Klasifikasi Citra Penyakit Pada Daun Tanaman Tomat (*Lycopersicum Esculentum Mill.*) Menggunakan Metode CNN Berbasis Android

LAPORAN TUGAS AKHIR

Laporan ini disusun guna memenuhi salah satu syarat untuk menyelesaikan program studi Teknik Informatika S-1 pada Fakultas Teknologi Industri Universitas Islam Sultan Agung

Disusun oleh :

Irfan Eka Mahdy

32602000034

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS ISLAM SULTAN AGUNG

SEMARANG

2025

Classification Of Disease Images On The Leaves Of Tomato Plants (Lycopersicum Esculentum Mill.) Using The Android-Based CNN Method

This Report is Prepared to Fulfill One of the Requirements to Obtain a Bachelor's Degree (S1) in the Informatics Engineering Study Program Faculty of Industrial Technology, Sultan Agung Islamic University

Arranged By :

Irfan Eka Mahdy

32602000034

MAJORING OF INFORMATICS ENGINEERING

INDUSTRIAL TECHNOLOGY FACULTY

SULTAN AGUNG ISLAMIC UNIVERSITY SEMARANG

2025

LEMBAR PENGESAHAN PEMBIMBING

Laporan Tugas Akhir dengan judul **"Klasifikasi Citra Penyakit Pada Daun Tanaman Tomat (*Lycopersicum Esculentum Mill.*) Menggunakan Metode CNN Berbasis Android"** ini disusun oleh :

Nama : Irfan Eka Mahdy

NIM : 32602000034

Program Studi : Teknik Informatika

Telah disetujui oleh Dosen Pembimbing pada :

Hari : Selasa

Tanggal : 25 Februari

Mengesahkan,

Pembimbing

Andi Riansyah, S.T, M.Kom

NIDN. 0609108802

Mengetahui,

Ketua Program Studi Teknik Informatika

Fakultas Teknologi Industri

Universitas Islam Sultan Agung

Moch Taufik, ST, MIT

NIDN. 0622037502

LEMBAR PENGESAHAN
TUGAS AKHIR

Klasifikasi Citra Penyakit Pada Daun Tanaman Tomat (*Lycopersicum Esculentum Mill.*) Menggunakan Metode CNN Berbasis Android

IRFAN EKA MAHDY
32602000034

Telah dipertahankan di depan tim penguji ujian sarjana tugas akhir
Program Studi Teknik Informatika
Universitas Islam Sultan Agung
Pada tanggal : .....

TIM PENGUJI UJIAN SARJANA :

Moch Taufik, ST, MIT NIDN.0622037502 (Ketua Penguji)	.....	25-02-25 .....
Bagus Satrio WP, S.Kom. M.Cs NIDN.1027118801 (Anggota Penguji)	.....	25-02-25 .....
Andi Riansyah, S.T, M.Kom NIDN. 0609108802 (Pembimbing)	.....	25-02-25 .....

Semarang, 24 Februari 2025

Mengetahui,

Kaprodi Teknik Informatika
Universitas Islam Sultan Agung

Moch Taufik, ST, MIT
NIDN. 0622037502

SURAT PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan dibawah ini :

Nama : Irfan Eka Mahdy

NIM : 32602000034

Judul Tugas Akhir : Klasifikasi Citra Penyakit Pada Daun Tanaman Tomat (*Lycopersicum Esculentum Mill.*) Menggunakan Metode CNN Berbasis Android.

Dengan bahwa ini saya menyatakan bahwa judul dan isi Tugas Akhir yang saya buat dalam rangka menyelesaikan Pendidikan Strata Satu (S1) Teknik Informatika tersebut adalah asli dan belum pernah diangkat, ditulis ataupun dipublikasikan oleh siapapun baik keseluruhan maupun sebagian, kecuali yang secara tertulis diacu dalam naskah ini dan disebutkan dalam daftar pustaka, dan apbila di kemudian hari ternyata terbukti bahwa judul Tugas Akhir tersebut pernah diangkat, ditulis ataupun dipublikasikan, maka saya bersedia dikenakan sanksi akademis. Demikian surat pernyataan ini saya buat dengan sadar dan penuh tanggung jawab.

Semarang,

Yang Menyatakan,

Irfan Eka Mahdy

PERNYATAAN PERSETUJUAN PUBLIKASI KARYA ILMIAH

Saya yang bertanda tangan dibawah ini :

Nama : Irfan Eka Mahdy

NIM : 32602000034

Program Studi : Teknik Informatika

Fakultas : Teknologi industri

Dengan ini menyatakan Karya Ilmiah berupa Tugas akhir dengan Judul : Klasifikasi Citra Penyakit Pada Daun Tanaman Tomat (*Lycopersicum Esculentum Mill.*) Menggunakan Metode CNN Berbasis Android. Menyetujui menjadi hak milik Universitas Islam Sultan Agung serta memberikan Hak bebas Royalti Non-Eksklusif untuk disimpan, dialihmediakan, dikelola dan pangkalan data dan dipublikasikan diinternet dan media lain untuk kepentingan akademis selama tetap menyantumkan nama penulis sebagai pemilik hak cipta. Pernyataan ini saya buat dengan sungguh-sungguh. Apabila dikemudian hari terbukti ada pelanggaran Hak Cipta/Plagiarisme dalam karya ilmiah ini, maka segala bentuk tuntutan hukum yang timbul akan saya tanggung secara pribadi tanpa melibatkan Universitas Islam Sultan agung.

Semarang,

Yang menyatakan,

Irfan Eka Mahdy

KATA PENGANTAR

Dengan mengucap syukur alhamdulillah atas kehadirat Allah SWT yang telah memberikan Rahmat dan karunianya kepada penulis, sehingga dapat menyelesaikan Tugas Akhir dengan judul "Klasifikasi Citra Penyakit Pada Daun Tanaman Tomat (*Lycopersicum Esculentum Mill*.) Menggunakan Metode CNN Berbasis Android" ini untuk memenuhi salah satu syarat menyelesaikan studi serta dalam rangka memperoleh gelar sarjana (S-1) pada Program Studi Teknik Informatika Fakultas Teknologi Industri Universitas Islam Sultan Agung Semarang.

Tugas Akhir ini disusun dan dibuat dengan adanya bantuan dari berbagai pihak, materi maupun teknis, oleh karena itu saya selaku penulis mengucapkan terima kasih kepada:

1. Rektor Universitas Islam Sultan Agung Bapak Prof. Dr. H. Gunarto, S.H., M.H., yang mengizinkan penulis menimba ilmu di kampus ini.
2. Dekan Fakultas Teknologi Industri Ibu Dr. Ir. Hj. Novi Marlyana, ST., M.T., IPU., ASEAN.Eng.
3. Dosen pembimbing penulis Bapak Andi Riansyah, S.T, M.Kom yang telah membimbing dan memberikan banyak nasehat dan saran.
4. Orang tua dan keluarga penulis yang senantiasa memberikan semangat serta doa agar tugas akhir ini berjalan dengan lancar.
5. Dan kepada teman-teman Kost Mirza yang telah memberikan semangat dan membantu dalam penyusunan tugas akhir ini.

Dengan segala kerendahan hati, penulis menyadari masih terdapat banyak kekurangan dari segi kualitas maupun dari ilmu pengetahuan dalam penyusunan laporan, sehingga penulis mengharapkan adanya saran dan kritikan yang bersifat membangun demi kesempurnaan laporan ini di masa mendatang.

Semarang, 06 Januari 2025

Irfan Eka Mahdy

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

ABSTRAK

Penyakit pada daun tanaman tomat merupakan salah satu penyebab utama gagal panen dalam budidaya tomat. Dampaknya dapat merugikan petani dalam hal produksi dan kualitas tanaman. Penyakit daun tomat memiliki penyebab yang berbeda-beda, mulai dari hama hingga virus. Diagnosis dini dan penggunaan varietas tahan penyakit adalah langkah efektif untuk mengendalikan risiko gagal panen. Gejala infeksi penyakit pada daun tomat dapat dikenali melalui perubahan warna, bentuk dan kondisi daun. Dengan deteksi yang cepat dan tepat, pengendalian penyakit dapat dilakukan dengan lebih efektif untuk mencegah kerugian yang lebih besar. Salah satu solusi *preventif* yang dapat diterapkan adalah klasifikasi gambar daun tomat menggunakan *machine learning*. Metode *Convolutional Neural Network* (CNN) dipilih karena kemampuannya untuk mengenali pola visual secara akurat. Dalam penelitian ini, arsitektur *MobileNetV2* digunakan untuk membangun model klasifikasi gambar penyakit daun tomat yang diimplementasikan dalam aplikasi *Android* menggunakan bahasa pemrograman *Kotlin*. Model ini dikembangkan menggunakan 1.200 gambar daun tomat yang dibagi menjadi lima kelas. Hasil pelatihan menunjukkan bahwa model memiliki kinerja yang baik, dengan akurasi 94%.
Kata Sandi: Penyakit Daun Tomat, *Convolutional Neural Network* (CNN), Klasifikasi Citra, Aplikasi *Android*

ABSTRACT

Diseases on the leaves of tomato plants are one of the main causes of crop failure in tomato cultivation. The impact can be detrimental to farmers in terms of production and quality of crops. Tomato leaf diseases have different causes, ranging from pests to viruses. Early diagnosis and the use of disease-resistant varieties are effective measures to control the risk of crop failure. Symptoms of disease infection on tomato leaves can be recognized through changes in leaf color, shape and condition. With quick and precise detection, disease control can be done more effectively to prevent greater losses. One of the preventive solutions that can be applied is the classification of tomato leaf images using machine learning. The Convolutional Neural Network (CNN) method was chosen because of its ability to accurately recognize visual patterns. In this research, MobileNetV2 architecture is used to build a tomato leaf disease image classification model implemented in an android application using Kotlin programming language. The model was developed using 1,200 tomato leaf images divided into five classes. The training results show that the model has good performance, with 94% accuracy.
Keyword: Tomato Leaf Diseases, Convolutional Neural Network (CNN), Image Classification, Android application

BAB I
PENDAHULUAN

1.1 Latar Belakang

Tomat merupakan tipe tumbuhan yang sangat diminati di pasar, hampir semua orang menyukai karena mudah didapatkan baik sebagai sayuran maupun buah dalam keadaan segar maupun setelah dimasak (Hanif, 2023). Tomat sering digunakan sebagai bahan utama dalam berbagai olahan makanan seperti saus, salad, dan pasta (Ali dkk., 2021). Tomat mengandung kaya akan vitamin A, B, dan C. Selain itu tanaman ini mudah untuk di tanam dan dirawat, serta memiliki biaya yang terjangkau bagi masyarakat (Pantang dkk., 2021). Di Indonesia, terdapat sekitar 63.369 hektar lahan yang digunakan untuk budidaya tomat. Pada tahun 2022, Indonesia berhasil menghasilkan sekitar 1.168.744 ton tomat (Badan Pusat Statistik, 2024).

