0.3 dan jangkauan maksimum yaitu 1.
- Pengaburan gambar dilakukan dengan bantuan *library* cv2.blur() dengan *size kernel* bersifat acak. Jangkauan minimum yaitu 1 dan jangkauan maksimum yaitu 3.
- Pengubah rentang *zoom* dilakukan secara acak dengan jangkauan minimum yaitu 0.5 dan jangkauan maksimum yaitu 1.

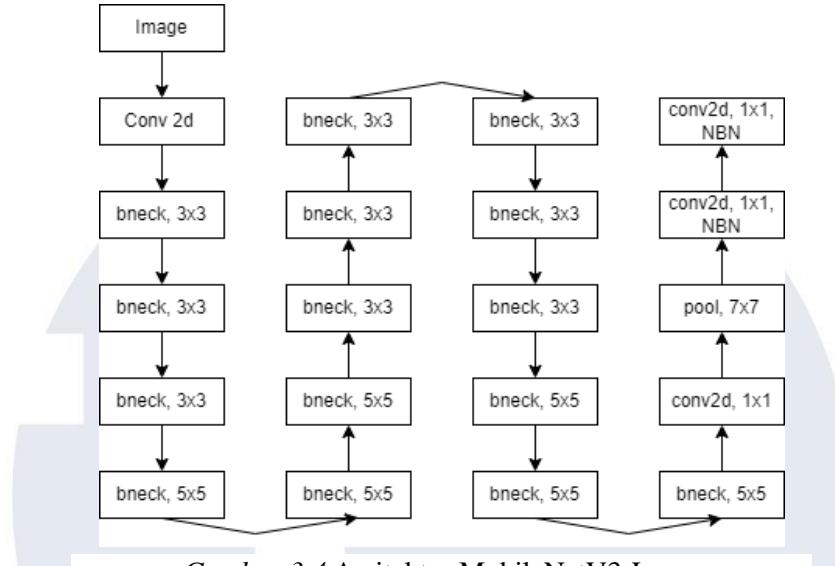
Tahap *pre-process* lainnya terletak pada penyesuaian ukuran *pixel* pada gambar. Mulanya ukuran gambar yaitu 480x640 *pixels* akan diubah menjadi 224x224 *pixel*. Perubahan ukuran *pixels* didasari pada ukuran gambar default terbaik untuk model EfficientNetV2B0 dan MobileNetV3-Large yaitu 224x224 *pixel*. Penelitian ini juga mencoba menggunakan ukuran gambar 256x256 *pixel* yang didasarkan atas penelitian yang dilakukan Vajira Tahmbawita et al. [15] yang mana peningkatan ukuran *pixel* selaras dengan peningkatan performa

model dan juga berdasarkan penelitian terdahulu yaitu Paddy Doctor yang menggunakan ukuran gambar 256x256 *pixel* [6]. Sehingga penulis dapat melihat dan membandingkan performa model antara ukuran gambar 224x224 *pixel* dan 256x256 *pixel*.

3.3.2 Perancangan Sistem Klasifikasi

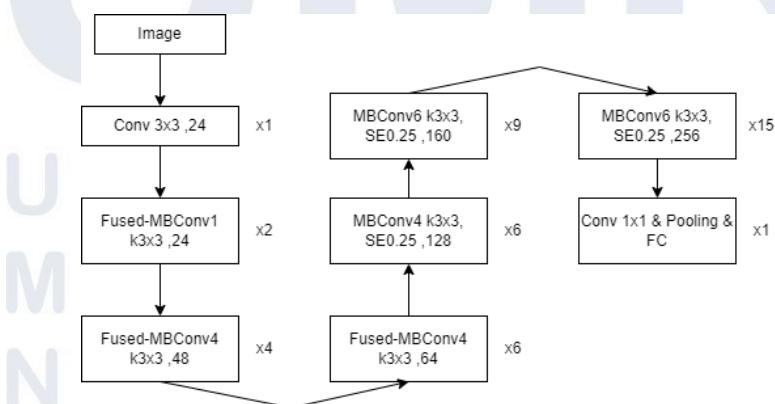
Perancangan sistem klasifikasi pada penelitian ini dilakukan dengan beberapa model *state-of-the-art* yang telah dipilih penulis untuk mengklasifikasi penyakit pada tanaman padi. Penelitian ini menggunakan dua model *pre-trained* yaitu MobileNetV3-Large dan EfficientNetV2B0 serta menggunakan dua metode *ensemble* yaitu Average Ensemble dan Concatenation Ensemble. Model yang akan digabungkan adalah MobileNetV3-Large dan EfficientNetV2B0.

Model MobileNetV3-Large dipilih penulis berdasarkan jurnal penelitian terdahulu [6] [8], memberikan performa yang cukup baik saat menggunakan model MobileNet. Penulis ingin mencoba untuk menaikkan performa dari model MobileNet dan memilih MobileNetV3-Large. Berdasarkan *website* keras [36], MobileNetV3-Large memiliki performa yang lebih baik dibandingkan MobileNet. Perbandingan parameter antar keduanya juga tidak berbeda jauh. Selain itu, penulis memilih MobileNetV3-Large daripada MobileNetV3-Small didasarkan pada tingkat akurasi yang didapat lebih baik dengan perbedaan parameter yang tidak terlalu signifikan. Jika dibandingkan dengan MobileNet versi sebelumnya, penggunaan struktur blok baru yaitu “*bottleneck residual block*” pada MobileNetV3-Large mampu mempertahankan representasi yang lebih kuat dan mengurangi kehilangan informasi dalam proses konvolusi. Selain itu terdapat juga pengoptimalan dengan menggunakan fungsi aktivasi *swish* yang lebih daripada penggunaan fungsi aktivasi ReLU. Swish dapat mengurangi efek kematian neuron yang cukup sering terjadi pada ReLU. Penggunaan konvolusi *depthwise* dengan faktor *stride* yang lebih besar mempercepat komputasi dengan tetap mempertahankan informasi yang penting. Arsitektur model MobileNetV3-Large dapat terlihat pada Gambar 3.4.



Gambar 3.4 Arsitektur MobileNetV3-Large

Model EfficientNetV2B0 dipilih penulis berdasarkan jurnal penelitian terdahulu [13] yang menggunakan EfficientNetB3 yang merupakan akurasi tertinggi pada penelitian tersebut yaitu 79.53% untuk klasifikasi penyakit tanaman padi. Akurasi tersebut tentu dapat ditingkatkan lagi dengan menggunakan EfficientNetV2B0 yang merupakan versi terbaru dari EfficientNet. Jika dibandingkan dengan EfficientNet versi sebelumnya, EfficientNetV2 melakukan pengurangan blok MBConv dan perubahan struktur residual pada blok MBConv yang dapat membantu meningkatkan performa dari model. Untuk pemilihan B0 juga didasarkan pada ringannya model dibandingkan dengan B1, B2, dan seterusnya dengan perbedaan akurasi yang tidak terlalu signifikan. Arsitektur model EfficientNetV2B0 dapat terlihat pada Gambar 3.5.

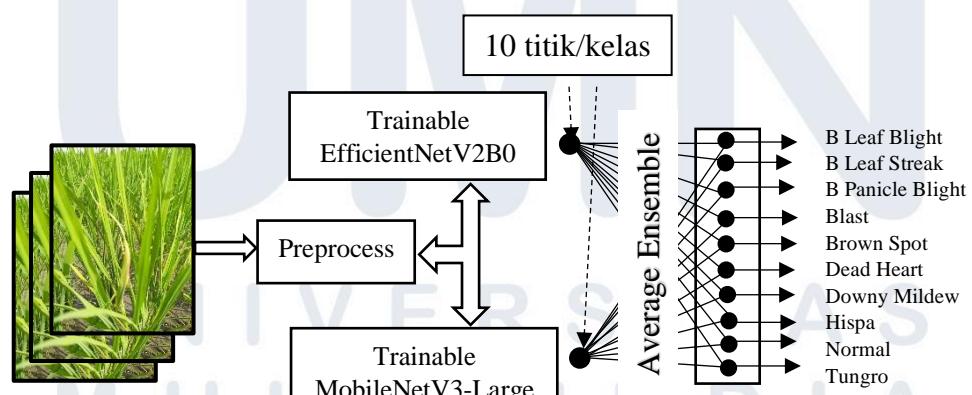


Gambar 3.5 Arsitektur EfficientNetV2B0

Average ensemble dipilih oleh penulis berdasarkan jurnal penelitian yang ditulis oleh Talib Iqbali dan M. Arif Wanim [33], membuktikan bahwa hasil yang didapat lebih baik daripada *pre-trained* model yang digunakan. *Average ensemble* memiliki konsep sederhana yang mengambil rata-rata hasil prediksi setiap model yang telah dilatih sebelumnya yang menghasilkan output prediksi ensemble yang dapat dilatih kembali. *Average ensemble* tidak memerlukan konfigurasi parameter yang rumit karena telah disediakan oleh *library* keras yaitu fungsi *average*, serta kompatibilitas yang baik untuk berbagai model seperti model EfficientNetV2B0 dan MobileNetV3-Large yang memiliki arsitektur berbeda, tetapi bisa digabungkan menggunakan metode *average ensemble*. Cara penulis menggunakan *average ensemble*, sebagai berikut:

- Menggunakan EfficientNetV2B0 dan MobileNetV3-Large sebagai *base model* untuk *average ensemble*.
- Model tersebut dilatih secara individu
- Memakai fungsi *average* pada *library* keras untuk menggabungkan *output* dari kedua model tersebut
- *Training ulang* model yang telah diproses

Skema penggunaan *average ensemble* pada penelitian ini dapat terlihat pada Gambar 3.6.

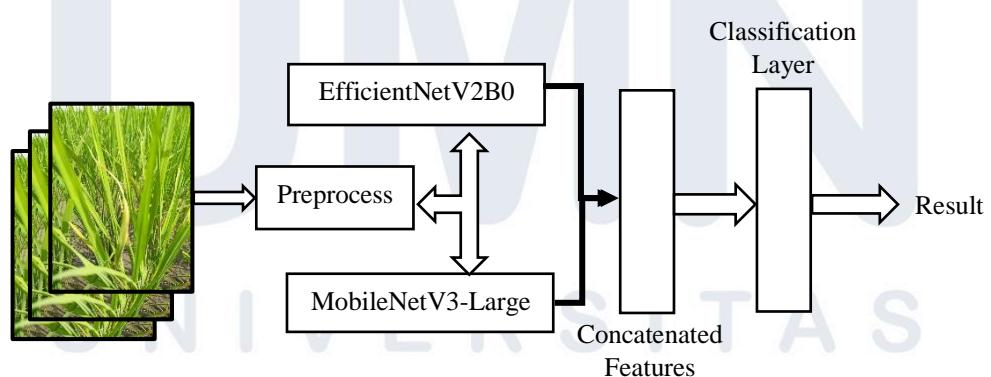


Gambar 3.6 Skema Penggunaan *Average Ensemble*

Concatenation ensemble dipilih oleh penulis berdasarkan jurnal penelitian yang ditulis oleh Oussama El Gannou, dkk [34], *concatenation ensemble* memberikan performa yang sangat bagus dan hampir melebihi semua model yang dicoba pada penelitian tersebut. *Concatenation ensemble* juga memiliki konsep yang sederhana, dengan menggabungkan setiap fitur dari model menjadi satu fitur besar yang nantinya akan digunakan pada layer selanjutnya yaitu layer klasifikasi. *Concatenation ensemble* juga tidak memerlukan konfigurasi parameter dan cukup menggunakan library yang telah disediakan oleh keras yaitu fungsi *concatenate*, serta memiliki kompatibilitas yang baik dengan berbagai model. Cara penulis menggunakan *concatenation ensemble*, sebagai berikut:

- Menggunakan EfficientNetV2B0 dan MobileNetV3-Large sebagai *base model* untuk *concatenate ensemble*
- Model tersebut langsung digabung menggunakan fungsi *concatenate*
- Penambahan beberapa layer untuk proses klasifikasi
- *Training* model yang telah diproses

Skema penggunaan *concatenation ensemble* pada penelitian ini dapat terlihat pada Gambar 3.7.



Gambar 3.7 Skema Penggunaan *Concatenation Ensemble*

Penggunaan dua *pre-trained* model dan 2 metode *ensemble* bertujuan untuk membandingkan performa antar model dan metode tersebut yang juga menggunakan beberapa parameter pengujian berbeda, seperti teknik *balancing*

dataset yang berbeda dan ukuran gambar berbeda. Untuk *pre-trained* model, alasan penggunaan MobileNetV3-Large adalah MobileNetV3-Large memiliki parameter yang mirip dengan EfficientNetV2B0. Untuk MobileNetV3-Large memiliki 5.4M parameter sedangkan EfficientNetV2B0 memiliki 7.2M parameter. Sehingga dapat menjadi bahan pembanding untuk akurasi serta kecepatan yang didapatkan. Selain itu, penggunaan kedua model tersebut dikarenakan menimbang banyaknya gambar yang dilatih serta ukuran dengan hardware yang disediakan oleh Google Colab Pro. Untuk *ensemble model*, alasan penggunaan teknik *ensemble* adalah *average ensemble* dan *concatenation ensemble* merupakan dua teknik *ensemble* yang sangat simpel dan mudah untuk diimplementasi. Performa yang diberikan dari teknik *ensemble* juga terbukti memiliki performa yang bagus berdasarkan jurnal penelitian terdahulu [33] [34]. Sehingga dapat menjadi bahan pembanding baik untuk model *ensemble* sendiri maupun dua *pre-trained* model.

Penyesuaian layer pada masing-masing model akan disamakan. Penyesuaian layer dibuat dikarenakan penelitian ini akan mengklasifikasi penyakit pada tanaman padi. Sehingga diperlukan *Fully Connected Layer* yang digunakan untuk proses klasifikasi. Pada penyesuaian layer, penulis menambahkan empat layer tambahan yang dapat terlihat pada Gambar 3.8.

```
#flatten layer
top_model = Flatten(name="flatten")(top_model)
#dense layer
top_model = Dense(1024, activation='relu')(top_model)
#dropout layer
top_model = Dropout(0.2)(top_model)
#dense layer
output_layer = Dense(classData, activation='softmax')(top_model)
```

Gambar 3.8 Penyesuaian Layer

Berikut penjelasan masing-masing layer yang ditambahkan oleh penulis untuk proses klasifikasi penyakit pada tanaman padi.

- *Flatten layer* digunakan untuk membuat input hasil ekstraksi fitur pada model pretrained yang memiliki banyak dimensi menjadi satu dimensi. Tentunya ini digunakan sebelum memasuki *fully connected*.
- *Dense layer* (1024, relu) digunakan untuk melakukan operasi *fully connected layer* yang menjadi inti dari proses klasifikasi. Layer ini akan memiliki 1024 neuron. Neuron tersebut akan menggunakan fungsi relu untuk menyelesaikan masalah non-linier pada hasil yang didapat pada setiap neuron.
- *Dropout layer* (0.2) digunakan untuk mencegah *overfitting* sehingga mengurangi koneksi dari neuron. Layer ini akan memilih beberapa neuron secara acak untuk dibuang atau tidak digunakan.
- *Dense layer* (10, softmax) digunakan untuk mengeluarkan *output* hasil klasifikasi menjadi 10 kelas dengan bantuan fungsi *softmax*. Fungsi *softmax* akan menyediakan probabilitas untuk setiap 10 kelas *output*.

Penggunaan *hyperparameter* pada masing-masing model juga akan disamakan. Berikut Tabel 3, *hyperparameter* yang digunakan serta fungsi parameter tersebut pada penelitian ini.

Table 3. Penggunaan Hyperparameter

Hyperparameter	Value	Fungsi & Alasan
Optimizer	Adam	Untuk menemukan nilai minimum lokal dari sebuah fungsi. Adam adalah kombinasi antara RMSprop dan Stochastic Gradient Descent dengan momentum. Pemilihan <i>optimizer</i> Adam juga dilihat dari performa yang dihasilkan dari penelitian sebelumnya.
Learning Rate	0.0001	Untuk menghitung nilai koreksi bobot pada waktu proses <i>training</i> . Semakin besar nilai <i>learning rate</i> , proses <i>training</i> akan semakin cepat tetapi

		keterlitian berkurang dan begitu sebaliknya. Penggunaan nilai 0.0001 pada penelitian ini untuk mendapatkan keterlitian yang lebih baik.
Loss	Categorical Crossentropy	Untuk menghitung perbedaan antara keluaran yang dihasilkan dan keluaran yang diharapkan. <i>Loss</i> ini banyak digunakan pada tugas klasifikasi terutama <i>categorical</i> untuk tugas <i>multiclass</i> dan membutuhkan label untuk dikodekan sebagai kategori. Hal ini bersesuaian dengan penelitian ini, dimana terdapat 10 kelas yang telah diberi label.
Batch_Size	32	Jumlah sampel data yang melewati jaringan saraf pada satu waktu. Menambah <i>batch size</i> akan menurunkan performa begitu pula sebaliknya.
Epoch	10	Untuk membuat seluruh dataset melalui proses <i>training</i> pada jaringan saraf dalam sekali putaran. Penggunaan 10 <i>epoch</i> dikarenakan percobaan yang dilakukan penulis, hanya menggunakan 3 - 4 <i>epoch</i> saja sudah mendapatkan akurasi yang bagus. Sehingga penulis menetapkan 10 <i>epoch</i> pada setiap model.

Training yang dilakukan akan mengimplementasikan *custom callback*. *Custom callback* yang penulis gunakan adalah *EarlyStopping*. Tujuan

penggunaan *custom callback* agar hasil training agar mencegah terjadinya *overfitting*. Berkut Tabel 4, parameter yang digunakan pada *custom callback* *earlystopping* beserta fungsinya.

Table 4. Penggunaan Parameter *EarlyStopping*

Parameter	Value	Fungsi
Monitor	Val_Loss	Memonitor <i>validation loss</i> pada saat <i>training</i>
Mode	Min	Menghentikan training ketika <i>validation loss</i> berhenti berkurang
Patience	3	Akan memberhentikan <i>training</i> jika <i>validation loss</i> tidak ada perkembangan atau berhenti berkurang dan menunggu sebanyak 3 <i>epoch</i> .

3.3.3 Evaluasi Metrik pada Model

Evaluasi metrik pada model memiliki peran penting dalam menganalisa performa model yang digunakan, baik untuk mengukur kinerja, membandingkan antara model satu dengan model yang lain, mengoptimalkan model, dan membantu penulis dalam mengambil kesimpulan uji model. Penting untuk memilih evaluasi metrik yang sesuai dengan penelitian yang dilakukan. Berikut evaluasi metrik yang penulis gunakan dalam penelitian ini.

