`

KLASIFIKASI PENYAKIT DAUN PADI MENGGUNAKAN METODE *DEEP LEARNING* MODEL ARSITEKTUR *CONVOLUTIONAL NEURAL NETWORK* (CNN)

SKRIPSI

UIN SUNAN AMPEL
S U R A B A Y A

Disusun Oleh:

MOH. WAFIYUL AHDI
H96219051

PROGRAM STUDI SISTEM INFORMASI
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI SUNAN AMPEL
SURABAYA
2023

PERNYATAAN KEASLIAN

Saya yang bertanda tangan di bawah ini,

NAMA	:	MOH. WAFIYUL AHDI
NIM	:	H96219051
PROGRAM STUDI	:	SISTEM INFORMASI
ANGKATAN	:	2019

Menyatakan bahwa saya tidak melakukan plagiat dalam penulisan skripsi saya yang berjudul: "KLASIFIKASI PENYAKIT DAUN PADI MENGGUNAKAN METODE DEEP LEARNING MODEL ARSITEKTUR CONVOLUTIONAL NEURAL NETWORK (CNN)". Apabila suatu saat nanti terbukti saya melakukan tindakan plagiat, maka saya bersedia menerima sanksi yang telah ditetapkan.

Demikian pernyataan keaslian ini saya buat dengan sebenar-benarnya.

Surabaya, 5 Juli 2023

Yang menyatakan,

Moh. Wafiyul Ahdi

NIM: H96219051

Dipindai dengan CamScanner

LEMBAR PERSETUJUAN PEMBIMBING

Skripsi oleh

NAMA : MOH. WAFIYUL AHDI

NIM : H96219051

JUDUL : KLASIFIKASI PENYAKIT DAUN PADI MENGGUNAKAN METODE DEEP LEARNING MODEL ARSITEKTUR CONVOLUTIONAL NEURAL NETWORK (CNN)

Ini telah diperiksa dan disetujui untuk diujikan.

Surabaya, 5 Juli 2023

Dosen Pembimbing 1	Dosen Pembimbing 2
Dr. Eng. Anang Kunaefi, M.Kom.	Khalid, M.Kom.
NIP 197911132014031001	NIP 197906092014031002

PENGESAHAN TIM PENGUJI SKRIPSI

Skripsi Moh. Wafiyul Ahdi ini telah dipertahankan
di depan tim penguji skripsi
di Surabaya, 5 Juli 2023.

Mengesahkan,

Dewan Penguji

Dosen Penguji 1	Dosen Penguji 2
Ahmad Yusuf, M.Kom.	Bayu Adhi Nugroho, Ph.D.
NIP 199001202014031003	NIP 197905182014031001
Dosen Penguji 3	**Dosen Penguji 4**
Dr. Eng. Anang Kunaefi, M.Kom.	Khalid, M.Kom.
NIP 197911132014031001	NIP 197906092014031002

Mengetahui,

Dekan Fakultas Sains dan Teknologi
UIN Sunan Ampel Surabaya

Dr. A. Saepul Hamdani, M.Pd.
NIP 196507312000031002

Dipindai dengan CamScanner

KEMENTERIAN AGAMA
UNIVERSITAS ISLAM NEGERI SUNAN AMPEL SURABAYA
PERPUSTAKAAN
Jl. Jend. A. Yani 117 Surabaya 60237 Telp. 031-8431972 Fax.031-8413300
E-Mail: perpus@uinsby.ac.id

LEMBAR PERNYATAAN PERSETUJUAN PUBLIKASI KARYA ILMIAH UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademika UIN Sunan Ampel Surabaya, yang bertanda tangan di bawah ini, saya:

Nama	: Moh. Wafiyul Ahdi
NIM	: H96219051
Fakultas/Jurusan	: Sains dan Teknologi / Sistem Informasi
E-mail address	: h96219051@student.uinsby.ac.id

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Perpustakaan UIN Sunan Ampel Surabaya, Hak Bebas Royalti Non-Eksklusif atas karya ilmiah :
☑ Sekripsi ☐ Tesis ☐ Desertasi ☐ Lain-lain (.....)
yang berjudul :

KLASIFIKASI PENYAKIT DAUN PADI MENGGUNAKAN METODE DEEP LEARNING MODEL ARSITEKTUR CONVOLUTIONAL NEURAL NETWORK (CNN)

beserta perangkat yang diperlukan (bila ada). Dengan Hak Bebas Royalti Non-Ekslusif ini Perpustakaan UIN Sunan Ampel Surabaya berhak menyimpan, mengalih-media/format-kan, mengelolanya dalam bentuk pangkalan data (database), mendistribusikannya, dan menampilkan/mempublikasikannya di Internet atau media lain secara ***fulltext*** untuk kepentingan akademis tanpa perlu meminta ijin dari saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan atau penerbit yang bersangkutan.

Saya bersedia untuk menanggung secara pribadi, tanpa melibatkan pihak Perpustakaan UIN Sunan Ampel Surabaya, segala bentuk tuntutan hukum yang timbul atas pelanggaran Hak Cipta dalam karya ilmiah saya ini.

Demikian pernyataan ini yang saya buat dengan sebenarnya.

Surabaya, 5 Juli 2023

Penulis

(Moh. Wafiyul Ahdi)
nama terang dan tanda tangan

ABSTRAK

KLASIFIKASI PENYAKIT DAUN PADI MENGGUNAKAN METODE *DEEP LEARNING* MODEL ARSITEKTUR *CONVOLUTIONAL NEURAL NETWORK* (CNN)

Oleh:

Moh. Wafiyul Ahdi

Permintaan beras yang terus meningkat dari tahun ke tahun menjadikan beras sebagai salah satu komoditas pangan penting Indonesia untuk ketahanan pangan. Jika penyakit, terutama pada daun padi, mengakibatkan gagal panen, peran penting padi ini akan terganggu, sehingga diperlukan identifikasi dini dengan akurasi yang wajar menggunakan mesin komputer. Pada penelitian ini, tahap preprocessing, modeling, dan evaluasi digunakan untuk mengklasifikasikan berbagai jenis penyakit daun padi menggunakan convolutional neural network (CNN), salah satu kelas dalam metodologi deep learning. Dengan menggunakan model dengan sedikit parameter, penelitian ini bertujuan untuk mengklasifikasikan penyakit daun padi secara cepat dan akurat. Hasil dari beberapa percobaan menggunakan metode ini dengan akurasi tertinggi dilakukan dengan menggunakan augmentasi data, normalisasi layer, dan fungsi optimalisasi RMSProp learning rate 0,001, menghasilkan akurasi sebesar 98,93 persen. Menurut hasil percobaan, persiapan data dan optimasi model sangat penting untuk proses klasifikasi, terutama untuk dataset dengan variasi yang besar dan tidak seimbang.

Kata kunci: *Convolutional Neural Network (CNN), image processing, deep learning, EfficientNet, rice leaf diseases*.

UIN SUNAN AMPEL
S U R A B A Y A

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

ABSTRACT

CLASSIFICATION OF RICE LEAF DISEASES USING THE CONVOLUTIONAL NEURAL NETWORK (CNN) DEEP LEARNING MODEL ARCHITECTURE METHOD

***By*:**

Moh. Wafiyul Ahdi

The demand for rice has been rising over the years, making it one of Indonesia's essential food commodities for food security. If disease, especially on rice leaves, results in crop failure, this crucial role of rice will be disrupted, so early identification with reasonable accuracy using a computer machine is required. In this study, the preprocessing, modeling, and evaluating stages were used to classify different types of rice leaf disease using the convolutional neural network (CNN), one of the classes in the deep learning methodology. Using a model with few parameters, this study aimed to quickly and accurately classify diseases of rice leaves. The results of several experiments using this method with the highest accuracy were carried out using data augmentation, layer normalization, and the optimization function RMSProp learning rate 0.001, yielding an accuracy of 98.93 percent. According to the results of the experiments, data preparation and model optimization are crucial to the classification process, particularly for datasets with large and unbalanced variations.

***Keyword**: Convolutional Neural Network (CNN), image processing, deep learning, EfficientNet, rice leaf diseases.*

UIN SUNAN AMPEL
S U R A B A Y A

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

DAFTAR ISI

UIN SUNAN AMPEL
S U R A B A Y A
http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

UIN SUNAN AMPEL
S U R A B A Y A

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

DAFTAR GAMBAR

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

UIN SUNAN AMPEL
S U R A B A Y A

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

DAFTAR TABEL

UIN SUNAN AMPEL
S U R A B A Y A

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

`

BAB I
PENDAHULUAN

1.1 Latar Belakang

Padi adalah salah satu komoditi tanaman budidaya paling besar di Indonesia. Hal ini menunjukan bahwa Indonesia adalah negara agraris. Tanaman padi memiliki peran yang penting dalam menjaga stabilitas dan ketahanan pangan pokok nasional (Mohidem dkk., 2022). Pada tahun 2022, sebesar 10,61 juta hektar lahan padi di panen dengan berat total yang dihasilkan adalah 55,67 juta ton. Hasil tersebut mengalami peningkatan pada luas lahan panen sebesar 194,71 ribu hektar dan berat 1,25 juta ton dari tahun 2021. Sebesar 32,07 juta ton padi menjadi konsumsi pangan penduduk Indonesia pada tahun 2022 (*Badan Pusat Statistik*, 2022). Oleh sebab itu, dibutuhkan hasil yang baik dalam memenuhi permintaan dan ketersediaan padi.

Peningkatan permintaan terhadap produksi padi semakin meningkat seiring bertambahnya tahun. Namun terdapat tantangan dalam memenuhi aspek permintaan dan ketersediaan padi. Masalah yang ada adalah produktivitas petani dipengaruhi oleh cuaca yang kadang tidak menentu serta munculnya berbagai penyakit pada padi yang bisa mengakibatkan gagal panen (Arifah dkk., 2022). Apabila hal ini terjadi, maka akan dapat mengganggu ketahanan pangan nasional.

Untuk mengidentifikasi penyakit pada padi, salah satu aspek yang bisa dilakukan adalah pemeriksaan pada daun padi (Rahman dkk., 2020). Secara umum, penyakit pada daun padi akan menyebabkan perubahan warna di area sekitar daun. Gangguan ini disebabkan oleh mikroorganisme kecil yang merusak tanaman padi sehingga mengganggu pertumbuhan padi (Shivappa dkk., 2021). Hal ini tentu mempengaruhi kualitas dan kuantitas padi yang dipanen. Oleh sebab itu, identifikasi penyakit pada padi akan sangat membantu dalam proses pencegahan dan penanganannya.

Identifikasi penyakit pada padi dapat dilakukan secara kasat mata oleh manusia dengan melihat pola kerusakan pada daun padi. Namun penilaian yang dihasilkan mungkin punya peluang besar untuk meleset yang disebabkan oleh

UIN SUNAN AMPEL
S U R A B A Y A

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

beberapa faktor seperti kelelahan, gangguan mata, atau pengalaman yang kurang. Oleh sebab itu, diperlukan sistem untuk melakukan klasifikasi dengan akurasi yang baik. Komputer dapat melakukan hal ini dengan cara menganalisa tekstur dari suatu citra digital untuk mengetahui pola suatu objek berdasarkan pada karakteristik yang diperoleh melalui perhitungan matematis (Armi & Fekri Ershad, 2019). Salah satu cara yang bisa dicoba adalah dengan menggunakan metode *deep learning*.

Deep learning dapat digunakan untuk melakukan tindakan seperti manusia secara alami bermodalkan pengajaran yang terstruktur (McClelland & Botvinick, 2020). Salah satu metode *deep learning* yang baik untuk klasifikasi citra gambar adalah *Convolutional Neural Network* (CNN). Metode ini bekerja dengan cara meniru sistem pengenalan citra sehingga mampu mengolah informasi citra tersebut (Anggraeni dkk., 2022).

Pada penelitian sebelumnya mengenai klasifikasi penyakit tanaman padi menggunakan *Convolutional Neural Network* (CNN), mengungkapkan bahwa klasifikasi yang dilakukan menggunakan *Convolutional Neural Network* (CNN) dengan model arsitektur VGG19 untuk mendeteksi penyakit padi menghasilkan tingkat akurasi 95.24% (Priyangka & Kumara, 2021). Hal ini sejalan dengan penelitian klasifikasi padi bagus dan padi rusak menggunakan *Convolutional Neural Network (CNN)* menghasilkan tingkat akurasi sebesar 83% (Indra dkk., 2022). pada penelitian lain, Klasifikasi pada daun kopi menggunakan model Arsitektur CNN *Alexnet* menghasilkan akurasi sebesar 81,6% (Irfansyah dkk., 2021).

Berdasarkan penelitian terdahulu tersebut, terdapat beberapa permasalahan selama melakukan proses klasifikasi menggunakan *Convolutional Neural Network* (CNN) disamping dari tingginya akurasi yang didapat. Beberapa diantara permasalahan tersebut adalah *runtime* yang cukup lama dan penggunaan memori yang berlebih dikarenakan parameter yang banyak terutama pada model arsitektur *Convolutional Neural Network* (CNN) klasik seperti VGG dan AlexNet. Banyak peneliti berusaha memperbaiki permasalahan ini dengan melakukan beberapa cara seperti *data augmentation*, penggunaan *pooling layer*, penggunaan fungsi aktivasi seperti *dropout*, optimasi dan banyak lagi (Gu dkk., 2018). Oleh sebab itu

UIN SUNAN AMPEL
S U R A B A Y A
http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

dibutuhkan suatu model yang memiliki *runtime* lebih cepat dengan jumlah parameter yang lebih sedikit (Beheshti & Johnsson, 2020).

Berdasar permasalahan tersebut di atas, penelitian ini mengajukan penggunaan model arsitektur *Convolutional Neural Network* (CNN) yang memiliki parameter dan *runtime* lebih sedikit namun dengan akurasi yang setara. Penggunaan model ini diharapkan dapat mengurangi *runtime* dan penggunaan memori dengan hasil akurasi yang setara pada klasifikasi penyakit daun padi.

1.2 Perumusan Masalah

Berdasarkan uraian dari latar belakang diatas, maka rumusan masalah yang digunakan untuk penelitian ini adalah sebagai berikut:

1. Bagaimana klasifikasi menggunakan metode *deep learning* dengan model arsitektur *Convolutional Neural Network* (CNN) pada penyakit daun padi?
2. Bagaimana tingkat akurasi dan *runtime* yang dihasilkan pada klasifikasi menggunakan metode *deep learning* arsitektur *Convolutional Neural Network* (CNN) pada penyakit daun padi?

1.3 Batasan Masalah

Adapun batasan masalah yang digunakan untuk membatasi ruang lingkup masalah yang terlalu luas pada penelitian ini adalah sebagai berikut:

1. Penelitian ini berfokus pada klasifikasi penyakit pada daun padi.
2. Format citra berkas yang diklasifikasi adalah JPG.
3. *Dataset* yang digunakan adalah *dataset* penyakit daun padi yang diperoleh melalui link https://www.kaggle.com/datasets/shayanriyaz/riceleafs pada situs www.kaggle.com.
4. Jenis penyakit yang diidentifikasi adalah *brown spot, hispa, leaf blast* dan *healthy*.

1.4 Tujuan Penelitian

Berdasarkan deskripsi pada rumusan masalah, dapat diketahui tujuan pada penelitian ini adalah sebagai berikut:

UIN SUNAN AMPEL
S U R A B A Y A
http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

1. Dapat mengetahui bagaimana implementasi metode *deep learning* dengan model arsitektur *Convolutional Neural Network* (CNN) untuk melakukan klasifikasi penyakit pada daun padi.
2. Dapat mengetahui tingkat akurasi dan *runtime* dari klasifikasi menggunakan metode *deep learning* model arsitektur *Convolutional Neural Network* (CNN) pada penyakit daun padi.

1.5 Manfaat Penelitian

Berdasarkan deskripsi dari tujuan penelitian, dapat diketahui manfaat dari penelitian ini adalah sebagai berikut:

1 Manfaat Akademis

a Penelitian ini diharapkan dapat menjadi referensi atau acuan dalam melakukan klasifikasi penyakit pada daun padi menggunakan metode *deep learning*.

2 Manfaat Praktis

a Bagi Peneliti

Penelitian ini dapat memberikan sumbangan ilmiah dari inovasi dalam melakukan klasifikasi penyakit daun padi menggunakan metode *deep learning* dengan model arsitektur *Convolutional Neural Network* (CNN).

b Bagi Pembaca

Penelitian ini dapat dijadikan sebagai referensi dalam mengetahui tingkat akurasi dan *runtime* dalam klasifikasi penyakit daun padi menggunakan metode *deep learning* pada model arsitektur *Convolutional Neural Network* (CNN) yang digunakan.

1.6 Sistematika Penulisan Skripsi

Bagian ini berisi tentang paparan garis-garis besar isi tiap bab.

1. BAB I PENDAHULUAN

Berisi tentang latar belakang dari permasalahan terkait penyakit pada daun padi dan model *deep learning* yang digunakan untuk mendeteksi penyakit

UIN SUNAN AMPEL
S U R A B A Y A
http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

tersebut, perumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, dan sistematika penulisan.

2. BAB II TINJAUAN PUSTAKA

   Berisi landasan teori mengenai penyakit pada daun padi, *data pre-processing*, metode *deep learning*, model *Convolutional Neural Network*, serta metode evaluasi *cross validation* dan *confusion matrix*.

3. BAB III METODE PENELITIAN

   Berisi tentang jenis penelitian, sumber data dan alur penelitian yang meliputi proses perumusan, literatur pengumpulan, pengolahan, dan analisa data.

4. BAB IV HASIL DAN PEMBAHASAN

   Berisi tentang hasil dan pembahasan dari rumusan masalah serta penjabaran proses implementasi, analisis dan penarikan kesimpulan.

5. BAB V PENUTUP

   Berisi tentang kesimpulan serta saran dari penelitian kepada pembaca.

UIN SUNAN AMPEL
S U R A B A Y A

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

`

BAB II
TINJAUAN PUSTAKA

2.1 Tinjauan Penelitian Terdahulu

Bagian ini berisikan beberapa pemaparan penelitian terdahulu yang memiliki relevansi dengan judul sebagai referensi dan bahan acuan untuk melakukan penelitian. Beberapa penelitian tersebut adalah sebagai berikut:

Tabel 2.1 Pemaparan Penelitian Terdahulu

No	Kategori	Hasil	Relevansi
1	*Deep Learning Classification* (Anggraeni dkk., 2022)	CNN menjadi yang paling unggal dalam perbandingan antara CNN, RCNN, dan CLSTM dengan nilai presisi klasifikasi *multilabel* sebesar 88,54%. Hal ini dikarenakan data yang berlabel sangat membantu proses komputasi dalam metode CNN.	CNN memiliki akurasi terbaik dalam melakukan klasifikasi menggunakan *labelled data* termasuk *image processing*
2	*Rice Plant classification using* CNN (Priyangka & Kumara, 2021)	Model arsitektur CNN (VGG19) memiliki fitur untuk meningkatkan akurasi hasil citra gambar dengan bantuan variasi *epoch* dan augmentasi data dengan akurasi sebesar 95,24%.	Model arsitektur pada metode CNN mampu membantu dalam meningkatkan akurasi hasil klasifikasi citra gambar
3	CNN *architecture* (Bezdan & Bacanin, 2019)	CNN adalah metode *deep learning* paling baik dalam melakukan *computer vision* karena kinerja dalam melakukan *image processing*, adanya Varian skala dan jaringan. ResNet menjadi model arsitektur modern yang paling umum digunakan dan AlexNet menjadi model arsitektur	Adanya perbandingan jumlah parameter antara model klasik dan modern sangat memberikan pandangan terhadap *runtime* dan penggunaan memori selama masa *training*

UIN SUNAN AMPEL
S U R A B A Y A

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

		klasik yang paling umum digunakan.	
4	CNN *Architecture comparison* (Setiawan, 2020)	Percobaan ini melibatkan beberapa arsitektur seperti VGG, DenseNet, ResNet50, ResNet101, GoogleNet, InceptionV3, AlexNet, InceptionResNetV2 dan SqueezeNet menggunakan citra gambar fundus retina untuk melakukan klasifikasi pada 2 kelas dengan akurasi tertinggi di dapat oleh VGG19.	Terdapat model dengan jumlah parameter dan *runtime* paling sedikit namun mampu mengungguli model dengan jumlah parameter dan *runtime* yang jauh lebih besar. Hal ini menunjukan bahwa jumlah parameter dan *runtime* tidak selalu memiliki hasil yang lebih baik.
5	*Light Model classification* (Hassanpour & Malek, 2019)	Klasifikasi menggunakan model yang "ringan" terkadang bekerja dengan baik pada tugas klasifikasi gambar karena lebih mampu mempelajari fitur-fitur pada gambar yang berguna untuk klasifikasi.	Performa model yang "ringan" memang tidak menjadi yang terbaik, namun pembelajaran yang mendalam mengenai fitur pada citra menjadi nilai lebih.

UIN SUNAN AMPEL
S U R A B A Y A

Pada penelitian yang dilakukan oleh (Anggraeni dkk., 2022), CNN menjadi metode terbaik diantara RCNN dan CLSTM dalam melakukan klasifikasi *multilabel* dengan akurasi rata-rata 88,54%. Hal ini dikarenakan CNN mampu membedakan label data dengan sangat baik yang sebelumnya telah diolah menggunakan teknik model NeuroNER. Sedangkan CLSTM memiliki hasil paling rendah karena metode ini lebih cocok untuk data sekuensial. Hal tersebut juga dibuktikan oleh penelitian (Priyangka & Kumara, 2021) terkait klasifikasi pada tanaman padi menggunakan metode CNN. Akurasi yang dihasilkan terbilang tinggi, yaitu 95,24%. Namun, hasil akurasi tinggi ini dibarengi dengan penggunaan model arsitektur klasik pada CNN yaitu VGG19. Model arsitektur ini terbukti mampu

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

meningkatkan hasil akurasi dari proses klasifikasi yang dilakukan menggunakan metode CNN.