Permintaan akan sayuran atau buah tomat di Indonesia terus meningkat seiring dengan pertumbuhan populasi. Oleh karena itu, penting untuk meningkatkan produksi tomat dan mencegah penurunan produksi agar dapat memenuhi kebutuhan yang semakin meningkat (Dwitomo dkk., 2023). Permintaan tomat untuk konsumsi meningkat setiap tahun, tetapi produksi tomat tidak selalu mengikuti peningkatan tersebut. Hal ini menunjukkan bahwa tomat sering kali mengalami perubahan dan cenderung mengalami penurunan dari waktu ke waktu.

Meskipun dapat terlihat dengan jelas mengidentifikasi penyakit pada tanaman tomat memerlukan waktu dan bisa sulit karena adanya kemiripan antara penyakit satu dengan yang lain (Khultsum & Subekti, 2021). Berbagai jenis penyakit yang sering menyerang tomat diantaranya *bacterial spot*, *late blight, target spot,* dan *tomato yellow leaf curl virus* (Riansyah dkk., 2021). Penyakit ini dapat menyebabkan perubahan warna dan bentuk pada seluruh bagian tanaman tomat, termasuk daun, batang, akar, hingga buah. Namun gejala penyakit pada tanaman tomat lebih sering diamati melalui perubahan pada daun (D. H. N. Aini dkk., 2023). Penyakit pada tanaman dapat

mengakibatkan kerugian panen setiap tahun, yang merupakan ancaman serius bagi produksi pertanian dan mengakibatkan penurunan hasil atau bahkan gagal panen (Li dkk., 2022).

MobileNetV2 adalah suatu struktur arsitektur *Convolution*al *Neural Network* (CNN) yang dirancang khusus untuk perangkat seluler. *MobileNetV2* terbagi menjadi dua lapisan, yaitu *depthwise convolution* dan *pointwise convolution*. *Depthwise convolution* menerapkan filter tunggal pada setiap saluran input, sementara *pointwise convolution* membuat kombinasi linear dari *output layer depthwise* (Li dkk., 2022). Untuk menyelesaikan kasus deteksi objek, segmentasi semantik, klasifikasi audio, klasifikasi gambar, pendeteksian wajah, metode *MobileNetV2* telah banyak dimanfaatkan oleh peneliti untuk menyelesaikan berbagai macam kasus saat ini.

Berdasarkan keunggulan arsitektur *MobileNetV2* dalam menyelesaikan permasalahan pada berbagai macam kasus serta penelitian klasifikasi/identifikasi penyakit tanaman tomat belum memanfaatkan metode terkini maka pada penelitian ini akan mencoba membangun model *Machine Learning* untuk menyelesaikan kasus klasifikasi penyakit tanaman tomat yang memanfaatkan data gambar daun tomat dengan judul "Klasifikasi Citra Penyakit Pada Daun Tanaman Tomat (*Lycopersicum Esculentum Mill*.) Menggunakan Metode CNN Berbasis *Android*".

1.2 Perumusan Masalah

Bagaimana mengimplementasikan metode *Convolutional Neural Network* dengan arsitektur *MobileNetV2* untuk mendeteksi penyakit pada tanaman tomat yaitu *bacterial spot, late blight, target spot,* dan *tomato yellow leaf curl virus* berbasis *Android*.

1.3 Pembatasan Masalah

Berdasarkan masalah tersebut maka batasan masalah yang akan dibuat adalah sebagai berikut :

1. Penelitian ini memanfaatkan *dataset* yang tersedia dari *Kaggle* (https://www.kaggle.com/datasets/ahmadzargar/tomato-leaf-disease-dataset-segmented) yang mencakup berbagai penyakit daun tanaman tomat. Dalam penelitian ini hanya akan difokuskan pada klasifikasi 4 jenis penyakit daun tanaman tomat, yaitu *Bacterial Spot, Late Blight, Target Spot,* dan *Tomato Yellow Leaf Curl Virus* serta 1 jenis tanaman tomat sehat.
2. Penelitian ini menggunakan algoritma *Convolutional Neural Network* dengan arsitektur *MobileNetV2*.
3. Total data yang digunakan sebanyak 1200 citra, yang terdiri dari 240 citra untuk masing-masing class.
4. Penelitian ini menggunakan *platform Android*.
5. Pengambilan gambar menggunakan kamera belakang.

1.4 Tujuan

Tujuan penelitian ini adalah mengimplementasikan metode CNN dengan arsitektur *MobileNetV2* untuk mengklasifikasikan penyakit daun tanaman tomat yaitu *bacterial spot, late blight, target spot,* dan *tomato yellow leaf curl virus,* serta daun tomat sehat dengan sistem berbasis *Android*.

1.5 Manfaat

Berdasarkan analisis permasalahan dan tujuan penelitian, manfaat dari penelitian ini adalah memudahkan identifikasi berbagai jenis penyakit pada daun tanaman tomat.

1.6 Sistematika Penulisan

Sistematika penulisan yang akan digunakan oleh penulis dalam sebuah pembuatan laporan tugas akhir adalah sebagai berikut:

BAB I : PENDAHULUAN

Pada bab I penulis mengutarakan latar belakang pemilihan judul, rumusan masalah, batasan masalah, tujuan penelitian, metodologi penelitian, serta sistematika penulisan.

BAB II : TINJAUAN PUSTAKA DAN DASAR TEORI

Pada bab II memuat penelitian-penelitian sebelumnya dan dasar teori untuk membantu penulis memahami bagaimana teori yang berhubungan dengan Algoritma *Convolutional Neural Network*.

BAB III : METODE PENELITIAN

Pada bab III menjelaskan tentang metode penelitian yang merupakan tahapan-tahapan penelitian yang dilakukan. Tahapan penelitian terdiri dari tahap pengumpulan data, augmentasi data, data *preprocessing*, klasifikasi menggunakan arsitektur *MobileNetV2*, evaluasi dan implementasi berbasis *Android*.

BAB IV : HASIL PENELITIAN

Pada bab IV menjelasakan mengenai hasil dan pembahasan yang dilakukan yaitu mengenai tahap pengumpulan data, augmentasi data, data *preprocessing*, klasifikasi menggunakan arsitektur *MobileNetV2*, evaluasi dan implementasi.

BAB V : KESIMPULAN DAN SARAN

Pada bab V membahas mengenai kesimpulan yang diperoleh dari hasil penelitian yang telah dilakukan serta saran-saran yang dapat diterapkan dari hasil penelitian yang dapat menjadi masukan yang berguna kedepannya.

BAB II
TINJAUAN PUSTAKA DAN DASAR TEORI

2.1 Tinjauan Pustaka

Pada penelitian ini, peneliti menggunakan metode *MobileNetv2*. Mereka membuat sistem yang dapat mengklasifikasikan tingkat keparahan penyakit daun pada tanaman padi yang terkena penyakit blas daun dengan bantuan algoritma *machine learning*. Pada penelitian tersebut, *dataset* yang digunakan merupakan hasil murni observasi dari penelitian yang sudah divalidasi oleh ahli dengan total 300 data asli. *MobileNetV2* ternyata sangat berhasil dalam mengklasifikasikan objek, dengan akurasi 78,33%. sehingga, petani dapat terbantu dalam mengenali tingkat keparahan penyakit *leafblast* pada tanaman padi dan pemberian bahan kimia berupa fungisida dapat sesuai dengan dosis anjuran tingkat keparahan (Annur dkk., 2023).

Dalam sebuah penelitian tahun 2023, dengan menggunakan arsitektur *MobileNetv2*. Program berhasil mendeteksi objek pada delapan kelas yaitu *Guava, Pear, Apple, Sugar Apple, Banana, Orange, Pineapple,* dan *Dragon fruits* yang menghasilkan tingkat akurasi yang baik (Prima, 2023).

Kemudian terdapat penelitian yang diterapkan dengan teknik *fine tune* menggunakan metode *MobileNetv2*. Teknik *fine tune* digunakan untuk meningkatkan performa model. Beberapa percobaan dilakukan dengan parameter yang berbeda seperti *epoch* dan *layer fine tune* untuk mendapatkan hasil terbaik. Hasil penelitian mendapatkan akurasi 97% *training*, 96% validasi dan 93% pengujian (Hendri Butar-Butar & Marpaung, 2023).

Penelitian selanjutnya, peneliti melakukan klasifikasi jenis penyakit daun teh menggunakan arsitektur CNN *MobileNetV2*. *MobileNetV2* memiliki *score* akurasi cukup tinggi, jumlah *training* parameters dan model size yang kecil namun memiliki performa yang baik. Arsitektur *MobileNetV2* menghasilkan akurasi sebesar 98,33% dan nilai loss 0,0307% yang menunjukkan bahwa arsitektur ini mampu melakukan klasifikasi penyakit dauh teh dengan baik (Febriady Marpaung, 2024).

Penelitian berikutnya, peneliti menggunakan arsitektur CNN *MobileNetV2*. Arsitektur *MobileNetV2* mampu melakukan klasifikasi jenis tumor otak dengan baik yang dibuktikan dengan hasil akurasi sebesar 88.64%, nilai loss 0.3424, pada *Confusion Matrix* yang menunjukkan hasil klasifikasi yang benar lebih banyak daripada hasil klasifikasi yang salah dan dari 32 citra sample yang terlihat sejumlah 28 citra terdeteksi sesuai (benar) (Winnarto dkk., 2022)

Berdasarkan berbagai penelitian di atas, penggunaan CNN dengan arsitektur *MobileNetV2* sangat cocok untuk diterapkan pada aplikasi *Android*. Keberhasilan model CNN, khususnya *MobileNetV2*, dalam menganalisis gambar dengan akurasi yang tinggi membuktikan efektivitasnya dalam berbagai aplikasi pengolahan citra. Oleh karena itu, penelitian ini akan mengembangkan sistem klasifikasi penyakit daun tanaman tomat menggunakan arsitektur *MobileNetV2* yang akan diimplementasikan pada *platform Android*.