- Penggunaan grafik pembelajaran atau performa untuk melihat kinerja model baik dari sisi kinerja pelatihan maupun validasi. Sumbu x akan menampilkan jumlah *epoch* yang dipakai saat proses pelatihan. Sumbu y akan menampilkan hasil akurasi dan *loss* dari hasil pelatihan yang dilakukan. Tujuan penggunaan grafik ini untuk melihat stabilitas dan generalisasi model.
- *Classification Report* merupakan satu set evaluasi metrik yang berisi akurasi, presisi, recall, F-1 *score*, dan *support*. Akurasi memiliki tujuan untuk melihat seberapa akurat model yang telah kita latih dalam mengklasifikasikan penyakit tanaman padi atau dapat dikatakan jumlah

prediksi yang benar dibandingkan dengan jumlah total sampel. Presisi menggambarkan akurasi antara data yang diminta dengan hasil prediksi yang diberikan oleh model. *Recall* bertujuan untuk menggambarkan keberhasilan model dalam menemukan kembali sebuah informasi. F-1 score bertujuan untuk menggambarkan perbandingan rata-rata presisi dan *recall*. Berikut rumus dari akurasi, presisi, *recall*, dan F-1 score.

$$\text{Akurasi} = (\text{TP} + \text{TN}) / (\text{TP} + \text{FP} + \text{FN} + \text{TN})$$

$$\text{Presisi} = (\text{TP}) / (\text{TP} + \text{FP})$$

$$\text{Recall} = \text{TP} / (\text{TP} + \text{FN})$$

$$\text{F-1 Score} = (2 * \text{Recall} * \text{Precision}) / (\text{Recall} + \text{Precision})$$

- *Confusion Matrix* atau matrik kebingungan merupakan evaluasi metrik yang digunakan untuk menggambarkan performa model pada dataset uji dengan melihat jumlah sampel yang benar dan salah. *Confusion matrix* memiliki empat kategori yang dapat terjadi yaitu *True Positive* (TP), *False Positive* (FP), *True Negative* (TN), dan *False Negative* (FN). Sebenarnya empat kategori tersebut yang nantinya akan digunakan untuk mendapatkan *classification report*. Implementasi *confusion matrix* akan menggunakan matrik *heat map*.

Ketiga evaluasi metrik tersebut akan digunakan penulis untuk membantu proses analisa terhadap hasil penelitian yang dilakukan.

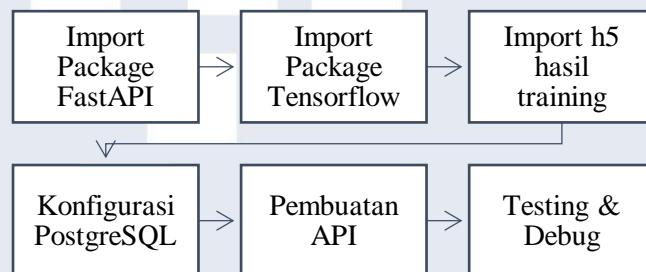
3.4. Perancangan Aplikasi

Perancangan aplikasi berbasis sistem PWA untuk klasifikasi penyakit tanaman padi terdiri dari beberapa proses, yaitu perancangan backend sistem PWA, perancangan tampilan sistem PWA, proses *deploy* aplikasi, dan analisa kecepatan rata-rata prediksi.

3.4.1 Perancangan Backend Sistem PWA

Perancangan sistem *backend* menggunakan *framework* FastAPI dengan tujuan memproses permintaan dari tampilan sistem PWA dan menjalankan prediksi model. Pemilihan FastAPI dikarenakan FastAPI memiliki performa yang lebih baik daripada Flask dan lebih ringan dari Django. Penulis akan menyiapkan direktori baru untuk pembuatan proyek *backend* serta mengimport

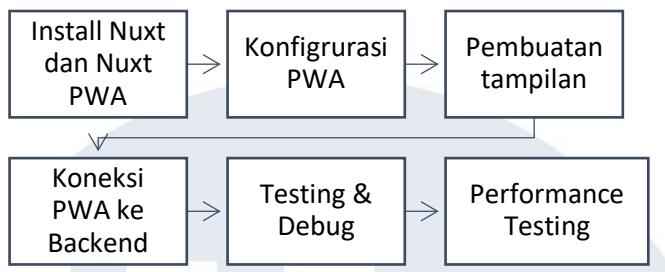
library yang dibutuhkann seperti fastapi, uvicorn, dan lainnya. Penulis juga menggunakan library tensorflow yang digunakan untuk menjalankan hasil model latih untuk mengklasifikasi gambar yang dikirim oleh tampilan sistem PWA. Selain itu penuli sjuga menggunakan PostgreSQL sebagai database untuk menyimpan data penyakit padi dan saran yang dikirimkan dari tampilan sistem PWA. Testing akan dengan bantuan *swagger* dan *postman* untuk memastikan API berhasil dijalankan. Berikut merupakan diagram alur perancangan sistem *backend* yang dapat terlihat pada Gambar 3.6.



Gambar 3.9 Alur Perancangan Sistem *Backend*

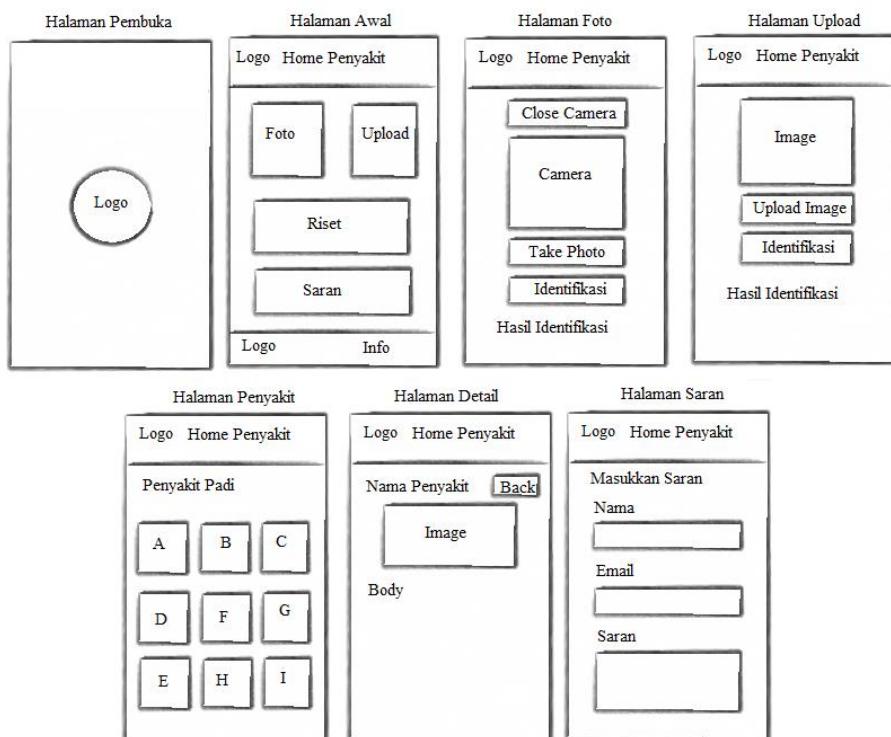
3.4.2 Perancangan Tampilan Sistem PWA

Perancangan tampilan sistem PWA menggunakan *framework* NuxtJs 2 dengan tujuan untuk membuat tampilan dan fungsi yang mampu melakukan *upload* dan foto tanaman padi oleh pengguna. Pemilihan NuxtJs 2 sebagai *framework* tampilan karena NuxtJs 2 dipercaya memiliki performa lebih baik daripada *framework* SSR lainnya dan faktor terpenting adalah NuxtJs memiliki *dependencies* Nuxt-PWA yang membantu dalam pengembangan sistem PWA. Penulis akan melakukan instalasi terhadap NuxtJs 2 dan import Nuxt-PWA module. Setelah itu penulis akan mengkonfigurasi PWA pada file konfigurasi nuxt. Selanjutnya penulis akan membuat tampilan yang memperhatikan aspek kemudahan dalam pengaksesan dengan tampilan yang cukup. Langkah selanjutnya adalah koneksi NuxtJs 2 ke FastAPI yang telah dibuat. Untuk memastikan koneksi antar tampilan dan *backend* berjalan, penulis menguji serta debug sistem dengan bantuan *network waterfall*.



Gambar 3.10 Alur Perancangan Sistem PWA

Untuk perancangan tampilan PWA, penulis akan merancang sesuai *wireframe user interface* yang telah penulis buat seperti pada Gambar



Gambar 3.11 Perancangan Sistem *User Interface* Sistem PWA

Alur penggunaan PWA, dimulai ketika *user* membuka PWA yang akan menampilkan halaman awal. *User* dapat melakukan pengambilan gambar padi melalui foto maupun *upload* gambar. Terlihat pada halaman foto dan *upload* gambar, *user* dapat menekan tombol identifikasi yang akan menghasilkan hasil prediksi di bawah tombol tersebut. Tentunya hanya terdapat nama penyakit dan jika *user* ingin melihat secara detail, dapat menekan tombol detail yang ada dibawah nama penyakit. Tombol tersebut akan mengarahkan ke halaman detail

penyakit padi. Penulis juga membuat halaman saran dengan tujuan agar *user* dapat memberikan saran untuk kemajuan sistem PWA Paddyist.

3.4.3 Proses Deploy Aplikasi

Terdapat dua aplikasi yang dapat *deploy* ke *internet* pada sistem PWA, yaitu NuxtJs 2 dan FastAPI. Untuk NuxtJs2, penulis menggunakan bantuan Netlify karena tempat hosting yang gratis, mudah, dan cepat. Tujuan penulis melakukan *deploy* NuxtJs2 ke Netlify agar dapat membuktikan bahwa sistem PWA yang dibuat oleh penulis dapat diinstalasi ke ponsel dan dapat mengakses kamera melalui ponsel. Sedangkan untuk FastAPI, tidak penulis *deploy* ke Internet. Alasan penulis dikarenakan penelitian yang dilakukan tidak berfokus pada sistem PWA yang sudah siap jadi. Penelitian ini berfokus pada analisa performa akurasi dan kecepatan prediksi setiap model pada sistem PWA. Sehingga untuk analisa kecepatan prediksi dilakukan secara lokal menggunakan laptop penulis.

3.4.4 Analisa Kecepatan Rata-Rata Prediksi

Penelitian ini akan menganalisa kecepatan rata-rata prediksi masing-masing percobaan model terhadap sistem PWA yang telah dibuat. Selain itu, penulis juga akan menganalisa hubungan antara kecepatan rata-rata prediksi dengan ukuran *file* model format h5 dengan menggunakan metode korelasi. Tujuan penulis melakukan analisa tersebut adalah untuk melihat adakah pengaruh antara model, teknik *balancing*, dan ukuran gambar yang berbeda terhadap kecepatan rata-rata prediksi serta korelasi antara kecepatan rata-rata prediksi dan ukuran file model h5.

Untuk pengecekan kecepatan prediksi, penulis akan menganalisis waktu Tensorflow memprediksi gambar ditambah waktu sistem PWA mengirim gambar ke sistem *backend* dan menerima kembali respon. Percobaan dilakukan sebanyak 5 kali untuk masing-masing percobaan model dan menghitung rata-rata dari 5 kali percobaan tersebut. Sehingga akan terlihat model yang memiliki waktu pemrosesan lebih cepat.

Korelasi juga digunakan untuk melihat hubungan antara waktu predksi rata-rata dan ukuran *file* model h5. Korelasi adalah hubungan antara dua variabel yang berbeda yang memiliki rentang nilai -1 sampai +1. Jika nilai yang didapat lebih besar dari 0 maka terdapat hubungan yang positif antara kedua variabel. Jika nilai adalah 0 maka tidak ada hubungan antara dua variabel. Jika nilai lebih kecil dari 0 maka terdapat hubungan yang negatif antara dua variabel. Berikut rumus untuk mencari koefisien korelasi yang akan digunakan pada penelitian ini.

$$r = \frac{n \sum_{i=1}^n X_i Y_i - \sum_{i=1}^n X_i \sum_{i=1}^n Y_i}{\sqrt{n \sum_{i=1}^n X_i^2 - (\sum_{i=1}^n X_i)^2} \sqrt{n \sum_{i=1}^n Y_i^2 - (\sum_{i=1}^n Y_i)^2}}$$



BAB IV

IMPLEMENTASI DAN PENGUJIAN SISTEM

4.1 Spesifikasi Sistem

Untuk melakukan penelitian, penulis menggunakan laptop penulis untuk tugas yang dapat dieksekusi pada laptop penulis dan untuk tugas yang tidak mampu dieksekusi di laptop penulis atau memerlukan *resource* yang besar, akan menggunakan Google Colab Pro.

4.1.1 Spesifikasi Hardware Laptop Penulis

Penulis menggunakan laptop dengan tujuan untuk menghemat *compute unit* serta biaya yang dikeluarkan dalam pemakaian Google Colab Pro. Penggunaan laptop penulis digunakan untuk melatih model EfficientNetV2B0 dan MobileNetV3-Large. Untuk model *ensemble concatenation* dan *average* tidak dapat dilakukan di laptop penulis karena spesifikasi laptop penulis kurang mumpuni. Penulis menggunakan aplikasi jupyter notebook untuk melakukan *training* pada laptop penulis. Penulis juga menggunakan laptop pribadi untuk menganalisa wakta rata-rata prediksi model ketika diimplementasikan ke PWA.

Laptop yang penulis pakai adalah Lenovo Yoga 530-14IKB dengan spesifikasi sebagai berikut:

- CPU: Intel® Core™ i7-8550U CPU @ 1.80GHz 1.99Ghz
- GPU: NVIDIA GeForce MX130
- RAM: 16 GB
- Storage: 512 GB PCIe SSD
- OS: Windows 10 Home

4.1.2 Spesifikasi Hardware Google Colaboratory Pro

Penulis menggunakan Google Colab Pro untuk melakukan penelitian yang tidak sanggup dilakukan di laptop penulis. Pemilihan Google Colab Pro untuk penelitian karena kemudahan dalam mengolah data (Google Drive), tanpa proses instalasi karena berbentuk *cloud*, serta mesin yang ditawarkan cukup

untuk penelitian yang dilakukan penulis. Biaya langganan penggunaan Colab Pro per bulan adalah \$9.99.

Google Colab Pro memiliki spesifikasi sebagai berikut:

- CPU: Intel Xeon Processor@2.3GHz
- GPU: A100 Nvidia. GPU RAM 40GB
- RAM: 83.5 GB
- Disk: 166.8 GB

4.2 Implementasi Sistem

Implementasi sistem terdiri dari pengolahan dataset, proses klasifikasi, dan perancangan sistem PWA yang terdiri dari perancangan *backend* dan pwa

4.2.1 Proses Pengolahan Dataset

Tahap awal adalah mempersiapkan dataset untuk penelitian teknik undersampling dan oversampling. Untuk undersampling, penulis melakukan pemotongan data secara manual untuk data yang lebih dari 337 gambar. Sedangkan untuk oversampling, penulis mengambil 100 gambar per kelas dengan total 1000 gambar untuk data test dan melakukan pemotongan data secara manual untuk data yang lebih dari 1000 gambar. Dikarenakan oversampling dataset belum seimbang, terdapat data kelas penyakit kurang dari 1000 gambar, maka diperlukan proses augmentasi data.

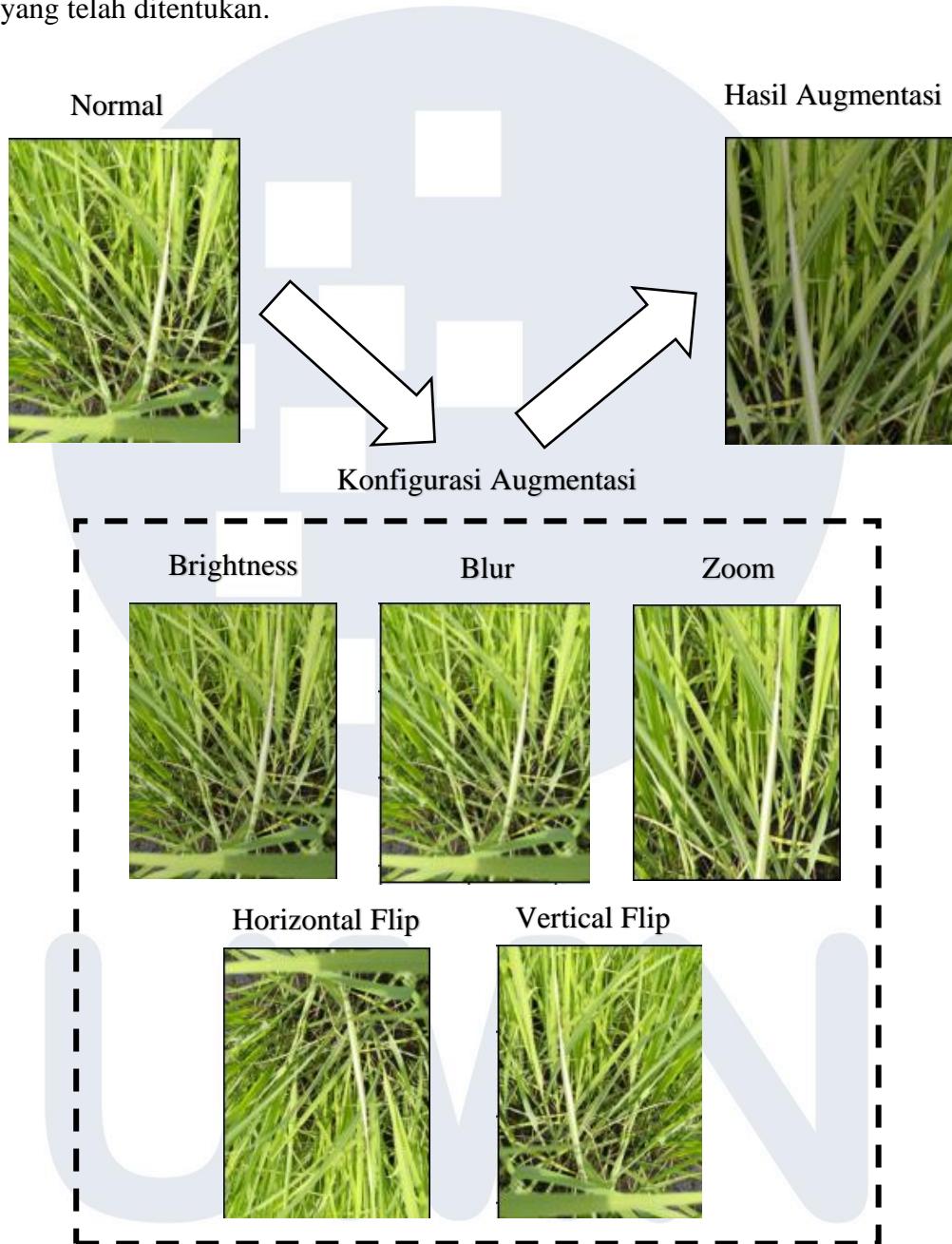
Proses augmentasi meliputi kombinasi penggunaan pengubah cahaya, pembalikan gambar, pengaburan gambar, dan pengubah kedekatan gambar (*zoom*). Berikut Gambar 4.1 kode augmentasi data yang penulis lakukan.

```
def blur(img):
    return (cv2.blur(img,(random.randint(1, 3),random.randint(1, 3)))) 

datagen = ImageDataGenerator(
    brightness_range=[0.3,1],
    horizontal_flip=True,
    vertical_flip=True,
    fill_mode='nearest',
    preprocessing_function=blur,
    zoom_range=[0.5,1.0]
)
```

Gambar 4.1 Potongan Kode Konfigurasi Augmentasi

Berikut Gambar 4.2, contoh hasil augmentasi menggunakan konfigurasi yang telah ditentukan.