(Bezdan & Bacanin, 2019) dalam penelitiannya memaparkan mengenai layer dan arsitektur pada CNN. Arsitektur CNN terbagi menjadi 2 yaitu model arsitektur klasik dan modern. Perbedaan kedua arsitektur ini ada pada *layer* yang digunakan. *Layer* pada model arsitektur CNN klasik lebih banyak karena parameter yang digunakan juga lebih banyak. Hal ini menyebabkan penggunaan memori dan *runtime* yang berlebih. Beberapa model arsitektur klasik yang ada adalah LeNet, AlexNet, dan VGG. Sedangkan arsitektur modern memiliki lebih sedikit parameter dan *runtime* namun dengan akurasi yang setara. Beberapa model arsitektur modern adalah GoogleNet, ResNet, dan SqueezeNet.

Pada penelitian (Julianto & Sunyoto, 2021), dilakukan komparasi semua model arsitektur pada CNN. Percobaan ini dilakukan menggunakan 2 skenario yaitu dengan menggunakan optimasi dan tidak. Hasil dari komparasi tersebut mengungkapkan bahwa VGG menjadi yang terbaik dari segi akurasi. Namun, ada hal unik saat melihat hasil percobaan pada model SqueezeNet. Model arsitektur Squeezenet dengan penggunaan memori paling sedikit memiliki akurasi yang baik bahkan sempat menyaingi akurasi VGG dan AlexNet saat percobaan menggunakan optimasi. Menurut (Hassanpour & Malek, 2019), Hal ini dapat terjadi karena akurasi model dapat dipengaruhi oleh kemampuan yang tinggi dalam melakukan pengenalan dan pembelajaran fitur meskipun dengan parameter dan *runtime* yang lebih sedikit.

UIN SUNAN AMPEL
S U R A B A Y A

Berdasarkan data yang terdapat pada tabel 2.1, dapat diketahui bahwa klasifikasi citra gambar dapat dilakukan menggunakan CNN sebagai metode *deep learning* yang terbaik untuk *computer vision*. Dalam metode CNN, terdapat beberapa model arsitektur lain yang memiliki kekurangan dan kelebihan masing-masing. Pemilihan model arsitektur dilakukan dengan alasan kemampuannya dalam mengenali informasi mendalam dari citra dibanding model arsitektur lain, parameter yang lebih sedikit dengan akurasi yang setara, dan *runtime* yang relatif lebih singkat.

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

2.2 Teori Dasar

2.2.1 Padi

Padi atau dalam nama latin biasa disebut *Oryza sativa* ini adalah salah satu jenis tanaman yang biasa dibudidayakan dan sangat penting bagi peradaban dunia, terutama Indonesia (Gross & Zhao, 2014). Diduga bahwa padi adalah tanaman yang berasal dari India dan dibawa masuk ke Indonesia pada kisaran tahun 1500 sebelum masehi (Gunawan, 2018). Terdapat dua tipe padi budidaya yaitu padi gogo dan padi rawa. Padi gogo biasa dibudidaya di daerah bersawah yang memiliki tempat untuk tadah air dan cenderung kering (Januarti dkk., 2021).

Padi memiliki beberapa jenis penyakit yang dapat menyerang di beberapa bagian seperti batang, daun dan biji selama proses pertumbuhan. Pada daerah daun, umumnya penyakit yang ada disebabkan oleh mikroorganisme kecil. Bisa juga disebabkan oleh gangguan cuaca, kekurangan kandungan gizi, obat-obatan yang tidak cocok dan jenis tanah (Nuryanto, 2018).

Pengendalian penyakit pada padi sebenarnya dapat dilakukan dengan berbagai cara tergantung dengan akar permasalahan. Namun hal yang paling penting adalah dengan mengetahui jenis tanaman, jenis tanah, jenis penyakit yang menyerang dan cara penanggulangannya. Penyebaran penyakit dapat diminimalisir dengan beberapa cara seperti penanaman varietas yang lebih tahan, penanaman yang berjarak, pemupukan berimbang, dan pestisida (Nuryanto, 2018).

2.2.2 Penyakit pada Daun Padi

Penyakit pada daun padi memiliki beberapa jenis tipe dan gejala sesuai dengan penyebab dan kondisi daerah tempat penanaman padi. Berikut adalah penjelasan dari penyakit daun padi yang digunakan pada penelitian ini:

1. *Brown Spot*

   Penyakit *brown spot* atau biasa disebut bercak coklat pada daun padi ini adalah salah satu penyakit yang bisa disebabkan oleh tanah yang kurang subur. Penyakit ini dapat menyebabkan tanaman padi mati karena pembusukan pada daerah yang diserang. Selain kesuburan tanah, faktor lain munculnya penyakit ini adalah munculnya mikroorganisme dari sistem pengairan yang kurang baik sehingga menghambat penyerapan unsur hara (Prasetyo, 2017).

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

Gejala dari penyakit ini adalah munculnya bercak berwarna coklat dengan titik tengah berwarna abu-abu pada permukaan daun sebesar biji wijen yang merata. Letak dari bercak ini umumnya terdapat di area yang sejajar dengan ibu tulang daun. Gejala ini akan terlihat mulai dari 2 hingga 4 minggu setelah padi dipindah (Norjamilah dkk., 2021).

Mikroorganisme yang menyebabkan penyakit ini terjadi adalah *Drechslera oryzae*. Penyebaran dari mikroorganisme ini bisa melalui angin. *Drechslera oryzae* adalah salah satu jenis jamur. Pengendalian penyakit ini dapat dilakukan dengan pemberian nutrisi yang cukup baik dari segi unsur hara, pengairan dan obat-obatan kepada padi. Penanaman varietas yang lebih tahan juga diperlukan apabila cuaca sedang tidak bersahabat (Norjamilah dkk., 2021).

2. *Hispa*

   Penyakit *hispa* pada daun padi adalah salah satu penyakit pada padi yang disebabkan oleh hama yang berkembang biak dari gulma rumput di sekitar sawah. Pemupukan secara berlebih dapat memperparah keadaan dari daun yang terinfeksi. Keadaan cuaca hujan juga sangat menguntungkan bagi hama untuk berkembang biak dan menyebar lebih luas pada padi (Sayuthi dkk., 2020).

   Penyakit ini dapat diidentifikasi dengan munculnya bercak putih memanjang dengan posisi yang sejajar dengan urat daun. Penyebabnya adalah pengikisan hingga jaringan epidermis oleh larva yang ditaruh oleh hama sehingga daun menjadi layu dan rusak (Mahmud, 2006).

   Pengendalian dari penyakit ini dapat dilakukan dengan menghindari pemupukan yang berlebih. Mengatur jarak tanam yang berdekatan juga meminimalisir perkembangbiakan hama. Pemotongan ujung pucuk juga dapat mengurangi populasi hama sebesar 70-90% (Mahmud, 2006).

3. *Leaf Blast*

   Penyakit ini adalah penyakit paling berbahaya bagi tanaman padi. *Leaf blast* disebabkan oleh cendawan yang menyebabkan munculnya lesi pada daun. Penyakit ini dapat terjadi pada seluruh bagian tanaman secara bersamaan. Penyakit ini disebabkan oleh jamur *Magnaporthe oryzae* (Nasution, 2014).

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

Identifikasi penyakit ini adalah dengan melihat bentuk bercak berbentuk berlian di area daun. Pada daun, penyakit ini dapat bervariasi sesuai dengan keadaan lingkungan, umur dan ketahanan varietas tanaman. Pada varietas yang rentan, penyakit ini mulanya akan berwarna abu kehijauan dan meluas dengan cepat menjadi warna coklat sepanjang beberapa sentimeter (Nasution, 2014).

Pengendalian penyakit ini dapat dilakukan dengan merotasi tanaman, pemupukan yang baik dan benar serta pengairan yang cukup. Penggunaan benih yang berkualitas juga mampu meminimalisir penyakit ini (Ulate dkk., 2020).

4. *Healthy*

   Daun padi yang sehat dan tidak terinfeksi oleh penyakit apapun adalah yang memiliki warna daun hijau tua. Tidak layu dan tampak segar adalah ciri lain dari daun padi yang sehat. Juga tidak terdapat kondisi yang ganjil atau perubahan warna yang aneh pada daun (Haris, 2020).

2.2.3 *Artificial Intelligence*

Artificial Intelligence (AI) atau yang biasa disebut sebagai kecerdasan buatan adalah salah satu rancangan dimana kecerdasan ditambahkan ke dalam sistem yang memiliki konfigurasi sebagai bentuk kecerdasan ilmiah. AI dimasukkan ke dalam sistem komputer dengan tujuan untuk melakukan tindakan layaknya manusia secara natural dengan dibekali kemampuan penalaran (Joiner, 2018). AI memiliki dua basis dalam bagian komputasinya, yaitu *motor* dan *knowledge base*. *Knowledge base* berisikan pemikiran, fakta dan teori manusia, sedangkan *motor base* melakukan penarikan kesimpulan dari pengetahuan yang berasal dari *knowledge base*. Bermodalkan hal tersebut, AI mampu menyelesaikan suatu permasalahan yang diberikan dan menjadi tolak ukur kecerdasan (Jones, 2009).

Dalam dunia komputasi, AI mampu membentuk cabang yang berkaitan dengan perilaku, pembelajaran dan adaptasi pada suatu sistem. Dalam dunia penelitian, AI biasa digunakan untuk mengotomasi sistem dalam menyelesaikan masalah dan menjadi disiplin ilmu yang memiliki fokus untuk menyediakan solusi

UIN SUNAN AMPEL
S U R A B A Y A
http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

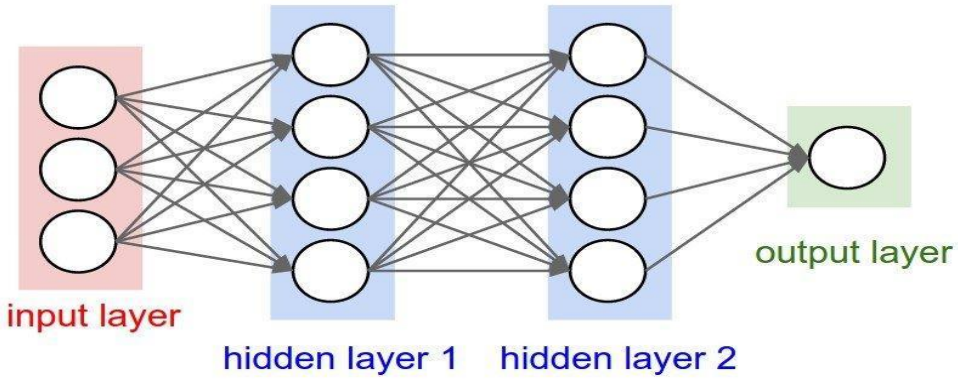
riil pada kehidupan nyata (Endang & R, 2017). Dewasa ini, kebanyakan AI digunakan pada beberapa bidang seperti militer, ekonomi, farmasi, dan *game*.

Garis besar dalam AI dibagi menjadi dua yaitu AI Konvensional dan *Computational Intelligence* (CI). AI Konvensional banyak melibatkan pembelajaran berbasis analisa statistik dengan beberapa metode seperti *bayesian network,* sistem pakar dan sebagainya. Sedangkan CI lebih melibatkan pembelajaran iteratif yang berasal dari data empiris dengan menggunakan beberapa metode seperti *fuzzy*, *neural network* dan komputasi evolusioner (Jones, 2009).

2.2.4 *Deep Learning*

Deep learning merupakan cabang ilmu dari *machine learning* yang lebih memiliki basis kepada jaringan syaraf tiruan untuk melatih sistem agar mampu meniru perilaku manusia secara natural. Contoh dari tindakan yang dimaksud adalah seperti mesin melakukan pembelajaran dari data yang diinputkan sehingga mampu melakukan pengambilan keputusan. *Deep learning* biasa digunakan untuk melakukan pemrosesan dari berbagai media seperti teks, gambar dan suara (Sharma dkk., 2021).

Layer pada *deep learning* sendiri terdapat 3 bagian yaitu *input layer*, *hidden layer* dan *output layer*. *Input layer* berisikan *node* atau titik yang memiliki nilai. Lalu *hidden layer* berisikan algoritma untuk meminimalisasi terjadinya *error* waktu *output*. Sedangkan *output layer* bertugas untuk menampilkan hasil perhitungan matematis dari *hidden layer* sebesar *input* yang diterima. Semakin banyak *layer* dan lengkap label dalam model pembelajaran, maka model pembelajaran akan semakin baik (Sarker, 2021).



Gambar 2.1 Arsitektur *Deep Learning*

Sumber: (J. Kim dkk., 2022)

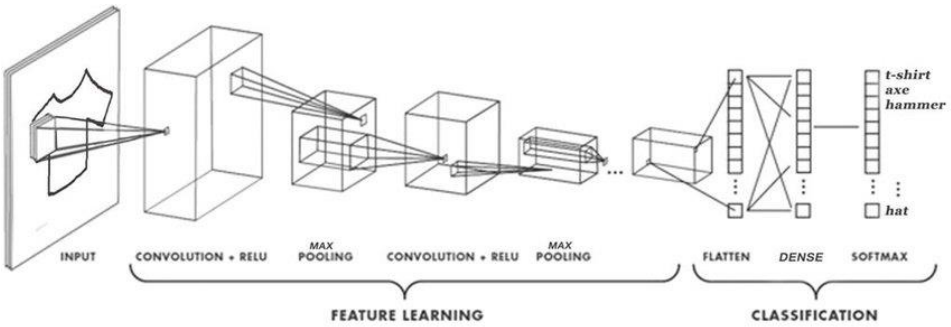
http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

Pada gambar di atas dapat diketahui bahwa bahwa setiap *node* pada setiap *layer* di *deep learning* akan langsung terhubung dengan *node* lain pada *layer* berikutnya. Arsitektur tersebut dapat disebut sebagai *Multi Layer Perceptron* (MLP). Semakin banyak *layer* yang terdapat pada *hidden layer* akan membantu optimasi dalam melakukan proses klasifikasi.

Oleh sebab itu, *deep learning* mampu mempelajari berbagai model abstraksi pada data dengan menggunakan proses pada model komputasi yang terdiri dari beberapa *layer*. Hal ini tentu sangat membantu peneliti dalam melakukan pengenalan objek, suara, deteksi objek dan lain sebagainya (Ilin dkk., 2017). Berikut adalah beberapa algoritma *deep learning* dan kegunannya antara lain *Deep Neural Network* (DNN) untuk *voice recognition*, *Convolutional Neural Network* (CNN) untuk *image processing*, *Recurrent Neural Network* (RNN) untuk translasi dan lain sebagainya (Anggraeni dkk., 2022).

2.2.5 *Convolutional Neural Network* (CNN)

Convolutional Neural Network (CNN) merupakan salah satu metode dalam *deep learning* untuk melakukan pengolahan citra yang dirancang dari dari cara kerja sistem saraf manusia. Metode ini digunakan untuk melakukan ekstraksi fitur dan pengklasifikasian data berupa gambar. CNN mempelajari *input* dari suatu citra dengan melakukan perhitungan bobot terlebih dahulu untuk dilakukan pengelompokan (Lindsay, 2020). Oleh sebab itu, CNN telah digunakan oleh beberapa orang untuk melakukan komputasi pada bidang segmentasi, pengenalan pola, deteksi dan klasifikasi (Bezdan & Bacanin, 2019). CNN memiliki dua tahap dalam proses pembelajarannya, yaitu proses feature learning dan tahap klasifikasi seperti pada gambar 2.2.



Gambar 2.2 Tahapan *Convolutional Neural Network* (CNN)

Sumber: (Murthy, 2019)

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

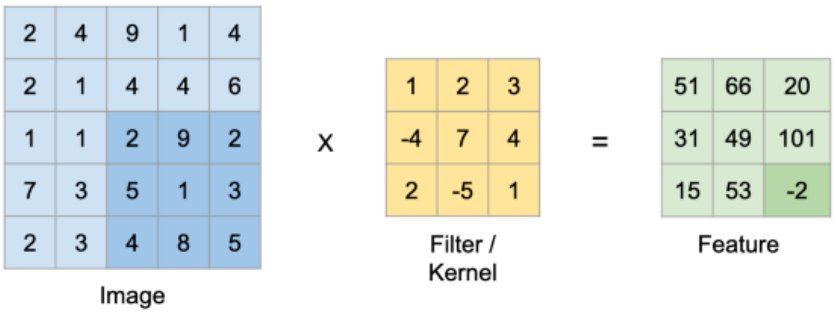
Gambar di atas memperlihatkan bahwa terdapat tiga tahap dalam CNN, yaitu *input*, *feature learning* dan *classification*. Pada *feature learning*, terdapat beberapa *layer* di dalamnya seperti *convolution layer*, ReLU, dan *pooling layer*. *Layer* ini akan menghasilkan vektor yang akan dijadikan data *input* untuk melakukan klasifikasi. Lalu pada tahap *classification*, CNN akan menggunakan metode *neural network* untuk mencari nilai probabilitas pada setiap klasifikasi dengan bantuan dari fully connected layer, fungsi aktivasi softmax, dan dropout (Yamashita dkk., 2018).

2.2.6 Tahap *Feature Learning*

Tahap *feature learning* adalah tahap dimana mesin mempelajari fitur pada citra dengan mengolah nilai pixel dan menghasilkan vektor sebagai bahan input pada layer selanjutnya (Anton dkk., 2021). Tahap ini memiliki beberapa jenis layer di dalamnya, yaitu:

1. *Convolution Layer*

   Layer ini adalah tempat dimana input dari citra akan melalui proses translasi untuk mempelajari fitur tepi, warna, dan bentuk menjadi fitur-fitur berdasarkan pada citra. Operasi ini akan melalui dua matriks yaitu matriks pixel pada citra dan matriks *kernel* atau *filter*. Matriks *filter* pada CNN berisikan nilai random antara -1 hingga 1 yang berfungsi untuk melakukan pembelajaran pada area kecil di dalam citra sesuai ukuran *filter*. Ukuran *filter* ini tergantung pada jenis arsitektur apa yang dipakai. Citra yang telah diinputkan akan diolah hingga tercipta *output* berupa tumpukan *feature map* dari semua lapisan matriks *filter*. Operasi ini dapat dilihat pada gambar 2.3.

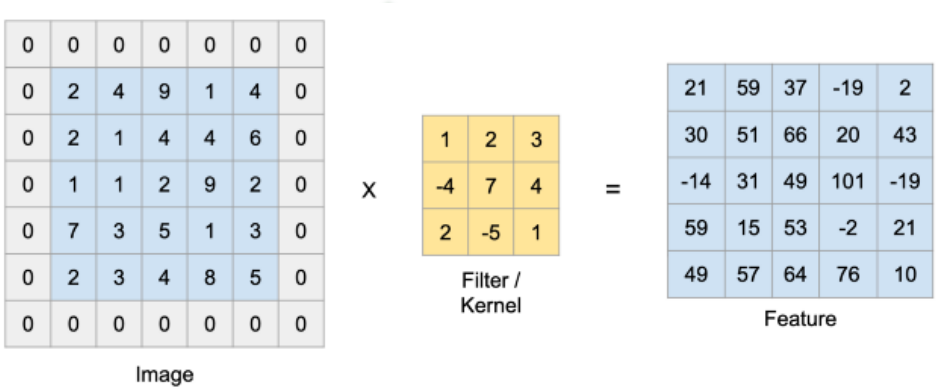


Gambar 2.3 *Convolution Operation*

Sumber: (Patel, 2020)

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

Untuk mencari nilai dalam *feature map*, perhitungan yang dilakukan adalah perkalian setiap elemen dalam pixel input dengan elemen dan ukuran kernel. Pada perhitungan jumlah *feature map*, terdapat pertimbangan perhitungan lain bernama *stride* atau jumlah pergeseran pixel dan *padding* atau parameter penentu jumlah pixel yang berisikan nilai 0 dan ditambahkan di setiap sisi dari citra yang di *input*. Contoh perhitungan dari operasi pada *convolution layer* ini dapat dilihat pada gambar 2.4 Dibawah.



Gambar 2.4 *Convolutional Operation* dengan *kernel*, *stride* dan *padding*

Sumber : (Patel, 2020)

Input citra yang mulanya berukuran 5x5 berubah menjadi 7x7 dikarenakan penambahan *padding* sebesar 1 pada setiap sisi dari citra. Lalu ukuran *filter* yaitu 3x3 dan *stride* sebesar 1 sehingga ditemukan ukuran *feature map* yaitu 5x5 dengan persamaan:

$$H = \left(\frac{I + 2P - K}{S}\right) + 1$$

Penjelasan:

I = ukuran *input* citra gambar

P = nilai *padding*

K = ukuran *kernel*

S = nilai *stride*

H = ukuran *feature map*

2. *Rectified Linear Unit* (ReLU)

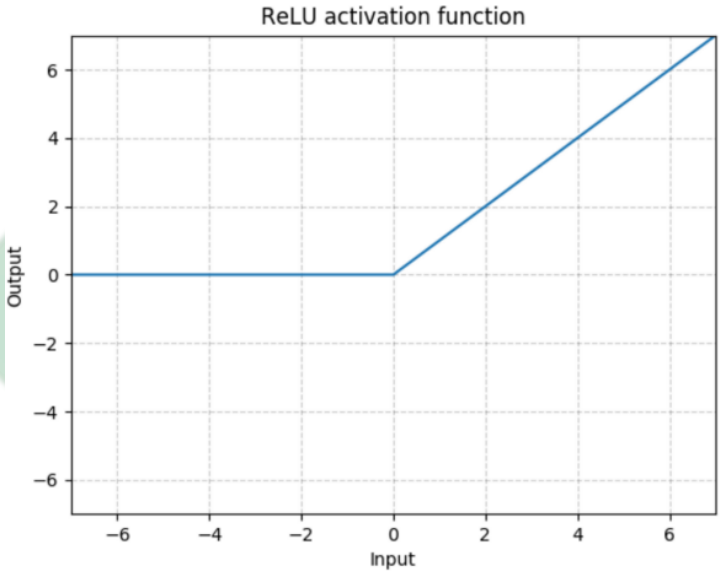
   ReLU merupakan salah satu fungsi aktivasi yang berfungsi untuk merubah nilai *feature map* pada *convolution layer* dalam range 0 hingga tak terbatas.