2.2 Dasar Teori

2.2.1 Penyakit Tanaman Tomat

Tanaman tomat sangat rentan terhadap serangan penyakit. Jika lingkungan sekitar tanaman tomat tidak bersih maka patogen seperti jamur, bakteri atau virus akan sangat mudah untuk menyerang tanaman tomat. Dalam *dataset* yang penulis gunakan, terdapat 4 jenis penyakit daun tanaman tomat dan 1 jenis daun tanaman tomat sehat. Berikut merupakan penjelasan penyakit daun tanaman tomat dan daun sehat yang digunakan penulis : (Soekarta dkk., 2023).

1. ***Bacterial Spot***

   Penyakit ini disebabkan oleh bakteri *Xanthomonas vesicatoria*, yang menyerang tomat hijau tetapi tidak pada tomat merah. Penyakit ini lebih sering terjadi selama musim hujan. Tanaman yang terinfeksi menunjukkan kerusakan seperti bercak pada daun dan buah, yang pada akhirnya dapat mengurangi hasil panen dan membuat buah rentan

terhadap kerusakan akibat sinar matahari. Gejala yang muncul termasuk bintik-bintik kecil yang berbentuk tidak beraturan dan tampak basah pada daun, serta bintik-bintik kudis pada buah. Bintik-bintik pada daun seringkali dikelilingi oleh lingkaran cahaya kuning, dengan pusat yang mengering dan sering kali robek (Adhikari dkk., 2020).

Gambar 2. 1 Penyakit *Bacterial Spot*

Pada Gambar 2.1 merupakan penyakit *bacterial spot* yang menyerang tanaman tomat. Tomat yang terkena penyakit bacterial spot dapat dilihat dari gejalanya melalui daun. Di mana pada tomat yang terkena penyakit ini akan mengalami gangguan pertumbuhan dan kesuburan pada tomat sehingga berdampak pada hasil panen bahkan kematian pada tanaman tomat.

2. ***Late Blight***

Pada daun yang terkena penyakit ini, dapat ditandai dengan munculnya bercak cokelat yang dikelilingi oleh tepi berwarna kuning. Di bagian bawah bercak, sering kali tampak lapisan putih yang menyerupai beludru. Seiring waktu, bercak ini akan menyebar dan meluas ke seluruh permukaan daun. Penyakit ini disebabkan oleh jamur *Phytophthora infestans* dan penyebarannya sangat cepat terutama di musim hujan, terutama ketika suhu berkisar antara 18-20ºC dan kelembapan mencapai 91-100% (Vincent dkk., 2023).

Gambar 2. 2 Penyakit *Late Blight*

Pada Gambar 2.2 merupakan citra daun tomat yang terjangkit penyakit *late blight*. Pada tanaman tomat yang terkena penyakit *late blight* umumnya dapat dilihat dari gejala yang terjadi pada daun. Penyakit *late blight* disebabkan oleh jamur air, Penyakit ini sangat merusak, terutama dalam kondisi lingkungan yang lembap dan basah. Hal ini dapat menyebabkan tomat mengalami gangguan pertumbuhan bahkan kematian pada tanaman tomat.

3. ***Target Spot***

Target spot pada daun tomat ditandai dengan munculnya banyak bintik bulat berwarna krem dengan diameter hingga 4 mm. Bintik-bintik ini sering kali berbentuk tidak teratur atau bersudut dan dibatasi oleh pembuluh darah. Daun yang terinfeksi akan mengering dan rontok sebelum waktunya. Jamur penyebabnya biasanya berasal dari tanaman lain yang telah terinfeksi, sisa-sisa tanaman sebelumnya, atau spesies inang lainnya. Infeksi ini sangat umum terjadi pada daun tomat, menyebabkan bintik-bintik sudut berwarna cokelat muda atau abu-abu dengan diameter sekitar 2 mm, kadang-kadang dikelilingi oleh tepi berwarna kuning. Pusat bintik-bintik ini sering kali rontok, menghasilkan efek "lubang tembakan" pada daun (Abdulridha dkk., 2020).

Gambar 2. 3 Penyakit *Target Spot*

Pada Gambar 2.3 merupakan citra gambar daun tomat yang terjangkit penyakit *target spot.* Gejala utama berupa bercak cokelat bundar pada daun yang menyerupai pola cincin konsentris seperti target. Bercak ini biasanya berawal kecil, namun seiring waktu dapat membesar, menyebabkan jaringan daun di sekitarnya menguning dan akhirnya mati.

4. *Yellow Leaf Curl Virus*

Daun muda yang terinfeksi virus ini akan tampak mengerut atau keriting dan berubah warna menjadi kuning. Virus ini terutama menyebar melalui kutu kebul sebagai vektor utamanya. Untuk mengendalikan penyebaran penyakit, tanaman yang terinfeksi harus segera dicabut dan dibakar. Langkah selanjutnya adalah mengendalikan populasi kutu kebul, yang dapat dilakukan dengan menyemprotkan insektisida berbahan aktif *abamektin* (Yan dkk., 2021).

Gambar 2. 4 Penyakit *Yellow Leaf Curl Virus*

Pada Gambar 2.4 merupakan citra gambar daun tomat yang terkena penyakit *yellow leaf curl virus.* Gejala utama yang dapat terlihat adalah perubahan warna pada daun menjadi kuning dan mengerut yang mengakibatkan gangguan pertumbuhan pada tanaman.

5. Daun Tomat Sehat

Daun tanaman tomat yang sehat memiliki warna hijau segar yang merata tanpa bercak kuning, cokelat, atau kehitaman, serta permukaan yang bersih dan halus atau sedikit berbulu. Bentuknya oval dengan ujung runcing, bergerigi di tepi, dan tumbuh secara proporsional sesuai varietasnya. Daun yang sehat tampak kokoh, tidak layu, serta ditopang oleh tangkai yang kuat tanpa tanda-tanda kerapuhan. Pertumbuhannya teratur dan subur, tanpa deformasi, kerdil, atau kerusakan seperti lubang dan keriting. Ciri-ciri ini menunjukkan tanaman mendapatkan nutrisi, cahaya, dan kondisi lingkungan yang optimal (Mungki Astiningrum dkk., 2020).

Gambar 2. 5 Daun Tomat Sehat

Pada Gambar 2.5 merupakan gambar daun tomat sehat yang memiliki ciri-ciri warna hijau yang merata, bentuk *oval* dengan ujung sedikit runcing, tanpa adanya bercak, atau kerusakan akibat hama dan penyakit. Daun tomat yang sehat menunjukkan kemampuan *fotosintesis* yang baik.

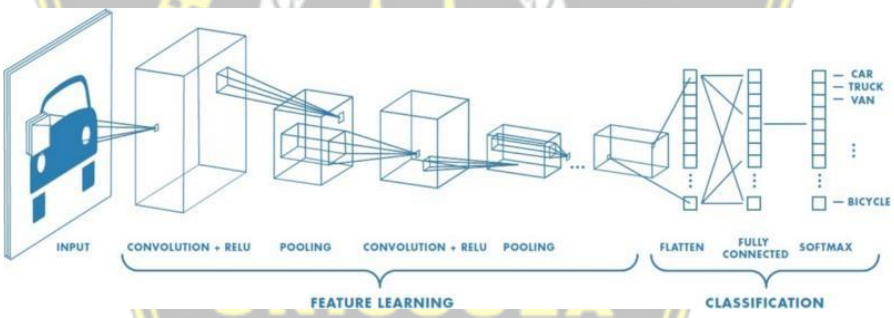
2.2.2 *Machine learning*

Machine learning (ML) atau pembelajaran mesin adalah pendekatan dalam bidang Kecerdasan Buatan (*Artificial Intelligence*) yang kini banyak digunakan untuk meniru cara manusia dalam memecahkan masalah dan mengotomatisasi tugas. *Machine learning* bekerja dengan meniru proses belajar manusia atau makhluk cerdas lainnya, memungkinkan komputer untuk terus meningkatkan kinerjanya seiring dengan bertambahnya data. Proses ini mencerminkan cara manusia belajar dari pengalaman, di mana algoritma menganalisis pola dalam data pelatihan untuk membuat prediksi

atau keputusan berdasarkan data baru. *Machine learning* menjadi fondasi teknologi *modern* dengan kontribusinya di berbagai industri, menunjukkan kemampuan untuk meniru proses pembelajaran yang biasanya dikaitkan dengan kecerdasan manusia atau makhluk cerdas lainnya (Morgan & Jacobs, 2020).

2.2.3 *Convolutional Neural Network* (CNN)

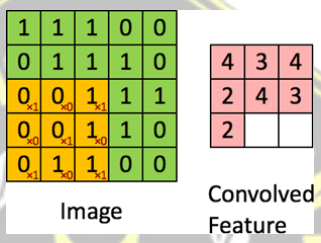
Convolutional Neural Network (CNN) adalah jenis jaringan saraf tiruan yang dirancang khusus untuk mengolah data yang berbentuk *grid*, seperti gambar dan video. Dengan memanfaatkan operasi konvolusi, *Convolutional Neural Network* dapat secara otomatis mengekstrak fitur-fitur penting dari data, seperti tepi, bentuk, dan tekstur. Teknologi ini sangat efektif dalam mengenali pola-pola *spatial*, sehingga sangat berguna untuk tugas-tugas seperti pengenalan objek dan analisis gambar (Alzubaidi dkk., 2021).



Gambar 2. 6 Alur Proses metode CNN

Secara umum, struktur CNN terbagi menjadi dua bagian utama, *feature extraction* dan *classification*. Pada tahap *feature extraction*, jaringan menggunakan lapisan *convolutional*, fungsi aktivasi seperti *ReLU*, dan lapisan *pooling* untuk mengecilkan ukuran gambar atau fitur yang dihasilkan, seperti menggunakan *max pooling*. Selanjutnya, pada tahap *classification*, hasil fitur yang telah diekstrak diubah menjadi vektor panjang melalui proses *flattening*. Vektor ini kemudian diproses melalui *fully connected layer* dan akhirnya menggunakan *softmax* untuk menentukan probabilitas setiap kelas dalam tugas klasifikasi.

Convolution layer adalah bagian penting dalam CNN tempat sebagian besar proses komputasi terjadi. Lapisan ini terdiri dari neuron yang diatur untuk membentuk filter dengan ukuran panjang dan lebar dalam piksel. Saat mendesain *convolution layer*, ada beberapa aspek yang perlu diatur, seperti ukuran kernel dan *stride*. Ukuran kernel, atau filter size, menentukan dimensi kernel dan area lokal saat proses konvolusi. Jika ukuran kernel sama dengan ukuran input, lapisan ini bisa dianggap sebagai *fully connected layer*. *Stride*, di sisi lain, adalah parameter yang mengatur seberapa jauh filter berpindah, dan jika *stride* diatur ke 1, filter konvolusi akan bergerak satu piksel pada setiap langkah baik secara horizontal maupun vertikal.