Gambar 4.2 Contoh Hasil Augmentasi Padi

Augmentasi akan dilakukan satu per satu kelas penyakit yang kurang dari 1000 gambar. Tentunya perlu dilakukan perhitungan perulangan yang sesuai agar gambar yang diaugmentasi mendekati 1000 gambar. Contohnya pada kelas penyakit bacterial_leaf_blight, memiliki total 379 gambar. Sehingga memerlukan sekitar 621 gambar agar mencapai 1000 gambar.

Penulis menginginkan setiap gambar diaugmentasi sebanyak 3 kali. Sehingga didapatkan iterasi yaitu 621 dibagi 3 adalah 207. Didapatkanlah perulangan augmentasi sebanyak 3 kali setiap gambar dan perulangan gambar yang akan digunakan sebanyak 207 gambar. Namun, pada prakteknya gambar yang berhasil dibuat tidak selalu pas 1000 gambar. Sehingga penulis mengakalinya dengan menambahkan perulangan, contohnya 207 penulis tambah menjadi 210 kali. Berikut kode yang penulis gunakan untuk melakukan hal tersebut dapat terlihat pada Gambar 4.3.

```
#Bacterial_Leaf_Blight - 379
#1000 - 379 = 621
#207 Gambar diaugment dengan iterasi 3 kali ==> 621
folder_dir = "aug_oversampling_v2_dataset/separate_to_1000_image/bacterial_leaf_blight/"
save_here = "aug_oversampling_v2_dataset/balanced_oversampling/bacterial_leaf_blight/"
i = 1
for image in os.listdir(folder_dir):
    image_path = 'aug_oversampling_v2_dataset/separate_to_1000_image/bacterial_leaf_blight/' + image
    image = np.expand_dims(imageio.imread(image_path), 0)
    datagen.fit(image)
    j=1
    for x in datagen.flow(image,
                           batch_size=1,
                           save_to_dir=save_here,
                           save_prefix='aug',
                           save_format='jpg'):
        j += 1
        if j > 3:
            break

    i += 1
    if i > 210:
        break
```

Gambar 4.3 Potongan Kode Augmentasi Kelas Padi

Tentu saja setiap kelas penyakit yang kurang dari 1000 gambar memiliki perhitungan perulangan yang berbeda-beda. Hal tersebut tergantung pada jumlah gambar pada kelas penyakit tersebut.

Setelah dataset baik dengan teknik undersampling dan oversampling sesuai yang diharapkan, maka akan dilakukan pemisahan dataset. Disini terdapat perbedaan perlakuan untuk dataset teknik undersampling dan dataset teknik oversampling. Untuk dataset teknik undersampling akan dilakukan pemisahan menjadi data latih dan data uji. Pemisahan dibantu dengan *library python* yaitu *splitfolders* dengan perbandingan 90:10. Berikut kode yang penulis gunakan untuk melakukan hal tersebut dapat terlihat pada Gambar 4.4.

```

import splitfolders
input_folder = 'aug_undersampling_dataset/balanced_undersampling'
splitfolders.ratio(input_folder, output="aug_undersampling_dataset", ratio=(.9,.1),
group_prefix=None)

Copying files: 3370 files [00:18, 181.68 files/s]

```

Gambar 4.4 Potongan Kode Pemisahan Dataset Latih dan Uji

Untuk *dataset* teknik *oversampling* tidak dilakukan proses pemisahan, karena data uji telah dipisahkan di awal. Pembagian data validasi akan dilakukan saat proses klasifikasi. Sebenarnya pemisahan data validasi baik dengan potongan kode di atas maupun saat klasifikasi sama saja karena perbandingan akhir tetap 80:10:10 (data latih : data validasi : data uji).

4.2.2 Proses Klasifikasi

Proses klasifikasi diawali dengan mengimport model EfficientNetV2B0 dan juga model MobileNetV3-Large. Model tersebut akan disesuaikan kembali layer outputnya agar sesuai dengan dataset yang penulis gunakan. Selain itu penulis juga mengubah *hyperparameter* sesuai yang telah penulis paparkan pada Bab III. Berikut kode yang penulis gunakan untuk melakukan hal tersebut dapat terlihat pada Gambar 4.5.

```

def create_model(base_model, classData):
    for layer in base_model.layers:
        layer.trainable = False
    top_model = base_model.output
    #flatten layer
    top_model = Flatten(name="flatten")(top_model)
    #dense layer
    top_model = Dense(1024, activation='relu')(top_model)
    #dropout layer
    top_model = Dropout(0.2)(top_model)
    #dense layer
    output_layer = Dense(classData, activation='softmax')(top_model)
    model = Model(inputs=base_model.input, outputs=output_layer)

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

    return model

train_path = 'aug_oversampling_dataset/train'
test_path = 'aug_oversampling_dataset/test'
IMG_SHAPE = (224, 224, 3)
batch_size = 32
classData = 10

base_model = EfficientNetV2B0(
    include_top=False,
    weights="imagenet",
    input_shape=IMG_SHAPE,
)
model = create_model(base_model, classData)

```

Gambar 4.5 Potongan Kode Penyesuaian Layer dan *Hyperparameter*

Setelah itu penulis melakukan pemisahan data latih menjadi data latih dan data validasi menggunakan bantuan library Tensorflow Keras *preprocessing image_dataset_from_directory* dengan perbandingan 90:10. Berikut potongan kode yang penulis gunakan untuk melakukan hal tersebut dapat terlihat pada Gambar 4.6.

```
train_ds = tf.keras.preprocessing.image_dataset_from_directory(  
    train_path,  
    validation_split=0.1,  
    subset="training",  
    seed=123,  
    label_mode='categorical',  
    image_size=(224,224),  
    batch_size=batch_size  
)  
  
val_ds = tf.keras.preprocessing.image_dataset_from_directory(  
    train_path,  
    validation_split=0.1,  
    subset="validation",  
    seed=123,  
    label_mode="categorical",  
    image_size=(224,224),  
    batch_size=batch_size  
)
```

Gambar 4.6 Potongan Kode Pembagian Data Latih dan Validasi

Setelah model, data latih, dan data uji telah disiapkan, penulis akan melakukan proses *training* dengan tambahan parameter *EarlyStopping*. Berikut kode yang penulis gunakan untuk menjalankan proses tersebut dapat terlihat pada Gambar 4.7.

```
early_stop = EarlyStopping(monitor='val_loss', mode='min', patience=3)  
history = average_ensemble_model.fit(  
    train_ds,  
    validation_data=val_ds,  
    epochs=10,  
    callbacks=[early_stop]  
)  
  
Epoch 1/10  
282/282 [=====] - 2016s 7s/step - loss: 0.0881 - accuracy: 0.9824 - val_loss: 0.6618 - val_accuracy: 0.0500  
Epoch 2/10  
282/282 [=====] - 41s 141ms/step - loss: 0.0942 - accuracy: 0.9832 - val_loss: 0.3549 - val_accuracy: 0.9100  
Epoch 3/10  
282/282 [=====] - 43s 146ms/step - loss: 0.0488 - accuracy: 0.9922 - val_loss: 0.3435 - val_accuracy: 0.9250  
Epoch 4/10  
282/282 [=====] - 41s 143ms/step - loss: 0.0388 - accuracy: 0.9957 - val_loss: 0.2968 - val_accuracy: 0.9300  
Epoch 5/10  
282/282 [=====] - 41s 141ms/step - loss: 0.0239 - accuracy: 0.9970 - val_loss: 0.2983 - val_accuracy: 0.9260  
Epoch 6/10  
282/282 [=====] - 41s 141ms/step - loss: 0.0225 - accuracy: 0.9966 - val_loss: 0.2789 - val_accuracy: 0.9350  
Epoch 7/10  
282/282 [=====] - 41s 143ms/step - loss: 0.0288 - accuracy: 0.9942 - val_loss: 0.4476 - val_accuracy: 0.9190  
Epoch 8/10  
282/282 [=====] - 41s 144ms/step - loss: 0.0432 - accuracy: 0.9956 - val_loss: 0.3679 - val_accuracy: 0.9140  
Epoch 9/10  
282/282 [=====] - 41s 143ms/step - loss: 0.0291 - accuracy: 0.9960 - val_loss: 0.3562 - val_accuracy: 0.9200
```

Gambar 4.7 Potongan Kode *Training* Model

Terdapat beberapa perbedaan saat menggunakan *ensemble learning* dalam mengklasifikasi model. Perbedaan paling utama adalah cara mengolah model untuk dilatih. Dikarenakan pada penelitian ini menggunakan teknik *average* dan *concatenation*, terdapat perbedaan dalam mengolah model.

Untuk teknik *average ensemble*, penulis mengimport kembali model MobileNetV3-Large dan EfficientNetV2B0 yang telah dilatih. Output dari kedua model tersebut diambil dan dirata-ratakan menggunakan metode *average*. Setelah itu akan dibuat model baru hasil dari rata-rata tersebut. Model baru yang telah dibuat tersebut akan dilatih dengan *hyperparameter* yang sama dengan model *deep learning* sebelumnya. Berikut kode yang penulis gunakan untuk menerapkan *average ensemble* dapat terlihat pada Gambar 4.8.

```
from tensorflow.keras.models import Model, load_model
from tensorflow.keras.layers import Input, Average
model_1 = load_model('PaddyDisease_EfficientNetV2B0_OverSampling_224.h5')
model_1 = Model(inputs=model_1.inputs,
                 outputs=model_1.outputs,
                 name='EfficientNetV2B0')
model_2 = load_model('PaddyDisease_MobileNetV3Large_OverSampling_224.h5')
model_2 = Model(inputs=model_2.inputs,
                 outputs=model_2.outputs,
                 name='MobileNetV3Large')
models = [model_1, model_2]
model_input = Input(shape=(224, 224, 3))
model_outputs = [model(model_input) for model in models]
ensemble_output = Average()(model_outputs)
average_ensemble_model = Model(inputs=model_input, outputs=ensemble_output,
                                 name='ensemble')
```

Gambar 4.8 Potongan Kode Inisialisasi Average Ensemble

Untuk teknik *concatenation ensemble*, penulis tidak mengimport model yang telah penulis latih sebelumnya, melainkan *import base* model MobileNetV3-Large dan EfficientNetV2B0. Penulis akan mengabungkan kedua model tersebut dengan metode *concatenate*. Setelah itu penulis menambahkan layer penyesuaian. Model *concatenation ensemble* ini juga menggunakan *hyperparameter* yang sama dengan model sebelumnya. Berikut kode yang penulis gunakan dapat terlihat pada Gambar 4.9.

```
    inp = Input((224,224,3))

    mobileNetV3Large_process = Lambda(process_mobileNetV3Large)(inp)
    MobileNetV3Large = model_MobileNetV3Large(mobileNetV3Large_process)

    efficientNetV2B0_process = Lambda(process_efficientNetV2B0)(inp)
    EfficientNetV2B0 = model_EfficientNetV2B0(efficientNetV2B0_process)

    x = Concatenate()([MobileNetV3Large, EfficientNetV2B0])
    x = Flatten(name="flatten")(x)
    x = Dense(1024, activation='relu')(x)
    x = Dropout(0.2)(x)
    out = Dense(10, activation='softmax')(x)

concatenation_ensemble_model = Model(inp, out)
```

Gambar 4.9 Potongan Kode Inisialisasi *Concatenation Ensemble*

Perbedaan perlakuan antara *average ensemble* dan *concatenation ensemble* dilakukan dikarenakan paper yang didapat penulis untuk *average ensemble*, mengambil rata-rata dari model yang telah dilatih. Sedangkan *concatenation ensemble*, menggabungkan base model dan dilatih seperti yang telah penulis jelaskan pada Bab III. Selai itu, saat penulis mencoba melakukan perlakuan yang penulis lakukan ke *average ensemble* pada *concatenation ensemble*, terdapat error akibat layer yang bertabrakan antara model EfficientNetV2B0 dan MobileNetV3-Large yang telah penulis latih. Pada kedua model tersebut penulis telah menambahkan beberapa layer pada layer output yang tentunya layer tersebut sama persis. Sehingga terjadi *duplicate layer* pada saat menggabungkan kedua model tersebut.

Model yang telah dilatih akan disimpan dalam format .h5 yang akan digunakan untuk membuat *classification report*, *confusion matrix*, dan analisa kecepatan pemrosesan pada sistem PWA. Total model yang tersimpan dari hasil latih adalah 16 model berformat .h5 yang akan diimplementasi satu per satu pada sistem PWA. Ukuran file setiap model h5 juga beragam sesuai dengan jenis model dan ukuran gambar yang digunakan.

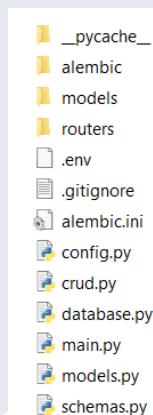
4.2.3 Implementasi Aplikasi

Pada bagian ini akan dijelaskan secara singkat penulis membuat sistem PWA yang nantinya akan diimplementasikan pada model untuk menganalisa kecepatan pemrosesan masing-masing model.

4.2.3.1. Pembuatan Sistem Backend

Sistem backend dibuat menggunakan *framework* FastAPI dimulai dengan menginstall FastAPI dan *uvicorn package*. Setelah itu penulis mulai membuat berbagai endpoint untuk mendukung penelitian, salah satunya adalah endpoint yang dapat digunakan PWA dalam memprediksi gambar. Penulis juga menginstall *tensorflow package* yang digunakan untuk mengimport model dan menjalankannya.

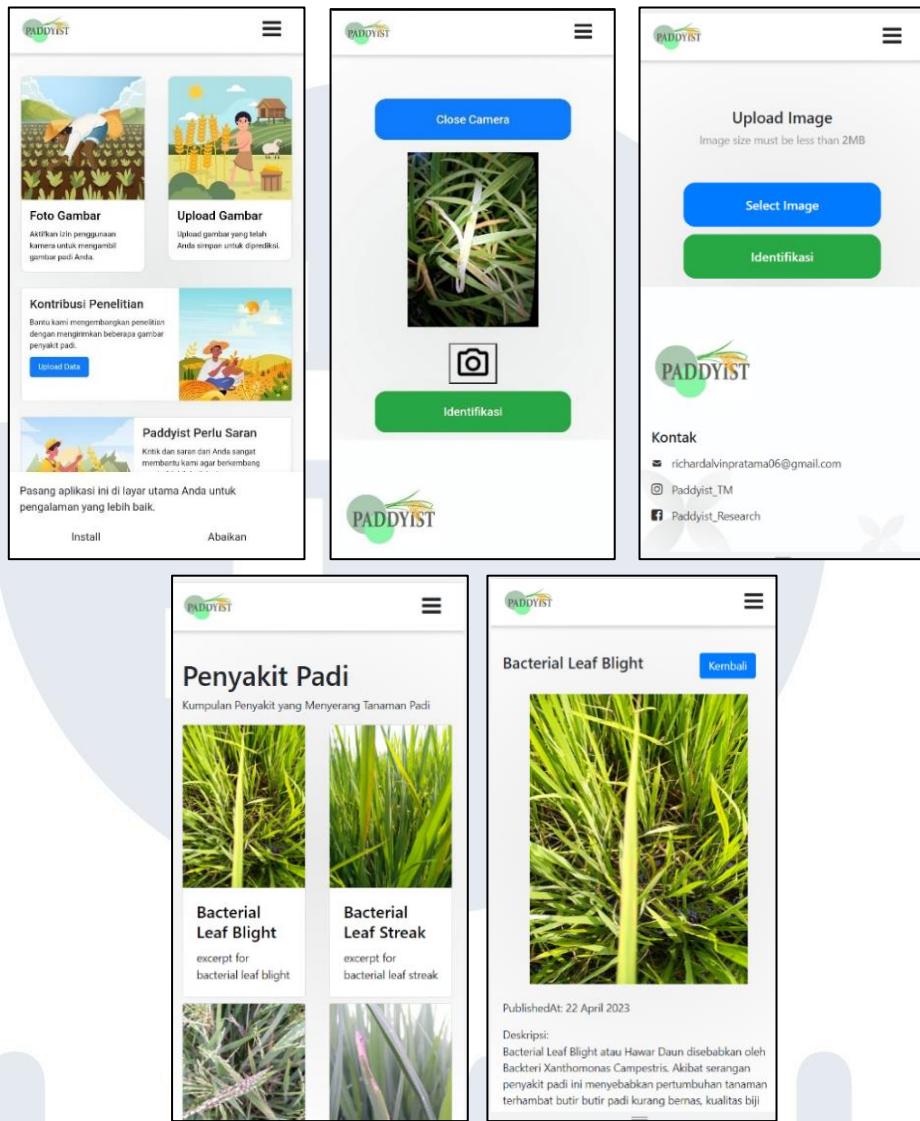
Dapat terlihat gambar di samping merupakan struktur folder *backend* yang penulis buat untuk mengklasifikasi penyakit tanaman padi. Folder alembic berisikan *version* atau *migration* tabel. Folder models berisikan 16 model yang telah penulis latih. Folder routers digunakan penulis untuk merapikan *endpoint*. File-file lainnya digunakan untuk menjalankan prediksi, *load* model, konfigurasi FastAPI ke *database* PostgreSQL, pembuatan request dan respons API yang akan digunakan PWA, dan lain sebagainya.



Gambar 4.10 Struktur Folder Backend

4.2.3.2. Pembuatan Tampilan Sistem PWA

Tampilan sistem PWA dibuat dengan bantuan framework NuxtJs 2. Pertama penulis akan menggunakan perintah “npm init nuxt-app <project_name>” untuk menginstall semua *dependency*. Setelah itu penulis akan membuat kode tampilan, konfigurasi PWA, serta koneksi ke *backend*. Setelah proses perancangan PWA serta koneksi ke *backend* telah berhasil, penulis akan menggunakan perintah “nuxt build” yang akan membuat folder dist. Folder tersebut akan diupload ke Netlify agar sistem PWA dapat diakses secara publik. Berikut tampilan sederhana yang penulis buat yang dapat terlihat pada Gambar 4.11.