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

Sederhananya, ReLU adalah fungsi yang menggantikan nilai negatif pada feature map menjadi 0 dan nilai selain negatif atau lebih besar sama dengan 0 akan tetap. Perhitungan ini dapat dilihat pada persamaan berikut:

$$f(x) = max(0, x)$$

f(x) memiliki arti hasil dari nilai 0 dan *x*. *X* adalah nilai dari feature map. Ketika *x* memiliki nilai yang lebih kecil daripada 0, maka *f(x)* akan bernilai 0. Ketika *x* bernilai lebih atau sama dengan 0, maka *f(x)* bernilai *x*. Hal ini dapat dilihat pada gambar 2.5 Di bawah.



Gambar 2.5 Fungsi Aktivasi ReLU

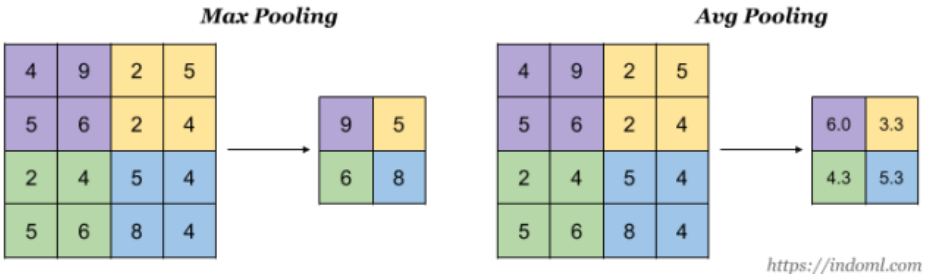
Sumber: (Patel, 2020)

UIN SUNAN AMPEL
S U R A B A Y A

3. *Pooling Layer*

Pooling layer berfungsi sebagai pengurang ukuran matriks input dengan mereduksi data spasial. Perhitungan *output* pada *layer* ini menggunakan ukuran *filter* dan nilai pergeseran *stride*. Terdapat 2 jenis operasi dalam *pooling layer*, yaitu *Max pooling* dan *average pooling*. *Max pooling* memiliki cara kerja dengan menghitung nilai maksimum pada baris kolom sesuai dengan ukuran *filter* dan nilai *stride*. Sedangkan *average pooling* melakukan perhitungan dengan cara mencari rata-rata pada baris kolom sesuai ukuran *filter* dan nilai *stride*. Hal ini dapat dilihat pada gambar 2.6 Di bawah.

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/



Gambar 2.6 *Max* dan *Average Pooling Layer*

Sumber: (Pokhrel, 2019)

Dengan adanya *pooling layer*, proses perhitungan komputasi akan menjadi lebih cepat karena proses ini melakukan perhitungan dengan mengurangi volume tumpukan *feature map* tanpa mengabaikan informasi penting di dalamnya untuk melakukan pengendalian *overfitting*. CNN menggunakan *pooling layer* dengan alasan agar ukuran dari citra gambar dapat direduksi sehingga dapat diganti dengan *convolutional layer* dengan jumlah *stride* yang sama secara mudah.

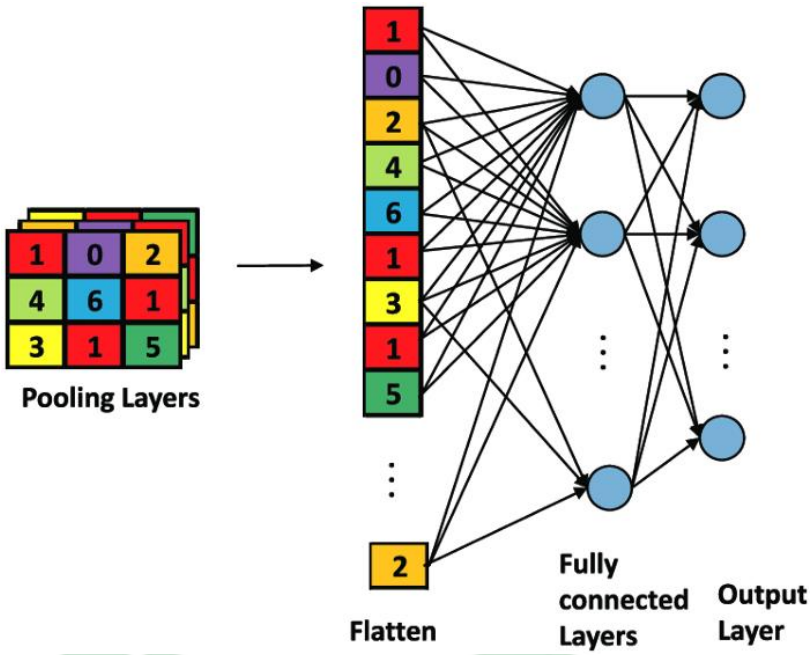
2.2.7 Tahap *Classification*

Tahap ini adalah tahap dimana menggunakan metode *neural network* untuk mencari nilai probabilitas pada setiap klasifikasi (Lindsay, 2020). Tahap ini memiliki beberapa jenis layer di dalamnya, yaitu:

1. *Fully Connected Layer*

   Layer ini menjadi tempat dimana *feature* dianalisis untuk menentukan hubungan atau korelasi antar kelas dan *feature*. *Layer* ini juga menjadi tempat dimana *feature* akan dibentuk ulang (*reshape feature map*) menggunakan *flatten* menjadi suatu vektor agar dapat menjadi *input* bagi *fully connected layer*. Hal ini dikarenakan *feature map* hasil dari *feature learning* masih berupa multidimensional array. Ilustrasi dari tahap ini dapat dilihat pada gambar 2.7 di bawah.

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/



Gambar 2.7 *Fully Connected Layer* dan *Flatten*

Sumber: (Lasky, 2022)

Perhitungan dari *fully connected layer* ini dapat dilihat pada persamaan berikut berikut.

$$y = f(W^T x + b)$$

Dimana *y* adalah nilai dari vektor satu dimensi yang berisikan fungsi *W* sebagai matriks sederhana dengan *T* atau bobot yang dikalikan dengan *x* sebagai vektor input lalu ditambah dengan *b* sebagai vektor bias.

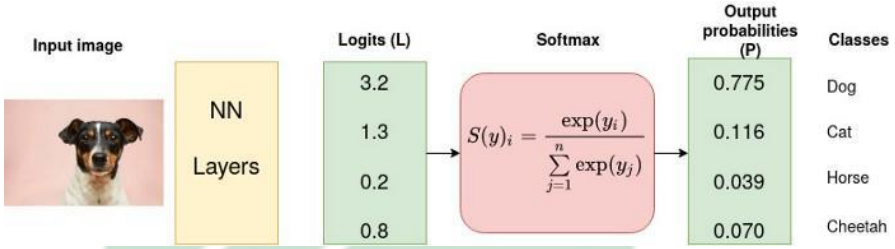
2. *Softmax*

Softmax adalah suatu fungsi aktivasi yang melakukan fungsi probabilitas eksponensial yang dinormalisasi pada kelas yang diamati. Tujuan dari *layer* ini adalah untuk memprediksi *output* hasil klasifikasi menjadi suatu probabilitas terbesar sebagai hasil dari prediksi. Keuntungan dari *softmax* adalah dapat mengubah vektor skor nilai riil menjadi rentang 0 hingga 1 yang berjumlah 1. Oleh sebab itu, fungsi ini dapat membantu mengembalikan peluang dari setiap kelas sehingga probabilitas target akan lebih tinggi. Perhitungan dari fungsi aktivasi ini dapat dilihat pada persamaan berikut:

$$S(y)_i = \frac{exp\ (y_i)}{\sum_{j=1}^{n} exp\ exp\ (y_i)}$$

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

Dimana y adalah vektor input dan y_i adalah elemen ke-i pada vektor *input*. Exp (y_i) memiliki arti bahwa fungsi standar eksponensial yang diaplikasikan ke y_i dan $\sum_{j=1}^{n} exp\ exp\ (y_i)$ memiliki arti bahwa bentuk normalisasi dari Exp (y_i). Ilustrasi dari fungsi aktivasi softmax dapat dilihat pada gambar 2.8 berikut.



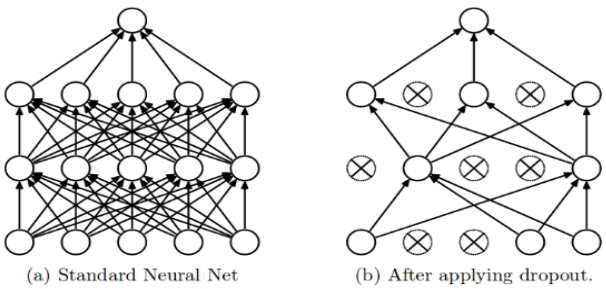
Gambar 2.8 Fungsi Aktivasi *Softmax*

Sumber: (Koech, 2022)

Gambar di atas menjelaskan cara kerja dari fungsi aktivasi ini adalah dengan merubah input citra menjadi angka logis atau vektor yang akan dimasukkan ke dalam perhitungan komputasi dan diurutkan berdasarkan label dengan probabilitas paling tinggi.

3. *Dropout*

Layer ini berfungsi untuk mengatasi *overfitting* akibat kompleksitas dari susunan *hidden layer* pada CNN. Teknik ini dapat membantu kinerja selama proses *testing*. Layer ini memiliki cara kerja dengan menghilangkan beberapa *node* acak dengan probabilitas pada setiap iterasi selama proses *training*. Akibatnya, jaringan akan menjadi lebih tipis sehingga proses *testing* akan menjadi lebih cepat dengan kemungkinan *overfitting* yang kecil. Ilustrasi dari *dropout* dapat dilihat pada gambar 2.9 berikut.



Gambar 2.9 *Dropout*

Sumber: (Novindasari, 2021)

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

2.2.8 *Data Augmentation*

Data Augmentation merupakan salah satu teknik yang digunakan untuk memecahkan keterbatasan pada suatu penelitian. Hal ini dikarenakan dapat mempengaruhi kinerja sistem dalam melakukan proses klasifikasi (Mikołajczyk & Grochowski, 2018). Teknik ini juga membantu dalam mengatasi overfitting atau kekompleksan data yang menyebabkan rendahnya akurasi waktu dilakukan pengujian (Romero Aquino dkk., 2017). Secara umum, *data augmentation* dibagi menjadi dua yaitu modifikasi citra dan transformasi geometri. Modifikasi citra yang dimaksud adalah melakukan modifikasi pada warna citra dengan memainkan kontras, dan sebagainya. Sedangkan maksud dari transformasi geometri adalah melakukan modifikasi pada citra berupa rotasi, refleksi, *resize* dan sebagainya (Kim dkk., 2020).

2.2.9 *Normalization Layer*

Normalization Layer adalah salah satu jenis *data pre-process* yang biasa digunakan untuk mendistribusikan data secara baik agar dapat meningkatkan performa model dan menghindari *overfitting* pada data. Menurut (Ioffe & Szegedy, 2015), *normalization layer* memiliki tujuan untuk mengatasi masalah pada pelatihan model dengan mengurangi kovariat internal pada *neural network* agar dapat menggali lebih banyak informasi pada data hingga berpengaruh pada peningkatan akurasi. Proses ini cocok digunakan pada jenis *dataset* yang memiliki variasi yang banyak karena dapat meningkatkan kecepatan pelatihan yang relevan dengan data (Ba dkk., 2016).

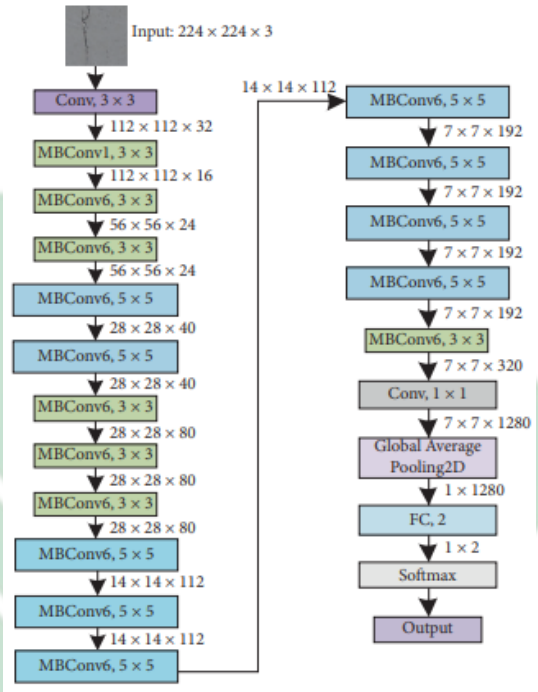
2.2.10 *EfficientNet-B0*

Pada mulanya, EfficientNet adalah model yang pertama kali diperkenalkan oleh Mingxing Tan dan Quoc V. Le pada tahun 2019. Menurut (Tan & Le, 2020), model ini adalah model dengan FLOPS paling sedikit dengan akurasi yang tinggi untuk klasifikasi gambar. Hal ini mengartikan bahwa model ini memiliki efisiensi energi, *scalability* dan akurasi yang baik.

Model ini menyediakan 8 model lain dengan skala yang berbeda dimulai dari EfficientNetB0 hingga B7. Setiap model tersebut mewakili efisiensi dan akurasi pada berbagai skala. B0 adalah model dasar dengan efisiensi yang mengungguli model lain pada setiap skala yang ada (Tan & Le, 2020). Untuk hal

UIN SUNAN AMPEL
S U R A B A Y A
http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

optimasi bobot, model ini menggunakan FixRes yang dikembangkan pada tahun 2020 oleh *Peking University*. Teknik optimasi bobot ini membantu model dalam memperkuat kemampuan representasi dalam *feature learning* yang memiliki kompleksitas pada data gambar. Berikut adalah gambar dari arsitektur dari model EfficientNetB0:



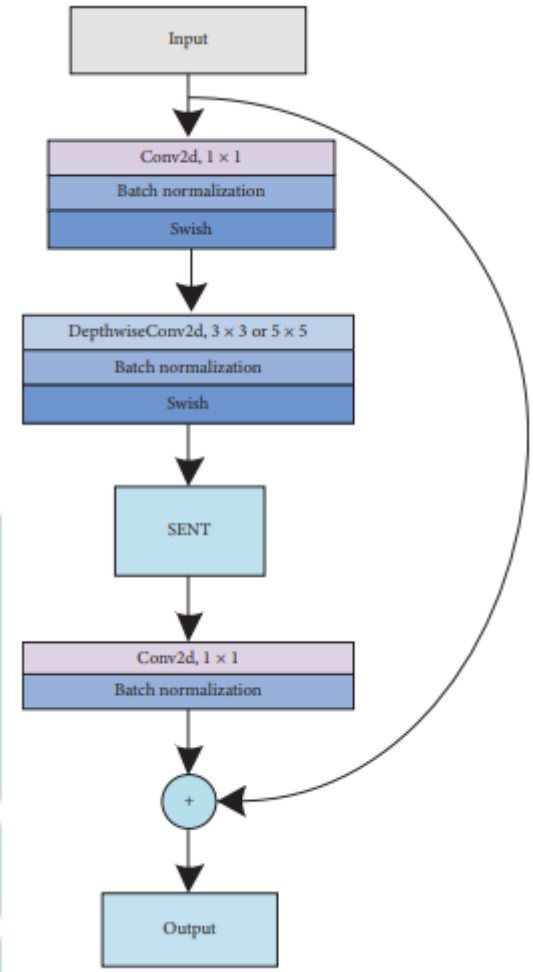
Gambar 2.10 Arsitektur EfficientNetB0

Sumber: (Li & Zhao, 2019)

Berdasarkan gambar diatas, pada *classification layer* model ini memiliki ukuran 3x3 *convolution operation* yang terdapat setelah proses *input image*. Setelah itu, model ini menggunakan *mobile inverted bottleneck* (MBConv) untuk mengekstrak fitur dari gambar. Pada *layer* akhir, model ini menggunakan *convolution* berukuran 1x1 dan *GlobalAverangePool2D* untuk selanjutnya masuk ke *fully connected layer*.

Komponen inti dari model ini ada pada modul *Mobile inverted Bottleneck Convolution* (MBConv). *Framework* dari modul ini terdiri dari *convolution layer, batch normalization* dan *swish activation function* untuk mengoptimalkan jaringan. Berikut adalah *framework* dari modul MBConv:

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/



Gambar 2.11 Ilustrasi *Framework* MBConv

Sumber: (Li & Zhao, 2019)

Desain modul ini terinspirasi dari *inverted residual block* yang memiliki struktur terbaik untuk alasan efisiensi. Proses dari modul ini diawali dengan *convolution operation* dengan ukuran 1x1 untuk mencari lebih banyak informasi saat *feature extraction.* Setelah itu, data akan di *expand* menjadi 3x3 atau 5x5. Kedua operasi ini menggunakan *batch normalization* dan *swish activation function*. Kemudian data akan ditingkatkan performanya dengan cara di *squeeze* menggunakan *Squeeze and Excitation* (SENT). Akhirnya, *convolution operation* dengan ukuran 1x1 dapat digunakan untuk mereduksi dimensi dan menambah *residual connection.*

2.2.11 *Optimizer*

Optimizer atau optimasi dalam *deep learning* adalah fungsi atau algoritma yang mengadaptasi atribut dalam sistem *neural network* seperti *learning rate* dan

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

weight. Hal ini dikarenakan saat mesin melakukan pelatihan pada model, penyesuaian setiap *epoch* harus dilakukan untuk meminimalisir *loss function*. Oleh sebab itu, optimasi ini memiliki peran dalam meningkatkan akurasi dan mengurangi total *loss*. Beberapa optimasi yang memiliki kemampuan baik adalah seperti Adam, RMSprop dan SGD (Mehmood dkk., 2023). Beberapa modifikasi yang dilakukan oleh peneliti untuk mengoptimalkan kinerja optimasi ini juga menghasilkan beberapa terobosan baru seperti modifikasi dari Adam yang menghasilkan AdaMax, Nadam, EAdam dan beberapa modifikasi lain pada optimasi yang berbeda.

2.2.12 *Cross Validation*

Cross Validation adalah salah satu teknik yang digunakan untuk mengevaluasi model deep learning dengan membagi *dataset* menjadi subset-subset tertentu untuk dilakukan *training* dan *testing* sesuai iterasi yang ditetapkan. Proses ini membantu dalam validasi jenis data pada model apakah dapat digeneralisasikan pada data baru (OShea & Hoydis, 2017). Dengan kata lain, proses ini digunakan untuk mengevaluasi performa pada data *testing* dan bukan pada data *training* (Chollet, 2018).

2.2.13 *Confusion Matrix*

Confusion matrix merupakan salah satu teknik yang sering digunakan untuk mengetahui tingkat kesalahan pada proses klasifikasi data. *Confusion matrix* biasanya di interpretasi dengan angka 0 hingga 1. Semakin kecil angka yang didapat, maka semakin baik pula sistem klasifikasi yang digunakan (Beauxis-Aussalet & Hardman, 2014). Dengan begitu peneliti akan mampu mengetahui keakuratan dan tingkat *error* sistem klasifikasi. Cara *confusion matrix* dalam menentukan akurasi adalah dengan menilai beberapa parameter berikut:

Tabel 2.2 Parameter *Confusion Matrix*

Kelas ril / Klasifikasi	*Predicted*	*Predicted*
Actual	*True Positive* (TP)	*True Negative* (TN)
Actual	*False Positive* (FP)	*False Negative* (FN)

Penjelasan:

TP (*True Positive*) = data bernilai positif yang diklasifikasi dengan benar

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

TN (*True Negative*) = data bernilai negatif yang diklasifikasi dengan benar
FN (*False Negative*) = data bernilai negatif yang tidak diklasifikasi dengan benar
FP (*False Positive*) = data bernilai positif yang tidak diklasifikasi dengan benar

Teknik ini menilai suatu klasifikasi berdasarkan *accuracy*, *sensitivity* dan *specificity* berdasarkan nilai dari parameter di atas (Gupta, 2013). Berikut adalah penjelasan dari tiap taraf evaluasi tersebut:

1. *Accuracy*

   Nilai yang didapat dengan membandingkan data lalu diklasifikasi dalam jumlah semua *dataset* yang menunjukan keakuratan dari suatu klasifikasi. Semakin banyak data yang digunakan dalam melakukan klasifikasi, semakin baik pula nilai akurasinya. Rumus dari perhitungan akurasi dapat dilihat pada persamaan di bawah:

   $$Accuracy = \left(\frac{TP + TN}{TP + TN + FP + FN}\right) X\ 100\%$$

2. *Specificity*

   Nilai yang merepresentasi jumlah data bernilai negatif yang terklasifikasi sebagai kelas negatif. Semakin besar nilai spesifitas, maka semakin baik pula proses klasifikasi pada kelas negatif suatu data. rumus dari spesvitas dapat dilihat pada persamaan di bawah:

   $$Specificity\ = \left(\frac{TP}{TP + FP}\right) X\ 100\%$$

3. *Sensitivity*

   Nilai yang merepresentasi jumlah data bernilai positif dan terklasifikasi sebagai kelas positif. Semakin tinggi nilai sensitivitas, maka akan semakin baik pula klasifikasi klas positif. rumus dari sensitivitas dapat dilihat pada persamaan di bawah:

   $$Sensitivity\ = \left(\frac{TP}{TP + FN}\right) X\ 100\%$$

2.3 Integrasi Keilmuan

Bermanfaat untuk kepentingan bersama adalah salah satu hal yang perlu dilakukan umat muslim. Kebutuhan pangan pokok adalah salah satu aspek yang

UIN SUNAN AMPEL
S U R A B A Y A
http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

penting bagi keberlangsungan hidup masyarakat. Padi adalah salah satu komoditas paling dicari di Indonesia. Hal ini dikarenakan harga yang murah dan kekayaan kandungan gizinya yang baik bagi tubuh. Berdasarkan hasil wawancara yang dilakukan dengan KH. Abdul Khaliq salah seorang kyai di Desa Wangen Kecamatan Glagah, menyebutkan bahwa makanan yang baik dalam artian makanan yang terhindar dari penyakit adalah salah satu cara untuk membersihkan laku dan hati. Oleh sebab itu, memakan makanan yang baik adalah anjuran dalam islam dan dengan memakan makanan yang baik, maka kesehatan dan kenikmatan akan datang seiring dengan rasa syukur.