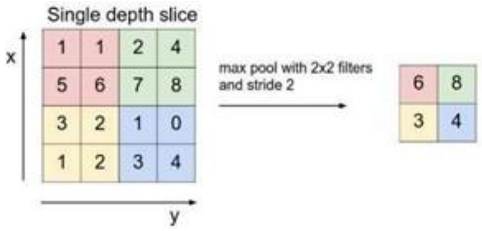


Gambar 2. 7 Proses *Convolution Layer*

Seperti yang ditunjukkan pada Gambar 2.7, gambar input memiliki ukuran 5 piksel lebar dan 5 piksel tinggi, serta 3 channel RGB. Lapisan konvolusi akan memproses bagian-bagian kecil dari gambar, misalnya area 3x3 piksel yang berwarna kuning. Kernel akan bergerak dari sudut kiri atas ke sudut kanan bawah gambar (Kayhan & van Gemert, 2020).

Pooling layer, atau yang juga dikenal sebagai *sub-sampling layer*, berfungsi untuk mengurangi ukuran dimensi dari *feature map* yang dihasilkan oleh *convolutional layer*. Ini membantu mengurangi beban komputasi pada *layer* berikutnya. *Pooling layer* bekerja dengan menggunakan filter yang memiliki dimensi dan pergeseran tertentu yang bergerak melintasi seluruh *feature map*. Ada dua jenis pooling yang sering digunakan yaitu *Max Pooling* dan *Average Pooling*. *Max Pooling* memilih nilai maksimum dari setiap area lokal yang dianalisis, sehingga hanya bagian yang paling penting dari input yang diteruskan. Sementara itu, *Average Pooling* menghitung rata-rata dari

semua nilai di area lokal, sehingga semua bagian input diteruskan dengan proporsi yang sama (Jie & Wanda, 2020).



Gambar 2. 8 Operasi *Max Pooling*

Pooling layer biasanya menggunakan filter berukuran 2x2 dengan pergeseran sebesar 2. Filter ini diterapkan pada setiap bagian dari input, dan pola ini dapat mengurangi ukuran *feature map* asli hingga 75%.

Aktivasi ReLU (*Rectified Linear Unit*) adalah jenis layer aktivasi yang sering digunakan dalam model CNN. Layer ini menerapkan efek *thresholding* dengan menetapkan nilai nol untuk semua piksel yang bernilai negatif pada citra input. Dengan kata lain, aktivasi ReLU akan mengubah semua nilai piksel negatif dalam citra menjadi nol.

Fully Connected Layer adalah lapisan yang biasanya digunakan dalam model *Multilayer Perceptron (MLP)* dan berfungsi untuk mengubah dimensi data menjadi jumlah neuron atau unit yang ada dalam lapisan tersebut, sehingga data bisa dianalisis secara linear. Sebelum data masuk ke dalam *fully connected layer*, setiap neuron pada *convolution layer* harus diubah menjadi data satu dimensi terlebih dahulu. Karena langkah ini mengakibatkan hilangnya informasi spasial yang tidak bisa dipulihkan, *fully connected layer* hanya ditempatkan di akhir jaringan untuk menjaga agar informasi spasial tetap utuh pada tahap-tahap awal (Ding dkk., 2021).

Softmax berfungsi untuk menghitung probabilitas setiap kelas target di antara semua kemungkinan kelas target. Tujuannya adalah untuk membantu menentukan kelas target yang paling mungkin untuk input yang diberikan. Berikut adalah rumus yang digunakan untuk menghitung *softmax*:

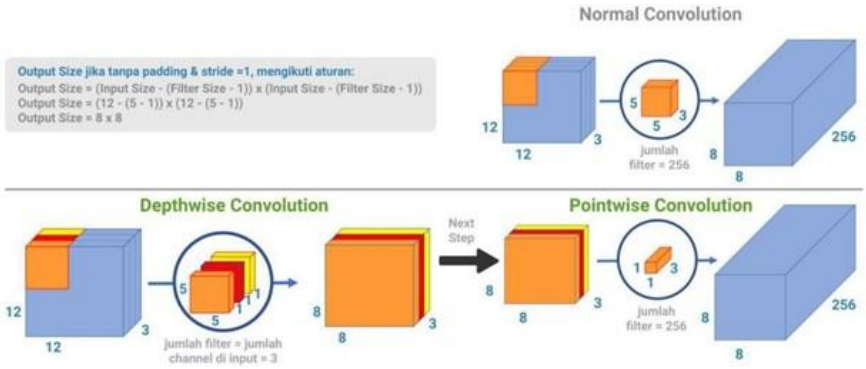
$$f_{j(z)} = \frac{e^{z_j}}{\Sigma_k e^{z_k}}$$

Notasi f_j menunjukkan hasil fungsi untuk setiap elemen ke-j pada output vektor kelas. Argumen z adalah hipotesis yang dihasilkan oleh model pelatihan, yang akan diklasifikasikan oleh fungsi *Softmax*. *Softmax* memberikan hasil yang lebih intuitif dan memungkinkan interpretasi probabilistik yang lebih baik dibandingkan dengan algoritma klasifikasi lainnya. Fungsi ini memungkinkan kita menghitung probabilitas untuk setiap label, dengan mengambil vektor nilai riil dari label yang ada dan mengubahnya menjadi vektor dengan nilai antara nol dan satu. Total nilai dalam vektor tersebut akan selalu berjumlah satu.

2.2.4 *MobileNetV2*

MobileNetV2 adalah arsitektur CNN yang dirancang untuk tugas klasifikasi gambar dan deteksi objek. Dikembangkan oleh Google sebagai penyempurnaan dari versi sebelumnya, *MobileNetV1*. *MobileNetV2* unggul dalam efisiensi komputasi, ringan, dan dirancang khusus untuk perangkat dengan sumber daya terbatas, seperti perangkat *mobile* (Annur dkk., 2023). Arsitektur ini memiliki lapisan khusus yang disebut *depthwise separable convolution*, yang terdiri dari dua tahap yaitu *depthwise convolution* dan *pointwise convolution*. Fungsi lapisan ini adalah untuk mengurangi komputasi dengan mengurangi jumlah parameter, sehingga menghasilkan model yang lebih kecil dalam ukuran (Saputra dkk., 2021).

Berikut adalah ilustrasi perbandingan antara *normal convolution* dan *depthwise convolution* pada Gambar 2.1. Gambar yang digunakan memiliki dimensi 12x12x3 dengan 256 kernel berukuran 5x5x3.

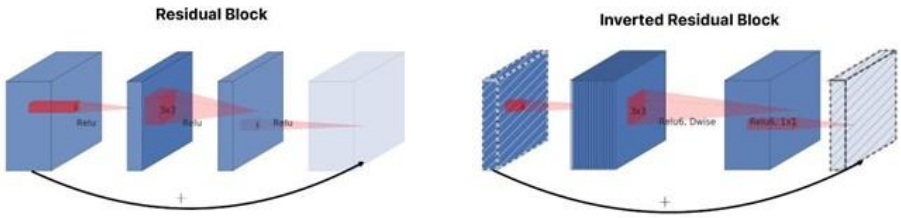


Gambar 2. 1 Perbandingan Normal *Convolution* dan *Depthwise Separable Convolution*

Dalam *normal convolution*, filter berinteraksi dengan seluruh channel pada input gambar dan menghasilkan feature map. Namun, dalam *depthwise convolution*, setiap filter hanya berinteraksi dengan satu channel dari input gambar, menghasilkan feature map dengan jumlah channel yang sama dengan jumlah filter. Setelah itu, untuk mencapai output yang sama seperti pada *normal convolution*, dilakukan *pointwise convolution* dengan filter 1x1. Filter ini memiliki kedalaman yang sama dengan jumlah channel pada feature map sebelumnya.

Dalam perbandingan jumlah komputasi antara *normal convolution* dan *depthwise separable convolution* pada gambar 2.1, dengan 256 filter berukuran 5x5x3 dan pergeseran sebanyak 8x8 kali, total perkalian selama proses konvolusi adalah 1.228.800. Namun, untuk *depthwise separable convolution*, tahap *depthwise convolution* menggunakan 3 filter berukuran 5x5x1 dengan pergeseran yang sama, hanya menghasilkan 4.800 perkalian. Kemudian, pada tahap *pointwise convolution*, dengan 256 filter berukuran 1x1x3 dan pergeseran yang sama, menambahkan 49.152 perkalian. Secara keseluruhan, jumlah perkalian pada *depthwise separable convolution* adalah 52.952, yang setara dengan sekitar 22,7% dari jumlah perkalian pada *normal*

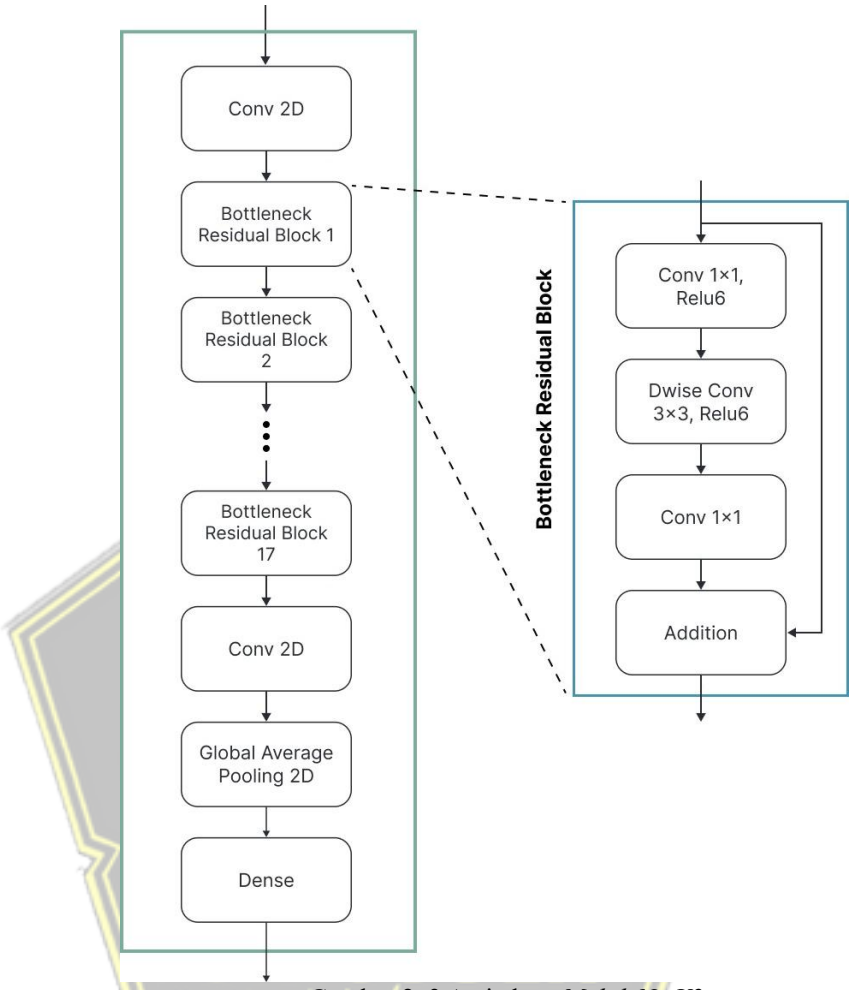
convolution. Oleh karena itu, *depthwise separable convolution* terbukti jauh lebih efisien dalam hal komputasi dibandingkan dengan *normal convolution*.