Gambar 4.11 Tampilan User Interface PWA Paddyist

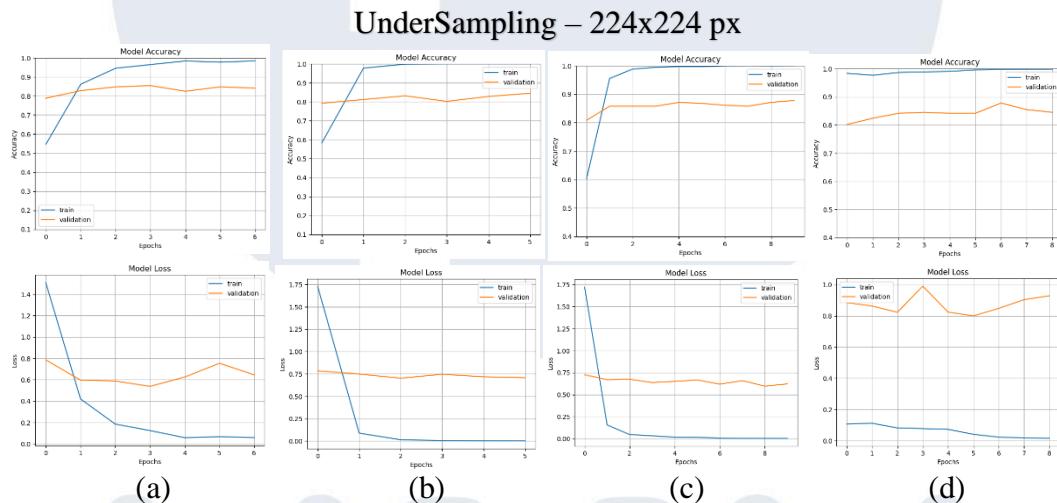
Hal yang perlu diperhatikan dalam mengimplementasikan PWA pada *framework* NuxtJs adalah konfigurasi pwa tersebut. Penulis perlu menginstall *dependency* nuxt-pwa dengan perintah “`npm i --save-dev @nuxtjs/pwa`”. Perintah tersebut akan membuat file nuxt.config.js. Di file ini penulis bisa mengatur berbagai hal terkait PWA baik logo yang akan tampil saat instalasi dan lain sebagainya. Selain itu pengimplementasian kamera juga penting, penulis menggunakan *MediaDevices* agar dapat mengakses fitur kamera di ponsel maupun di laptop.

4.3 Hasil dan Analisis Training Model

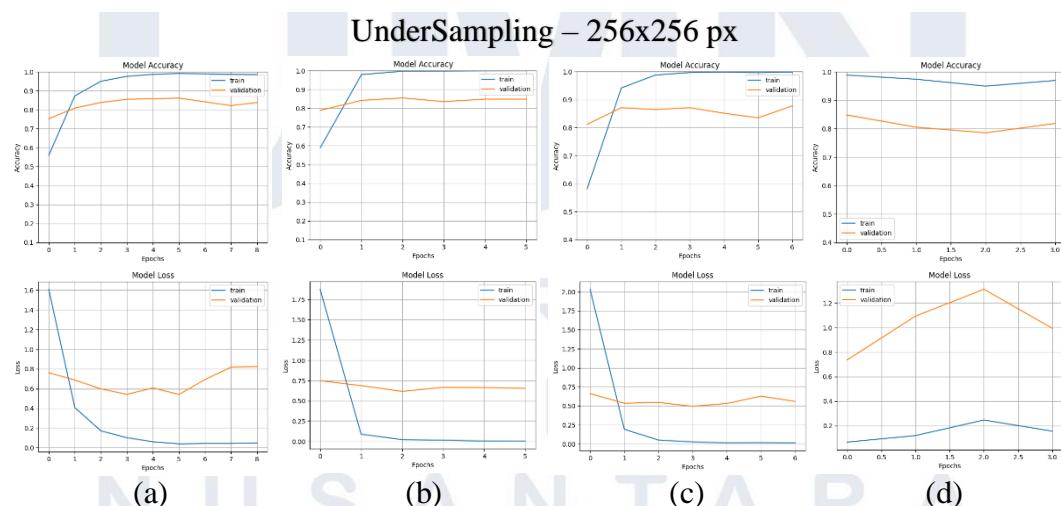
Dari proses *training* yang telah dilakukan pada berbagai percobaan, didapatkanlah hasil performa model serta waktu pelatihan. Pembahasan terdiri dari beberapa sub-bab yaitu performa *training* model pada dataset teknik *undersampling* dan performa *training* model pada dataset teknik *oversampling*.

4.3.1 Performa Training Model Pada Dataset - UnderSampling

Hasil akurasi dan loss saat training setiap model dengan 32 batch size, Adam optimizer, 0.0001 learning rate, earlystop patience 3, 224 px & 256 px ukuran gambar dan undersampling dataset dapat terlihat pada Gambar 4.12 & 4.13.



Gambar 4.12 Grafik Train & Loss - UnderSampling (224px); (a) EfficientNetV2B0
(b) MobileNetV3-Large (c) Concatenation (d) Average



Gambar 4.13 Grafik Train & Loss – UnderSampling (256px); (a) EfficientNetV2B0
(b) MobileNetV3Large (c) Concatenation (d) Average

Dapat terlihat pada grafik diatas, dapat dianalisa bahwa akurasi training yang didapat setiap model selalu meningkat dan hampir menyentuh 100%. Namun, akurasi validasi yang didapat berada antara 80% - 85%. Hal ini disebabkan karena data yang digunakan hanya 303 data per kelas, sehingga semua model yang diuji menghasilkan grafik yang *overfitting*. Gambar padi yang digunakan pada penelitian ini juga masih memiliki *background* sehingga terlalu banyak informasi yang tidak sebanding dengan jumlah data per kelas.

Hal unik terlihat pada hasil akurasi validasi pada model *ensemble*. Terlihat pada *concatenation ensemble*, akurasi validasi terus meningkat dan mendekati 88%. Namun pada *average ensemble*, pada *epoch* keenam, akurasi validasi mendekati 88%, tapi selanjutnya mengalami penurunan akurasi validasi. Perbedaan akurasi validasi kedua model *ensemble* ini disebabkan oleh *concatenation ensemble* tidak menggunakan hasil pada model latih dua model *pre-trained* sebelumnya sedangkan *average ensemble* melakukan *training* ulang berdasarkan hasil latih yang didapat pada model latih dua model *pre-trained*. Sehingga membuat hasil *average ensemble* tidak efektif untuk dataset teknik *undersampling*. Hal menarik lainnya dapat terlihat pada hasil validasi loss pada metode *average ensemble* yang terlihat cukup tinggi dan berbanding terbalik dengan validasi akurasi yang didapat. Hal ini disebabkan oleh *average ensemble* juga mengambil rata-rata dari *loss* pada *pre-trained* model. Sehingga jika kita melihat *loss* pada EfficientNetV2B0, ketika *epoch* mendekati terakhir terdapat kenaikan yang cukup signifikan.

Jika dibandingkan dengan penggunaan dataset teknik *undersampling* dan ukuran gambar 256x256 *pixel*, tidak terdapat perbedaan yang signifikan dengan penggunaan dataset teknik *undersampling* dan ukuran gambar 224x224 *pixel*. Sedikit perbedaan yang dapat dianalisa adalah model MobileNetV3Large yang memiliki validasi akurasi sedikit lebih baik. Keanehan kembali terjadi pada validasi loss *average ensemble* dan bahkan sedikit lebih buruk. Terlihat bahwa validasi loss mencapai > 1.2 yang pada penggunaan ukuran gambar 224x224 *pixel* hanya mencapai 1.0. Sehingga dapat diberikan kesimpulan bahwa

penggunaan dataset teknik *undersampling* tidak banyak perubahan yang terjadi antara penggunaan ukuran gambar 224x224 *pixel* dan 256x256 *pixel*.

Berikut hasil validasi akurasi terbaik dan loss terbaik setiap model dengan dataset *undersampling* dapat terlihat pada Tabel 5.

Table 5. Validasi Akurasi & Loss Terbaik - UnderSampling

Model	Ukuran	Val Akurasi	Val Loss
EfficientNetV2B0	224	85.48%	0.5378
	256	86.14%	0.5399
MobileNetV3-Large	224	84.49%	0.7019
	256	85.48%	0.6171
Average Ensemble	224	87.79%	0.7998
	256	84.82%	0.7354
Concatenation Ensemble	224	87.79%	0.5958
	256	87.79%	0.4942

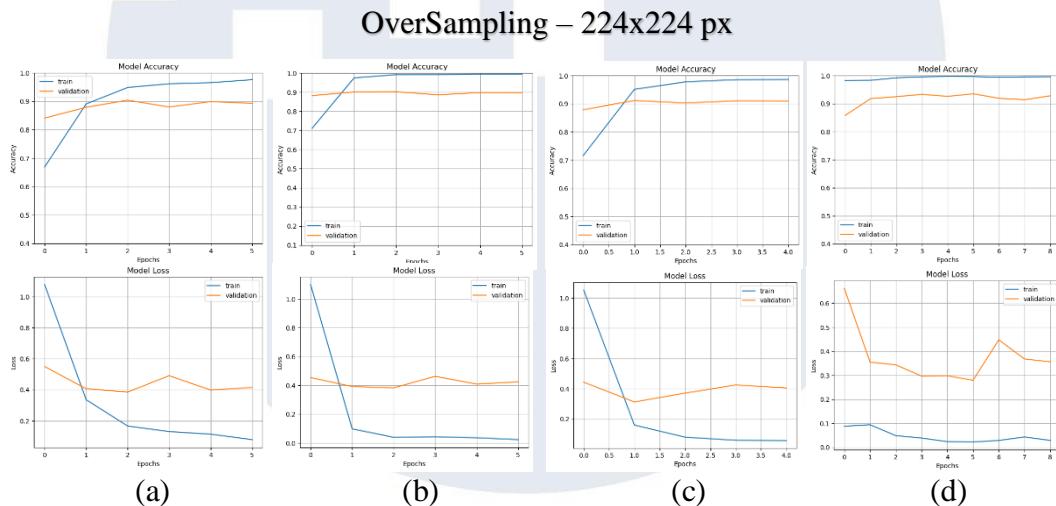
Terlihat pada tabel 5, kombinasi validasi akurasi dan loss terbaik didapatkan oleh *concatenation ensemble* pada ukuran 256x256 *pixel* dengan validasi akurasi 87.79% dan loss 0.4942 dan didukung ketika menggunakan ukuran 224x224 *pixel* pada model yang sama yang performa yang mirip. Hal ini menjadikan *concatenation ensemble* merupakan metode yang paling sesuai jika digunakan pada dataset teknik *undersampling*. Hal ini dapat terjadi karena *concatenation ensemble* melakukan ekstraksi fitur dari awal tanpa perlu mengambil hasil keluaran *pre-trained* yang telah dilatih.

Berbeda halnya dengan *average ensemble*, memiliki akurasi yang bagus tetapi juga mendapatkan loss yang paling buruk diantara model yang dianalisa. Hal ini diakibatkan konsep *average ensemble* yang mengambil rata-rata prediksi dari model *pre-trained* yang telah dilatih, padahal model tersebut masih memiliki loss yang buruk. Untuk akurasi validasi terburuk didapatkan model MobileNetV3-Large dengan ukuran gambar 224x224 *pixel*. Hal ini dikarenakan parameter paling sedikit dari model yang dianalisa dan juga pengaruh dari

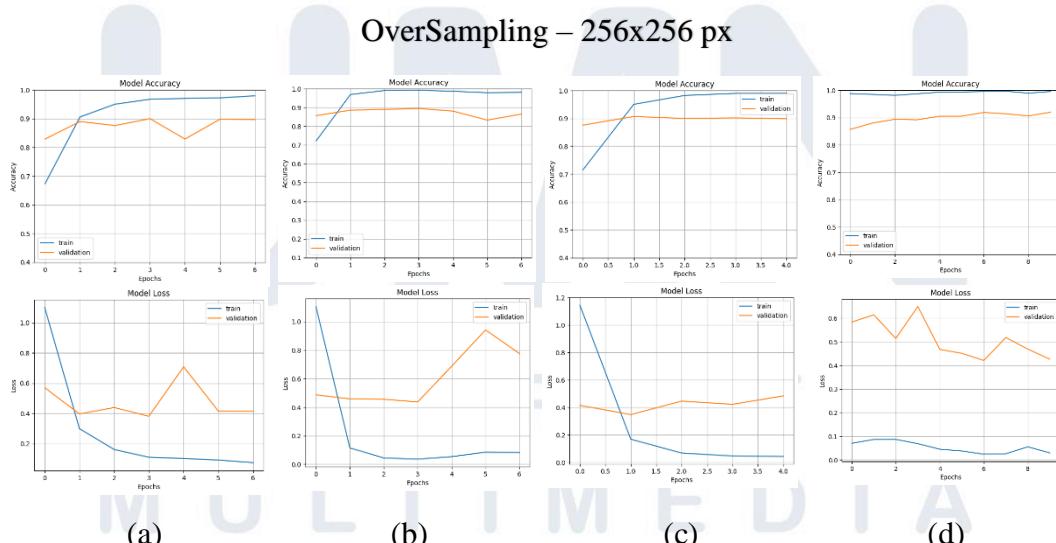
penggunaan ukuran gambar. Terbukti ketika menggunakan ukuran gambar 256×256 pixel, terdapat peningkatan baik dari validasi akurasi maupun loss.

4.3.2 Performa Training Model Pada Dataset - OverSampling

Hasil akurasi dan loss saat training setiap model dengan 32 batch size, Adam optimizer, 0.0001 learning rate, earlystop patience 3 , 224×224 & 256×256 ukuran gambar dan oversampling teknik dataset dapat terlihat pada Gambar 4.14 & 4.15.



Gambar 4.14 Grafik Train & Loss – OverSampling (224px); (a) EfficientNetV2B0 (b) MobileNetV3-Large (c) Concatenation (d) Average



Gambar 4.15 Grafik Train & Loss – OverSampling (256px); (a) EfficientNetV2B0 (b) MobileNetV3-Large (c) Concatenation (d) Average

Dapat terlihat pada grafik diatas, dapat dianalisa bahwa akurasi yang didapat setiap model selalu meningkat dan hampir menyentuh 100%. Namun terdapat perbedaan grafik akurasi training antara dataset teknik *undersampling* dan dataset teknik *oversampling*. Dataset teknik *oversampling* memiliki grafik akurasi training yang sedikit lebih baik dibandingkan dataset teknik *undersampling*. Selain itu, validasi akurasi yang didapat juga mengalami peningkatan pada penggunaan dataset teknik *oversampling*. Pada dataset teknik *oversampling*, validasi akurasi yang didapat di rentang 80% - 97%. Sedangkan dataset teknik *undersampling*, validasi akurasi yang didapat di rentang 80% - 87%. Sehingga dapat dikatakan bahwa penggunaan dataset teknik *oversampling* dapat membantu mengurangi *overfitting* pada suatu model dan dapat meningkatkan validasi akurasi pada suatu model.

Penggunaan dataset teknik *oversampling* juga memperbaiki validasi *loss* yang sebelumnya pada dataset *undersampling* memiliki validasi *loss* yang kurang baik. Terlihat pada validasi loss dataset teknik oversampling, setiap model berada di bawah 0.5. Sedangkan pada dataset teknik *undersampling*, setiap model berada di atas 0.5. Hal positif lainnya adalah validasi loss untuk model *average ensemble* memiliki hasil grafik yang jauh lebih baik daripada ketika menggunakan dataset teknik *undersampling*. Hal ini dikarenakan penggunaan dataset teknik *oversampling* memiliki jumlah total gambar per kelas lebih banyak yaitu 1.000 gambar. Sedangkan dataset teknik *undersampling* memiliki total gambar per kelas yaitu 303 gambar.

Jika dibandingkan dengan penggunaan ukuran gambar 256x256 *pixel* pada dataset teknik *oversampling*, terdapat beberapa perbedaan yang dapat dianalisa. Perbedaan pertama adalah pada hasil validasi akurasi cenderung menurun pada semua model ketika menggunakan ukuran gambar 256x256 *pixel*. Hal ini disebabkan, model EfficientNetV2B0 dan MobileNetV3Large memang memiliki ukuran gambar yang disarankan yaitu 224x224 *pixel*.

Dapat terlihat pada grafik di atas, dapat dianalisa bahwa akurasi yang didapat setiap model juga selalu meningkat dan hampir menyentuh 100%.

Terdapat perbedaan antara grafik *oversampling* (224x224 *pixel*) dan *oversampling* (256x256 *pixel*). Perbedaannya terletak pada hasil validasi akurasi EfficientNetV2B0 dengan ukuran gambar 256x256 *pixel* yang cenderung menurun jika dibandingkan dengan penggunaan ukuran gambar 224x224 *pixel*. Hal ini disebabkan, model EfficientNetV2B0 memang memiliki ukuran gambar yang disarankan yaitu 224x224 *pixel*.

Berikut hasil validasi akurasi terbaik dan *loss* terbaik setiap model dengan dataset *oversampling* dapat terlihat pada Tabel 6.

Table 6. Validasi Akurasi & Loss Terbaik - OverSampling

Model	Ukuran	Val Akurasi	Val Loss
EfficientNetV2B0	224	90.40%	0.3849
	256	90.00%	0.3813
MobileNetV3-Large	224	90.20%	0.3823
	256	89.50%	0.4374
Average Ensemble	224	93.50%	0.2789
	256	92.00%	0.4262
Concatenation Ensemble	224	91.20%	0.3125
	256	90.70%	0.3475

Terlihat pada Tabel 6, validasi akurasi dan loss terbaik didapat *average ensemble* dengan validasi akurasi 93.50% dan 0.2789. Hal ini dapat terjadi dikarenakan model *pre-trained* yang telah dilatih, memiliki hasil akurasi dan loss yang bagus. Sehingga ketika dilakukan rata-rata prediksi, hasil *average ensemble* juga menjadi bagus. Untuk validasi dan loss terburuk didapatkan oleh MobileNetV3-Large, hal ini memiliki persamaan hasil ketika menggunakan dataset *undersampling*.

Namun, terdapat perbedaan antara penggunaan dataset *undersampling* dan dataset *oversampling* adalah penggunaan ukuran 224x224 *pixel* lebih bagus digunakan pada dataset *oversampling*. Hal ini terjadi karena *pre-trained* model yang digunakan, sebelumnya telah dilatih oleh dataset ImageNet yang

menggunakan ukuran gambar 224x224 *pixel*. Sehingga arsitektur model *pre-trained* telah dioptimalkan untuk mengelola gambar dengan ukuran tersebut. Pada dataset *undersampling*, mendapatkan akurasi dan loss yang lebih baik ketika menggunakan ukuran 256x256 *pixel*. Hal ini terjadi karena dataset yang sedikit memberikan performa akurasi kecil tetapi dibantu penggunaan ukuran gambar 256x256 *pixel* yang memiliki resolusi informasi gambar lebih banyak.

4.4. Hasil Classification Report Testing Model

Pembahasan hasil *classification report* model terdiri dari beberapa sub-bab yaitu hasil classification report model pada dataset teknik *undersampling*, hasil classification report model pada dataset teknik *oversampling*, dan rangkuman kombinasi model terbaik.

4.4.1 Hasil Classification Report Model Pada Dataset - *UnderSampling*

Data uji yang digunakan pada sub-bab ini adalah data yang diambil dari dataset teknik *undersampling* atau dataset yang hanya memiliki 337 gambar per kelas. Penelitian ini mengambil 10% dari dataset teknik *undersampling* untuk dijadikan data uji. Sehingga didapatkanlah 34 gambar per kelas yang akan diuji dengan *classification report*. Berikut Tabel 7, hasil *classification report* dengan dataset teknik *undersampling*.