Dewasa ini, *Artificial intelligence* menjadi topik yang hangat diperbincangkan karena manfaat yang diberikan. *Convolutional Neural Network* (CNN) menjadi salah satu model yang ikut memberi sumbangsih manfaat karena kemampuannya dalam mengklasifikasi citra gambar. Banyak studi kasus yang menggunakan metode untuk melakukan klasifikasi pada bahan pangan pokok. Klasifikasi ini juga berguna dalam mengidentifikasi kelainan pada suatu objek untuk mengatasi atau mencegah hal yang tidak baik pada bahan pangan.

Al-Qur'an menjelaskan bahwa Allah swt telah memerintahkan umatnya agar mengkonsumsi makanan yang baik di Bumi. Hal ini mengisyaratkan bahwa apa yang kita makan sebaiknya adalah makanan yang baik tanpa ada penyakit yang terkandung di dalamnya. Sebab dari pandangan agama, makanan yang baik akan membantu kita dalam mengingat mukjizat dan tidak melupakan rasa syukur kepada Allah swt. Hal ini tertulis dalam Al-Qur'an surah Al Baqarah ayat 168 (*Surah Al-Baqarah* - سُورَة البقرة | *Qur'an Kemenag*, 2022) yang berbunyi:

يٰٓاَيُّهَا النَّاسُ كُلُوْا مِمَّا فِى الْاَرْضِ حَلٰلًا طَيِّبًاۖ وَّلَا تَتَّبِعُوْا خُطُوٰتِ الشَّيْطٰنِۗ اِنَّهٗ لَكُمْ عَدُوٌّ مُّبِيْنٌ

Artinya: "*Wahai manusia! Makanlah dari (makanan) yang halal dan baik yang terdapat di bumi, dan janganlah kamu mengikuti langkah-langkah setan. Sungguh, setan itu musuh yang nyata bagimu*".

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

Ayat lain juga menyebutkan hal yang sama bahwa makanan adalah salah satu bentuk rezeki, maka dari itu mengucap rasa syukur adalah bentuk terima kasih dari apa yang diterima. Sebab hakikat dari hidup manusia adalah untuk mencari ridho Allah swt. Rasa syukur akan lebih membantu umat Islam agar lebih mampu mendekatkan diri kepada Allah swt. Perkara ini tertuang dalam Al Qur'an surah Al Baqarah ayat 172 (*Surah Al-Baqarah* - سُورَة البقرة | *Qur'an Kemenag*, 2022) yang berbunyi:

يٰٓاَيُّهَا الَّذِيْنَ اٰمَنُوْا كُلُوْا مِنْ طَيِّبٰتِ مَا رَزَقْنٰكُمْ وَا شْكُرُوْا لِلّٰهِ اِنْ كُنْتُمْ اِيَّاهُ تَعْبُدُوْنَ

Artinya: "*Wahai orang-orang yang beriman! Makanlah dari rezeki yang baik yang Kami berikan kepada kamu dan bersyukurlah kepada Allah jika kamu hanya menyembah kepada-Nya*".

Berdasarkan paparan dari ayat di atas, dapat disimpulkan bahwa apa yang masuk ke tubuh kita berupa makanan haruslah suatu hal yang baik yang tidak membawa penyakit. Ajaran dalam Al Qur'an untuk selalu bersyukur atas rezeki yang diterima seperti makanan yang baik akan membantu umat islam agar menjadi lebih dekat dengan Allah swt. Rasa syukur juga menjadi bentuk taat oleh orang beriman atas nikmat yang diberikan kepada mereka.

UIN SUNAN AMPEL
S U R A B A Y A

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

`

BAB III
METODOLOGI PENELITIAN

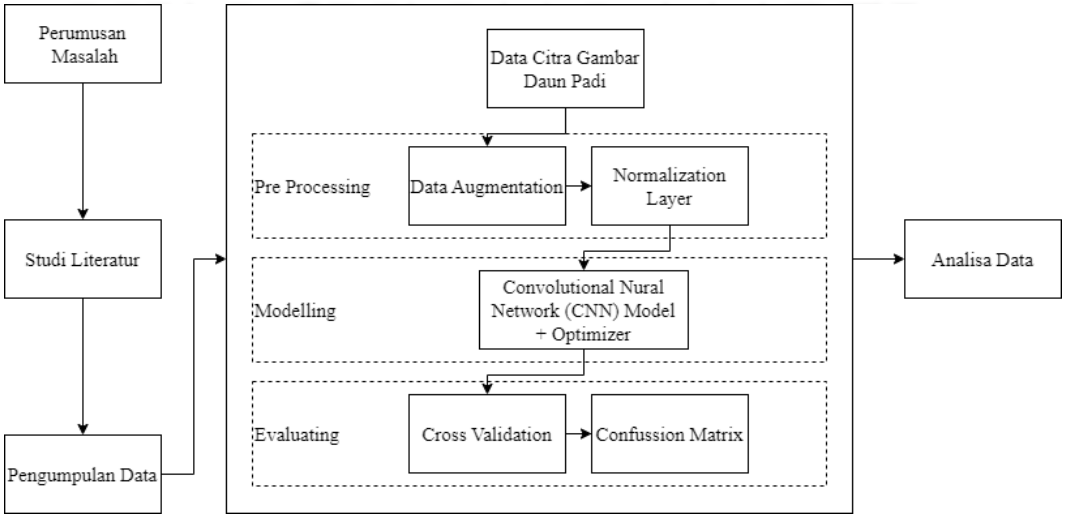
3.1 Jenis Penelitian dan Data

Penelitian ini adalah jenis penelitian kuantitatif dikarenakan data yang diperoleh diukur atau diproses dengan menggunakan teknik matematis. Pada penelitian ini data berupa citra yang berisikan informasi dalam bentuk matriks yang kemudian dilakukan perhitungan matematis. Tujuan dari proses tersebut adalah untuk melakukan klasifikasi sehingga mengetahui jenis penyakit pada daun padi yang diteliti.

Jenis data pada penelitian ini adalah menggunakan tipe jenis data yang bersifat sekunder. Jenis data ini biasa diperoleh melalui media, perantara atau pihak lain. Maksud dari perantara adalah media yang mampu mendukung segala bentuk keperluan dari kredibilitas suatu data seperti buku, literatur dan *website*.

3.2 Tahap Penelitian

Tahap penelitian akan dipaparkan pada media berbentuk diagram alur di bawah. Hal ini dilakukan untuk lebih memudahkan dalam memahami setiap langkah atau proses penelitian secara rinci. Tahap penelitian dapat dilihat pada gambar 3.1.



Gambar 3.1 Tahapan Penelitian

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

3.2.1 Perumusan Masalah

Landasan dari penelitian ini adalah perumusan masalah dengan mengangkat masalah yang terdapat di latar belakang, yaitu klasifikasi penyakit pada daun padi. Klasifikasi ini dilakukan untuk melakukan prediksi dari data yang dimasukkan. Hal ini akan membantu untuk melakukan pencegahan atau penanggulangan sebelum padi memasuki fase penyakit yang lebih buruk.

3.2.2 Studi Literatur

Untuk melakukan pembelajaran terkait topik penelitian, dilakukan proses studi literatur dari berbagai sumber. Pembelajaran yang dilakukan adalah terkait klasifikasi menggunakan *deep learning*, penyakit pada daun padi, *data pre-process*, *modelling* dan evaluasi data. Sumber pembelajaran diperoleh melalui buku, *paper* atau jurnal, dan internet dengan mempertimbangkan kredibilitas dan relevansi informasi yang ada.

3.2.3 Pengumpulan Data

Data pada penelitian ini diperoleh dari situs www.kagle.com yaitu *Rice Leafs Dieses* dengan jumlah data sekunder yang digunakan adalah 3355 berkas dengan masing-masing pembagian 2684 berkas untuk *training* dan 671 berkas untuk *validation*. ini memiliki 4 label untuk setiap kondisi daun yang ada yaitu:

1. *Brown Spot*

   Kondisi daun dimana mengalami perubahan warna bercak coklat memanjang yang sejajar dengan ibu daun tulang yang disebabkan oleh jamur *Drechslera oryzae*.

2. *Hispa*

   Kondisi pada daun padi dimana muncul garis paralel berwarna putih di sepanjang sumbu padi. Gejala lain adalah daun yang menjadi layu. Hal ini disebabkan oleh kumbang yang memakan bagian luar epidermis atas daun padi.

3. *Leaf Blast*

   Kondisi dimana daun terinfeksi oleh jamur *Pyricularia Grisea* yang menyebabkan munculnya bercak coklat pada daun berbentuk belah ketupat.

4. *Healthy*

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

Kondisi dimana daun padi tidak mengalami kelainan apapun dan dinyatakan sehat. Hal ini ditunjukan dengan warna daun yang tidak terdapat perubahan dan kondisi fisik daun yang normal.

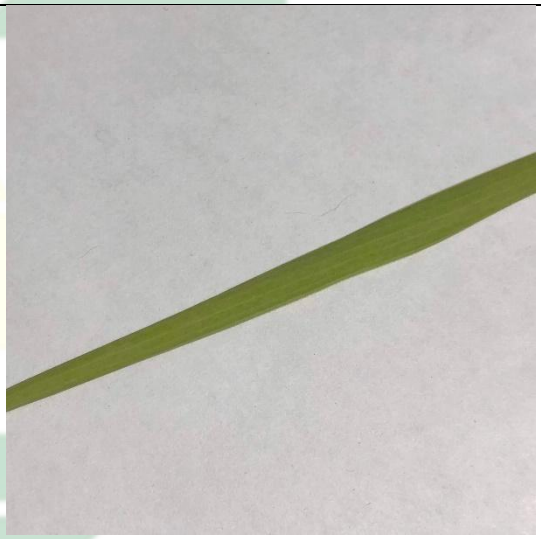
Untuk lebih jelasnya, bentuk dan gambar dari setiap kondisi daun yang dipaparkan dapat dilihat pada tabel 3.1.

Tabel 3.1 Bentuk Kondisi Daun

No	Nama Kondisi Daun	Gambar
1	*Brown Spot*	
2	*Hispa*	

UIN SUNAN
SURABAYA

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

3	*Leaf Blast*	
4	*Healthy*	

3.2.4 Pengolahan Data

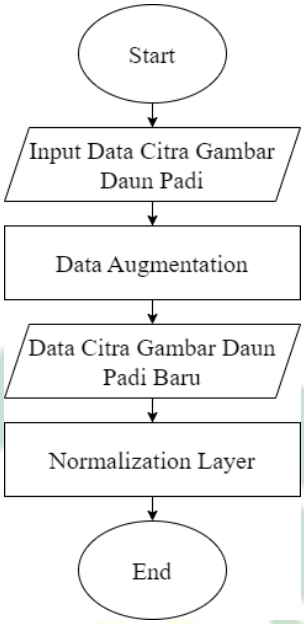
Setelah berhasil memperoleh data yang diinginkan, langkah selanjutnya dalam penelitian ini adalah melakukan pengolahan dari data yang didapat yang meliputi 3 tahap yaitu *pre-processing*, *modelling* dan *evaluating*.

1. *Pre-Processing*

   Tahap ini adalah tahap awal dimana data akan dirapikan sedemikian rupa untuk kelancaran pada tahap selanjutnya. Tahap ini akan menentukan kelayakan data yang diproses. Penelitian ini menggunakan teknik *data augmentation* dan *normalization layer*. Tujuan dari proses ini adalah untuk menyiapkan data agar siap digunakan pada tahap *modelling* serta mengurangi resiko terjadinya *overfitting* pada data selama masa *training.*

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

Langkah-langkah dalam tahap *pre-processing* ini dapat dilihat pada gambar 3.2.



Gambar 3.2 Alur *Preprocessing* Data

Teknik pertama yang digunakan pada tahap pre-processing yaitu *data augmentation*. Jenis teknik *data augmentation* yang digunakan adalah *rotation, translation, flip* dan *contrast*. Teknik ini digunakan untuk mengurangi resiko adanya overfitting dengan cara membuat data tambahan dengan melakukan perubahan pada data asli. Lalu setelah itu ditambahkan *normalization layer*. Teknik ini digunakan untuk menyiapkan *dataset* agar dapat digunakan pada tahap *modelling*.

2. *Modelling*

Tahap ini adalah tahap dimana data yang sudah dilakukan *pre-process* dimasukkan ke dalam *deep learning* untuk dilakukan klasifikasi menggunakan model arsitektur *Convolutional Neural Network* (CNN). Komposisi perbandingan rasio *split* data yaitu 80% untuk *training* dan 20% untuk *validation*.

Selama pemrosesan pada model, dilakukan penambahan fungsi optimasi untuk meningkatkan akurasi prediksi dan mengurangi *loss* pada model dengan melakukan *update* pada setiap iterasi yang disesuaikan dengan *learning rate*.

UIN SUNAN AMPEL
S U R A B A Y A
http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

Untuk model yang digunakan adalah model arsitektur EfficientNet-B0. Pemilihan model ini didasarkan pada peringkat *imagenet bencmark* pada *website* https://paperswithcode.com/sota/image-classification-on-imagenet. Kriteria yang dicari oleh peneliti adalah model dengan *top 1 accuracy* di atas 80% dengan jumlah parameter paling sedikit. Hal ini berkaitan dengan tujuan awal peneliti mengenai model dengan *runtime* dan akurasi yang baik juga dengan parameter yang lebih sedikit.

Sedangkan pemrosesan model dilakukan dengan menggunakan 3 proses yaitu *scratch, pre-train* dan *fine-tuning*, dan *unfreeze layer*. *Scratch Model* adalah proses awal pada tahap pelatihan model dimana model dibuat tanpa menggunakan bobot yang sudah tersedia sebelumnya agar memiliki fleksibilitas pada pengaturan arsitektur yang sesuai dengan *task*.

Proses pre-trained dan fine tunind adalah proses dimana model di *load* dengan bobot yang sudah dilatih sebelumnya untuk membantu model dalam melakukan ekstraksi fitur serta agar dapat disesuaikan dengan *dataset* yang digunakan untuk studi kasus. Pada umumnya, hasil akurasi pada saat validasi selama training akan menjadi lebih baik daripada hasil akurasi training dikarenakan proses ini sangat membantu pengenalan fitur dan pengenalan pada data baru.

Sedangkan *Unfreeze layer* adalah proses lanjutan dari proses *fine tuning* yang dilakukan untuk menyesuaikan model dengan kecepatan pembelajaran. Hal ini ditujukan agar ekstraksi fitur bekerja lebih baik dengan model yang digunakan. Menurut peneliti, proses ini dirasa penting karena penelitian ini menggunakan kumpulan data yang agak berbeda dari kumpulan data dalam *dataset* "`imagenet`". Oleh sebab itu, proses tersebut akan membantu proses ekstraksi fitur lebih optimal dikarenakan memiliki data penyesuaian yang lebih banyak.

3. *Evaluating*

Tahap ini dilakukan untuk mengukur bagaimana akurasi, konsistensi dan kinerja model dalam melakukan prediksi data. Proses ini dilakukan dengan melalui pengujian secara objektif kepada model agar dapat menjamin bahwa

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

model memiliki kemampuan dalam memecahkan permasalahan sesuai kebutuhan.

Evaluasi hasil klasfikasi dilakukan menggunakan teknik *cross validation* dan *confusion matrix* untuk menilai tingkat akurasi dari model arsitektur yang diterapkan. Pengukuran evaluasi akan didasarkan pada nilai yang dihasilkan.

Teknik evaluasi *cross validation* dialkukan dengan cara membagi *dataset* menjadi beberapa bagian. Bagian data yang sudah dibagi kemudian dilakukan iterasi. Hasil akhir evaluasi diambil dari rata-rata hasil dari setiap iterasi dari proses yang dilakukan. Sedangkan confusion matrix ini digunakan untuk mengukur performa dengan melakukan perbandingan prediksi dari model dan label yang sebenarnya dari *dataset*. Untuk memudahkan dalam melakukan analisa, pada umumnya hasil dari *confusion matrix* akan divisualisasikan

Terdapat beberapa skenario pengolahan data yang dilakukan pada penelitian ini yaitu skenario normal yang berisikan konfigurasi kode *modelling* dan evaluasi tanpa *pre-process*, lalu dilanjut dengan penambahan *teknik pre-proccessing data augmentation* untuk skenario *data augmentation*, dan penambahan teknik *pre-process normalizational layer* untuk skenario *normalizational layer*. Dilakukan juga skenario perbandingan fungsi optimasi pada model menggunakan tiga jenis fungsi optimasi umum yaitu SGD, RMSProp, dan Adam untuk melihat pengaruh fungsi ini terhadap kinerja model.

3.2.5 Analisa Data

Analisa yang dilakukan adalah dengan melihat hasil evaluasi menggunakan *confusion matrix* dengan menilai berdasarkan tingkat akurasi dan *runtime* klasifikasi menggunakan model arsitektur *Convolutional Neural Network* (CNN). Sajian dari confusion matrix adalah berupa tabel perhitungan klasifikasi dengan pengukuran evaluasi. Hasil analisa ini nantinya akan dapat melihat seberapa baik metode yang diterapkan untuk melakukan proses klasifikasi.

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

`

BAB IV
HASIL DAN PEMBAHASAN

Setelah dilakukan perumusan masalah, studi literatur dan pengumpulan data, pada bab ini akan dilakukan tahap pengolahan dan analisa data. Tahap pengolahan data dilakukan dengan melakukan *pre-process*, *modelling* dan *evaluating model*. Sedangkan tahap analisa data dilakukan dengan menganalisa hasil dari tahap pengolahan data lalu menarik kesimpulan dari hasil analisa tersebut.

Data yang akan digunakan pada tahap pengolahan data akan dibagi menjadi dua variabel *train* sebagai data *training* dan *val* sebagai data *validation*. Data citra juga dilakukan perubahan pada ukuran *pixel* menggunakan teknik *resize* dengan ukuran 224 x 224 *pixel* untuk semua data. Perubahan ukuran *pixel* dilakukan karena ada kaitannya dengan model yang digunakan pada tahap pengolahan data yang memiliki *default input size* berukuran 224 x 224 *pixel*. Berikut adalah penjelasan lebih detail mengenai tahap pengolahan dan analisa data:

4.1 *Pre-Processing*

Tahap *pre-process* dilakukan kepada keseluruhan citra yang dimasukan sebelumnya dengan menggunakan teknik yaitu *data augmentation* dan *normalization layer*.

4.1.1 *Data Augmentation*

Teknik pertama yang digunakan pada tahap *pre-processing* yaitu *data augmentation*. Berikut adalah konfigurasi kode untuk proses *data augmentation*:

```
data_augmentation = tf.keras.Sequential(
    [
        tf.keras.layers.RandomRotation(factor=0.15),
        tf.keras.layers.RandomTranslation(height_factor=0.1, w
        idth_factor=0.1),
        tf.keras.layers.RandomFlip(),
        tf.keras.layers.RandomContrast(factor=0.1),
    ],
    name="img_augmentation",
```

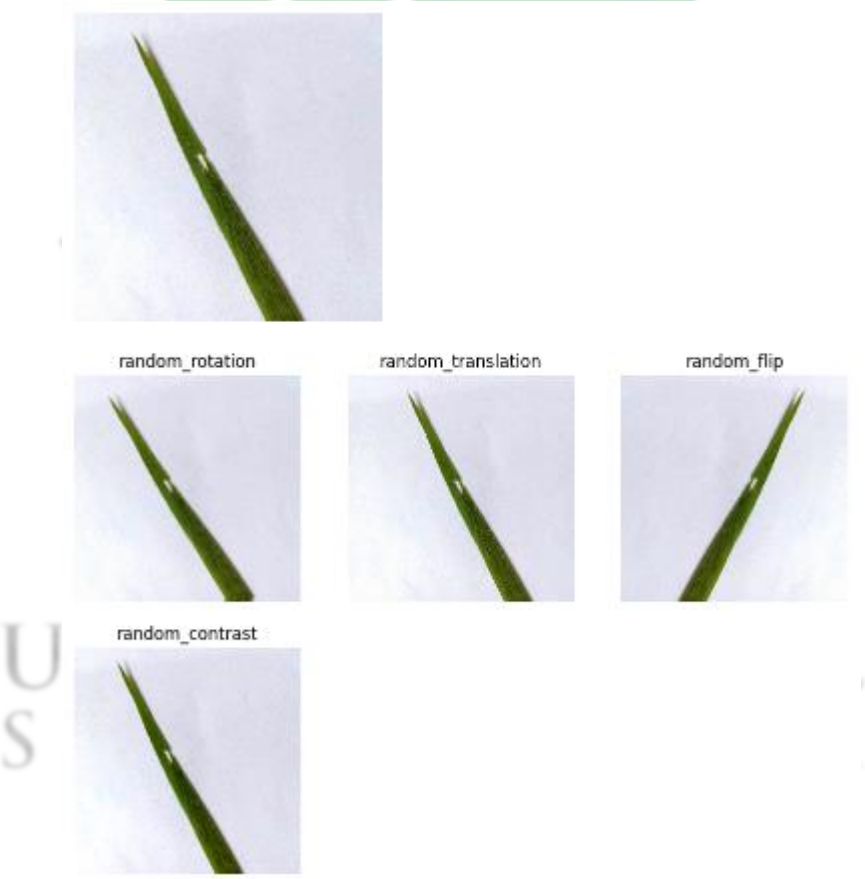
UIN SUNAN AMPEL
S U R A B A Y A

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

```
)
```

Kode di atas digunakan untuk membuat fungsi dari *data augmentation* dengan nama "`data_augmentation`" menggunakan *TensorFlow* dengan *Sequential API*. Secara singkat, kode di atas menggunakan beberapa teknik modifikasi data yaitu dengan melakukan rotasi pada kisaran 15 derajat, pergeseran acak pada kisaran 10 derajat, pemutaran acak secara vertikal atau horizontal dan perubahan kontras pada kisaran 10 derajat.

Untuk mengetahui bagaimana pengaruh fungsi *data augmentation* pada *dataset* yang digunakan dapat dilihat pada gambar berikut:



Gambar 4.1 Contoh Citra yang Telah dilakukan *Data Augmentation*

Gambar di atas adalah contoh bagaimana teknik dari *data augmentation* bekerja pada data citra yang melewati proses ini. Setiap data baru dari hasil teknik ini juga akan membantu memperkecil kemungkinan dari adanya resiko *overfitting*.