Gambar 2. 2 Perbedaan *Residual Block* dan *Inverted Residual Block*

Dalam *MobileNetV2*, terdapat dua ide utama yang ditambahkan, yaitu *inverted residual* dan *linear bottleneck*. *Linear bottleneck* adalah desain *residual block* di mana output dari konvolusi terakhir dalam *residual block* dapat dihubungkan ke input awal menggunakan *shortcut connection*. *Inverted residual block* mirip dengan *residual block* seperti yang terlihat pada gambar 2.2. Perbedaannya adalah bahwa *inverted residual block* dalam *MobileNetV2* menerapkan pendekatan sempit → lebar → sempit dalam hal jumlah channel, seperti membalikkan urutan operasi dalam *residual block* konvensional. Pada *inverted residual block*, langkah pertama adalah memperluas jaringan menggunakan konvolusi 1x1. Kemudian, dilakukan *depthwise convolution*

3x3, diikuti oleh fungsi aktivasi *ReLU*. Akhirnya, jumlah channel dikurangi kembali menggunakan konvolusi 1x1 lainnya.



Gambar 2. 3 Arsitektur *MobileNetV2*

Dengan menggabungkan konsep *inverted residual* dan *linear bottleneck*, arsitektur *MobileNetV2* terdiri dari serangkaian lapisan seperti yang terlihat pada gambar 2.3. Pada proses pertama, citra dimasukkan melalui lapisan *convolution* 2D, dan selanjutnya data diproses melalui lapisan *bottleneck residual block*. Di sinilah konsep *linear bottleneck* dan *inverted residual block* diterapkan. Setelah melewati *bottleneck residual block*, data masuk ke lapisan *Global Average Pooling* (GAP) untuk menghasilkan output model berupa tensor 2D. Tensor 2D ini kemudian diproses melalui lapisan *Dense*, yang mengubah matriks 2D menjadi representasi yang lebih padat dan kompleks, sehingga model dapat mempelajari pola-pola yang lebih detail dari data yang diberikan.

2.2.5 *TensorFlow*

Gambar 2. 4 *TensorFlow*

TensorFlow adalah kerangka kerja *open-source* yang dikembangkan oleh Google untuk komputasi numerik dan kecerdasan buatan (*Artificial Intelligence*). *TensorFlow* menyediakan berbagai alat dan sumber daya yang memungkinkan pengguna untuk membuat dan melatih model jaringan saraf (*neural networks*) yang digunakan dalam berbagai aplikasi, seperti pengenalan gambar, pengenalan suara, dan pemrosesan bahasa alami.

Sedangkan *TensorFlowLite* adalah solusi mudah untuk perangkat seluler dan *embedded devices*. Ini memungkinkan untuk menjalankan model yang dipelajari mesin pada perangkat seluler dengan latensi rendah, sehingga dapat memanfaatkannya untuk melakukan klasifikasi, regresi, atau apa pun yang mungkin di inginkan tanpa harus melakukan perjalanan bolak-balik ke server.

TensorFlowLite terdiri dari runtime tempat anda dapat menjalankan model yang sudah ada sebelumnya, dan seperangkat alat yang dapat Anda gunakan untuk mempersiapkan model anda untuk digunakan pada perangkat seluler dan *embedded devices*. Meskipun demikian, ini berfungsi dengan model Klasifikasi Gambar umum (Sallang dkk., 2021).

2.2.6 *Android*

Android adalah sistem operasi mobile berbasis *Linux* yang mencakup sistem operasi, *middleware*, dan aplikasi. Sistem operasi ini dirancang khusus untuk perangkat mobile seperti smartphone. Salah satu keunggulan *Android* adalah fleksibilitasnya dalam pengembangan aplikasi. Pengembang dapat

membuat aplikasi dengan berbagai fungsi seperti media sosial, permainan. (Monteiro dkk., 2021).

Gambar 2. 5 *Kotlin*

Kotlin adalah bahasa pemrograman modern yang dikembangkan oleh *JetBrains*. Bahasa ini beroperasi di atas platform *Java Virtual Machine* (JVM) dan mendukung interoperabilitas dengan *Java*, memungkinkan penggunaan keduanya dalam satu proyek. Keunggulan *Kotlin* meliputi manajemen yang lebih baik terhadap *NullPointerException*, penulisan kode yang lebih ringkas dan mudah dimengerti, serta dukungan penuh dari IDE. *Kotlin* juga telah mendapatkan dukungan resmi dari *Google* untuk pengembangan aplikasi *Android* sejak diumumkan pada acara *Google* I/O 2017. Hal ini memperkuat statusnya sebagai bahasa pemrograman yang sangat relevan dan didukung secara kuat untuk pengembangan aplikasi *Android*.

2.2.7 *Flowchart*

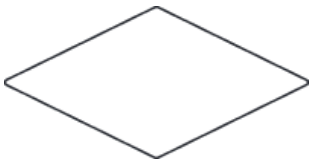
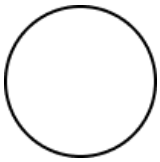
Flowchart adalah gambaran diagram alir dari algoritma dalam suatu program. Dengan menggunakan *flowchart*, penyelesaian suatu masalah khususnya yang memerlukan pemahaman dan evaluasi lebih lanjut dapat lebih mudah. Untuk membuat *flowchart* secara efektif, penting untuk memperhatikan beberapa pedoman berikut:

Simbol-simbol dalam *flowchart* digunakan untuk menjelaskan berbagai aspek dalam pembuatan diagram alir secara rinci. Diagram alir biasanya disusun dari atas ke bawah dan dari kiri ke kanan, menggambarkan proses secara urut. Setiap aktivitas yang digambarkan harus dijelaskan dengan jelas

dan mudah dipahami oleh pembaca, dengan penentuan waktu awal dan akhir yang pasti. Langkah-langkah aktivitas dijelaskan menggunakan kata kerja dan disusun dalam urutan yang benar. Penting untuk memeriksa lingkup dan rentang setiap aktivitas yang digambarkan. Jika ada percabangan dalam aktivitas, tidak selalu perlu dimasukkan dalam *flowchart* yang sama. Penggunaan simbol penghubung dan pengaturan percabangan yang tepat membantu mengelola percabangan tersebut. Percabangan yang tidak relevan dengan sistem bisa dihilangkan atau ditempatkan di halaman terpisah. Dengan menggunakan simbol-simbol *flowchart* yang standar, alur logika program dapat diilustrasikan secara efektif. Tabel 2.1 Akan menjelaskan simbol-simbol *flowchart* yang biasanya digunakan:

Tabel 2. 1 Simbol-simbol Flowchart

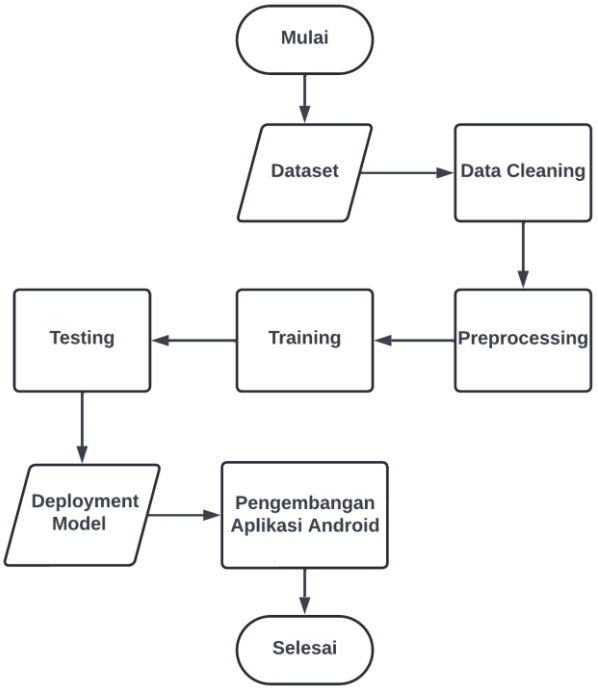
Simbol	Fungsi
	***Terminator Symbol*** Permulaan atau akhir dari suatu program.
	***Processing Symbol*** Proses yang dilakukan oleh sistem.
	***Input-Output*** Menunjukkan proses *input* dan *output*, terlepas dari jenis peralatan yang digunakan.
	***Predefine Proses*** Simbol untuk menjalankan sub-program atau prosedur.

	***Decision*** Simbol yang menandakan suatu keadaan tertentu, di mana menghasilkan dua pilihan.
	***Connector Symbol*** Keluar-masuk atau menghubungkan proses di dalam lembar / halaman yang sama.
	***Flow*** Juga disebut dengan *connecting line*, untuk menghubungkan satu simbol dengan simbol lainnya.

BAB III
METODE PENELITIAN

3.1 Metode Penelitian

Pada penelitian ini memiliki beberapa tahapan yang perlu dilakukan dalam penelitian, dapat terlihat pada Gambar 3. 1



Gambar 3. 1 *Flowchart* Metode Penelitian

3.1.1 *Dataset*

Dataset penelitian dalam klasifikasi penyakit tanaman tomat. *Dataset* ini berasal dari *kaggle* yang dapat diakses melalui (https://www.kaggle.com/datasets/ahmadzargar/tomato-leaf-disease-dataset-segmented). Jumlah data citra adalah 10.000 dengan 10 class jenis penyakit daun tanaman tomat. Dari *dataset* ini, dipilih 5 kelas, yaitu 4 kelas penyakit daun tomat yaitu *bacterial spot, late blight, target spot,* dan *tomato yellow leaf curl virus* dan 1 kelas sehat daun tomat. Masing-masing *class* memiliki 240 gambar, sehingga jumlah *dataset* saat ini adalah 1200. *Dataset* ini berisi gambar tanaman tomat sehat dan yang terinfeksi penyakit.