Table 7. Hasil Classification Report Model – Dataset Teknik UnderSampling

Model	Image Size	Average Precision	Average Recall	Average F1-score	Accuracy
EfficientNetV2B0	224	0.8421	0.8412	0.8404	0.8412
	256	0.8286	0.8147	0.8137	0.8147
MobileNetV3-Large	224	0.8189	0.8176	0.8167	0.8176
	256	0.8407	0.8353	0.8359	0.8353
Average Ensemble	224	0.8576	0.8529	0.8535	0.8529
	256	0.8189	0.8000	0.8028	0.8000
Concatenation Ensemble	224	0.8578	0.8529	0.8531	0.8529
	256	0.8606	0.8559	0.8537	0.8559

Terlihat pada Tabel 7, hasil akurasi yang didapat berkisar antara 80-86%. Hal ini selaras dengan validasi akurasi yang didapat yaitu sekitar 80-87%. Penggunaan ukuran gambar yang berbeda antara 224x224 *pixel* dan 256x256 *pixel* ternyata memberikan dampak performa yang berbeda antara keempat model tersebut. Untuk EfficientNetV2B0 dan *average ensemble*, memberikan performa lebih baik di ukuran 224x224 *pixel*. Sedangkan MobileNetV3Large dan *concatenation ensemble*, memberikan performa lebih baik di ukuran 256x256 *pixel*. Hasil *classification report* yang didapat, selaras dengan grafik pembelajaran, yang membuktikan bahwa *concatenation ensemble* memiliki hasil uji yang paling baik. Hasil terburuk didapat oleh *average ensemble* dengan ukuran gambar 256x256 *pixel*, juga selaras dengan grafik pembelajaran yang memiliki validasi akurasi terburuk kedua dan validasi loss terburuk kedua dari semua model yang dianalisa.

Dalam perbandingan dengan penelitian yang dilakukan oleh Paddy Doctor [34], hasil performa penggunaan dataset teknik undersampling masih memiliki performa yang cukup jauh dibawah penelitian Paddy Doctor. Hal ini disebabkan dataset teknik undersampling (3.370 gambar) memiliki perbandingan jumlah data yang sangat jauh jika dibandingkan dengan dataset Paddy Doctor (sekitar 16.225 gambar). Namun hasil performa yang didapat pada penggunaan dataset teknik *undersampling*, cukup memuaskan. Hal ini didukung oleh penggunaan *pre-trained* model yang lebih baru dibandingkan model yang digunakan Paddy Doctor serta penggunaan metode *ensemble*.

4.4.2 Hasil Classification Report Model Pada Dataset - *OverSampling*

Data uji yang digunakan pada sub-bab ini adalah data yang diambil dari dataset *original* sebelum dilakukan augmentasi. Penelitian ini mengambil 100 gambar per kelas dan didapatkanlah 1000 total gambar uji. Berikut Tabel 8, hasil *classification report* dengan dataset teknik *oversampling*.

Table 8. Hasil Classification Report Model - Dataset Teknik *OverSampling*

Model	Image Size	Average Precision	Average Recall	Average F1-score	Accuracy
EfficientNetV2B0	224	0.9129	0.9080	0.9085	0.9080
	256	0.9105	0.9050	0.9059	0.9050
MobileNetV3-Large	224	0.8933	0.8870	0.8877	0.8870
	256	0.8900	0.8760	0.8776	0.8760
Average	224	0.9344	0.9320	0.9318	0.9320
Ensemble	256	0.9339	0.9330	0.9328	0.9330
Concatenation	224	0.9148	0.9140	0.9141	0.9140
	256	0.9108	0.9040	0.9046	0.9040

Dapat terlihat pada Tabel 8, terjadi peningkatan hasil performa *testing* pada setiap model dan ukuran gambar pada dataset teknik *oversampling* daripada menggunakan dataset teknik undersampling. Rentang akurasi yang didapat dengan menggunakan dataset teknik *oversampling* adalah 87-93%. Sedangkan saat menggunakan dataset teknik *undersampling* mendapatkan rentang akurasi 80-86%. Pengaruh dari kuantitas gambar yang dilatih ternyata sangat berpengaruh terhadap hasil *classification report* yang didapat. Hasil *classification report* yang didapat juga selaras dengan grafik pembelajaran. Terlihat hasil terbaik didapat oleh model *average ensemble* dan untuk hasil terburuk didapat oleh model MobileNetV3-Large.

Dalam perbandingan dengan penelitian yang dilakukan oleh Paddy Doctor [34], hasil performa penggunaan dataset teknik *oversampling* masih dibawah performa yang didapat oleh penelitian Paddy Doctor. Contohnya terlihat pada model MobileNet Paddy Doctor memiliki akurasi 92%. Sedangkan model MobileNetV3-Large pada penelitian ini mendapatkan akurasi 88.7%. Padahal terlihat bahwa model MobileNetV3-Large merupakan versi terbaru dari model MobileNet. Hal ini dikarenakan kekurangan pada penggunaan dataset. Dataset teknik *oversampling* penelitian ini memiliki total 10.000 gambar. Sedangkan dataset Paddy Doctor memiliki total 16.255 gambar.

Namun dengan adanya penggunaan metode *ensemble*, hasil akurasi yang didapat bisa mencapai 92%. Bahkan jika dibandingkan dengan model CNN yang dibuat oleh Paddy Doctor sendiri, hanya mendapatkan sekitar 88.84%. Sehingga dapat dikatakan penggunaan metode *ensemble* dapat meningkatkan hasil performa tanpa perlu menghabiskan waktu membuat model baru dari awal. Walaupun hasil performa metode *ensemble* yang dilakukan pada penelitian ini masih dibawah model Resnet34 dan Xception yang mendapat hasil performa 96.58% dan 97.50%. Hal ini disebabkan kompleksitas arsitektur model dari Resnet34 dan Xception serta kuantitas dataset dengan total 16.255 gambar yang membuat performa menjadi lebih baik.

4.4.3 Rangkuman Model Terbaik Berdasarkan *Classification Report*

Dari semua performa model yang telah penulis uji, penulis akan mengambil tiga kombinasi model terbaik berdasarkan akurasi yang didapat dari *classification report*. Kombinasi model yang diambil penulis terdiri dari dua kombinasi model menggunakan dataset teknik *undersampling* dan dua kombinasi model menggunakan dataset teknik *oversampling*. Berikut Tabel 9, hasil performa model terbaik berdasarkan akurasi pada penelitian yang dilakukan penulis.

Table 9. Hasil Performa Model Terbaik Berdasarkan Akurasi

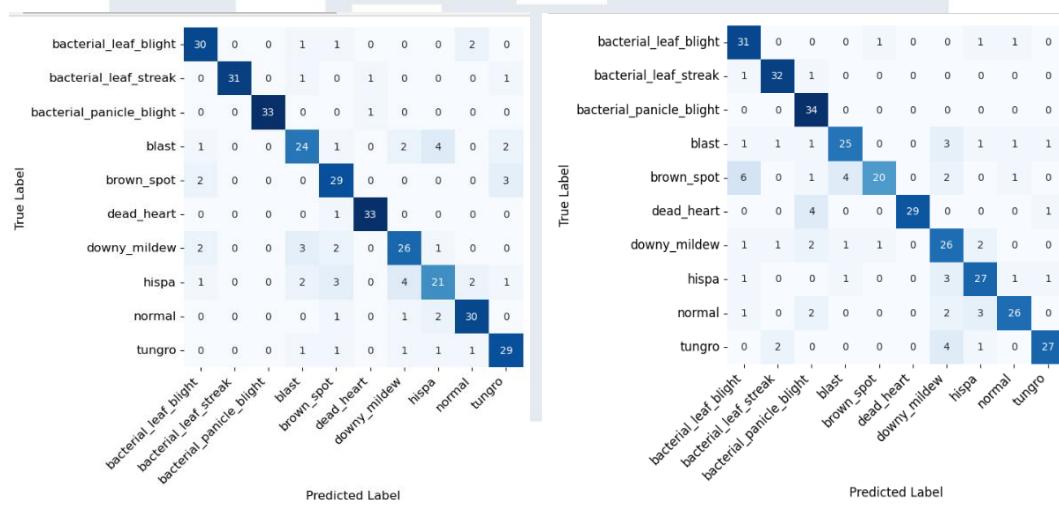
Model	Dataset	Image Size	Average Precision	Average Recall	Average F1-Score	Accuracy
Average Ensemble	OverSampling	256	0.9339	0.9330	0.9328	0.9330
Average Ensemble	OverSampling	224	0.9344	0.9320	0.9318	0.9320
Concatenation Ensemble	OverSampling	224	0.9148	0.9140	0.9141	0.9140

Terlihat dari hasil yang didapat penggunaan metode *ensemble* terutama *average ensemble* menjadi model dengan performa terbaik pada penelitian ini. Hal ini kembali lagi didukung oleh metode *average ensemble* yang menggabungkan

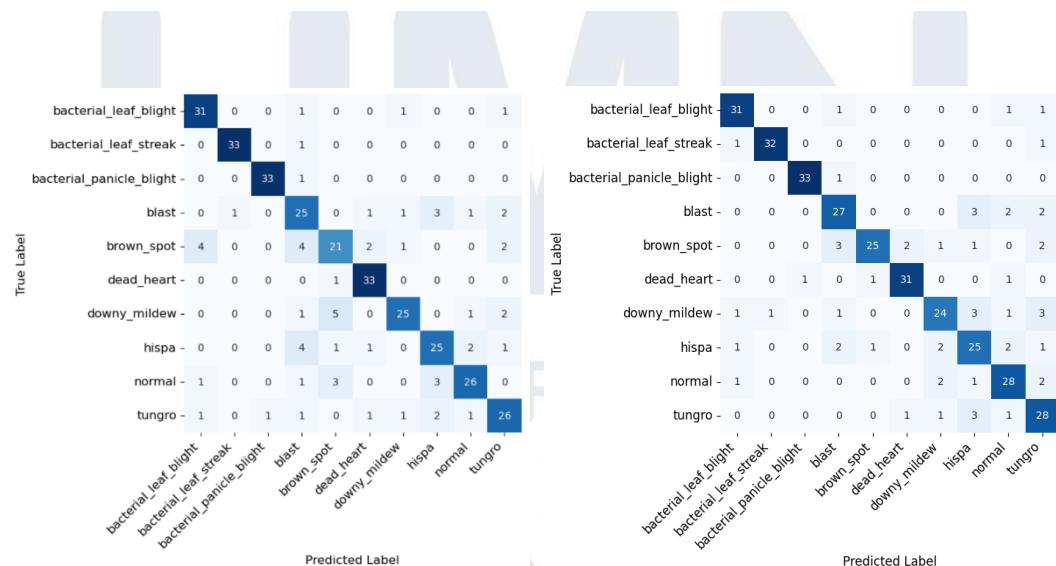
hasil rata-rata kedua hasil terbaik dari dua *pre-trained* model. Disini juga terlihat penggunaan dataset teknik *oversampling* atau banyaknya kuantitas gambar berpengaruh terhadap hasil performa. Dapat terlihat tidak ada dataset teknik *undersampling* pada tiga kombinasi model terbaik pada penelitian ini.

4.5. Analisa Confusion Matrix Terhadap Karakteristik Kelas

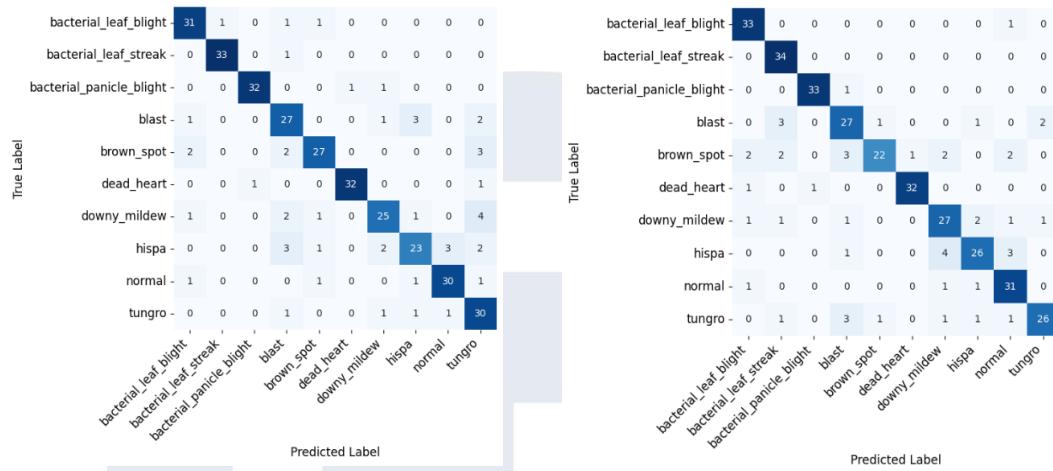
Pada bagian ini, penulis akan menganalisa pengaruh karakteristik kelas penyakit padi terhadap hasil *confusion matrix* yang didapat. Berikut merupakan *plot heatmap confusion matrix* model dengan dataset teknik *undersampling*.



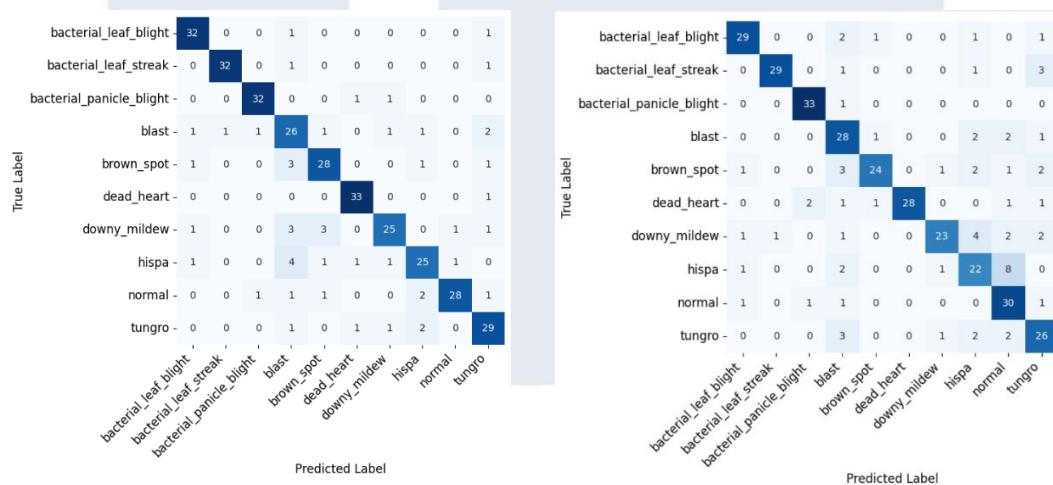
Gambar 4.16 Confusion Matrix EfficientNetV2B0 UnderSampling: (kiri) 224 (kanan) 256



Gambar 4.17 Confusion Matrix MobileNetV3Large UnderSampling; (kiri) 224 (kanan) 256



Gambar 4.18 Confusion Matrix Concatenation UnderSampling; (kiri) 224 (kanan) 256



Gambar 4.19 Confusion Matrix Average UnderSampling; (kiri) 224 (kanan) 256

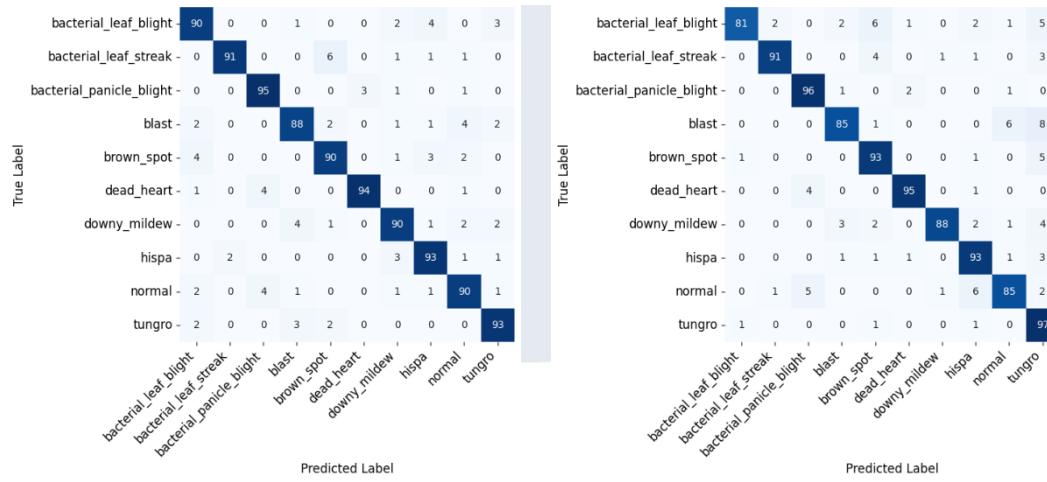
Berikut merupakan *plot heatmap confusion matrix* model dengan dataset teknik oversampling.



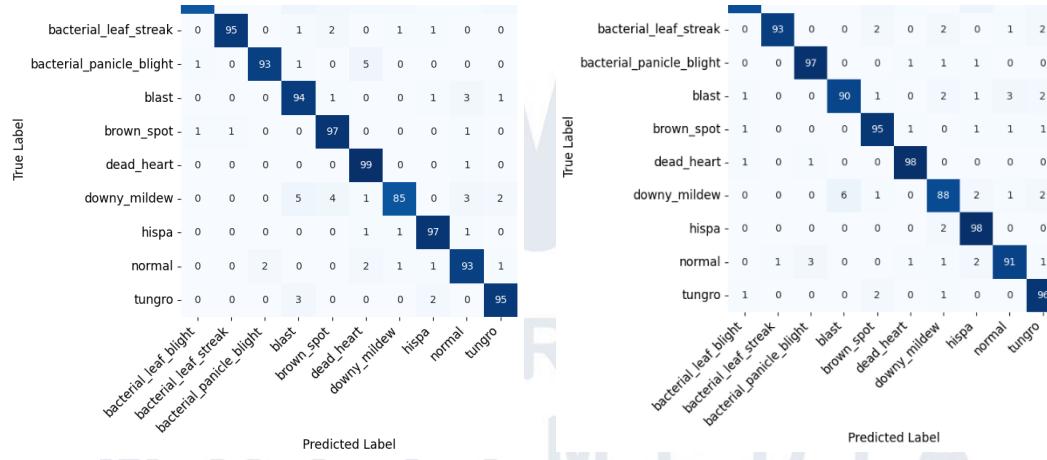
Gambar 4.20 Confusion Matrix EfficientNetV2B0 OverSampling; (kiri) 224 (kanan) 256



Gambar 4.21 Confusion Matrix MobileNetV3Large OverSampling; (kiri) 224 (kanan) 256



Gambar 4.22 Confusion Matrix Concatenation OverSampling; (kiri) 224 (kanan) 256



Gambar 4.23 Confusion Matrix Average OverSampling; (kiri) 224 (kanan) 256

Dari hasil confusion matrix dataset teknik *undersampling* dan teknik *oversampling*, penulis akan menganalisa kelas penyakit padi yang mayoritas memiliki TP (*True Positive*) terburuk pada semua percobaan model. Untuk memudahkan analisa, berikut Tabel 10, ringkasan TP terburuk pada semua percobaan model.