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

4.1.2 *Normalization Layer*

Teknik kedua yang digunakan adalah *normalization layer*. Berikut adalah konfigurasi kode untuk proses *normalization layer*:

```
normalization_layer = layers.experimental.preprocessing.Rescal
                      ing(1./256)
normalized_ds = train_ds.map(lambda x, y: (normalization_layer
                (x),y))
image_batch, labels_batch = next(iter(normalized_ds))
first_image = image_batch[0]
```

Secara ringkas, kode diatas digunakan untuk membuat layer normalisasi, menerapkan normalisasi pada dataset pelatihan, dan mengambil gambar pertama dari dataset yang telah dinormalisasi setelah diperiksa apakah proses normalisasi sudah berjalan dengan baik.

Dikarenakan proses *normalization layer* ini memakan cukup banyak waktu, dilakukan proses yang dapat mengoptimalkan *data loading* yaitu dengan menggunakan "`tf.data.AUTOTUNE`". Proses ini bekerja dengan cara menyesuaikan proses thread pada saat *data loading* dengan jumlah CPU yang tersedia.

4.2 Modelling

Data yang sudah melalui tahap *pre-processing*, selanjutnya akan masuk ke tahap *modelling*. Tahap ini terdiri dari tahap pemilihan model dan pelatihan model. Untuk pemilihan model, EfficientNetB0 menjadi model yang dipilih untuk digunakan pada tahap *modelling*. Lalu untuk proses pelatihan model, peneliti melakukan proses *scratch*, *pre-train* dan *fine-tuning*.

4.2.1 *Scratch Model*

Proses pemodelan yang pertama dilakukan adalah proses *scratch*. Berikut adalah konfigurasi kode proses *scratch model* pada model EfficientNetB0:

```
from tensorflow.keras.applications import EfficientNetB0
strategy = tf.distribute.MirroredStrategy()
with strategy.scope():
    inputs = layers.Input(shape=(IMG_SIZE, IMG_SIZE, 3))
```


http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

```
    x = data_augmentation(inputs)
    outputs = EfficientNetB0(include_top=True, weights=None, c
              lasses=NUM_CLASSES)(x)
    model = tf.keras.Model(inputs, outputs)
    model.compile(optimizer=tf.keras.optimizers.Adam(learning_
                  rate=LEARNING_RATE,
                  loss='categorical_crossentropy',
                  metrics=['accuracy'])
```

Secara singkat, kode diatas digunakan untuk membuat sebuah model menggunakan arsitektur EfficientNetB0 disertai proses *parallel processing* dengan input layer sesuai variabel "`IMG_SIZE`" dan 3 channel. Selain itu, teknik *data augmentation* juga ditambahkan ke dalam model lalu memasukkannya ke dalam variabel *output*. Setelah itu model di compile menggunakan *optimizer* Adam, *loss categorical crossentrophy function* dan *metric accuracy* untuk mengevaluasi proses *training*. Pada proses ini, peneliti memilih untuk tidak melakukan proses *fitting* pada *model*.

4.2.2 *Pre-Train* dan *Fine Tuning*

Setelah dilakukan *scratch model*, selanjutnya adalah proses *pre-train* dan *fine tuning*. Proses ini dapat dilakukan dengan mengunduh model *pre-train* dari *library* yang ada. Pada konfigurasi kode di bawah, peneliti menginisiasi model dengan bobot menggunakan *dataset* "`imagenet`" yang telah dilatih sebelumnya.

```
def build_model(num_classes, input_shape):
    inputs = layers.Input(shape=input_shape)
    x = inputs
    model = EfficientNetB0(include_top=False, input_tensor=x,
            weights="imagenet")

    # Freeze the pretrained weights
    model.trainable = False

    # Rebuild top
    x = layers.GlobalAveragePooling2D(name="avg_pool")(model.o
        utput)
    x = layers.BatchNormalization()(x)

    top_dropout_rate = 0.2
```


http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

```
    x = layers.Dropout(top_dropout_rate, name="top_dropout")(x
)
    outputs = layers.Dense(num_classes, activation="softmax",
              name="pred")(x)

    # Compile
    model = tf.keras.Model(inputs, outputs, name="EfficientNet
")
    model.compile(optimizer=tf.keras.optimizers.Adam(learning_
                  rate=LEARNING_RATE,
                  loss='categorical_crossentropy',
                  metrics=['accuracy'])

    return model
```

Secara singkat, maksud dari konfigurasi kode diatas adalah untuk membuat fungsi *pre-train* bernama "`build_model`" dengan dua parameter yaitu variabel "`num_classes`" sebagai jumlah *class* pada *dataset* dan "`input_shape`" sebagai ukuran *input* data yang dibutuhkan. Untuk mempermudah proses *transfer learning*, proses *training* hanya dilakukan pada *top layer* dengan cara melakukan *freeze layer*.

Penambahan *pooling*, *normalization layer* dan fungsi *dropout* juga dilakukan untuk lebih mengurangi resiko *overfitting* pada model. Sedangkan *softmax* digunakan untuk melakukan perhitungan probabilitas hasil *output*. Setelah itu dilakukan proses *compile* yang sama seperti pada proses *scratch model* sebelumnya dengan tujuan yang sama.

Setelah melakukan *pre-train*, langkah selanjutnya adalah melakukan *fine-tuning* dengan *dataset* baru menggunakan konfigurasi kode berikut:

```
with strategy.scope():
    model = build_model(num_classes=NUM_CLASSES, input_shape=(
            IMG_SIZE, IMG_SIZE, 3))
epochs = 40
hist = model.fit(train_ds, epochs=epochs, validation_data=val_
       ds, verbose=2)
```

Maksud dari kode di atas yaitu proses *fine-tuning* dilakukan menggunakan fungsi "`build_model`" dengan bantuan fungsi "`strategy.scope()`" agar semua variabel dapat direplika dan memungkinkan dilakukan *parallel process*. Untuk jumlah iterasi yang ditetapkan pada proses ini adalah sebanyak 40 kali iterasi pada

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

proses *training model* dengan informasi progres yang detail menggunakan *verbose*. Setelah itu, hasil dari setiap iterasi akan di plot dengan menggunakan fungsi "`plot_hist`" dengan konfigurasi kode berikut:

```
def plot_hist(hist):
    plt.plot(hist.history["accuracy"])
    plt.plot(hist.history["val_accuracy"])
    plt.title("model accuracy")
    plt.ylabel("accuracy")
    plt.xlabel("epoch")
    plt.legend(["train", "validation"], loc="upper right")
    plt.show()
plot_hist(hist)
```

Kode di atas menunjukan bahwa visualisasi plot akan berisikan sejumlah informasi seperti *train accuracy, validation accuracy,* judul plot, label plot dan legenda plot.

4.2.3 *Unfreeze Layer*

Proses terakhir adalah proses *unfreeze layer*. Berikut adalah konfigurasi kode untuk proses *unfreeze model*:

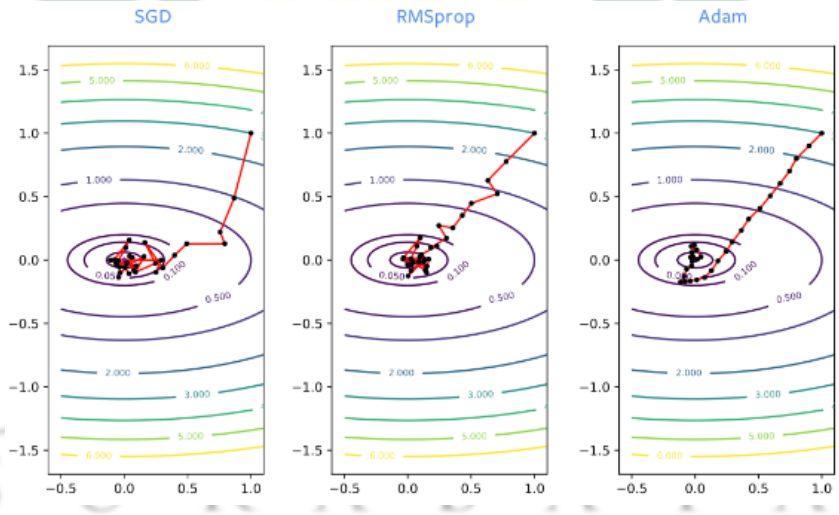
```
def unfreeze_model(model):
    # We unfreeze the top 20 layers while leaving BatchNorm la
      yers frozen
    for layer in model.layers[-20:]:
        if not isinstance(layer, layers.BatchNormalization):
            layer.trainable = True
    model.compile(optimizer=tf.keras.optimizers.Adam(learning_
                  rate=LEARNING_RATE,
                  loss='categorical_crossentropy',
                  metrics=['accuracy'])
unfreeze_model(model)
epochs = 30
hist = model.fit(train_ds, epochs=epochs, validation_data=val_
                 ds, verbose=2)
plot_hist(hist)
```


http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

Secara singkat, kode diatas digunakan untuk membuat fungsi proses *unfreeze layer* pada fungsi "`build_model()`". *Layer* yang di *unfreeze* yaitu 20 *layer* terakhir pada model dengan tanpa menyertakan *layer batch normalization.* Selanjutnya model di *compile* sama seperti proses pada fungsi model sebelumnya. Setelah itu, fungsi *"*`unfreeze_model`*"* dipanggil dan di *training* ulang untuk mengupdate bobot pada seluruh *layer* model. Jumlah iterasi yang ditetapkan pada proses ini adalah sebanyak 30 kali pada dengan informasi progres yang detail menggunakan *verbose.* Setelah itu, dilakukan visualisasi akurasi menggunakan fungsi "`plot_hist`".

4.2.4 *Optimizer*

Penelitian ini menggunakan optimasi Adam untuk 3 skenario awal dan melakukan perbandingan 3 optimasi setelah itu menggunakan fungsi optimasi SGD, RMSprop dan Adam. Fungsi optimasi tersebut dapat diakses melalui library "`tf.keras. optimizers` ".



Gambar 4.2 SGD vs RMSprop vs Adam

Sumber: (*Optimisation Techniques II · Deep Learning*, t.t.)

Saat melakukan proses *training* pada model, SGD seringkali melakukan kesalahan arah untuk menuju konvergensi. Sedangkan RMSprop sering kali lebih unggul dalam permasalahan ini namun terkadang saat mendekati minimum konvergensi, arah optimasi ini sering memantul. Sedangkan Adam mampu menemukan arah yang tepat dengan konsistensi dan minim gangguan saat menuju

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

minimum konvergensi. Oleh sebab itu, dengan kemampuannya yang efisien dan stabil Adam lebih direkomendasikan daripada RMSprop dan SGD.

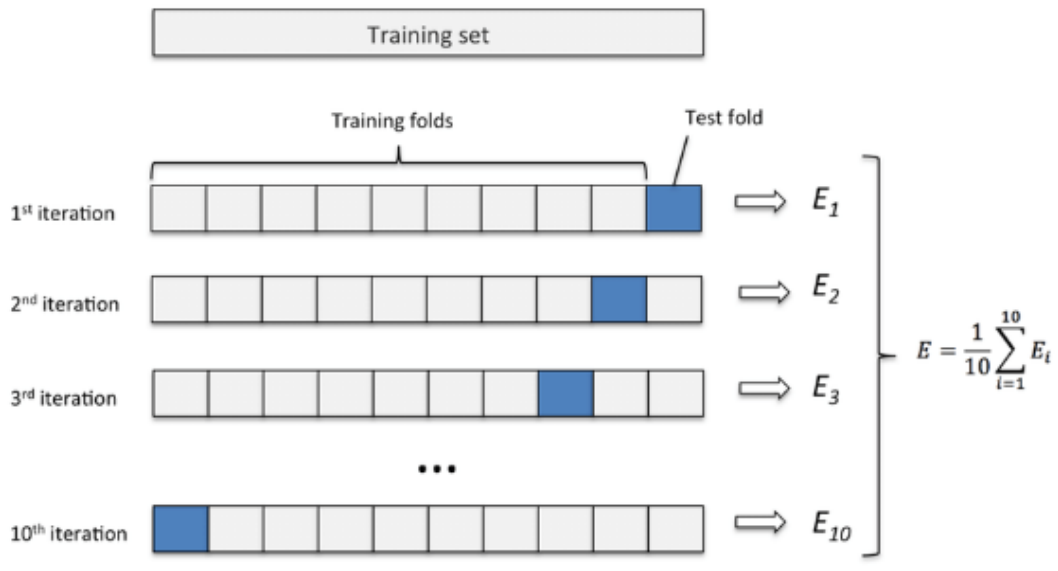
Menurut gambar di atas, secara keseluruhan kelebihan dari SGD ada pada sifatnya yang *simple* dan stabil, lalu kelebihan dari RMSprop ada pada kemampuannya dalam mengatasi permasalahan gradien rata-rata yang kecil, dan kelebihan Adam adalah mampu mencapai konvergensi yang lebih cepat dan stabil. Pemilihan fungsi optimasi ini tergantung pada karakteristik data dan model yang digunakan.

4.3 *Evaluating*

Tahap *evaluating* dilakukan setelah tahap *modelling*. Penelitian ini menggunakan dua teknik *evaluating* yaitu *cross validation* untuk menilai kinerja proses dan *confusion matrix* untuk melihat gambaran jumlah data yang telah diklasifikasi dengan tepat atau tidak pada setiap *class* yang ada.

4.3.1 *Cross Validation*

Penelitian ini menggunakan *k-fold cross validation* sebagai salah satu jenis dari teknik *cross validation*. Jenis *cross validation* ini membagi *dataset* menjadi "k" dengan ukuran yang sama. Maksud dari "k" disini adalah jumlah iterasi yang prinsipnya sama dengan penjelasan sebelumnya. Selama masa iterasi, pemilihan data *train* dan *testing* akan diurut hingga semua bagian data pernah menjadi data *testing*.



http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

Gambar 4.3 Ilustrasi K-Fold Cross Validation

Sumber: (Ashfaque Maat & Iqbal, 2019)

Jumlah iterasi atau "k" yang digunakan untuk penelitian ini berjumlah 5. Dengan kata lain, peneliti menetapkan 5 kali iterasi atau membagi data menjadi 5 bagian untuk evaluasi model.

4.3.2 *Confusion Matrix*

Pada penelitian ini, proses ini dilakukan menggunakan library "`seaborn`". Selain itu, teknik ini juga mampu menghitung nilai *accuracy, precision, recall* dan *f1 score* untuk menilai seberapa baik model dalam memprediksi data menggunakan *library* dari "`sklearn.metrics`". Pada proses *cross validation*, hasil perhitungan nilai *accuracy, precision, recall* dan f*1 score* dapat dicari menggunakan perhitungan dari setiap iterasi lalu merata-rata nilai yang dihasilkan.

4.4 Skenario Implementasi

Setelah penjelasan mengenai alur implementasi yang terdapat pada sub bab sebelumnya, pada sub bab ini akan dipaparkan mengenai seluruh proses implementasi pada beberapa skenario yang ditetapkan oleh peneliti. Beberapa skenario yang ditetapkan oleh peneliti diantaranya skenario normal, skenario *data augmentation,* skenario *normalization layer*, dan skenario *optimizer*. Pada implementasi ini, mesin yang digunakan untuk menjalankan konfigurasi kode adalah CPU dan bukan GPU. Selain itu, digunakan beberapa *library* untuk membantu proses penulisan kode diantaranya `tensorflow, matplotlib, os, random, seaborn, numpy, PIL, sklearn.metrics,` dan `keras`. Untuk *dataset* yang digunakan akan diimpor terlebih dahulu sebagai 2 variabel menggunakan konfigurasi kode:

```
train_dir = ...\\RiceLeaf \\train'
val_dir = '...\\RiceLeaf \\validation'
```

Setelah proses impor, selanjutnya dilakukan inisiasi beberapa variabel kunci diantaranya variabel `IMG_SIZE = 224, NUM_CLASSES = 4, batch_size = 64,` dan `LEARNING_RATE = 0,001`.

4.4.1 Skenario Normal

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

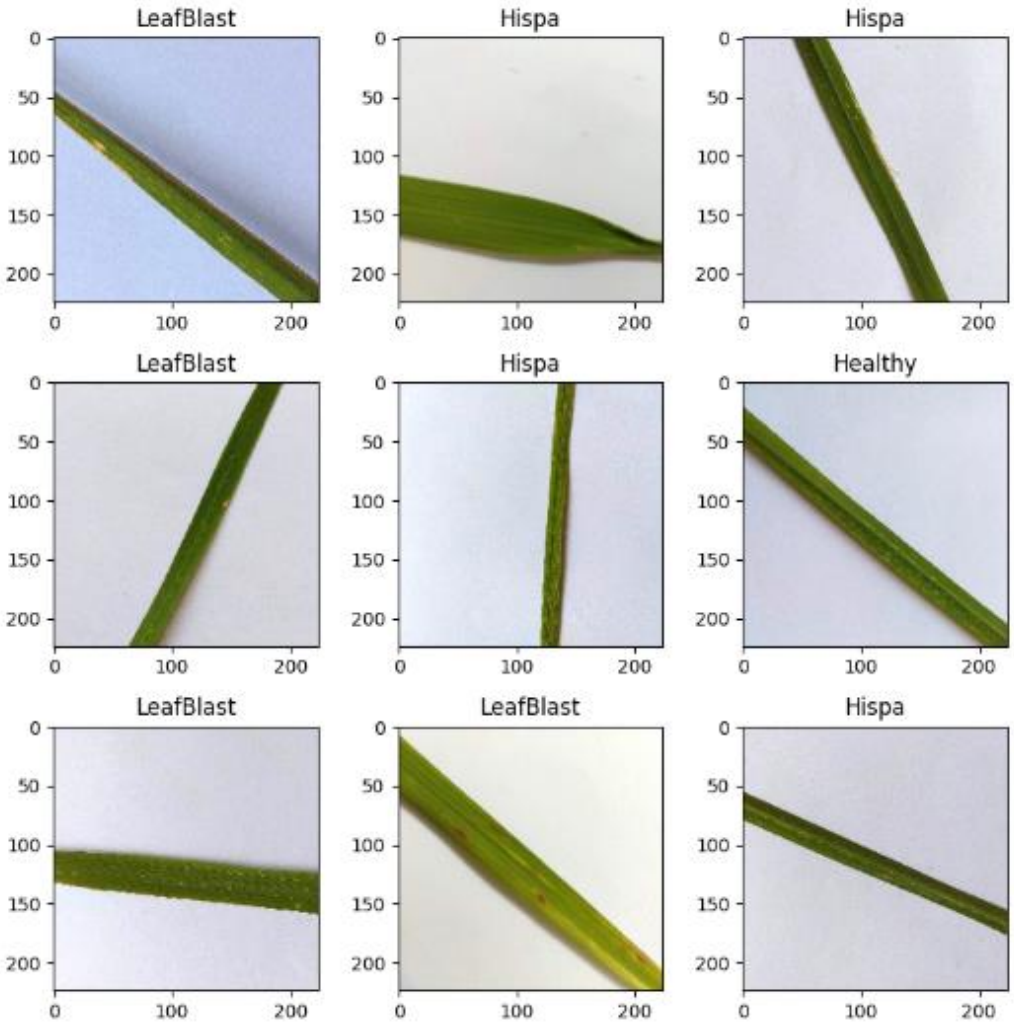
Skenario ini diawali dengan memuat *dataset* dengan format yang cocok untuk *training model* agar dapat digunakan pada tahap *modelling* nantinya menggunakan konfigurasi kode sebagai berikut:

```
train_ds = tf.keras.preprocessing.image_dataset_from_
directory(
    train_dir,
    labels="inferred",
    label_mode="categorical",
    image_size=(IMG_SIZE, IMG_SIZE),
    batch_size=batch_size,
)
val_ds = tf.keras.preprocessing.image_dataset_from_directory(
    val_dir,
    labels="inferred",
    label_mode="categorical",
    image_size=(IMG_SIZE, IMG_SIZE),
    batch_size=batch_size,
)
class_names = train_ds.class_names
print(class_names)
```

Kode di atas menunjukan adanya inisiasi dua variabel untuk memuat *dataset* dengan beberapa argumen yaitu `labels`, `label_mode`, `image_size` dan `batch_size`. Setelah memuat *dataset*, kemudian dilakukan inisiasi variabel `class_names` untuk mengetahui kesesuaian antara nama *class* yang dimuat dengan nama *class* pada *dataset*. berikut adalah tampilan dari beberapa data beserta label dan ukuran *pixel* dari *dataset* yang telah dimuat:

UIN SUNAN AMPEL
S U R A B A Y A

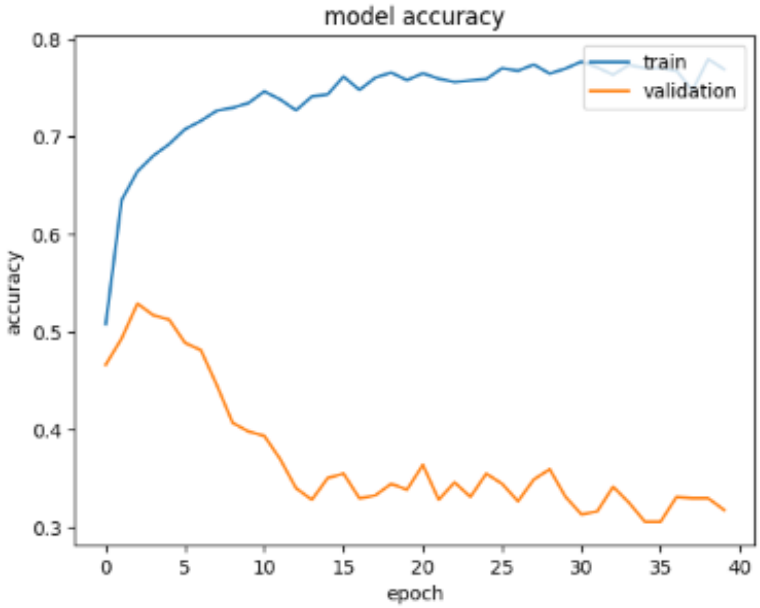
http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/