Gambar 3. 2 Data Gambar Penyakit Daun Tanaman Tomat

3.1.2 Data *Cleaning*

Pada tahap ini, dilakukan penyamarataan jumlah citra gambar pada setiap kelas untuk menghindari hasil yang buruk ketika melakukan prediksi model. Dari total sebelas (10 kelas), dipilihlah lima (5 kelas) yang akan dilakukan pendistribusian jumlah citra gambar yang sama untuk setiap kelasnya. Berikut adalah jumlah citra gambar yang belum dilakukan data *cleaning* dan citra gambar yang sudah dilakukan data *cleaning*.

No	Kelas	Jumlah Gambar
1	Tomato Bacterial Spot	1000
2	Tomato Healthy	1000
3	Tomato Late Blight	1000
4	Tomato Target Spot	1000
5	Tomato Yellow Leaf Curl Virus	1000
6	Tomato Early Blight	1000
7	Tomato Leaf Mold	1000
8	Tomato Septoria Leaf Spot	1000
9	Tomato Spider Mites Two Spotted Spider Mite	1000
10	Tomato Mosaic Virus	1000
Total	-	**10.000**

Gambar 3. 3 *Dataset* Sebelum Dilakukan data *cleaning*

Pada gambar 3.3 merupakan *dataset* tomat yang belum dilakukan data *cleaning* yang berjumlah 10.000 citra dan terdapat jumlah data yang tidak seimbang sehingga mengakibatkan prediksi yang kurang baik.

No	Kelas	Jumlah Gambar
1	Tomato Bacterial Spot	240
2	Tomato Healthy	240
3	Tomato Late Blight	240
4	Tomato Target Spot	240
5	Tomato Yellow Leaf Curl Virus	240
Total	-	**1.200**

Gambar 3. 4 *Dataset* Setelah Dilakukan data *cleaning*

Pada Gambar 3.4 merupakan *Dataset* yang telah dilakukan *data cleaning* dan berisikan 1.200 citra pada kelasnya. Hal ini bertujuan agar Ketika proses pengujian model arsitektur *MobileNetV2* dilakukan, dapat meminimalkan terjadinya *overfitting*. Pada proses pengujian model, *overfitting* sering terjadi dikarenakan banyak faktor diantaranya model terlalu mempelajari data yang sudah ada sehingga sukar untuk mempelajari data baru. Penyamarataan adalah alternatif untuk mengurangi terjadinya overfitting.

3.1.3 *Preprocessing*

Preprocessing adalah Langkah penting dalam proses pengolahan data yang melibatkan persiapan dan pembersihan data mentah yang sebelum dianalisis ataupun diproses lebih lanjut. Tujuan utama dari *preprocessing* adalah untuk memastikan bahwa data yang akan digunakan dalam analisis atau model *machine learning* berada dalam format yang bersih, konsisten, dan siap untuk digunakan sebelum masuk proses *training* atau pelatihan model. Langkah ini mencakup pembersihan, transformasi, reduksi dimensi, dan pembagian data, yang semuanya bertujuan untuk meningkatkan keakuratan dan keandalan hasil analisis atau model yang akan dibangun.

1) *Resize Data*

Resize data yaitu proses di mana gambar yang ada diubah ukurannya untuk memenuhi persyaratan *input* model. Dalam penelitian ini, semua gambar dalam *dataset* disesuaikan ke ukuran 224 piksel x 224 piksel sebelum diberikan ke model, untuk memudahkan proses *training*.

2) *Split* Data

Split data adalah proses membagi *dataset* menjadi beberapa bagian untuk tujuan pelatihan dan pengujian model dalam pengujian mesin atau analisis statistik. Pembagian ini dilakukan untuk memastikan bahwa model yang dibangun mampu melakukan generalisasi dengan baik terhadap data baru yang belum pernah dilihat sebelumnya. Tujuan utama dari *split* data adalah menguji dan mengevaluasi performa model secara objektif. Dengan memisahkan data pelatihan (*training* data) dan data pengujian (testing data), memastikan bahwa model tidak hanya belajar menghafal data, tetapi juga benar-benar memahami pola yang dapat diaplikasikan pada data baru.

3) *Normalization*

Normalization atau normalisasi yang dilakukan dengan tujuan untuk mengubah nilai fitur ke dalam skala yang seragam. Dalam penelitian ini, normalisasi dilakukan menggunakan metode *rescaling* citra. *Rescale* memiliki tujuan untuk mengubah ukuran citra dengan melakukan pembagian nilai piksel terkecil dengan nilai piksel terbesar. Penerapan dalam penelitian ini adalah melakukan transformasi pada ukuran piksel gambar RGB dari rentang nilai 0 hingga 255 sehingga nilai terkecil menjadi 0 dan nilai terbesar menjadi 1. Dengan menggunakan metode *Min-Max Scaling, Z-Score Normalization,* atau lainnya. Kita dapat meningkatkan kinerja dan akurasi model *Machine learning* serta mempercepat proses pelatihan.

4) *Image Augmentation*

Image augmentation adalah teknik yang digunakan dalam *machine learning* dan visi komputer untuk memperluas *dataset* gambar secara artifisial dengan membuat variasi gambar yang ada.Tujuan dari *image augmentation* adalah untuk meningkatkan jumlah dan keberagaman data pelatihan yang dapat membantu model menjadi lebih umum (*generalized*) dan mengurangi *overfitting,* terutama ketika

dataset asli kecil atau terbatas. Pada penelitian ini, variasi yang digunakan antara lain:

1. *Rotation Range,* Transformasi ini digunakan untuk memutar gambar secara acak sehingga dapat membantu model untuk mempelajari fitur yang tidak bergantung pada orientasi objek dalam gambar.
2. *Zoom Range,* Transformasi ini dapat secara acak memperbesar atau memperkecil gambar sehingga membantu model untuk mempelajari fitur yang relevan terlepas dari Tingkat zoom gambar.
3. *Vertical Flip,* Transformasi ini digunakan untuk membalik gambar secara vertikal dan secara acak (mencerminkan sepanjang sumbu atas-bawah). Dapat membantu model mempelajari fitur yang tidak bergantung pada orientasi atas-bawah objek.
4. *Horizontal Flip,* Transformasi ini dgunakan untuk membalik gambar secara horizontal dan secara acak (mencerminkan sepanjang sumbu kiri-kanan). Ini membantu model mempelajari fitur yang tidak bergantung pada orientasi kiri-kanan objek.
5. *Brightness Range,* Transformasi ini secara acak menyesuaikan kecerahan pada gambar, ini dapat membantu model mempelajari fitur yang tidak bergantung pada variasi pencahayaan.

3.1.4 *Training*

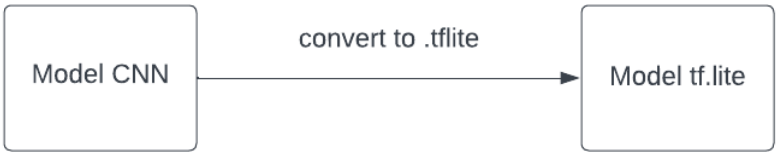
Dataset yang telah dilakukan penyamarataan jumlah citra gambar pada setiap kelas. Kemudian dibagi menjadi 3 bagian antara lain data *training*, data *testing,* dan data *validation.* Rasio pembagian data *training, testing,* serta *validation* masing-masing secara berturut-turut sebanyak 80% : 10% : 10%.

Tabel 3. 1 Rasio data

No	Kelas	Train	Test	Validation
1.	Bacterial Spot	80%	10%	10%
2.	Late Blight	80%	10%	10%

3.	Target Spot	80%	10%	10%
4.	Tomato Yellow Leaf Curl Virus	80%	10%	10%
5.	Healthy	80%	10%	10%
All	Count	1000	100	100

3.1.5 *Deployment Model MobileNetV2*



Gambar 3. 5 Alur *Deployment* Model

Model CNN yang telah dilatih untuk klasifikasi penyakit tomat akan dikonversi menjadi format *TensorFlow Lite* agar bisa diintegrasikan ke dalam aplikasi perangkat *Android*. Proses konversi model *MobileNetV2* ke *TensorFlow Lite* bertujuan untuk mengoptimalkan ukuran model sehingga dapat berjalan dengan cepat dan efisien di perangkat *mobile*. Hasil dari *compile* model kemudian akan disimpan kedalam format '.h5' kemudian akan di *convert* kedalam format '.tflite' menggunakan fungsi python yang nantinya akan digunakan untuk *deployment* aplikasi *mobile*.

3.1.6 Pengembangan Aplikasi *Android*

Pada tahap pengembangan aplikasi *Android*, *Android* Studio digunakan sebagai platform pengembangan utama. Proses yang dilakukan pada pengembangan aplikasi *Android* melibatkan integrasi model CNN dengan arsitektur *MobileNetV2* yang sudah berformat .tflite ke dalam aplikasi. Selanjutnya, dilakukan implementasi *interface* pengguna yang dirancang untuk memfasilitasi pengguna dalam mengambil gambar daun tomat dan melihat hasil deteksi. Antarmuka ini memungkinkan pengguna berinteraksi dengan aplikasi secara langsung dan menyediakan pengalaman yang mudah untuk dipahami.

3.2 Perancangan Alur Sistem

3.2.1 Analisis Kebutuhan

Pada bagian analisis kebutuhan, sistem ini akan dipelajari secara mendalam untuk menentukan segala tindakan yang harus dilakukan selama proses masukan untuk menghasilkan keluaran berupa klasifikasi. Selain itu, ada beberapa proses dan fungsi yang harus diterapkan dalam sistem ini, antara lain:

1) Tips penggunaan aplikasi

Pengguna perlu memahami cara menggunakan aplikasi dengan benar agar dapat menghindari kesalahan saat mengunggah gambar. Disarankan agar gambar yang diunggah hanya menampilkan bagian daun tomat yang terkena penyakit dan kualitas gambar citra yang bagus merupakan faktor tingginya model akan bisa mengenali citra.

2) *Upload* gambar

Dalam fitur *upload* gambar, terdapat dua opsi yang tersedia, yaitu meng-upload gambar dari galeri yang telah menyimpan beberapa gambar atau mengambil gambar langsung dari kamera.