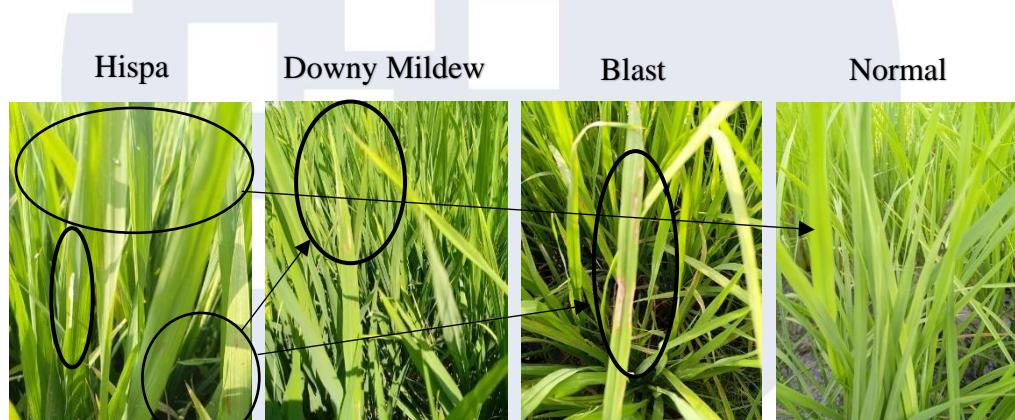
Table 10. TP Terburuk Pada Semua Percobaan Model

Confusion Matrix - Dataset UnderSampling		
Model	Ukuran	Terburuk
EfficientNetV2B0	224	Hispa
	256	Brown Spot
MobileNetV3Large	224	Brown Spot
	256	Downy Mildew
Concatenation Ensemble	224	Hispa
	256	Brown Spot
Average Ensemble	224	Downy Mildew dan Hispa
	256	Hispa
Confusion Matrix – Dataset OverSampling		
Model	Ukuran	Terburuk
EfficientNetV2B0	224	Bacterial Leaf Blight
	256	Brown Spot
MobileNetV3Large	224	Bacterial Panicle Blight dan Hispa
	256	Tungro
Concatenation Ensemble	224	Blast
	256	Bacterial Leaf Blight
Average Ensemble	224	Bacterial Leaf Blight
	256	Bacterial Leaf Blight

Dapat terlihat pada tabel diatas, terdapat perbedaan TP terburuk pada masing-masing penggunaan dataset. Untuk dataset teknik *undersampling*, TP paling rendah muncul pada empat percobaan penelitian yaitu Hispa yang disusul

Brown Spot. Sedangkan dataset teknik *oversampling*, TP paling rendah muncul pada empat percobaan penelitian yaitu Bacterial Leaf Blight.

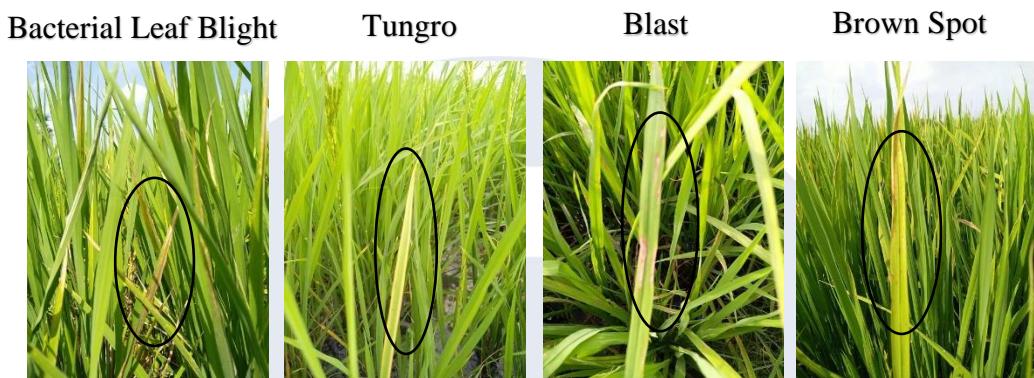
Dalam analisa kelas Hispa pada hasil confusion matrix dataset undersampling, terlihat bahwa FP (*False Positive*) mayoritas mengarah ke kelas Downy Mildew, Blast, dan Normal. Berikut merupakan salah satu gambar penyakit Hispa, Downy Mildew, Blast, dan Normal yang digunakan untuk data latih dataset teknik undersampling.



Gambar 4.24 Analisa Gambar Padi dengan TP Terburuk - *UnderSampling*

Terlihat pada Gambar 4.24, karakteristik dari penyakit tersebut memang hanya yang dilingkarkan penulis. Hispa layaknya goretan putih memanjang memang memiliki perbedaan terhadap Downy Mildew (bintik kuning diatas bagian daun), blast (bercak elips berwarna coklat kemerahan), maupun normal. Namun terlihat gambar tersebut memiliki background yang mirip yaitu terdapat daun padi lain yang normal dan mungkin punya juga sedikit bercak. Sehingga sangat dimungkinkan terdapat kesalahan prediksi pada kelas penyakit hispa.

Untuk analisa kelas Bacterial Leaf Blight sebagai kelas penyakit yang memiliki TP terendah pada dataset oversampling, terlihat bahwa FP (*False Positive*) mayoritas mengarah ke kelas tungro, blast, dan brown spot. Berikut merupakan salah satu gambar penyakit Bacterial Leaf Blight, Tungro, Blast, dan Brown Spot yang digunakan untuk data latih dataset teknik oversampling.



Gambar 4.25 Analisa Gambar Padi dengan TP Terburuk - *OverSampling*

Terlihat pada Gambar 4.25, karakteristik penyakit masing-masing kelas yang hanya dilingkarkan penulis. Jika dilihat dari kelas Bacterial Leaf Blight terlihat mirip dengan tungro sebuah bercak memanjang, tetapi memang letak perbedaan antara Bacterial Leaf Blight dan Tungro itu pada warna dan juga daun yang sedikit layu. Tungro memiliki warna lebih kuning dan sedikit layu. Jika dibandingkan juga dengan Blast dan Brown Spot, karakteristik warna memiliki kemiripan dengan Bacterial Leaf Blight. Sehingga hal ini memungkinkan terjadinya FP pada kelas Bacterial Leaf Blight.

Sebenarnya jika dianalisa kembali terjadi perbedaan TP terendah antara penggunaan dataset *undersampling* dan *oversampling*. Padahal seharusnya antara kedua dataset tersebut minimal Bacterial Leaf Blight terdeteksi sebagai TP terendah di dataset *undersampling* karena memang gambar yang digunakan sama hanya beda kuantitas. Ada beberapa faktor yang menjadi alasan terjadinya hal ini. Pertama diakibatkan oleh proses pengolahan jumlah dataset yang dimungkinkan terjadi perbedaan gambar antara data testing pada dataset teknik *undersampling* maupun *oversampling*. Kedua dapat dimungkinkan perbedaan jumlah gambar saat training, dataset *oversampling* menggunakan augmentasi, yang dapat mempengaruhi hasil *confusion matrix*. Ketiga diakibatkan oleh arsitektur *pre-trained*, terlihat baik EfficientNetV2B0 dan MobileNetV3Large mempunyai kelas TP terendah yang berbeda.

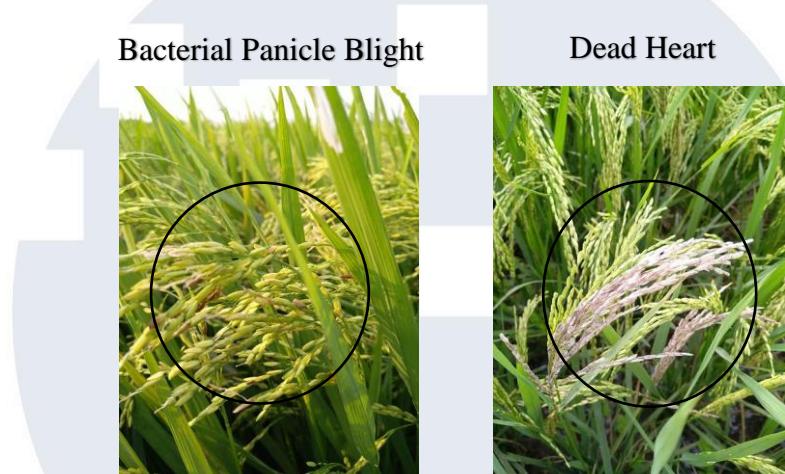
Dari hasil *Confusion Matrix* diatas, dapat dianalisa juga kelas yang memiliki TP terbaik pada setiap model percobaan. Untuk memudahkan analisa, berikut Tabel 11, ringkasan TP terbaik pada semua percobaan model.

Table 11. TP Terbaik Pada Semua Percobaan Model

Confusion Matrix - Dataset UnderSampling		
Model	Ukuran	Terbaik
EfficientNetV2B0	224	Bacterial Panicle Blight dan Dead Heart
	256	Bacterial Panicle Blight
MobileNetV3Large	224	Bacterial Leaf Streak, Bacterial Panicle Blight, dan Dead Heart
	256	Bacterial Panicle Blight
Concatenation Ensemble	224	Bacterial Leaf Streak
	256	Bacterial Leaf Streak
Average Ensemble	224	Dead Heart
	256	Bacterial Panicle Blight
Confusion Matrix – Dataset OverSampling		
Model	Ukuran	Terbaik
EfficientNetV2B0	224	Tungro
	256	Dead Heart dan Tungro
MobileNetV3Large	224	Dead Heart dan Tungro
	256	Hispa
Concatenation Ensemble	224	Bacterial Panicle Blight
	256	Tungro
Average Ensemble	224	Dead Heart
	256	Dead Heart dan Hispa

Terlihat pada Tabel 11, terdapat dua kelas yang mayoritas memiliki TP terbaik pada setiap percobaan model, yaitu Bacterial Panicle Blight dan Dead Heart. Alasan kedua kelas ini mempunyai TP terbaik pada mayoritas model

percobaan karena penyakit pada kedua kelas ini menyerang malai pada padi. Berbeda dengan penyakit lainnya yang menyerang daun pada tanaman padi. Berikut Gambar 4.26, kelas Bacterial Panicle Blight dan Dead Heart.

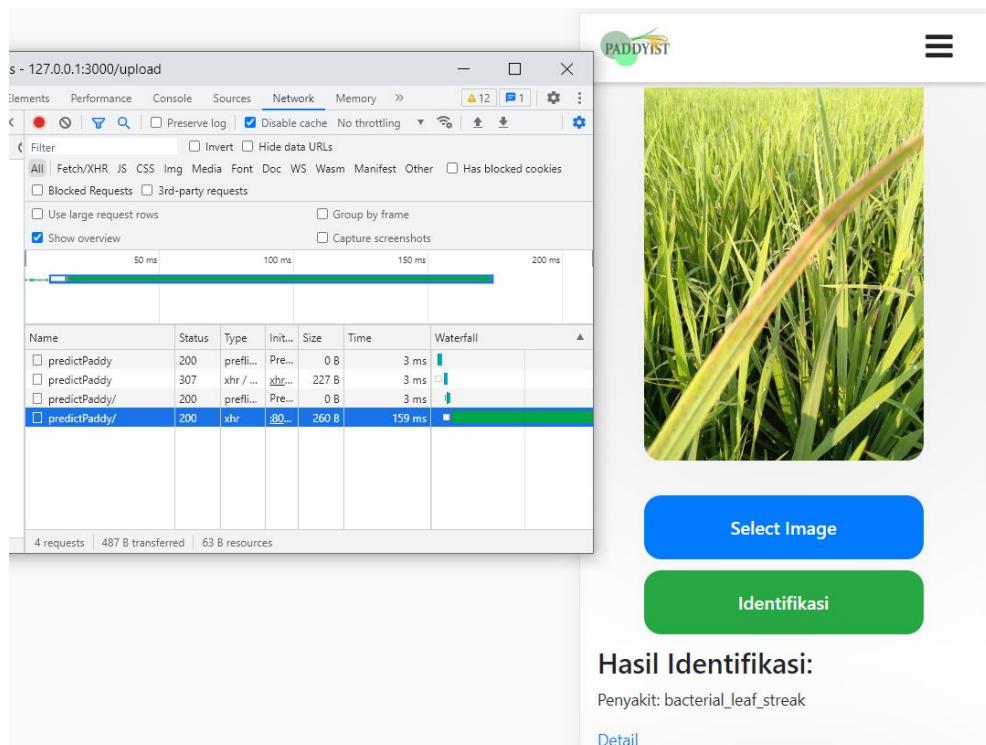


Gambar 4.26 Analisa Gambar Padi dengan TP Terbaik

Letak perbedaan kedua kelas tersebut adalah Bacterial Panicle Blight memiliki karakteristik warna ungu di sepertiga bagian bawah malai. Sedangkan Dead Heart memiliki karakteristik malai mengalami keputihan. Sehingga kedua kelas penyakit ini memiliki nilai *true positive* yang baik, karena berbeda dari kelas yang lainnya dan antar kelas memiliki karakteristik yang cukup dapat dibedakan. Jika kita kembali melihat hasil *confusion matrix*, *false positive* pada Bacterial Panicle Blight akan banyak mengarah ke Dead Heart dan begitu pun sebaliknya. Sehingga artinya kalaupun terjadi kesalahan prediksi pada kedua kelas ini, dia akan mengarah ke kelas Dead Heart atau Bacterial Panicle Blight.

4.6. Hasil dan Analisis Kecepatan Prediksi Model pada Sistem PWA

Hasil kecepatan prediksi model diambil dari halaman *network waterfall* pada sistem PWA yang diuji pada laptop secara lokal atau belum dideploy. Model akan diganti secara bergantian pada *backend* penulis. Berikut Gambar 4.27, cara penulis melakukan pengujian kecepatan prediksi model pada PWA.



Gambar 4.27 Cara Melihat Waktu Prediksi pada PWA – Network Waterfall

4.6.1 Hasil dan Analisa Kecepatan Prediksi Model - UnderSampling

Untuk pengujian tahap pertama, akan menguji model yang menggunakan dataset teknik undersampling. Setiap model akan diuji sebanyak 5 kali percobaan prediksi. Hasil yang didapat, dapat terlihat pada Tabel 12.

Table 12. Kecepatan Prediksi Model UnderSampling

Percobaan	Waktu Respons UnderSampling (ms)							
	EfficientNetV2B0		MobileNetV3 Large		Average Ensemble		Concatenation Ensemble	
	224	256	224	256	224	256	224	256
Tes 1	215	336	225	304	412	470	444	-
Tes 2	218	329	227	307	388	505	390	-
Tes 3	207	347	229	301	390	300	403	-
Tes 4	261	482	261	310	440	270	383	-
Tes 5	345	348	251	300	441	263	395	-
Rata-rata	249	368	239	304	414	362	403	-

Terlihat pada Tabel 12, pada model EfficientNetV2B0 dan MobileNetV3Large memiliki selisih waktu prediksi yang sedikit pada ukuran gambar 224x224 *pixel*. EfficientNetV2B0 memiliki waktu prediksi sedikit lebih lama daripada MobileNetV3Large dikarenakan parameter dan struktur model EfficientNetV2B0 yang lebih kompleks. Tentunya ini didukung juga dengan menggunakan ukuran gambar 256x256 *pixel*, EfficientNetV2B0 memiliki selisih waktu prediksi yang lebih banyak.

Hal lain yang dapat dianalisa adalah selisih waktu prediksi antara model *ensemble* dan model *pre-trained* yang memiliki selisih cukup jauh. Hal ini dapat terlihat pada model ensemble baik *average* maupun *concatenation* pada ukuran gambar 224x224 *pixel*, memilih selisih cukup jauh dari dua model *pre-trained*. Hal ini dikarenakan kompleksitas arsitektur model *ensemble* lebih tinggi dibandingkan dua model *pre-trained* tersebut. Namun terdapat ketidaksesuaian pada hasil waktu prediksi *average ensemble* dengan ukuran gambar 256x256 *pixel*. Terlihat waktu prediksinya mendekati waktu prediksi dan lebih cepat daripada EfficientNetV2B0 (256x256 *pixel*). Sehingga analisa yang dilakukan sebelumnya belum dapat dipastikan kesesuaiaannya 100%.

4.6.2 Hasil dan Analisa Kecepatan Prediksi Model - OverSampling

Untuk pengujian tahap kedua, akan menguji model yang menggunakan dataset teknik oversampling. Setiap model akan diuji sebanyak 5 kali percobaan prediksi. Hasil yang didapat dapat terlihat pada Tabel 13.

Table 13. Kecepatan Prediksi Model OverSampling

Percobaan	Waktu Respons Dataset OverSampling (ms)							
	EfficientNetV2B0		MobileNetV3-Large		Average Ensemble		Concatenation Ensemble	
	224	256	224	256	224	256	224	256
Tes 1	268	350	190	256	438	383	345	-
Tes 2	180	369	245	229	382	382	361	-
Tes 3	192	341	200	233	412	398	348	-
Tes 4	214	495	206	220	420	423	377	-

Tes 5	266	350	201	223	395	402	444	-
Rata-rata	224	381	208	232	409	398	375	-

Terlihat pada Tabel 13, memiliki hasil yang berkesesuaian dengan saat kita menggunakan dataset teknik *undersampling*. Ukuran gambar 256x256 *pixel* pada dua *pre-trained* model memiliki waktu prediksi lebih lama dibandingkan ukuran gambar 224x224 *pixel*. *Ensemble* model juga memiliki selisih waktu prediksi lebih lama dibandingkan kedua *pre-trained* model. Sehingga dapat diberikan analisa bahwa perbedaan penggunaan dataset teknik *undersampling* maupun *oversampling* tidak mempengaruhi waktu prediksi.

4.6.3 Analisa Hubungan Waktu Prediksi dan Ukuran File Model

Hasil waktu prediksi yang telah didapat, terlihat bahwa *concatenation ensemble* dengan ukuran gambar 256x256 *pixel* tidak tercantum hasil waktu prediksi. Penyebabnya adalah kemampuan laptop penulis yang tidak mampu menjalankan model tersebut pada FastAPI.