Gambar 4.4 Tampilan Beberapa Data dari *Dataset*

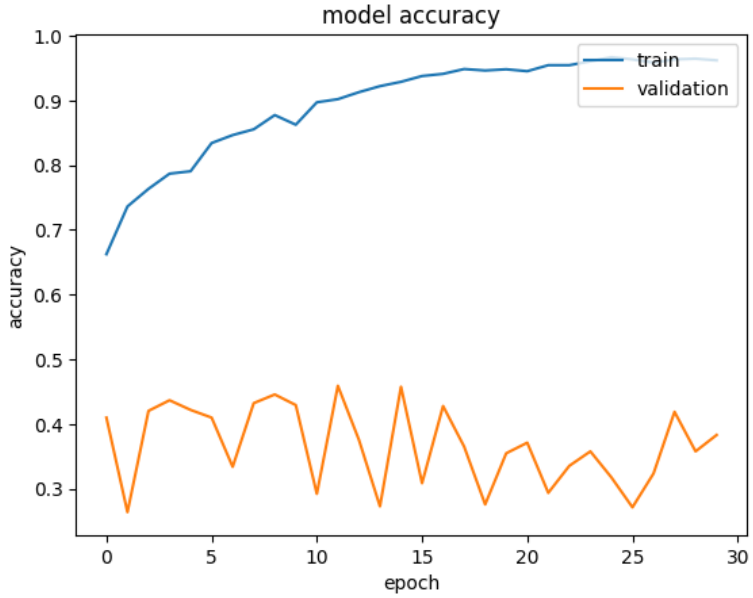
Setelah tahap memuat *dataset* selesai, tahap selanjutnya adalah tahap *modelling* dengan melalui urutan alur proses *scratch model*, *pre-train* dan *fine-tuning* yang memiliki konfigurasi kode sama seperti pada sub bab 4.2. Hasil *training* menggunakan konfigurasi kode pada proses *pre-train* dan *fine tuning* menunjukan bahwa model mengalami *overfitting* yang ditandai dengan adanya kenaikan akurasi dan penurunan *loss* pada waktu *train* sedangkan penurunan akurasi dan kenaikan *loss* waktu *validation*. Untuk mengetahui lebih detail, berikut adalah *accuracy plot* dari proses *training model* menggunakan konfigurasi kode *pre-train* dan *fine tuning* pada skenario normal:

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/



Gambar 4.5 *Accuracy Plot Pre-Train* dan *Fine Tuning* Skenario Normal

Untuk mengetahui apakah performa dan akurasi model dapat lebih ditingkatkan, dilakukanlah proses penyesuaian model dengan *learning rate* menggunakan proses *unfreeze layer* sebagai lanjutan dari proses *fine tuning*. Berikut adalah *accuracy plot* menggunakan konfigurasi kode *unfreeze layer* pada skenario normal:

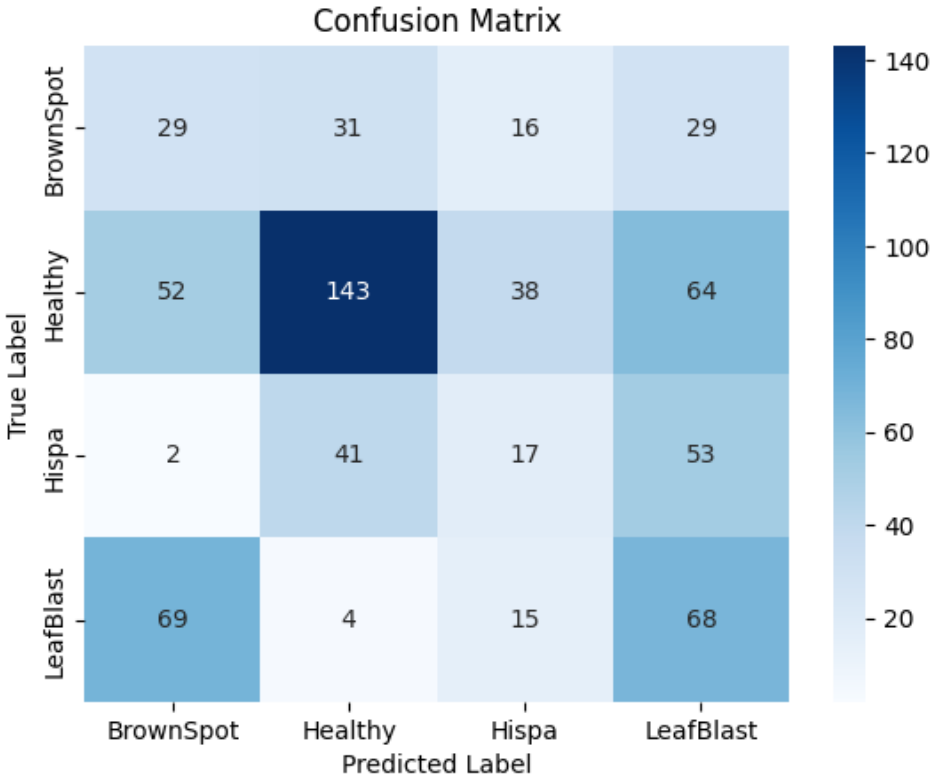


Gambar 4.6 *Accuracy Plot Unfreeze Layer* Skenario Normal

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

Setelah dilakukan proses *training* menggunakan konfigurasi kode pada proses *unfreeze layer*, hasil iterasi menunjukkan indikasi bahwa model masih mengalami *overfitting*. Namun pada hasil *training* kali ini memiliki perbedaan dari hasil *training* menggunakan konfigurasi kode sebelumnya. Pada hasil proses *training* kali ini, akurasi dan *loss* waktu *validation* tidak serta merta mengalami penurunan secara terus menerus. Namun terjadi kondisi dimana akurasi dan *loss* waktu *validation* mengalami kondisi naik dan turun yang cukup berjarak. Hal ini menandakan bahwa konfigurasi kode *unfreeze layer* memiliki pengaruh pada *overfitting* yang terjadi meskipun belum mampu menghindarkan dari terjadinya *overfit*.

Berdasarkan hasil *training* pada *unfreeze layer*, diperoleh nilai *accuracy* 69%, *precision* 33%, *recall* 33% dan *f1-score* 33% . Hasil prediksi label dengan label sebenarnya dapat dilihat pada visualisasi *confusion matrix* berikut:



Gambar 4.7 *Confusion Matrix Unfreeze Layer* Skenario Normal

Pada gambar di atas menunjukkan bahwa prediksi dari label dan label yang sebenarnya masih banyak terjadi kesalahan. Label dengan akurasi prediksi paling tinggi yaitu pada *class "healty*" dan label dengan akurasi prediksi paling rendah

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

yaitu pada *class "hispa*". Meskipun tingkat akurasi sudah di atas 65%, namun hal ini dirasa masih kurang dikarenakan akurasi yang diharapkan adalah sekitar 80% sesuai dengan pernyataan dari top 1 *accuracy* pada *website imagenet benchmark*. Selain itu permasalahan *overfitting* juga mengakibatkan perhitungan nilai *f1-score* menjadi rendah. Hal ini menandakan adanya ketidakseimbangan pada *class.*

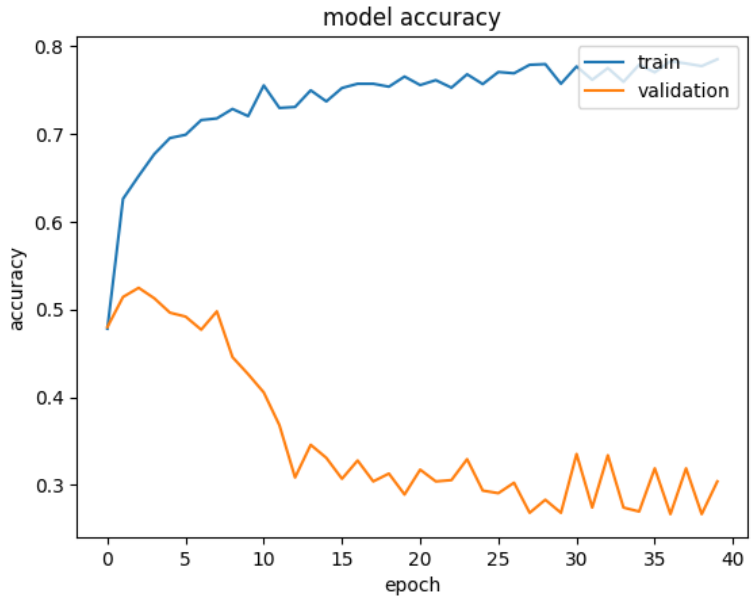
4.4.2 Skenario *Data Augmentation*

Setelah mengetahui hasil dari tahap *modelling* pada skenario normal, perlu dilakukan peningkatan pada akurasi dan performa model dengan menambahkan salah satu konfigurasi kode *pre-processing* yaitu *data augmentation*. Hal ini diharapkan mampu lebih membantu model menghindari *overfitting* pada waktu *training model*. konfigurasi kode ini ditambahkan setelah proses memuat *dataset* dan sebelum tahap *modelling*. konfigurasi kode proses *data augmentation* yang dipakai pada skenario ini sama dengan konfigurasi kode proses *data augmentation* pada sub bab 4.1.

Setelah masuk ke tahap *modelling* dan dilakukan proses *training model* menggunakan konfigurasi *kode pre-train* dan *fine tuning*, diketahui bahwa data masih terindikasi mengalami *overfitting* dengan gejala yang hampir sama seperti skenario sebelumnya. Tidak banyak perubahan yang terjadi setelah penambahan proses *data augmentation* ini. Untuk mengetahui lebih detail, berikut adalah *accuracy plot* dengan konfigurasi kode *pre-train* dan *fine tuning* pada skenario *data augmentation:*

UIN SUNAN AMPEL
S U R A B A Y A

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

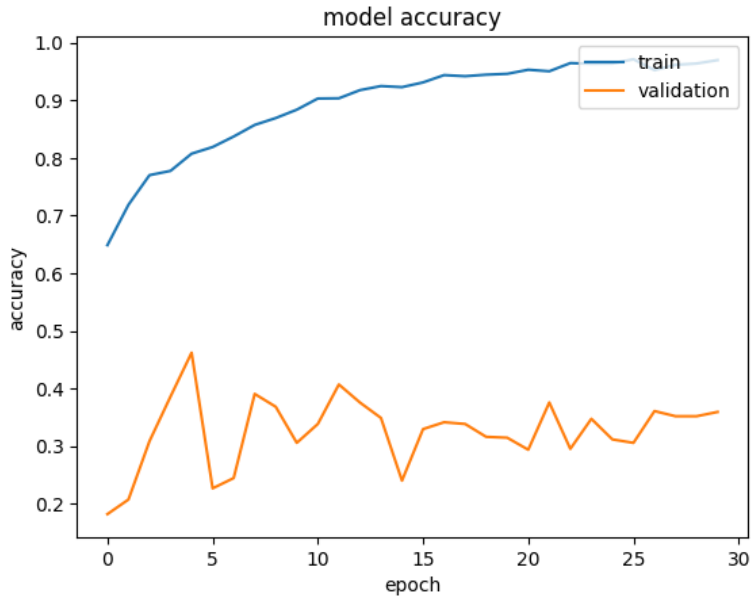


Gambar 4.8 *Accuracy Plot Pre-Train* dan *Fine Tuning* Skenario *Data Augmentation*

Jika diperhatikan, *accuracy plot* pada skenario ini dirasa mengalami penurunan daripada *accuracy plot* pada skenario sebelumnya. Hasil *accuracy plot* pada skenario sebelumnya masih berada di atas 30% untuk akurasi paling rendah. Sedangkan pada skenario ini, *accuracy plot* sampai berada di bawah 30% untuk akurasi paling rendah. Hal ini mungkin terjadi karena model semakin kesulitan dalam proses pengenalan fitur disebabkan oleh data baru yang diterima dari proses *data augmentation* tidak seimbang atau terdistribusi dengan baik. Oleh sebab itu, hasil dari proses *training* menjadi menurun dikarenakan pengenalan fitur data baru lebih kompleks daripada data asli.

Sama seperti skenario sebelumnya, akan dilakukan proses *unfreeze layer* untuk melakukan peningkatan performa dan akurasi pada model. Untuk lebih detail, berikut adalah *accuracy plot* proses *training* menggunakan konfigurasi kode *unfreeze layer* pada skenario *data augmentation*:

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/



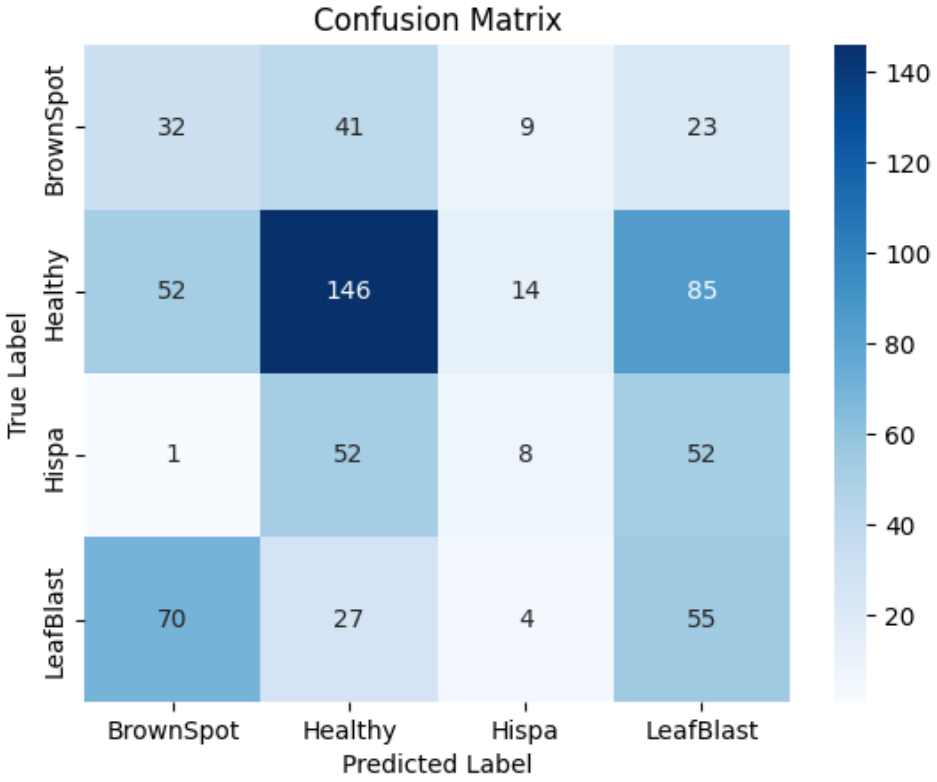
Gambar 4.9 *Accuracy Plot Unfreeze Layer* Skenario *Data Augmentation*

Setelah melihat hasil *accuracy plot* pada proses *training* kali ini, dapat dibandingkan bahwa lonjakan naik turun yang terjadi pada proses *training* kali ini lebih jarang daripada waktu proses *training* pada skenario sebelumnya. Hal ini menandakan bahwa dengan menambahkan fungsi *data augmentation* masih belum cukup untuk menghindarkan dari adanya *overfitting*.

Berdasarkan hasil dari proses *training* pada konfigurasi kode *unfreeze layer* kali ini, ditemui hasil perhitungan nilai *accuracy* 67%, *precision* 30%, *recall* 30% dan *f1-score* 30% . Hasil prediksi label dengan label sebenarnya dapat dilihat pada visualisasi *confusion matrix* berikut:

UIN SUNAN AMPEL
S U R A B A Y A

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/



Gambar 4.10 *Confusion Matrix Unfreeze Layer* Skenario Data Augmentation

Berdasarkan visualisasi *confusion matrix* dari skenario kali ini dan skenario sebelumnya, prediksi label dengan label sebenarnya masih dirasa kurang. Bahkan pada skenario ini, hasil prediksi lebih menurun daripada hasil prediksi sebelumnya. Oleh sebab itu, perlu dilakukan skenario lanjutan untuk menormalkan distribusi data.

4.4.3 Skenario *Normalization Layer*

Berdasarkan hasil dari tahap *modelling* dua skenario sebelumnya, hasil yang diterima masih dirasa berada di bawah ekspektasi. Setelah pada skenario sebelumnya ditambahkan *data augmentation*, pada skenario kali ini akan ditambahkan jenis *pre-process*ing lain yaitu proses *normalization layer*. Hasil dari skenario sebelumnya yang masih mengalami overfitting disebabkan oleh kurangnya performa ekstraksi fitur, pada skenario kali ini akan diperbaiki menggunakan teknik *normalization layer*. Selain itu, teknik ini juga membantu distribusi data dengan mempermudah proses ekstraksi fitur menggunakan cara pengurangan kovarian pada data.

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

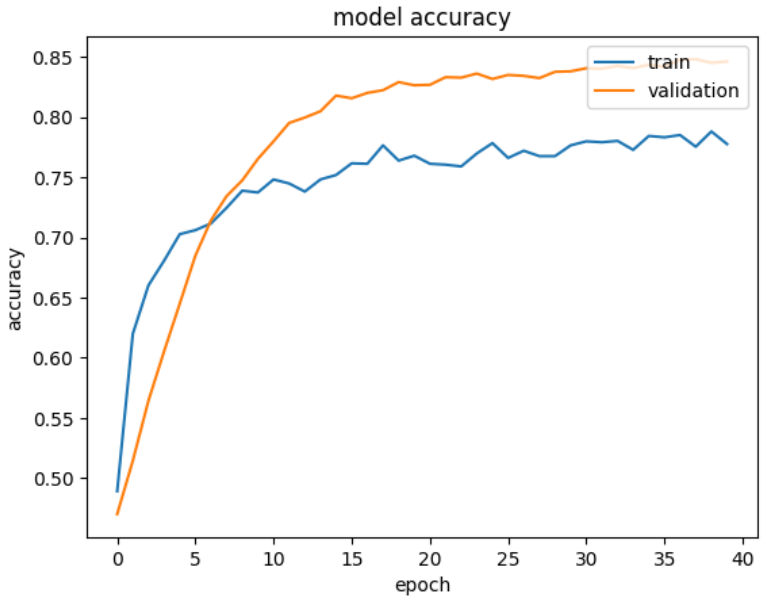
Teknik ini dirasa cocok untuk penelitian ini dikarenakan teknik ini memiliki pengaruh yang baik saat digunakan pada data yang memiliki banyak variasi seperti *dataset* yang digunakan. Dikarenakan data yang digunakan cukup besar dan banyak, perlu dilakukan optimasi pada proses *data loading* sebelum memasuki proses *normalization layer*. Proses ini dilakukan menggunakan "`tf.data.AUTOTUNE`" dengan konfigurasi kode sebagai berikut:

```
UTOTUNE = tf.data.AUTOTUNE
train_ds =
train_ds.cache().shuffle(1000).prefetch(buffer_size=UTOTUNE)
val_ds = train_ds =
train_ds.cache().prefetch(buffer_size=UTOTUNE)
```

Secara singkat, konfigurasi kode di atas adalah kode untuk melakukan beberapa operasi pada *dataset* seperti penyimpanan *cache* dalam memori untuk meningkatkan akses data selama *training*, pengacakan data dan *prefetch* data untuk mempercepat pemrosesan data. Baru setelah itu data dimasukkan ke dalam proses *normalization layer*. Konfigurasi kode proses *normalization layer* pada proses ini adalah konfigurasi kode yang sama seperti pada sub bab 4.1.

Setelah melewati proses *training* pada tahap *modelling* menggunakan konfigurasi kode *pre-train* dan *fine tuning*, terjadi peningkatan performa yang sangat jauh dibandingkan sebelumnya. Pada skenario ini, permasalahan *overfitting* telah berhasil diatasi dan akurasi juga melonjak sangat tinggi dengan akurasi *training* mencapai angka hampir 80% dan akurasi *validation* mencapai angka di atas 80%. Untuk lebih detail, berikut adalah *accuracy plot* proses *training model* menggunakan konfigurasi kode *pre-train* dan *fine tuning* pada skenario *normalization layer*:

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

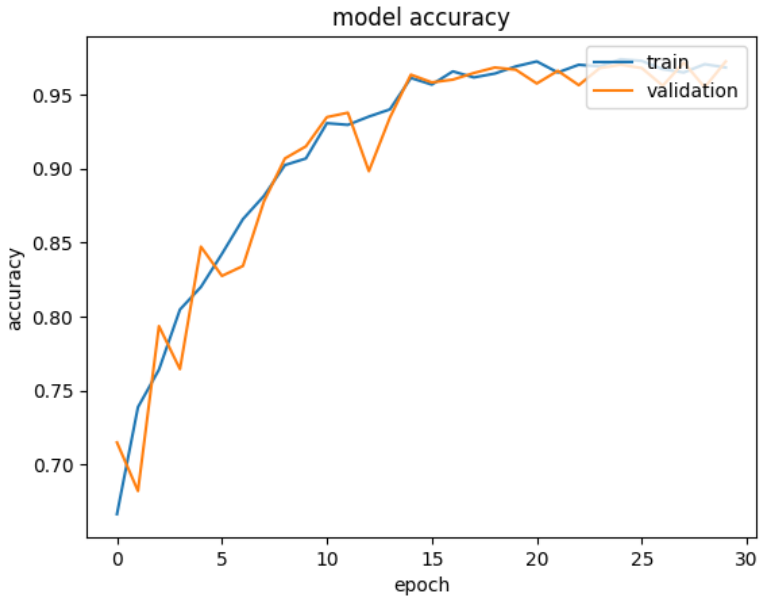


Gambar 4.11 *Accuracy Plot Pre-Train* dan *Fine Tuning* Skenario *Normalization Layer*

Pada gambar di atas, dapat dilihat bahwa akurasi mengalami peningkatan pesat di awal dan selanjutnya mengalami peningkatan konstan hingga akhir iterasi. Hal ini menunjukan bahwa pendistribusian dan *data loading* menjadi hal penting selama masa persiapan data sebelum masuk ke tahap *modelling* pada penelitian ini. Terbukti dengan hasil dari proses *training* pada skenario ini yang memiliki perbandingan yang jauh lebih baik daripada hasil *training* pada skenario sebelumnya.