3) Melakukan klasifikasi

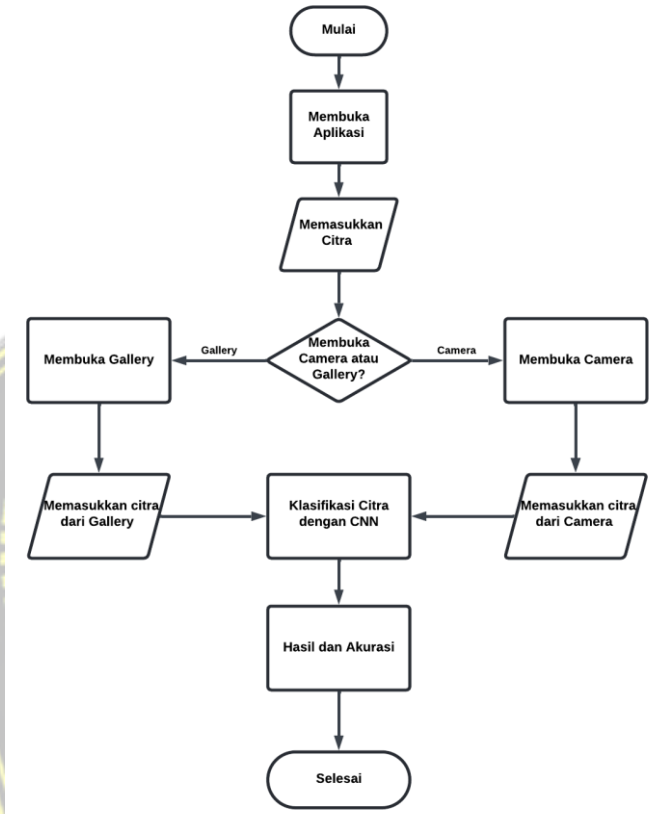
Klasifikasi merupakan fitur utama dalam aplikasi ini. Klasifikasi akan dilakukan secara komputasi oleh sistem ketika pengguna menekan tombol *camera* atau mengambil gambar terlebih dahulu dari *gallery*.

4) Menampilkan hasil klasifikasi

Fungsi ini adalah *output* akhir dari aplikasi. Sistem akan menampilkan hasil klasifikasi gambar yang diberikan pengguna. Proses klasifikasi akan dilakukan secara komputasi berdasarkan dari hasil model yang telah dibuat sebelumnya. Model dengan format '.tflite' yang telah ditambahkan kedalam pembuatan aplikasi akan melakukan prediksi, kemudian hasil prediksi berupa nama penyakit dan akurasi dari tiap input gambar.

3.2.2 Analisis Alur Sistem

Dalam analisis alur sistem ini, terdapat flowchart yang menjelaskan langkah- langkah urutan operasi dari sistem penelitian ini. Mulai dari proses input data hingga output berupa klasifikasi yang dihasilkan sistem. Flowchart ini dapat dilihat pada gambar 3.6.



Gambar 3. 6 *Flowchart* Sistem

Analisis sistem *flowchart* aplikasi dimulai dengan langkah pengguna membuka aplikasi. Pengguna akan diarahkan ke halaman *home*. Terdapat tombol "*Camera"* untuk mengambil gambar, sedangkan Tombol "*Delete"* digunakan untuk menghapus gambar dan tombol "*Gallery*" digunakan untuk menambahkan gambar dari galeri. Ketika pengguna menekan tombol *"Camera"* maka sistem akan otomatis melakukan prediksi gambar. Atau pengguna juga dapat memilih tombol *"Gallery"* untuk menambahkan gambar dari galeri yang kemudian akan otomatis menampilkan hasil dari

prediksi. Hasil deteksi tersebut kemudian ditampilkan pada antarmuka pengguna berupa hasil klasifikasi.

3.2.3 Analisis Sistem

Pada tahap analisis sistem, kebutuhan berbagai alat dan perangkat yang diperlukan untuk pengembangan sistem ini akan diidentifikasi. Informasi terperinci mengenai alat dan perangkat yang akan digunakan tersedia dalam tabel 3.2.

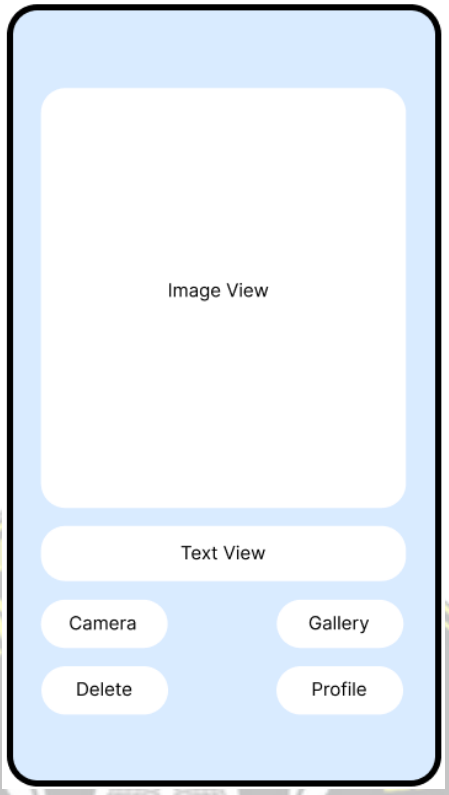
Tabel 3. 2 Analisis Sistem

No	Tool	Version
1	Laptop Acer Nitro AN515-57	*Windows* 11
2	Smartphone Samsung Galaxy A32	*Android* 11
4	*Google Collab*	T4 GPU
6	*Android* Studio	Koala, 2024.1.1
7	*Kotlin*	1.9.0
8	*Gradle*	8.7
9	SDK	34

3.3 Perancangan *User Interface*

Dalam proses perancangan *user interface*, dilakukan desain *mockup* yang akan diimplementasikan ke dalam sistem. Berikut adalah tampilan *user interface* yang telah dirancang untuk sistem Klasifikasi Citra Penyakit Pada Daun Tanaman Tomat (*Lycopersicum Esculentum Mill*.) Menggunakan Metode CNN Berbasis *Android*.

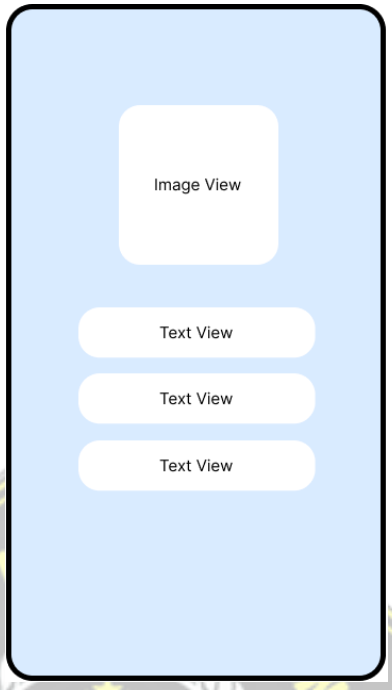
3.3.1 Halaman Home



Gambar 3. 7 Halaman *Home*

Halaman Home merupakan tampilan pertama dalam aplikasi ini. Pada halaman ini terdapat 4 menu utama yaitu *Camera, Gallery*, *Delete,* dan *Profile*. Di mana pada halaman tersebut, pengguna dapat melakukan klasifikasi secara langsung dengan menekan tombol "*Camera"* atau menambahkan gambar terlebih dahulu dengan menekan tombol Gallery yang kemudian akan diproses oleh sistem secara otomatis untuk melakukan prediksi gambar. Pada tombol "Delete" pengguna dapat menghapus gambar yang ada pada frame. Pada tombol "Profile" pengguna dapat melihat informasi singkat penulis. Tampilan halaman *Home* dapat dilihat pada gambar 3.7.

3.3.2 Halaman Profil



Gambar 3. 8 Halaman Profil

Pada Gambar 3. 8 merupakan desain UI pada bagian profil. Halaman profil ini memuat informasi penulis yang berisikan informasi singkat tentang penulis.

BAB IV
HASIL DAN ANALISIS PENELITIAN

4.1 Proses Penggunaan dan Cara Kerja Sistem

4.1.1 Upload Gambar

Pertama, *user* memasukkan gambar penyakit tomat yang akan diklasifikasi. Dalam sistem, terdapat dua pilihan untuk melakukan *upload* gambar: mengambil gambar dari kamera *smartphone user* atau memilih gambar dari galeri *user*.

4.1.2 Konversi Gambar ke TensorImage

Pada tahap ini sistem akan mengambil gambar yang telah di-*upload* pengguna dan menyimpannya dalam bentuk *Bitmap*. Kemudian gambar akan diubah menjadi TensorImage. Proses konversi ini dilakukan untuk mendapatkan gambar yang sesuai dengan ukuran yang diharapkan oleh model, yaitu me-*resize* gambar menjadi 224 piksel x 224 piksel.

4.1.3 Konversi TensorImage ke TensorBuffer

Selanjutnya sistem akan mengkonversi TensorImage menjadi TensorBuffer. TensorBuffer digunakan untuk menyimpan representasi gambar yang sudah diubah menjadi TensorImage. Selanjutnya, TensorBuffer siap untuk proses klasifikasi oleh model.

4.1.4 Proses Klasifikasi

Pada tahap ini, gambar yang sudah menjadi TensorBuffer akan diproses oleh model tflite untuk dilakukan klasifikasi. Kemudian sistem akan menampilkan hasil klasifikasi pada gambar yang telah di input melalui berbagai tahapan dan hasilnya akan menunjukkan label tentang penyakit dan persentase tingkat keakuratan pada penyakit tomat.

4.1.5 Pengujian Model *MobileNetV2* dengan *Hyperparameter*

Penggunaan *hyperparameter* seperti rescale 1/255, bertujuan untuk membagi nilai dari 3 channel RGB sehingga didapatkan nilai baru dengan *range* antara 0 sampai 1. Kemudian juga terdapat hyperparameter lainnya seperti flipping (Horizontal, Vertical) yang diterapkan sebagai berikut :

Tabel 4. 1 Parameter

NO	Parameter	Value
1.	Rescale	1./255
2.	Zoom range	0.2
3.	Vertical flip	True
4.	Horizontal flip	True
5.	Brightness range	[0.8, 1.2]

Pada tabel 4.1 merupakan pemberian parameter sebelum dilakukannya proses pelatihan dengan arsitektur *MobileNetV2*. Hyperparameter mengontrol perilaku pelatihan dan struktur model, seperti tingkat pembelajaran (*learning rate*), ukuran batch (*batch size*), jumlah lapisan(layers), dan sebagainya. Mereka berbeda dari parameter model, yang dipelajari secara langsung oleh model selama proses pelatihan.