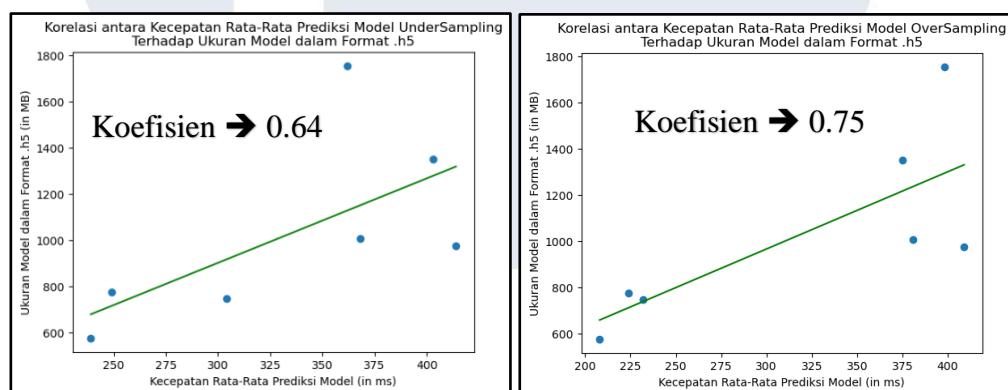
Namun, dengan adanya permasalahan ini, penulis mendapatkan hal lain yang dapat dianalisa yaitu keterkaitan antara waktu prediksi yang didapat dengan ukuran model yang telah dilatih (file .h5). Berikut gambar berbagai model yang telah penulis latih dan penulis gunakan untuk mendapatkan prediksi waktu yang dapat terlihat pada Gambar 4.28.

Name	Date modified	Type	Size
PaddyDisease_EfficientNetV2B0_OverSampling_224.h5	4/20/2023 7:58 PM	H5 File	776,635 KB
PaddyDisease_EfficientNetV2B0_OverSampling_256.h5	4/13/2023 10:54 AM	H5 File	1,007,036 KB
PaddyDisease_EfficientNetV2B0_UnderSampling_224.h5	4/20/2023 5:58 PM	H5 File	776,635 KB
PaddyDisease_EfficientNetV2B0_UnderSampling_256.h5	4/13/2023 6:38 AM	H5 File	1,007,036 KB
PaddyDisease_EnsembleAverage_OverSampling_224.h5	4/24/2023 12:23 PM	H5 File	976,584 KB
PaddyDisease_EnsembleAverage_OverSampling_256.h5	4/20/2023 8:08 PM	H5 File	1,756,198 KB
PaddyDisease_EnsembleAverage_UnderSampling_224.h5	4/24/2023 12:27 PM	H5 File	976,584 KB
PaddyDisease_EnsembleAverage_UnderSampling_256.h5	4/20/2023 8:11 PM	H5 File	1,756,198 KB
PaddyDisease_EnsembleConcatenation_OverSampling_224.h5	4/24/2023 10:18 AM	H5 File	1,352,869 KB
PaddyDisease_EnsembleConcatenation_OverSampling_256.h5	4/24/2023 11:36 AM	H5 File	1,756,070 KB
PaddyDisease_EnsembleConcatenation_UnderSampling_224.h5	4/24/2023 10:27 AM	H5 File	1,352,869 KB
PaddyDisease_EnsembleConcatenation_UnderSampling_256.h5	4/24/2023 11:46 AM	H5 File	1,756,069 KB
PaddyDisease_MobileNetV3Large_OverSampling_224.h5	4/20/2023 9:03 PM	H5 File	577,118 KB
PaddyDisease_MobileNetV3Large_OverSampling_256.h5	4/13/2023 8:17 AM	H5 File	749,922 KB
PaddyDisease_MobileNetV3Large_UnderSampling_224.h5	4/20/2023 6:17 PM	H5 File	577,118 KB
PaddyDisease_MobileNetV3Large_UnderSampling_256.h5	4/13/2023 7:19 AM	H5 File	749,922 KB

Gambar 4.28 Ukuran File Model-Model Latih Penelitian

Dapat terlihat pada Gambar 4.28, model *concatenation ensemble* dengan ukuran gambar 256x256 pixel, memiliki ukuran file yang sangat besar yaitu 1.75 GB. Ini menjadi salah satu faktor yang membuat penulis tidak mampu menjalankan model tersebut pada laptop penulis untuk proses prediksi.

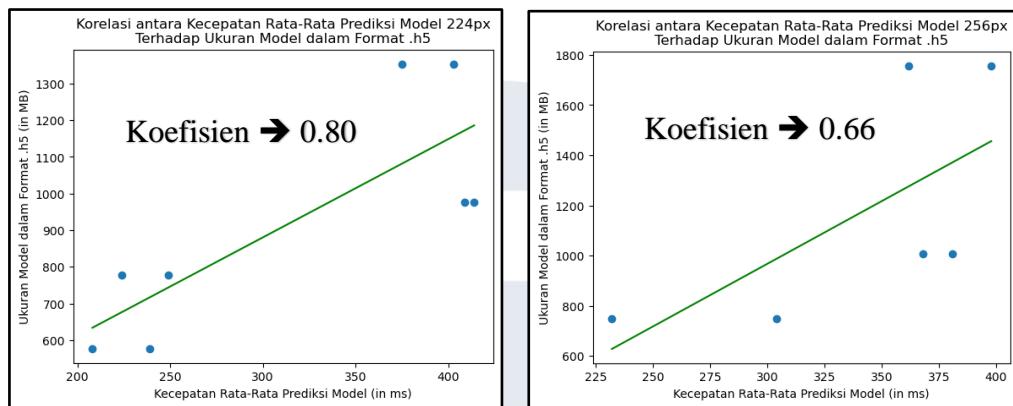
Untuk hubungan antara waktu prediksi dan ukuran file berformat .h5, penulis menggunakan metode korelasi. Variabel yang digunakan berjumlah dua variabel yaitu kecepatan rata-rata prediksi dan ukuran file berformat .h5 dari data yang telah dikumpulkan sebelumnya. Penulis akan menganalisa korelasi dengan berbagai parameter yaitu perbedaan teknik *balancing*, ukuran gambar, serta keseluruhan korelasi uji model. Berikut grafik korelasi antara kedua teknik *balancing* yang dapat terlihat pada Gambar 4.29.



Gambar 4.29 Grafik Korelasi antara Kecepatan Rata-Rata Prediksi Model Parameter Teknik *Balancing* terhadap Ukuran Model dalam Format .h5

Terlihat ketika menggunakan model yang menggunakan teknik *undersampling* memiliki koefisien lebih rendah yaitu 0.64 (kategori *moderate positive linear relationship*) daripada *oversampling* yaitu 0.75 (kategori *strong positive linear relationship*). Hal ini terjadi karena selisih kecepatan rata-rata antara EfficientNetV2B0 dan Average Ensemble lebih berelasi pada dataset teknik *oversampling* dibandingkan dataset teknik *undersampling*.

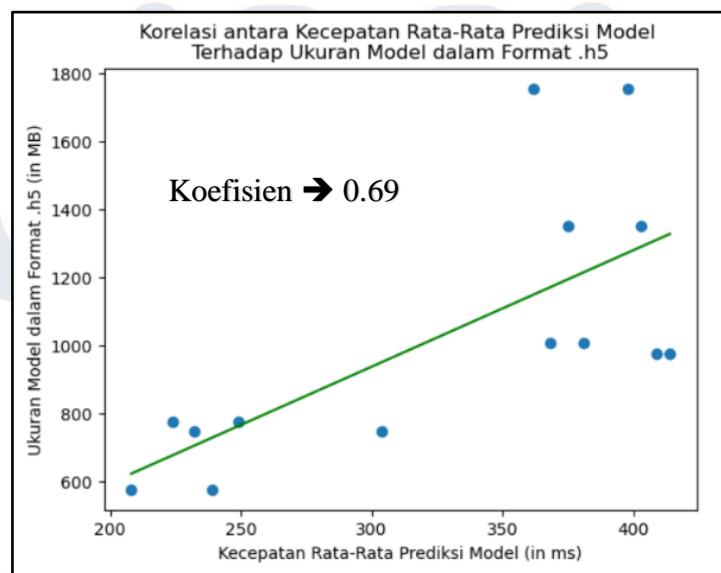
Selain korelasi dengan parameter teknik *balancing* dataset, berikut grafik korelasi antara kedua ukuran gambar yang berbeda yang dapat terlihat pada Gambar 4.30.



Gambar 4.30 Grafik Korelasi antara Kecepatan Rata-Rata Prediksi Model Parameter Ukuran Gambar terhadap Ukuran Model dalam Format .h5

Terlihat ketika menggunakan model yang menggunakan ukuran gambar 224x224 piksel memiliki koefisien lebih tinggi yaitu 0.80 (kategori *strong positive linear relationship*) daripada ukuran gambar 256x256 piksel yaitu 0.66 (kategori *moderate positive linear relationship*). Hal ini terjadi karena ketika ukuran gambar 256x256 piksel pada model *average ensemble* memiliki kecepatan rata-rata prediksi lebih cepat dibandingkan 224x224 piksel yang tidak selaras dengan ukuran file format .h5.

Berikut grafik korelasi semua uji model yang telah dibuat yang dapat terlihat pada Gambar 4.31.



Gambar 4.31 Grafik Korelasi antara Kecepatan Rata-Rata Prediksi Model terhadap Ukuran Model dalam Format .h5

Koefisien korelasi yang didapat adalah 0.696 yang terlihat pada grafik korelasi diatas, garis hijau tidak diagonal sempurna tapi sedikit menurun. Berdasarkan penelitian yang dilakukan oleh Bruce Ratner mengenai korelasi koefisien [39], nilai 0.696 dapat dikategorikan sebagai *moderate positive linear relationship*. Positif disini berarti bahwa kedua variabel berjalan ke arah yang sama. Ketika variabel kecepatan rata-rata naik, variabel ukuran file juga cenderung meningkat. Hal ini dapat terlihat pada model MobileNetV3-Large ukuran gambar 224x224 *pixel* yang memiliki ukuran 577 MB memiliki waktu prediksi rata-rata 224 ms. Sedangkan untuk model *average ensemble* ukuran gambar 224x224 *pixel* memiliki ukuran file 976 MB dengan waktu prediksi rata-rata 409 ms. *Moderate* dapat diartikan bahwa relasi antara kedua variabel tidak terlalu kuat tapi juga tidak terlalu lemah. Hal ini dapat terlihat pada model *average ensemble* dengan ukuran gambar berbeda, memiliki perbedaan yang sangat jauh dari segi ukuran file, tetapi kecepatan rata-rata prediksinya hampir sama bahkan ukuran gambar 256x256 *pixel* cenderung lebih cepat. Hal itu yang membuat hasil koefisien korelasi tidak terlalu kuat atau *moderate*.



BAB V

SIMPULAN DAN SARAN

5.1 Simpulan

Dari penelitian yang telah dilakukan setelah menggunakan berbagai model, dataset, dan ukuran gambar, *average ensemble* dengan dataset teknik *oversampling* dan penggunaan ukuran gambar 256x256 *pixel* memiliki hasil akurasi testing paling tinggi yaitu 93.3%. Hasil ini lebih baik daripada penggunaan *concatenation ensemble* dan *pre-trained* model pada penelitian yang telah dilakukan penulis. Jika membandingkan dengan penelitian terdahulu, berhasil melebihi performa dari penggunaan model VGG16 dengan dataset yang berjumlah 16.225 sampel. Sehingga dapat dikatakan penggunaan metode *ensemble* sangatlah membantu untuk meningkatkan performa dengan cara yang lebih efisien.

Perbedaan penggunaan jumlah sampel gambar pada penelitian ini terbukti memiliki pengaruh yang cukup signifikan pada performa yang didapat. Proses augmentasi dilakukan untuk mendapatkan dataset teknik *oversampling* dengan setiap kelas berjumlah 1.000 gambar. Konfigurasi augmentasi yang dilakukan meliputi *brightness*, *blur*, *zoom*, dan *flip*. Tentunya konfigurasi augmentasi yang digunakan menyesuaikan dari penelitian yang ada bahwa *brightness* dan *blur* menghasilkan performa terbaik dan menyesuaikan kondisi lapangan ketika petani mengambil gambar padi. Dengan penggunaan dataset *oversampling* (10.000 sampel data latih), membuat keempat model yang digunakan berhasil melampaui performa yang didapat ketika menggunakan dataset teknik *undersampling*. Hasil grafik pembelajaran yang didapat juga menunjukkan sedikit terjadinya penurunan *overfitting* pada akurasi *training*.

Selain itu, perbedaan penggunaan ukuran gambar juga mempengaruhi hasil yang didapat. Pada teorinya, model EfficientNetV2B0 dan MobileNetV3-Large memiliki ukuran gambar bawaan yaitu 224x224 *pixel* yang dimungkinkan dapat memberikan performa terbaik. Hasil yang didapat saat menggunakan dataset teknik *undersampling*, ukuran gambar 256x256 *pixel* memberikan performa lebih baik

dikarenakan penggunaan dataset yang sedikit, memberikan informasi sedikit, tetapi didukung oleh ukuran gambar yang lebih besar. Hasil yang didapat saat menggunakan dataset teknik *oversampling*, ukuran gambar 224x224 *pixel* memberikan performa lebih baik dikarenakan kompatibilitas model *pre-trained* yang dilatih menggunakan ImageNet ukuran gambar 224x224 *pixel*. Namun, jika menggunakan ukuran yang lebih tinggi dari 256x256 *pixel*, contohnya 512x512 *pixel* seperti pada penelitian yang dilakukan oleh V. Tahmbawita [15] dimungkinkan dapat meningkatkan performa.

Karakteristik gambar tanaman padi yang digunakan juga mempengaruhi hasil performa yang didapat. Pada analisa karakteristik gambar melalui hasil *confusion matrix* yang telah dijelaskan, terlihat bahwa terdapat gambar yang memiliki karakteristik sendiri dan juga memiliki karakteristik yang dimiliki kelas lainnya. Hal ini terlihat pada penyakit Hispa yang mempunyai *true positive* terburuk pada dataset teknik *undersampling*, gambar tersebut juga memiliki karakteristik terhadap kelas blast, downy mildew, dan juga normal. Hal ini disebabkan karena gambar yang digunakan memiliki background atau bukan gambar sehelai daun yang memiliki satu karakteristik kelas. Namun dengan menggunakan *pre-trained* model dan metode *ensemble*, hasil yang didapat sudah cukup baik.

Kecepatan prediksi setiap model dengan ukuran dan dataset teknik *balancing* yang berbeda ketika diimplementasi pada sistem PWA, juga memiliki waktu prediksi yang berbeda-beda. Perbedaan ukuran gambar mempengaruhi waktu prediksi tetapi tidak terlalu signifikan. Selisih yang didapat berkisar antar 30 – 180 mili detik dari hasil waktu prediksi yang telah dirata-rata. Namun perbedaan yang cukup signifikan adalah perbedaan waktu prediksi antara penggunaan *pre-trained* model dan metode *ensemble*. Dapat terlihat bahwa selisih yang didapat berkisar antara 150 – 200 mili detik. Untuk perbedaan waktu prediksi antara menggunakan dataset teknik *undersampling* maupun *oversampling* tidak dapat disimpulkan. Hal ini dikarenakan hasil yang didapat tidak stabil. Sehingga dapat disimpulkan bahwa kuantitas dataset yang dilatih tidak mempengaruhi waktu prediksi suatu model. Meskipun sebenarnya waktu prediksi dari sistem PWA yang

dihadirkan tidaklah terlalu mempengaruhi pengguna, tetapi bisa digunakan untuk penelitian maupun riset lainnya.

Poin lainnya yang dapat terlihat pada penelitian ini adalah korelasi antara kecepatan rata-rata waktu prediksi dan ukuran file model format .h5. Dengan hasil korelasi *moderate positive linear relationship*, yang artinya terdapat hubungan yang positif antar kedua variabel tersebut, tetapi tidak terlalu kuat maupun terlalu lemah. Selain itu tidak adanya waktu prediksi pada *concatenation ensemble* dengan ukuran gambar 256x256 px, dikarenakan keterbatasan *hardware* yang digunakan oleh penulis. Ini juga menjadi dasar penulis hanya menggunakan dua ukuran gambar yaitu 224x224 *pixel* dan 256x256 *pixel*.

5.2 Saran

Berdasarkan penelitian yang telah dilakukan penulis, terdapat beberapa saran yang dapat penulis sampaikan, yaitu:

1. Dapat mencoba menggunakan berbagai konfigurasi augmentasi lainnya yang mungkin dapat memberikan hasil akurasi yang lebih baik dari yang penulis gunakan pada penelitian ini.
2. Dapat mencoba menggunakan model yang memiliki layer lebih banyak daripada EfficientNetV2B0 atau MobileNetV3-Large agar bisa dibandingkan baik dari sisi performa akurasi dan waktu prediksi dengan model *ensemble* hasil gabungan dua *pre-trained* model yang memiliki layer lebih sedikit.
3. Mencoba untuk menambahkan dataset baik primer maupun sekunder ke dataset yang penulis gunakan agar mendapatkan hasil model yang dapat digunakan untuk semua kemungkinan karakteristik gambar padi yang berbeda-beda.
4. Mencoba menyelesaikan sistem PWA yang bisa langsung dipakai oleh masyarakat terutama petani. Sehingga tidak hanya sekedar meneliti tapi mampu membuat suatu sistem siap pakai.
5. Melakukan validasi data uji dengan petani maupun *experts* yang ahli dalam bidang pertanian.