Jika dilihat kembali, hasil *accuracy plot* di atas menunjukan bahwa akurasi pada saat *validation* lebih tinggi daripada akurasi pada saat *training*. Hal ini bisa jadi karena proses ekstraksi data *validation* menjadi lebih muda dikenali daripada data *training* akibat proses distribusi. Untuk melihat apakah hasil *training* ini masih dapat ditingkatkan, proses ini kemudian dilanjut ke proses *training* menggunakan konfigurasi kode *unfreeze layer*. Untuk melihat hasil dari proses *training*, berikut adalah *accuracy plot* dari proses *training model* menggunakan konfigurasi kode *unfreeze layer* pada skenario *normalization layer*:

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

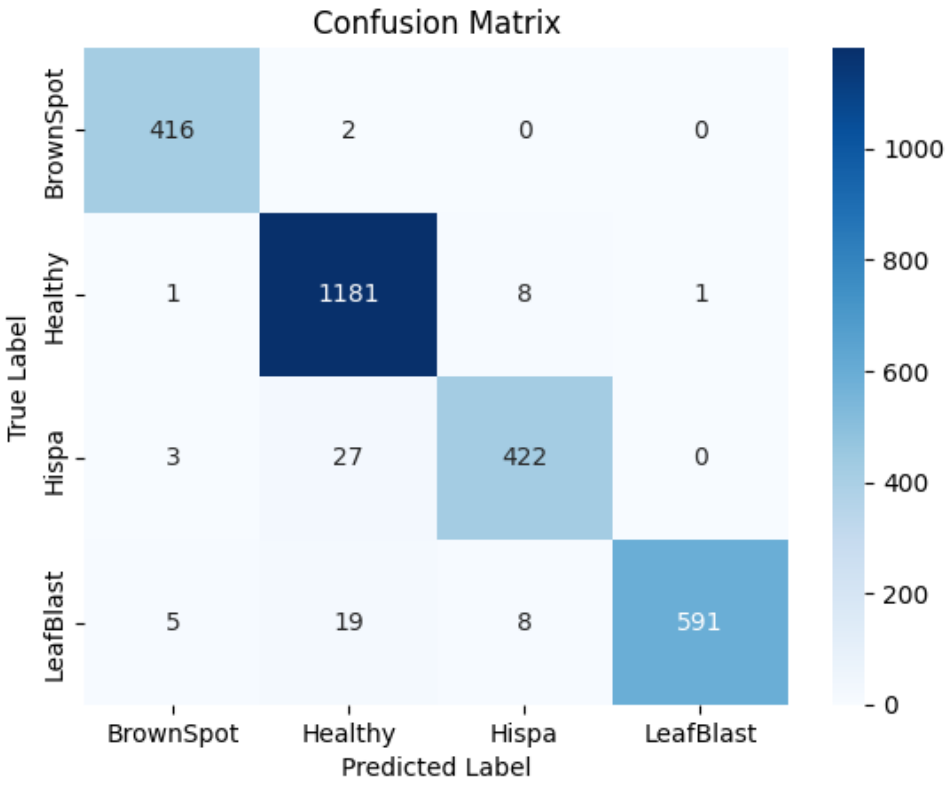


Gambar 4.12 *Accuracy Plot Unfreeze Layer* Skenario *Normalization Layer*

Pada gambar di atas menunjukan bahwa akurasi dari hasil *training* kembali mengalami peningkatan hingga di atas 95%. Hasil akurasi pada proses *fine tuning* yang sebelumnya memiliki selisih grafik antara akurasi *validation* dan *training*, kali ini sudah selaras. Hal ini mengartikan bahwa hasil dari distribusi data yang baik dengan konfigurasi kode yang lengkap telah menghasilkan akurasi dan performa yang baik pula selama masa *training*.

Berdasarkan hasil dari proses *training* menggunakan konfigurasi kode *unfreeze layer* pada skenario *normalization layer*, ditemui hasil perhitungan nilai *accuracy* 98%, *precision* 97%, *recall* 96% dan *f1-score* 97% . *Runtime* yang tercatat untuk seluruh proses pada skenario ini yaitu 1 jam 38 menit. Hasil prediksi label dengan label sebenarnya dapat dilihat pada visualisasi *confusion matrix* berikut:

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/



Gambar 4.13 *Confusion Matrix Unfreeze Layer Training* Skenario *Normalization Layer*

Pada skenario kali ini, disimpulkan bahwa konfigurasi kode tahap *pre-process* dengan jenis *data augmentation dan normalization layer*, konfigurasi kode tahap *modelling* dengan proses *scratch*, *pre-train* dan *fine tuning* dengan fungsi optimasi Adam, dan konfigurasi kode tahap evaluasi dengan teknik *confusion matrix* telah dirasa memiliki hasil yang memuaskan dari segi performa, dan *runtime*.

Untuk skenario selanjutnya, akan dibandingkan pengaruh dari setiap jenis optimasi yang telah ditetapkan terhadap performa dari konfigurasi kode pada skenario ini. Selain itu, pada skenario selanjutnya juga akan ditambahkan teknik evaluasi *k-fold cross validation* untuk melihat apakah model memiliki kemampuan yang baik jika digeneralisasi pada data baru.

4.4.4 Skenario *Optimizer*

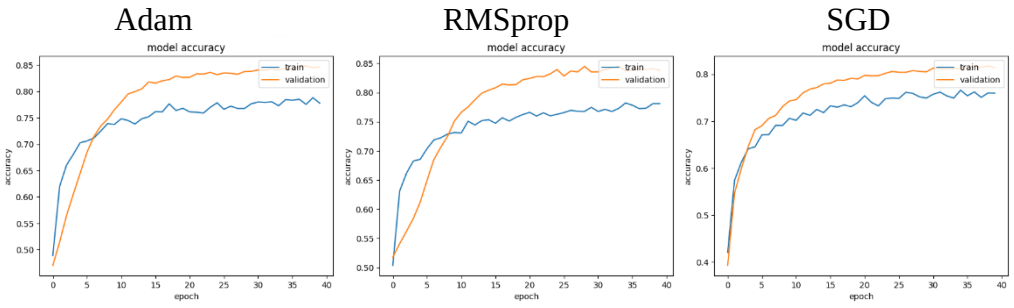
Skenario kali ini akan berisikan tingkat performa dan akurasi yang dicapai untuk melakukan klasifikasi gambar menggunakan beberapa jenis fungsi optimasi yang telah ditetapkan sebelumnya. Fungsi optimasi tersebut yaitu Adam, RMSprop,

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

dan SGD. Konfigurasi kode yang digunakan untuk mengukur performa fungsi optimasi pada skenario kali ini yaitu konfigurasi kode pada skenario *normalization layer* dengan penambahan teknik evaluasi *k-fold cross validation*. Lengkapnya, Proses klasifikasi ini akan dimulai dari tahap impor dan pemuatan *dataset*, lalu tahap *pre-processing* yang berisikan proses *data augmentation* dan *normalization layer*, selanjutnya disusul tahap *modelling* yang berisikan proses *scratch model*, *pre-train* dan *fine tuning model*, serta tahap evaluasi yang berisikan *proses k-fold cross validation* dan *confusion matrix*.

Untuk *learning rate* yang dipakai adalah angka 0,001. Hal ini dikarenakan *dataset* penelitian yang cukup besar dirasa cocok untuk menggunakan *learning rate* yang kecil sebab dapat membantu model dalam menyesuaikan parameter dan mencapai hasil lebih baik. Selain itu, proses optimasi yang stabil dan model yang kompleks menjadi alasan lain pemilihan angka *learning rate*. Khusus untuk fungsi optimasi SGD ditambahkan nilai momentum dengan angka 0.9. Angka ini dipilih dengan alasan bahwa angka ini banyak memberikan hasil yang baik pada kecepatan konvergensi, mengurangi fluktuasi dan menghindari stagnasi. Hal ini membantu SGD meminimalisir salah arah saat proses optimasi berlangsung.

Untuk melihat hasil proses training pada ketiga fungsi optimasi yang digunakan menggunakan konfigurasi kode *pre-train* dan *fine tuning*, berikut adalah visualisasi dari *accuracy plot* yang dihasilkan:



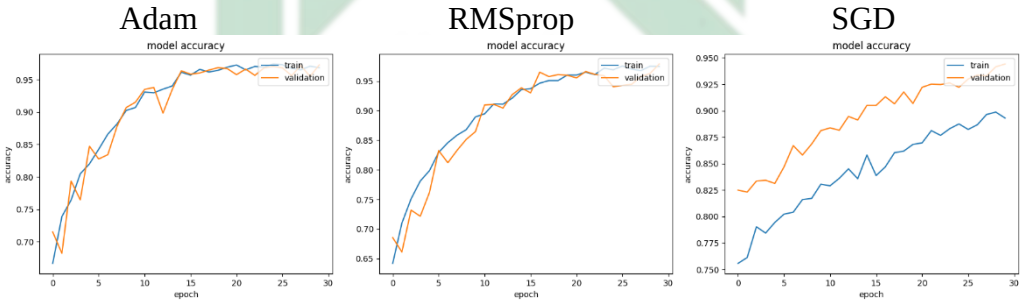
Gambar 4.14 *Accuracy Plot Pre-Train* dan *Fine Tuning* Masing-Masing Fungsi Optimasi Skenario *Optimizer*

Jika melihat gambar di atas, hasil akurasi dari fungsi optimasi Adam dan RMSprop dirasa mirip daripada hasil akurasi oleh fungsi optimasi SGD. Namun, hasil akurasi fungsi optimasi Adam lebih stabil daripada hasil akurasi fungsi

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

optimasi RMSprop. Hal ini mungkin terjadi dikarenakan kemampuan dari fungsi optimasi Adam yang memiliki kemampuan untuk mencapai konvergensi dengan lebih cepat dan stabil daripada fungsi RMSprop yang terkadang terjadi pantulan. Oleh sebab itu, grafik yang dihasilkan fungsi optimasi Adam lebih stabil daripada grafik pada fungsi optimasi RMSprop. Sedangkan fungsi optimasi SGD memiliki hasil akurasi di bawah dari kedua fungsi optimasi sebelumnya. Hal ini mungkin terjadi karena sifat dari SGD yang sering kali salah arah saat menuju konvergensi.

Untuk melihat lebih jauh terkait peningkatan performa pada model menggunakan ketiga fungsi optimasi, proses training dilanjutkan kembali menuju proses *unfreeze layer*. Berikut adalah *accuracy plot* dari ketiga hasil *training* fungsi optimasi menggunakan konfigurasi kode *unfreeze layer* pada skenario *optimizer*:



Gambar 4.17 *Accuracy Plot Unfreeze Laye*r Masing-Masing Fungsi Optimasi Skenario *Optimizer*

Jika kembali melihat gambar, fungsi optimasi Adam dan RMSprop kembali memberikan hasil yang mirip. Namun untuk saat ini, hasil akurasi dari fungsi Optimasi RMSprop menjadi sedikit lebih stabil dibandingkan dengan hasil akurasi fungsi optimasi Adam. Hal ini menunjukan bahwa pantulan saat menuju konvergensi sebelumnya telah diminimalisir seiring dengan lama iterasi. Selain itu selisih yang terjadi antara akurasi *train* dan *validation* yang sebelumnya terjadi juga telah menjadi lebih selaras. Hal ini mungkin terjadi bahwa proses ini membantu distribusi data menjadi lebih baik lagi. Sedangkan fungsi optimasi SGD masih berada di bawah kedua fungsi lain namun tidak memiliki selisih yang jauh. Untuk melihat hasil *accuracy, precision, recall, f1-score* serta runtime proses training menggunakan konfigurasi kode *unfreeze layer* pada masing-masing fungsi optimasi yang dapat dilihat pada tabel berikut:

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

Tabel 4.1 Hasil *Accuracy, Precision, Recall* dan *F1-Score Unfreeze Layer* pada Masing-Masing Fungsi Optimasi

	Accuracy	Precision	Recall	F1-Score	Runtime
Adam	98,62%	97,53%	96,72%	97,13%	1 Jam 38 menit
RMSprop	98,93%	97,83%	97,78%	97,80%	1 jam 43 menit
SGD	97,20%	94,07%	94,50%	94,28%	1 jam 58 menit

Hasil pada tabel di atas memperlihatkan bahwa RMSprop menjadi fungsi optimasi yang paling unggul dalam klasifikasi ini untuk hal akurasi. Disusul oleh fungsi optimasi Adam yang tidak berbeda jauh dari fungsi optimasi RMSprop dan unggul di bidang *runtime*. Terakhir disusul oleh fungsi optimasi SGD. Oleh karena hasil dari proses *training* sangat memuaskan, perlu dilakukan *cross validation* apakah model dengan hasil yang memuaskan ini dapat digeneralisasi ke data baru. Hal ini ditujukan untuk melihat apakah model hanya memiliki performa yang baik pada kasus ini ataukah juga memiliki performa yang baik untuk data baru pada kasus lain.

Setelah dilakukan tahap evaluasi menggunakan teknik *k-fold cross validation* sebanyak 5 kali *fold*, berikut adalah hasil *accuracy, precision, recall* dan *f1-score* proses *training* pada masing-masing fungsi optimasi yang dapat dilihat pada tabel berikut:

Tabel 4.2 Hasil *Accuracy, Precision, Recall* dan *F1-Score Unfreeze Layer* pada Masing-Masing Fungsi Optimasi

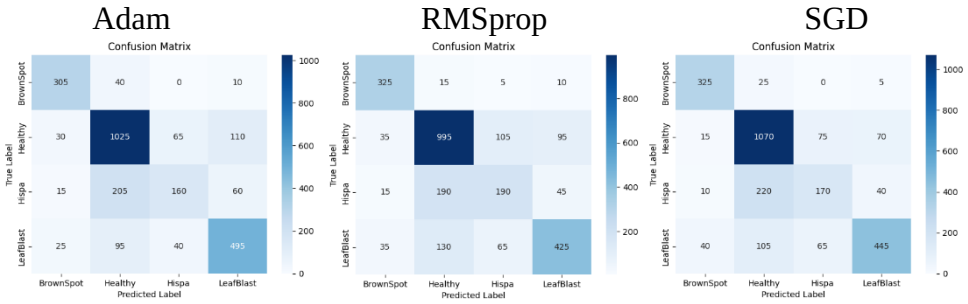
	Accuracy	Precision	Recall	F1-Score
Adam	± 74,14%	± 73,73%	± 69,65%	± 69,86%
RMSprop	± 71,98%	± 70,92%	± 68,55%	± 68,15%
SGD	± 74,40%	± 72,68%	± 70,87%	± 71,11%

Berdasarkan hasil pada tabel di atas, dapat dilihat bahwa fungsi optimasi SGD memiliki kemampuan paling unggul dalam penyesuaian dengan data baru. Sedangkan RMSprop menjadi yang paling rendah. Hal ini menunjukan bahwa kemampuan daripada setiap fungsi optimasi yang digunakan tergantung oleh beberapa faktor pada kasus yang dihadapi seperti konfigurasi kode, *dataset* dan beberapa hal lain. Setiap fungsi optimasi juga memiliki kelemahan dan kelebihan

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

masing-masing. Namun, fungsi optimasi dengan hasil paling stabil ditunjukan oleh fungsi optimasi adam berkat kemampuannya yang lebih stabil dan efisien untuk mencapai konvergensi.

Agar dapat melihat gambaran hasil klasifikasi pada setiap fungsi optimasi, berikut adalah visualisasi *confusion matrix* setelah proses *k-fold cross validation*:



Gambar 4.15 *Confusion Matrix K-Fold Cross Validation* Masing-Masing Fungsi Optimasi Skenario *Optimizer*

4.5 Analisa Hasil

Untuk mengetahui bagaimana performa oleh setiap skenario terhadap klasifikasi yang dilakukan, perlu dilakukan analisa hasil setiap skenario yang ada. Analisa ini bertujuan untuk mengetahui bagaimana pengaruh dan hasil setiap skenario terhadap klasifikasi yang dilakukan. Selain itu, kelebihan dan kekurangan dari setiap skenario juga perlu untuk bahas.

Pada skenario awal yaitu skenario normal, konfigurasi kode untuk klasifikasi yang dilakukan dirasa masih terlalu kosong dan hanya mengandalkan proses *modelling* untuk memacu performa model pada proses *training*. Hal ini mengabaikan hal penting terkait data. Tidak melakukan persiapan data adalah hal yang kurang bijak dalam melakukan klasifikasi khususnya pada klasifikasi gambar.

Pada skenario ini, data hanya dimuat dan langsung masuk ke tahap *modelling*. Meskipun hasil akurasi dari skenario ini dapat dikatakan baik, namun masih terjadi permasalahan pada saat proses training yaitu masalah *overfitting* dan masalah ketidakseimbangan data. Permasalahan ini mungkin terjadi karena kurangnya persiapan data, data yang kompleks, dan ekstraksi fitur yang kurang optimal.

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

Hasil dari skenario ini menunjukkan angka perhitungan nilai *accuracy* 69%, *precision* 33%, *recall* 33% dan *f1-score* 33%. Meskipun hasil akurasi berada di atas 65%, namun hasil *f1-score* hanya menunjukan angka 30%. Hal ini disebabkan oleh adanya permasalahan *overfit* dan ketidakseimbangan data.

Berdasarkan permasalahan ini, dilakukanlah skenario kedua yaitu skenario terkait tahap *pre-processing* yaitu *data augmentation*. Skenario ini memiliki tujuan agar data dapat dipersiapkan dan dapat membantu model dalam proses ekstraksi fitur. Cara kerja proses ini adalah dengan merubah data asli menjadi data baru yang telah dimodifikasi dengan berbagai teknik yang ada. Proses ini ditujukan agar model memiliki data evaluasi baru yang lebih banyak sehingga meningkatkan performa dan menghindarkan dari terjadinya masalah *overfitting*.

Hasil dari skenario kedua ini juga ternyata tidak berbeda jauh dari hasil pada skenario sebelumnya. Pada skenario ini, data memang sudah dipersiapkan dengan baik menggunakan teknik *data augmentation*. Namun persiapan data baru ini ternyata semakin menyulitkan ekstraksi fitur pada saat *modelling*. Data yang sangat berbeda mempersulit model dengan kompleksitas yang lebih tinggi pada saat ekstraksi. Hal ini mungkin terjadi karena kurang baiknya distribusi data pada data baru sehingga kompleksitas data semakin meningkat.

Perhitungan nilai yang tercatat pada skenario kali ini yaitu *accuracy* 67%, *precision* 30%, *recall* 30% dan *f1-score* 30%. Hasil perhitungan ini menjadi lebih menurun daripada skenario sebelumnya. Tapi bukan berarti penambahan proses *data augmentation* ini tidak memiliki pengaruh pada saat *training model*. Grafik yang tercatat pada *accuracy validation* menjadi lebih konstan dan fluktuatif yang ada juga menjadi berkurang. Arah grafik juga menjadi lebih baik.

Selanjutnya untuk menjawab permasalah ini, dilakukanlah skenario ketiga yang masih terkait tahap *pre-process* yaitu penambahan proses *normalization layer*. Tujuan skenario ini adalah untuk membantu pendistribusian data secara baik dan mengoptimalkan *data loading* pada waktu *training*.

Pada skenario ini, hasil yang ditampilkan dapat dikatakan sangat baik karena model telah terhindar dari permasalahan *overfit*. Pergerakan grafik kedua akurasi juga beranjak naik secara bersamaan dan selaras yang menandakan keseimbangan data. Hasil perhitungan nilai juga melonjak sangat tinggi yaitu nilai

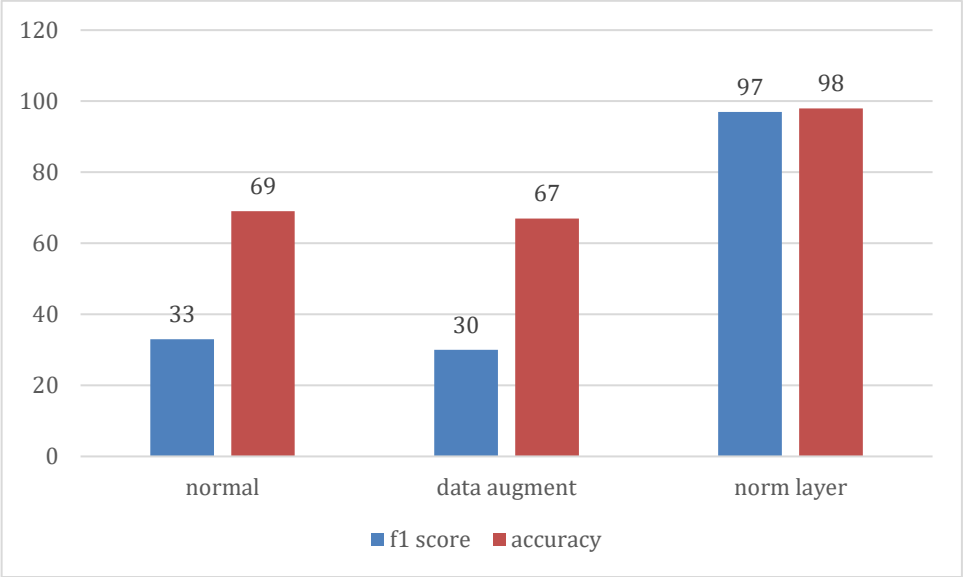
http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

accuracy 98%, *precision* 97%, *recall* 96% dan *f1-score* 97% pada konfigurasi kode menggunakan skenario fungsi optimasi RMSProp. Selain itu, *runtime* paling baik yang tercatat dari memulai hingga selesai proses klasifikasi menggunakan konfigurasi kode skenario fungsi optimasi Adam adalah 1 jam 38 menit. Waktu yang dirasa cukup baik dengan hasil performa yang cukup baik pula.

Skenario ini menunjukkan bahwa tahap *pre-processing* adalah tahap yang sangat penting pada proses klasifikasi. Selain itu, proses *data loading* juga cukup membantu model untuk melakukan *read and write* data dengan lebih optimal sehingga *runtime* yang dihasilkan juga cukup baik. Penormalan distribusi data menggunakan proses *normalization layer* menjadi kunci pada penelitian kali ini. Dengan adanya proses ini, proses klasifikasi dapat memberikan hasil yang memuaskan. Untuk melihat bagaimana peningkatan performa dari beberapa skenario sebelumnya, berikut disediakan tabel dan gambar komparasi hasil skenario normal hingga normalization layer.

Tabel 4.3 Komparasi Hasil Skenario Normal, *Data Augmentation* dan *Normalization Layer*

	Skenario Normal	Skenario Data Augment	Skenario Norm Layer
Accuracy	69%	67%	98%
Precision	33%	30%	97%
Recal	33%	30%	96%
F1-Score	33%.	30%	97%



http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

Gambar 4.16 Hasil Akurasi dan F1-Score dari ke-3 Skenario

Setelah menemukan konfigurasi kode yang dirasa memiliki performa baik, penelitian dilanjutkan pada skenario *optimizer* dengan melakukan perbandingan fungsi optimasi pada tahap *modelling*. Hal ini ditujukan untuk melihat apakah fungsi optimasi memiliki pengaruh pada performa pada saat klasifikasi berlangsung. Hasil skenario ini menunjukan bahwa fungsi optimasi memiliki pengaruh pada performa saat proses *training model*. Meskipun besar kecilnya pengaruh yang diberikan oleh fungsi optimasi bergantung pada beberapa faktor, namun hal ini cukup membantu model untuk mencapai titik optimal konvergensi pada setiap iterasi agar model dapat menjadi semakin efisien. Selain itu, fungsi optimasi juga dapat membantu dalam proses distribusi data dan meminimalisir resiko *overfitting*.

Setelah melakukan perbandingan pada tiga fungsi optimasi yang diantaranya adalah fungsi optimasi Adam, RMSprop, dan SGD, diperoleh hasil bahwa selama masa *training*, RMSprop menjadi fungsi optimasi yang paling unggul. Hal ini dikarenakan fungsi optimasi ini memiliki Kemampuan penyesuaian yang tinggi selama masa *training* yang membantunya dalam meminimalisir lompatan yang sering terjadi saat proses optimasi mulai mendekati konvergensi. Selain itu, Adam menjadi model dengan hasil paling stabil. Hal ini dikarenakan fungsi optimasi ini memiliki kemampuan untuk mencapai konvergensi dengan lebih cepat dan stabil. Sedangkan SGD menjadi fungsi optimasi yang paling unggul pada waktu evaluasi menggunakan *cross validation*. Hal ini menunjukan sifat dari SGD yang simple memiliki pengaruh yang baik pada waktu generalisasi data.

Selain itu, Hasil dari tahap evaluasi yaitu *k-fold cross validation* juga menunjukan bahwa klasifikasi ini dapat digeneralisasi pada data baru dengan cukup baik. Hal ini ditunjukan oleh hasil *cross validation* pada skenario *optimizer* menggunakan masing-masing fungsi optimasi. Hasil tersebut menunjukan bahwa akurasi yang di dapat berada pada angka 74% menggunakan fungsi optimasi SGD.

Kesimpulan yang bisa ditarik adalah bahwa tahap *pre-process* adalah proses yang penting untuk melakukan suatu klasifikasi. Pemilihan model untuk *training* juga menentukan bagaimana hasil dari klasifikasi yang dilakukan. Selain itu, fungsi optimasi dan *data loading* adalah proses yang tidak kalah penting untuk dilakukan

UIN SUNAN AMPEL
S U R A B A Y A
http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

saat proses klasifikasi, khususnya saat data yang digunakan berukuran besar dan memiliki banyak varian. Hasil dari klasifikasi pada waktu *training* yang memuaskan juga tidak serta merta dapat digeneralisasi pada data baru dan kasus lain. Perlu dilakukan tahap evaluasi yaitu *cross validation* untuk melihat apakah data memiliki performa yang baik jika digeneralisasikan pada data dan kasus baru.

UIN SUNAN AMPEL
S U R A B A Y A

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

`

BAB V
PENUTUP

5.1 Kesimpulan

Berdasarkan analisa dari setiap hasil pada proses yang dilakukan, diperoleh beberapa kesimpulan sebagai berikut:

1. Klasifikasi menggunakan metode CNN ini dilakukan menggunakan model arsitektur EfficientNetb0 dengan melalui tiga tahapan yaitu tahap *pre-processing, modelling* dan *evaluating.* Tahap *pre-processing* dilakukan dengan menggunakan teknik *data augmentation* dan *normalization layer*, tahap *modelling* dilakukan dengan melalui proses *scratch, pre-train* dan *fine tuning*, dan tahap *evaluating* dilakukan dengan melakukan *teknik k-fold cross validation* dan *confusion matrix.* Selain itu dilakukan perbandingan menggunakan beberapa jenis *optimizer* yang umum digunakan yaitu Adam, RMSprop dan SGD.
2. Terdapat beberapa tingkat akurasi yang dihasilkan pada klasifikasi kali ini yaitu tingkat akurasi *training* dan *cross validation*. Untuk tingkat akurasi *training* paling tinggi yaitu 98,93% pada klasifikasi menggunakan konfigurasi kode skenario fungsi optimasi RMSprop. Sedangkan untuk tingkat akurasi *cross validation* paling tinggi yaitu 74,40% pada klasifikasi menggunakan konfigurasi kode skenario fungsi optimasi SGD. Waktu *runtime* yang tercatat mulai dari dimulai proses klasifikasi hingga selesai waktu *training* yaitu 1 jam 38 menit menggunakan konfigurasi kode skenario fungsi optimasi Adam.

5.2 Saran

Sebagai upaya perbaikan dan pengembangan penelitian selanjutnya, maka peneliti memiliki saran sebagai berikut:

1. Pada penelitian ini, tahap *pre-processing* hanya dilakukan menggunakan teknik *data augmentation* dan *normalization layer*. Perlu dilakukan

UIN SUNAN AMPEL
S U R A B A Y A

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

klasifikasi dengan penambahan teknik lain yang relevan pada tahap *pre-process* agar proses persiapan data menjadi lebih optimal.

2. Mengingat tujuan dari penelitian ini, model arsitektur yang digunakan adalah model dengan jumlah parameter yang sedikit dan agak tua. Perlu dilakukan klasifikasi menggunakan model arsitektur lain yang lebih baru dan memiliki tingkat akurasi dan *runtime* yang lebih baik.
3. Penelitian ini menggunakan CPU untuk melakukan *running* pada program dikarenakan keterbatasan spesifikasi *device*. Untuk mengetahui tingkat *runtime* yang lebih baik, perlu dilakukan klasifikasi menggunakan GPU dengan spesifikasi yang lebih andal.
4. Penyakit daun pada yang diklasifikasi pada penelitian hanyalah empat penyakit. Perlu dilakukan klasifikasi pada semua penyakit daun padi yang ada agar penelitian yang ada dapat diterapkan dengan lebih universal.

UIN SUNAN AMPEL
S U R A B A Y A

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

DAFTAR PUSTAKA

Anggraeni, S. R., Ranggianto, N. A., Ghozali, I., Fatichah, C., & Purwitasari, D. (2022). Deep Learning Approaches for Multi-Label Incidents Classification from Twitter Textual Information. *Journal of Information Systems Engineering and Business Intelligence*, *8*(1), Art. 1. https://doi.org/10.20473/jisebi.8.1.31-41

Anton, A., Nissa, N. F., Janiati, A., Cahya, N., & Astuti, P. (2021). Application of Deep Learning Using Convolutional Neural Network (CNN) Method for Womenâ€™s Skin Classification. *Scientific Journal of Informatics*, *8*(1), Art. 1. https://doi.org/10.15294/sji.v8i1.26888

Arifah, Salman, D., Yassi, A., & Bahsar-Demmallino, E. (2022). Climate change impacts and the rice farmers' responses at irrigated upstream and downstream in Indonesia. *Heliyon*, *8*(12), e11923. https://doi.org/10.1016/j.heliyon.2022.e11923

Armi, L., & Fekri Ershad, S. (2019). *Texture image analysis and texture classification methods—A Review*. *2*, 1–29.

Ashfaque Maat, J., & Iqbal, A. (2019). *Introduction to Support Vector Machines and Kernel Methods*.

Badan Pusat Statistik. (2022). Pada 2022, luas panen padi diperkirakan sebesar 10,61 juta hektare dengan produksi sekitar 55,67 juta ton GKG. https://www.bps.go.id/pressrelease/2022/10/17/1910/pada-2022--luas-panen-padi-diperkirakan-sebesar-10-61-juta-hektare-dengan-produksi-sekitar-55-67-juta-ton-gkg.html

Ba, J., Kiros, J., & Hinton, G. (2016). *Layer Normalization*.

Beauxis-Aussalet, E., & Hardman, L. (2014, November 1). *Simplifying the Visualization of Confusion Matrix*.

Beheshti, N., & Johnsson, L. (2020). Squeeze U-Net: A Memory and Energy Efficient Image Segmentation Network. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 1495–1504. https://doi.org/10.1109/CVPRW50498.2020.00190

Bezdan, T., & Bacanin, N. (2019). *Convolutional Neural Network Layers and*

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

Architectures. 445–451. https://doi.org/10.15308/Sinteza-2019-445-451

Chollet, F. (2018). *Deep learning with Python*. Manning Publications Co.

Endang, A. H., & R, M. (2017). PERINGATAN JAM MALAM ANAK BERDASARKAN LOKASI HANDPHONE BERBASIS ANDROID. *Jurnal INSYPRO (Information System and Processing)*, *2*(2), Art. 2. https://doi.org/10.24252/insypro.v2i2.4078

Gross, B. L., & Zhao, Z. (2014). Archaeological and genetic insights into the origins of domesticated rice. *Proceedings of the National Academy of Sciences*, *111*(17), 6190–6197. https://doi.org/10.1073/pnas.1308942110

Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., Liu, T., Wang, X., Wang, G., Cai, J., & Chen, T. (2018). Recent advances in convolutional neural networks. *Pattern Recognition*, *77*, 354–377. https://doi.org/10.1016/j.patcog.2017.10.013

Gunawan, F. (2018). *PENGARUH PENGGUNAAN FAKTOR PRODUKSI TERHADAP PRODUKSI PADI DI DESA BARUGAE KABUPATEN BONE* [Diploma, UNIVERSITAS NEGERI MAKASSAR]. http://eprints.unm.ac.id/11202/

Gupta, N. (2013). Accuracy, Sensitivity and Specificity Measurement of Various Classification Techniques on Healthcare Data. *IOSR Journal of Computer Engineering*, *11*, 70–73. https://doi.org/10.9790/0661-1157073

Haris, N. (2020). Kombinasi Ciri Bentuk dan Ciri Tekstur Untuk Identifikasi Penyakit Pada Tanaman Padi. *JATISI (Jurnal Teknik Informatika dan Sistem Informasi)*, *7*, 237–250. https://doi.org/10.35957/jatisi.v7i2.239

Hassanpour, M., & Malek, H. (2019). Document Image Classification using SqueezeNet Convolutional Neural Network. *2019 5th Iranian Conference on Signal Processing and Intelligent Systems (ICSPIS)*, 1–4. https://doi.org/10.1109/ICSPIS48872.2019.9066032

Iandola, F., Moskewicz, M., Ashraf, K., Han, S., Dally, W., & Keutzer, K. (2016). *SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and textless1MB model size*.

Ilin, R., Watson, T., & Kozma, R. (2017). *Abstraction hierarchy in deep learning neural networks*. 768–774. https://doi.org/10.1109/IJCNN.2017.7965929

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

Indra, D., Fadlillah, H. M., Kasman, K., Ilmawan, L. B., & Lahuddin, H. (2022). Classification of good and damaged rice using convolutional neural network. *Bulletin of Electrical Engineering and Informatics*, *11*(2), Art. 2. https://doi.org/10.11591/eei.v11i2.3385

Irfansyah, D., Mustikasari, M., & Suroso, A. (2021). Arsitektur Convolutional Neural Network (CNN) Alexnet Untuk Klasifikasi Hama Pada Citra Daun Tanaman Kopi. *Jurnal Informatika: Jurnal Pengembangan IT*, *6*(2), Art. 2. https://doi.org/10.30591/jpit.v6i2.2802

Januarti, I., Junaidi, Y., & Mulyana, E. (2021). ANALISIS USAHATANI PADI GOGO DI LAHAN RAWA LEBAK (Studi Kasus: Desa Talang Dukun, Kecamatan Sungai Pinang, Propinsi Sumatera Selatan). *AGRISAINTIFIKA: Jurnal Ilmu-Ilmu Pertanian*, *5*(1), Art. 1. https://doi.org/10.32585/ags.v5i1.1109

Joiner, I. A. (2018). Chapter 1 - Artificial Intelligence: AI is Nearby. Dalam I. A. Joiner (Ed.), *Emerging Library Technologies* (hlm. 1–22). Chandos Publishing. https://doi.org/10.1016/B978-0-08-102253-5.00002-2

Jones, M. T. (2009). *Artificial intelligence: A systems approach*. Jones and Bartlett Publishers.

Julianto, A., & Sunyoto, A. (2021). A performance evaluation of convolutional neural network architecture for classification of rice leaf disease. *IAES International Journal of Artificial Intelligence (IJ-AI)*, *10*(4), Art. 4. https://doi.org/10.11591/ijai.v10.i4.pp1069-1078

Kim, E., Lee, H., Kim, J., & Kim, S. (2020). Data Augmentation Method by Applying Color Perturbation of Inverse PSNR and Geometric Transformations for Object Recognition Based on Deep Learning. *Applied Sciences*, *10*, 3755. https://doi.org/10.3390/app10113755

Kim, J., Jin, M., Homma, Y., Sim, A., Kroeger, W., & Wu, K. (2022). *Extract Dynamic Information To Improve Time Series Modeling: A Case Study with Scientific Workflow*. https://doi.org/10.48550/arXiv.2205.09703

Koech, K. E. (2022, Juli 16). *Cross-Entropy Loss Function*. Medium. https://towardsdatascience.com/cross-entropy-loss-function-f38c4ec8643e

Lasky, A. (2022, Mei 25). Deep Learning Project: Multilayer Perceptron. *Medium*.

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

https://medium.com/@artjovianprojects/deep-learning-project-multilayer-perceptron-e34017941918

Li, S., & Zhao, X. (2019). *Image-Based Concrete Crack Detection Using Convolutional Neural Network and Exhaustive Search Technique*. Advances in Civil Engineering, 2019, e6520620. https://doi.org/10.1155/2019/6520620

Lindsay, G. (2020). Convolutional Neural Networks as a Model of the Visual System: Past, Present, and Future. *Journal of Cognitive Neuroscience*, *33*, 1–15. https://doi.org/10.1162/jocn_a_01544

Loffe, S., & Szegedy, C. (2015). *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*.

Mahmud, T. (2006). *Identifikasi serangga di sekitar tumbuhan kangkungan (Ipomoea crassicaulis Roob.)* [Undergraduate, Universitas Islam Negeri Maulana Malik Ibrahim]. http://etheses.uin-malang.ac.id/4518/

McClelland, J. L., & Botvinick, M. M. (2020). *Deep Learning: Implications for Human Learning and Memory* [Preprint]. PsyArXiv. https://doi.org/10.31234/osf.io/3m5sb

Mehmood, F., Ahmad, S., & Whangbo, T. K. (2023). An Efficient Optimization Technique for Training Deep Neural Networks. *Mathematics*, *11*(6), Article 6. https://doi.org/10.3390/math11061360

Mikołajczyk, A., & Grochowski, M. (2018). *Data augmentation for improving deep learning in image classification problem*. 117–122. https://doi.org/10.1109/IIPHDW.2018.8388338

Mohidem, N. A., Hashim, N., Shamsudin, R., & Che Man, H. (2022). Rice for Food Security: Revisiting Its Production, Diversity, Rice Milling Process and Nutrient Content. *Agriculture*, *12*, 741. https://doi.org/10.3390/agriculture12060741

. *International Journal of Engineering and Advanced Technology*, *9*. https://doi.org/10.35940/ijeat.B4412.129219

Murthy, N. V. E. S. (2019). Freehand Sketch-Based Authenticated Security System using Convolutional Neural Network. *International Journal of Engineering and Advanced Technology*, *9*. https://doi.org/10.35940/ijeat.B4412.129219

UIN SUNAN AMPEL
S U R A B A Y A

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

Nasution, A. (2014). *Penyakit Blas Pyricularia grisea pada Tanaman Padi dan Strategi Pengendaliannya*. *9*(2).

Norjamilah, N., Mariana, & Budi, I. S. (2021). Ketahanan Penyakit Bercak Coklat (Helminthosporium sp.) pada Padi Beras Merah, Padi Beras Hitam, Lokal Siam, dan Unggul Ciherang. *JURNAL PROTEKSI TANAMAN TROPIKA*, *4*(3), Art. 3. https://doi.org/10.20527/jptt.v4i3.900

Novindasari, I. (2021, Februari 11). Mengapa Diperlukan Regularisasi pada Model Neural Network? *Medium*. https://idanovinda.medium.com/mengapa-diperlukan-regularisasi-pada-model-neural-network-d622ed98f9a8

Nuryanto, B. (2018). PENGENDALIAN PENYAKIT TANAMAN PADI BERWAWASAN LINGKUNGAN MELALUI PENGELOLAAN KOMPONEN EPIDEMIK. *Jurnal Penelitian Dan Pengembangan Pertanian*, *37*(1), Art. 1. https://doi.org/10.21082/jp3.v37n1.2018.p1-8

OShea, T., & Hoydis, J. (2017). An Introduction to Deep Learning for the Physical Layer. *IEEE Transactions on Cognitive Communications and Networking*, *PP*, 1–1.

Optimisation Techniques II · Deep Learning. (t.t.). Diambil 9 Mei 2023, dari https://atcold.github.io/pytorch-Deep-Learning/en/week05/05-2/

Patel, K. (2020, Oktober 18). *Convolution Neural Networks—A Beginner's Guide [Implementing a MNIST Hand-written Digit…*. Medium. https://towardsdatascience.com/convolution-neural-networks-a-beginners-guide-implementing-a-mnist-hand-written-digit-8aa60330d022

Pokhrel, S. (2019, September 20). *Beginners Guide to Understanding Convolutional Neural Networks*. Medium. https://towardsdatascience.com/beginners-guide-to-understanding-convolutional-neural-networks-ae9ed58bb17d

Prasetyo, M. S. H. (2017). *KAJIAN INTENSITAS PENYAKIT BERCAK COKLAT SEMPIT (Cercospora oryzae) DAN TEKNIK PENGENDALIANNYA PADA PERTANAMAN PADI DI KECAMATAN TANGGUL KABUPATEN JEMBER*. https://repository.unej.ac.id/xmlui/handle/123456789/82358

Priyangka, A. A. J. V., & Kumara, I. M. S. (2021). Classification Of Rice Plant Diseases Using the Convolutional Neural Network Method. *Lontar*

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

Komputer : Jurnal Ilmiah Teknologi Informasi, *12*(2), Art. 2. https://doi.org/10.24843/LKJITI.2021.v12.i02.p06

Rahman, R., Arko, P., Ali, M. E., Khan, M., Apon, S. H., Nowrin, F., & Wasif, A. (2020). Identification and recognition of rice diseases and pests using convolutional neural networks. *Biosystems Engineering*, *194*, 112–120. https://doi.org/10.1016/j.biosystemseng.2020.03.020

Rana, Y. (2019). *Python: Simple though an Important Programming language*. *06*(02).

Romero Aquino, M., Gutoski, M., Hattori, L., & Lopes, H. (2017, Oktober 30). *The Effect of Data Augmentation on the Performance of Convolutional Neural Networks*. https://doi.org/10.21528/CBIC2017-51

Sarker, I. H. (2021). Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions. *SN Computer Science*, *2*(6), 420. https://doi.org/10.1007/s42979-021-00815-1

Sayuthi, M., Hanan, A., Muklis, M., & Satriyo, P. (2020). Distribusi Hama Tanaman Padi (Oryza sativa L.) pada Fase Vegetatif dan Generatif di Provinsi Aceh. *Jurnal Agroecotania : Publikasi Nasional Ilmu Budidaya Pertanian*, *3*(1), Art. 1. https://doi.org/10.22737/agroecotania.v3i1.11286

Setiawan, W. (2020). PERBANDINGAN ARSITEKTUR CONVOLUTIONAL NEURAL NETWORK UNTUK KLASIFIKASI FUNDUS. *Jurnal Simantec*, *7*, 48–53.

Sharma, N., Sharma, R., & Jindal, N. (2021). Machine Learning and Deep Learning Applications-A Vision. *Global Transitions Proceedings*, *2*(1), 24–28. https://doi.org/10.1016/j.gltp.2021.01.004

Shivappa, R., Navadagi, D. B., Baite, M. S., Yadav, M. K., Rathinam, P. S., Umapathy, K., Pati, P., & Rath, P. C. (2021). Emerging Minor Diseases of Rice in India: Losses and Management Strategies. Dalam *Integrative Advances in Rice Research*. IntechOpen. https://doi.org/10.5772/intechopen.99898

Surah Al-Baqarah—البقرة سُورَة | Qur'an Kemenag. (2022). https://quran.kemenag.go.id/surah/2/167

Tan, M., & Le, Q. V. (2020). *EfficientNet: Rethinking Model Scaling for*

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/

Convolutional Neural Networks (arXiv:1905.11946). arXiv. https://doi.org/10.48550/arXiv.1905.11946

Thangarajah, V. (2019). *Python current trend applications-an overview*.

Ulate, D., Amanupunnyo, H. R. D., Umasangaji, A., Ririhena, R. E., & Leiwakabessy, C. (2020). Kejadian Penyakit Blas Pada Varietas Padi Inpari Sidenuk di Desa Waimital Kecamatan Kairatu, Kabupaten Seram Bagian Barat. *JURNAL PERTANIAN KEPULAUAN*, *4*(2), Art. 2.

Villanueva, A., Benemerito, R., Cabug-Os, M., Chua, R., Rebeca, C., & Miranda, M. (2019). *Somnolence Detection System Utilizing Deep Neural Network*. 602–607. https://doi.org/10.1109/ICOIACT46704.2019.8938460

Xu, D., Zhang, S., Zhang, H., & Mandic, D. (2021). Convergence of the RMSProp deep learning method with penalty for nonconvex optimization. *Neural Networks*, *139*. https://doi.org/10.1016/j.neunet.2021.02.011

Yamashita, R., Nishio, M., Do, R. K. G., & Togashi, K. (2018). Convolutional neural networks: An overview and application in radiology. *Insights into Imaging*, *9*(4), Art. 4. https://doi.org/10.1007/s13244-018-0639-9

UIN SUNAN AMPEL
S U R A B A Y A

http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/http://digilib.uinsby.ac.id/