DAFTAR PUSTAKA

- [1] M. Shahbandeh, “Rice Consumption Worldwide Leading Countries 2022/2023,” Statista, Februari 2023. [Online]. Tersedia: <https://www.statista.com/statistics/255971/top-countries-based-on-rice-consumption-2012-2013/>
- [2] L. J. Sembiring, “Produksi Beras RI Turun, Ini Biang Keroknya”, CNBC Indonesia, 1 Maret 2022. [Online]. Tersedia: <https://www.cnbcindonesia.com/news/20220301115731-4-319184/duh-produksi-beras-ri-turun-ini-biang-keroknya>
- [3] Administrator, “Hama Penyakit Bakteri Pada Tanaman Padi,” Pertanian Kabupaten Ngawi, 12 January 2023. [Online]. Tersedia: <https://pertanian.ngawikab.go.id/2023/01/12/hama-penyakit-bakteri-pada-tanaman-padi/>
- [4] A. E. Asibi, Q. Chai, J. A. Coulter., “Rice Blast: A Disease with Implications for Global Food Security”, MDPI, Vol.9, No.8, 15 August 2019. doi: <https://doi.org/10.3390/agronomy9080451>
- [5] “Rice Knowledge Bank Pests and Disease”, Knowledge Bank IRRI. [Online]. Tersedia: <http://www.knowledgebank.irri.org/step-by-step-production/growth/pests-and-diseases/diseases>
- [6] Petchiammal, B. Kiruba, D. Murugan, Pandarasamy., “Paddy Doctor: A Visual Image Dataset for Paddy Disease Classification”, Arxiv, 23 May 2022. doi: <https://arxiv.org/pdf/2205.11108v1.pdf>
- [7] R. Manavalan, “Automatic Identification of Diseases in Grains Crops through Computational Approaches: A Review”, Elsevier, 7 Oktober 2020. doi: <https://doi.org/10.1016/j.compag.2020.105802>
- [8] F. H. Hawari, et al. “Klasifikasi Penyakit Padi Menggunakan Algoritma CNN (Convolutional Neural Network)”, Jurnal Responsif, Vol.4, No.2, 2 Agustus 2022. doi: <https://ejurnal.ars.ac.id/index.php/jti/article/view/856/580>
- [9] *Paddy Doctor: Paddy Disease Classification*: Kaggle, 2022. Tersedia: <https://www.kaggle.com/competitions/paddy-disease-classification/overview>

- [10] A. Howard, et al. “*Searching for MobileNetV3*”, Arxiv, 20 November 2019. [Online]. Tersedia: <https://arxiv.org/pdf/1905.02244.pdf>
- [11] M. Tan, Q. V. Le., “*EfficientNetV2: Smaller Models and Faster Training*”, Arxiv, 23 June 2021. [Online]. Tersedia: <https://arxiv.org/pdf/2104.00298.pdf>
- [12] A. R. Wang, N. H. Shabrina., “*A Deep Learning-Based Mobile App System for Visual Identification of Tomato Plant Disease*”, IJECE, Vol.99, No.1, 2022. [Offline].
- [13] E. Anggiratih, et al. “Klasifikasi Penyakit Tanaman Padi Menggunakan Model Deep Learning EfficientNetB3 dengan Transfer Learning,” JIS, Vol.19, No.1, Januari 2021. doi: <https://doi.org/10.30646/sinus.v19i1.526>
- [14] O. V. Putra, et al. “*HiT-LIDIA: A Framework for Rice Leaf Disease Classification using Ensemble and Hierarchical Transfer Learning*”, Lontar Komputer, Vol.13, No.3, Desember 2022. Tersedia: <https://ojs.unud.ac.id/index.php/lontar/article/download/84026/47439>
- [15] V. Thambawita, et al. “*Impact of Image Resolution on Deep Learning Performance in Endoscopy Image Classification: An Experimental Study Using a Large Dataset of Endoscopic Images*”, Diagnostic (Basel), 24 November 2021. Tersedia: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8700246/#:~:text=In%20general%2C%20the%20resolutions%20for,64%20and%20256%20%C3%97%20256.>
- [16] L. Alzubaidi, et al. “Review of Deep Learning: Concepts, CNN Architectures, Challenges, Applications, Future Directions”, Springer, Vol.8, No.53, 31 Maret 2021, doi: <https://doi.org/10.1186/s40537-021-00444-8>
- [17] “*Deep Learning*”, IBM. [Online]. Tersedia: <https://www.ibm.com/id-en/topics/deep-learning>
- [18] Q. Lina, “Apa itu *Convolutional Neural Network?*”, Medium. [Online]. Tersedia: <https://medium.com/@16611110/apa-itu-convolutional-neural-network-836f70b193a4>
- [19] A. G. Howard, et al. “*MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications*”, Arxiv, 17 April 2017. [Online]. doi:

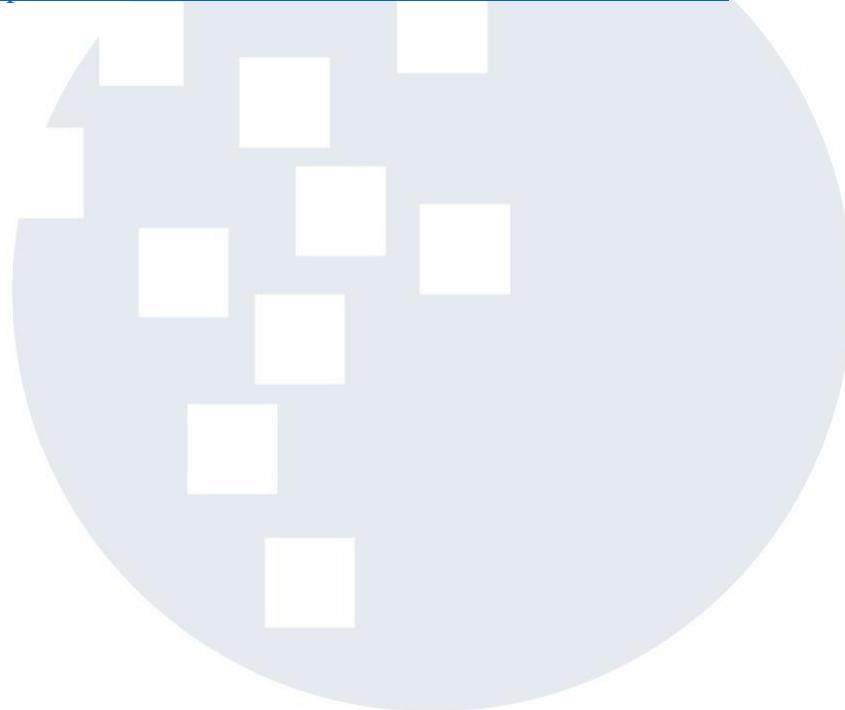
<https://arxiv.org/pdf/1704.04861.pdf>

- [20] M. Sandler, et al. “*MobileNetV2: Inverted Residuals and Linear Bottlenecks*”, Arxiv, 21 Maret 2019. [Online]. doi: <https://arxiv.org/pdf/1801.04381.pdf>
- [21] M. Tan, Q. V. Le, “*EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks*”, Arxiv, 11 September 2020. [Online]. doi: <https://arxiv.org/pdf/1905.11946.pdf>
- [22] A. Devasia, “*Combining Two Deep Learning Models*”, Control Automation, 2 December 2021. [Online]. Tersedia: <https://control.com/technical-articles/combining-two-deep-learning-models>
- [23] H. Tripathi, “*What is Balanced and Imbalanced Dataset?*”, Medium, 25 September 2019. [Online]. Tersedia: <https://medium.com/analytics-vidhya/what-is-balance-and-imbalance-dataset-89e8d7f46bc5>
- [24] “*Data Augmentation*”, Tensorflow, 27 Mei 2023. [Online]. Tersedia: [https://www.tensorflow.org/tutorials/images/data augmentation](https://www.tensorflow.org/tutorials/images/data_augmentation)
- [25] A. Barguzar, “*Python Flask versus FastAPI: Which Should You Choose?*”, Netguru, 21 April 2022. [Online]. Tersedia: <https://www.netguru.com/blog/python-flask-versus-fastapi>
- [26] Admin, “Inilah Review NuxtJS yang Wajib Anda Ketahui Sebagai Developer”, Codedelapan, 24 Januari 2022. [Online]. Tersedia: <https://codelapan.com/post/inilah-review-nuxtjs-yang-wajib-anda-ketahui-sebagai-developer>
- [27] S. Kurniawan, “Mengenal PWA – Progressive Web App untuk Website Lebih Cepat”, Niagahoster, 5 December 2022. [Online]. Tersedia: <https://www.niagahoster.co.id/blog/progressive-web-app/>
- [28] S. N. Aeni, “Ketahui, Ini Karakteristik Tanaman Padi, dari Akar sampai Buah”, Kompas, 22 Agustus 2022. [Online]. Tersedia: <https://agri.kompas.com/read/2022/08/22/131200284/ketahui-ini-karakteristik-tanaman-padi-dari-akar-sampai-buah?page=all>

- [29] “Downy Mildew”, Planet Natural Research Center. [Online]. Tersedia: <https://www.planetnatural.com/pest-problem-solver/plant-disease/downy-mildew/>
- [30] “Rice Knowledge Bank Pests Management – Rice Hispa”, Knowledge Bank IRRI. [Online]. Tersedia: <http://www.knowledgebank.irri.org/training/fact-sheets/pest-management/insects/item/rice-hispa>
- [31] “Rice Knowledge Bank Pests Management – Stem Borer”, Knowledge Bank IRRI. [Online]. Tersedia: <http://www.knowledgebank.irri.org/training/fact-sheets/pest-management/insects/item/stem-borer>
- [32] D. Groth, C. Hollier., “Bacterial Panicle Blight of Rice,” Isuagcenter. [Online]. Tersedia: <https://www.lsuagcenter.com/~/media/system/9/1/f/d/91fd6a36273fc8be55db2cfe7b9e1b53/pub3106bacterialpanicleblightlowres.pdf>
- [33] T. Iqball, M. A. Wani, “Weighted Ensemble Model for Image Classification”, Springer, Vol.15, No.2, 557-564. doi: <https://doi.org/10.1007/s41870-022-01149-8>
- [34] O. E. Gannour, et al. “Concatenation of Pre-Trained Convolutional Neural Networks for Enhanced COVID-19 Screening Using Transfer Learning Technique”, MDPI, Vol.11, No.1. doi: <https://doi.org/10.3390/electronics11010103>
- [35] L. Alzubaidi, et al. “Review of Deep Learning: Concepts, CNN Architectures, Challenges, Applications, Future Directions”, Springer. Vol.8, No.53, 31 Maret 2021. doi: <https://doi.org/10.1186/s40537-021-00444-8>
- [36] “Keras Applications”, Keras, [Online]. Tersedia: <https://keras.io/api/applications/>
- [37] M. S. Anggreany, “Confusion Matrix”, Binus, 1 November 2020, [Online]. Tersedia: <https://socbs.binus.ac.id/2020/11/01/confusion-matrix/>
- [38] “Paddy Doctor - Code”, Paddy Doctor. [Online]. Tersedia: <https://paddydoc.github.io/code/>
- [39] B. Ratner, “The Correlation Coefficient: Its Values Range Between +1/-1, or Do They?”, Springer, Vol.17, 18 Mei 2009. doi:

<https://link.springer.com/content/pdf/10.1057/jt.2009.5.pdf?pdf=button>

- [40] S. Candrawardhani, “Koefisien Korelasi: Pengertian, Rumus, Contoh, dan Cara Menghitungnya”, KitaLulus, 11 September 2022. [Online]. Tersedia: <https://www.kitalulus.com/bisnis/koefisien-korelasi-adalah>



UMN
UNIVERSITAS
MULTIMEDIA
NUSANTARA

LAMPIRAN

Lampiran A. Hasil Pengecekan Turnitin

Skripsi - Richard Alvin Pratama		
ORIGINALITY REPORT		
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS
11 %	8 %	6 %
STUDENT PAPERS		
3 %		
PRIMARY SOURCES		
1 Moch. Kholid, Heri Priya Waspada, Rafika Akhsani. "Klasifikasi Penyakit Infeksi Pada Ayam Berdasarkan Gambar Feses Menggunakan Convolutional Neural Network", SINTECH (Science and Information Technology) Journal, 2022 Publication	3 %	
2 kc.umn.ac.id Internet Source	1 %	
3 repository.ub.ac.id Internet Source	<1 %	
4 jurnal.stkipgritulungagung.ac.id Internet Source	<1 %	
5 repository.its.ac.id Internet Source	<1 %	
6 pubmed.ncbi.nlm.nih.gov Internet Source	<1 %	
7 kse.or.id Internet Source	<1 %	
8 www.mdpi.com Internet Source	<1 %	
9 www.researchgate.net Internet Source	<1 %	
10 pt.scribd.com Internet Source	<1 %	
11 123dok.com Internet Source	<1 %	
12 id.scribd.com Internet Source	<1 %	
13 eprints.sinus.ac.id Internet Source	<1 %	
14 Abdullah Ali Salamai, Nouran Ajabnoor, Waleed E. Khalid, Mohammed Maqsood Ali, Abdulaziz Ali Murayr. "Lesion-aware visual transformer network for Paddy diseases detection in precision agriculture", European Journal of Agronomy, 2023 Publication	<1 %	
15 www.slideshare.net Internet Source	<1 %	
16 Submitted to Ajou University Graduate School Student Paper	<1 %	
17 Submitted to Universitas Negeri Jakarta Student Paper	<1 %	
18 docplayer.info Internet Source	<1 %	
19 library.binus.ac.id Internet Source	<1 %	
20 www.diary-statistika.com Internet Source	<1 %	
21 deepai.org Internet Source	<1 %	
22 Submitted to Universitas Merdeka Malang Student Paper	<1 %	
23 adoc.pub Internet Source	<1 %	
24 es.scribd.com Internet Source	<1 %	
25 ojs.uajy.ac.id Internet Source	<1 %	
26 1001ilmupertanian.wordpress.com Internet Source	<1 %	
27 repository.uin-suska.ac.id Internet Source	<1 %	
28 media.neliti.com Internet Source	<1 %	
29 moonlight314.github.io Internet Source	<1 %	
30 Submitted to Sriwijaya University Student Paper	<1 %	
31 fr.scribd.com Internet Source	<1 %	
32 Hai Thanh Nguyen, Quyen Thuc Quach, Chi Le Hoang Tran, Huong Hoang Luong. "Chapter 66 Deep Learning Architectures Extended from Transfer Learning for Classification of Rice Leaf Diseases", Springer Science and Business Media LLC, 2022 Publication	<1 %	
33 Submitted to Universiti Malaysia Pahang Student Paper	<1 %	
34 ejournal.poltekgal.ac.id Internet Source	<1 %	
35 jifosi.upnjatim.ac.id Internet Source	<1 %	
36 jurnal.akba.ac.id Internet Source	<1 %	
37 Submitted to Universitas Sultan Ageng Tirtayasa Student Paper	<1 %	
38 repository.uph.edu Internet Source	<1 %	

NUSANTARA

A

Implementasi Ensemble Deep Learning Untuk Klasifikasi Penyakit Pada Tanaman Padi, Richard Alvin Pratama, Universitas Multimedia Nusantara

		Learning Efficientnet B3 dengan transfer Learning", Jurnal Ilmiah SINUS, 2021
39	Kurniawan Irfan Nauval, Sri Lestari. "Implementasi Deteksi Objek Penyakit Daun Kentang dengan Metode Convolutional Neural Network", Jurnal Aplikasi Teknologi Informasi dan Manajemen (JATIM), 2022	Publication <1 %
40	Submitted to Universitas Brawijaya	<1 %
41	Student Paper lib.unnes.ac.id	<1 %
42	Internet Source repository.unwira.ac.id	<1 %
43	repository.um-palembang.ac.id	<1 %
44	Internet Source www.gubukpintar.com	<1 %
45	A Irwanto, B Prasetyo. "High capacity image steganography using LSB with modified binary addition on RGB indicator", Journal of Physics: Conference Series, 2021	Publication <1 %
46	Submitted to Tarumanagara University	<1 %
47	Student Paper eprints.radenfatah.ac.id	<1 %
48	Internet Source jurnal.polgan.ac.id	<1 %
49	klikgss.com	<1 %
50	Internet Source repozitorij.unizg.hr	<1 %
51	Internet Source arxiv-export1.library.cornell.edu	<1 %
52	begawe.unram.ac.id	<1 %
53	Internet Source dspace.uii.ac.id	<1 %
54	garuda.kemdikbud.go.id	<1 %
55	Internet Source lib.uia.ac.id	<1 %
56	proceeding.unpkediri.ac.id	<1 %
57	Internet Source www.marketeers.com	<1 %
58	Endang Anggiratih, Sri Siswanti, Saly Kurnia Octaviani, Arum Sari. "Klasifikasi Penyakit Tanaman Padi Menggunakan Model Deep	<1 %
67	id.123dok.com	<1 %
68	Internet Source journal.uin-alauddin.ac.id	<1 %
69	jurnal.unimed.ac.id	<1 %
70	jurnalnasional.ump.ac.id	<1 %
71	koran,tempo.co	<1 %
72	moam.info	<1 %
73	repositori.buddhidharma.ac.id	<1 %
74	text-id.123dok.com	<1 %
75	www.binus.ac.id	<1 %
76	www.doria.fi	<1 %
77	www.scribd.com	<1 %
78	www.techradar.com	<1 %
79	Internet Source widuri.raharja.info	<1 %

Exclude quotes On
Exclude bibliography On

Exclude matches < 7 words

Lampiran B. Formulir Konsultasi Skripsi

Lampiran B. Formulir Konsultasi Skripsi

FORMULIR KONSULTASI SKRIPSI – FAKULTAS TEKNIK & INFORMATIKA

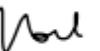
Dosen Pembimbing : Nabilah Husna Shabrina, S.T., M.T.
Jurusan : Teknik Komputer
Semester : Delapan (8)
Nama : Richard Alvin Pratama
NIM : 00000034813



Tanggal Konsultasi	Agenda/Pokok Bahasan	Saran Perbaikan	Paraf Dosen Pembimbing
04 / 02 / 2023	Konsultasi Pemilihan Topik	Disarankan untuk membaca dataset dan juga paper kakak kelas yang mirip dengan topik yang akan diteliti	
07 / 02 / 2023	Konsultasi Referensi dan Dataset	Dicoba tambah data primer	
09 / 02 / 2023	Permasalahan dalam Pembagian Dataset	Data untuk testing dapat dibantu ahli untuk validasi penyakit padinya	
11 / 02 / 2023	Penggunaan Aplikasi PWA	Melihat contoh aplikasi sejenis yang sudah ada dan mencari beberapa jawaban atas pertanyaan menyangkut PWA yang mungkin ditanya saat sidang	
16 / 02 / 2023	Bimbingan Progres Proposal	Beberapa perbaikan pada latar belakang dan identifikasi masalah, serta persimpel flowchart pada Bab III	
09 / 03 / 2023	Bimbingan Revisi Proposal	Diberikan referensi untuk menjawab pertanyaan "demand" pada hasil revisi proposal	
30 / 03 / 2023	Analisis Hasil Training	Coba untuk menggunakan augmentasi pada dataset yang tidak balance	
18 / 04 / 2023	Progress Training dan PWA	Bikin excel perbandingan, cek size model yang terlalu besar, dan ulang train pada ensemble model untuk mendapatkan akurasi lebih baik	
05 / 05 / 2023	Pengecekan Bab 4	Terdapat beberapa hal yang perlu diperbaiki dan dicari kembali referensi. Selain itu kesalahan tata cara penulisan dan penyusunan sub-bab	

C

Implementasi Ensemble Deep Learning Untuk Klasifikasi Penyakit Pada Tanaman Padi, Richard Alvin Pratama, Universitas Multimedia Nusantara

16 / 05 / 2023	Pengecekan Bab 2, 3, dan 4	Untuk mengulang training pada model oversampling dikarenakan terdapat kesalahan dalam proses augmentasi, menggabungkan beberapa sub-bab agar tidak terjadi pengulangan dan memudahkan untuk dilakukan analisa	
25 / 05 / 2023	Pengecekan Keseluruhan Laporan	Pergantian judul agar sesuai dengan yang diteliti, mempelajari kembali recall atau TP dll pada evaluasi metrik, dan mencoba memperbaiki kesimpulan agar sesuai dengan rumusan masalah (4 rumusan masalah = 4 paragraf kesimpulan)	
02 / 06 / 2023	Membahas Revisi Laporan	Memperbaiki latar belakang dan identifikasi masalah yang belum koheren, menambah beberapa flowchart untuk bab 3 dan bab 2. Menambah analisis dan korelasi di bab 4	

Catatan : Form ini wajib dibawa pada saat konsultasi & dilampirkan didalam skripsi (**Minimal 8 kali Konsultasi**)

Tangerang, 7 Juni 2023



Nabila Husna Shabrina, S.T., M.T.



D

Implementasi Ensemble Deep Learning Untuk Klasifikasi Penyakit Pada Tanaman Padi, Richard Alvin Pratama, Universitas Multimedia Nusantara