4.1.6 Pengujian dengan *Optimizer* SGD

Stochastic Gradient Descent (**SGD**) adalah algoritma pengoptimalan yang umum digunakan dalam pelatihan pembelajaran mesin dan model pembelajaran mendalam. SGD adalah variasi dari penurunan gradien klasik, tetapi lebih efisien untuk kumpulan data besar. Tujuan utama algoritma ini adalah meminimalkan fungsi kerugian dengan memperbarui parameter model secara berulang berdasarkan gradien yang dihitung dari data. Tidak seperti penurunan gradien biasa, yang menggunakan seluruh himpunan data pada setiap iterasi untuk menghitung gradien, SGD hanya menggunakan sampel data yang dipilih secara acak.

Proses kerja SGD dimulai dengan menginisialisasi parameter model dengan nilai acak. Kemudian, kami mengambil sampel data secara acak dari kumpulan data dan menghitung gradien fungsi kerugian terhadap parameter.

Salah satu keuntungan utama SGD adalah kecepatannya dalam memproses kumpulan data yang sangat besar. Setiap langkah pembaruan hanya membutuhkan satu sampel, menghasilkan waktu komputasi yang lebih cepat dibandingkan dengan penurunan gradien batch, di mana gradien dihitung menggunakan seluruh himpunan data. Selain itu, SGD hemat

memori karena tidak perlu menyimpan seluruh kumpulan data dalam memori selama komputasi.

Meskipun memiliki keterbatasan, SGD tetap menjadi pilihan pertama untuk mengoptimalkan model pembelajaran mesin karena kesederhanaan, kecepatan, dan kemampuannya menangani kumpulan data besar. Menggabungkan berbagai teknik perbaikan membuat pelatihan model terkini menjadi lebih efektif.

Penggunaan Optimizer yang berbeda dapat memberikan nilai akurasi yang berbeda juga. Penulis menggunakan *optimizer SGD* dengan learning rate '0.0001' yang nantinya akan digunakan saat meng-compile model dengan arsitektur *MobileNetV2*.

4.2 Implementasi User Interface

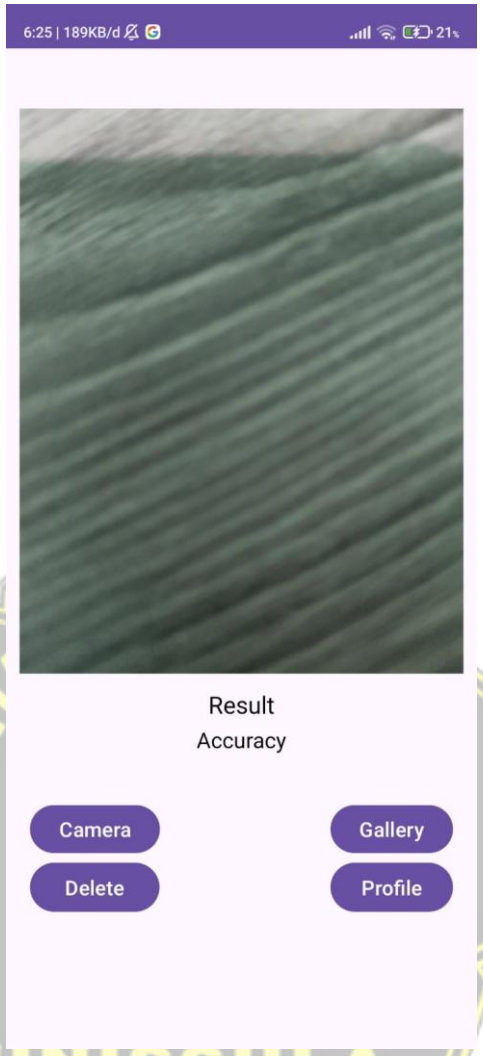
4.2.1 Halaman *SplashScreen*

Gambar 4. 1 Halaman *SplashScreen*

Pada halaman *SplashScreen* merupakan tampilan awal aplikasi, Di mana terdapat logo daun tomat. Terdapat jeda selama 3 detik pada halaman *splashscreen* sebelum menuju halaman *home.*

4.2.2 Halaman Home

Pada halaman Home terdapat beberapa tombol untuk memudahkan interaksi pengguna dengan aplikasi. Tampilan home terlihat seperti pada gambar 4.2.



Gambar 4. 2 Tampilan *Home*

Pada Gambar 4.2 aplikasi terdiri dari *Button, TextView,* serta *ImageView. Button* pada halaman ini dibagi menjadi empat, yaitu *Capture, Gallery, Delete,* dan *Profile. Capture* berfungsi untuk mengambil gambar secara langsung dan menampilkan hasil gambar pada *ImageView. Gallery* berfungsi untuk menambahkan gambar dari galeri yang kemudian akan diproses oleh sistem dan menampilkannya melalui *ImageView*. *Delete* berfungsi untuk menghapus gambar pada *ImageView*. *Profile* berfungsi untuk menampilkan informasi singkat tentang penulis. *ImageView* dalam halaman utama memiliki fungsi untuk menampilkan gambar yang nantinya akan di input oleh pengguna. Kemudian terdapat pula *TextView* yang berfungsi untuk menampilkan nama penyakit dan akurasi dari citra gambar.

4.2.3 *Output* Klasifikasi

Klasifikasi akan dimulai setelah pengguna meng-*upload* gambar. Sistem akan memulai proses klasifikasi untuk mengidentifikasi jenis penyakit tomat yang terdapat pada gambar yang diunggah. Setelah proses klasifikasi selesai, halaman akan menampilkan hasil klasifikasi jenis penyakit tomat pada *textview result* dan *accuracy.* Tampilan ini dapat dilihat pada gambar 4.3.

Gambar 4. 3 Tampilan *Output* Klasifikasi

Pada Gambar 4.3 merupakan tampilan *output* klasifikasi yang menunjukkan bahwa daun pada gambar terkena penyakit *Tomato Bacterial Spot* dengan akurasi 96,68%, dengan demikian daun tersebut positif terjangkit penyakit.

4.3 Pengujian Sistem

Pada tahap ini, digunakan metode *black box testing* yang merupakan pendekatan untuk menguji fungsionalitas sistem tanpa memperhatikan detail struktur atau logika internalnya. Metode ini bertujuan untuk memastikan bahwa sistem berfungsi sesuai dengan spesifikasi yang telah ditentukan. Dalam penelitian ini, *function testing* akan digunakan untuk menguji apakah setiap fungsi beroperasi dengan benar sesuai dengan yang diharapkan. Berikut tabel 4. 1 yang merupakan hasil *function testing* pada sistem ini.

Tabel 4. 2 Testing *Black Box* pada Sistem

Skenario Pengujian	**Kasus Pengujian**	**Hasil Pengujian**	**Kesimpulan**
Membuka aplikasi	Menampilkan halaman home	Sesuai	Normal
Upload gambar dari kamera	Melakukan *upload* gambar penyakit tomat dari kamera	Sesuai	Normal
Upload gambar dari galeri	Melakukan *upload* gambar penyakit tomat dari galeri	Sesuai	Normal
Menekan tombol *Delete*	Menghapus gambar yang terdapat pada frame	Sesuai	Normal
Menekan tombol *Profile*	Menampilkan halaman Profile	Sesuai	Normal
Testing gambar daun Tomat *Healty*	Melakukan *testing* pada gambar yang menunjukkan jenis daun tomat *Healty*	Sesuai	Normal

Testing gambar daun Tomat *Bacterial Spot*	Melakukan *testing* pada gambar yang menunjukkan jenis penyakit tomat *Bacterial Spot*	Sesuai	Normal
Testing gambar daun Tomat *Late Blight*	Melakukan *testing* pada gambar yang menunjukkan jenis penyakit tomat *Late Blight*	Sesuai	Normal
Testing gambar daun Tomat *Target Spot*	Melakukan *testing* pada gambar yang menunjukkan jenis penyakit tomat *Target Spot*	Sesuai	Normal
Testing gambar daun Tomat *Yellow Leaf Curl Virus*	Melakukan *testing* pada gambar yang menunjukkan jenis penyakit tomat *Yellow Leaf Curl Virus*	Sesuai	Normal

4.4 Hasil dan Analisis

4.4.1 *Training* dan *Validation*

Dataset yang dibagi menjadi data *training* dan *validation* berjumlah 1200 gambar. Data *training* terdiri dari 1000 gambar yang terdiri dari kelas *bacterial spot*, *late blight*, *target spot*, *tomato yellow leaf curl virus* dan *healthy*. Untuk data *validation* berjumlah 100 gambar untuk kelas *bacterial spot*, *late blight*, *target spot*, *tomato yellow leaf curl virus* dan *healthy*.

Hasil pelatihan dan validasi ditampilkan dalam grafik yang menunjukan accuracy dan loss. Grafik juga menunjukan hasil konfigurasi yang telah digunakan. Untuk mendapatkan pemahaman visual tentang bagaimana model

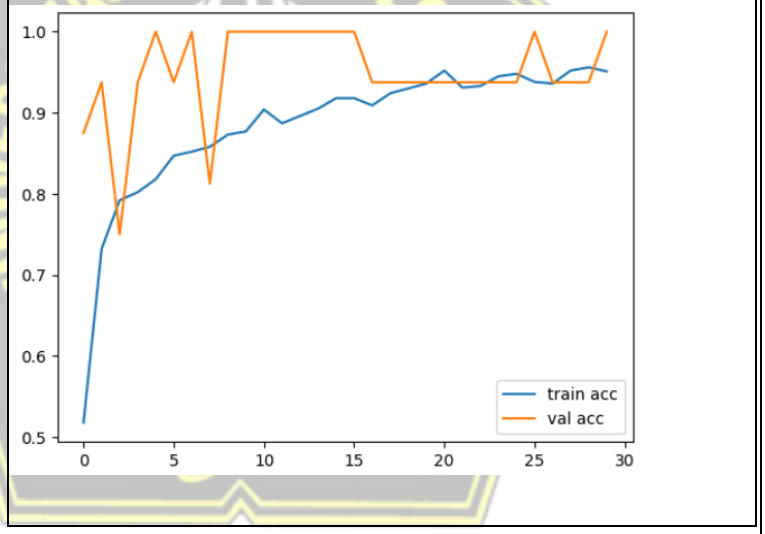
berfungsi selama diproses pelatihan dan validasi, grafik ketepatan dan kehilangan nilai dapat digunakan. Sebaliknya, konfigurasi yang digunakan dalam penelitian ini mengacu pada *optimizer* yang digunakan.

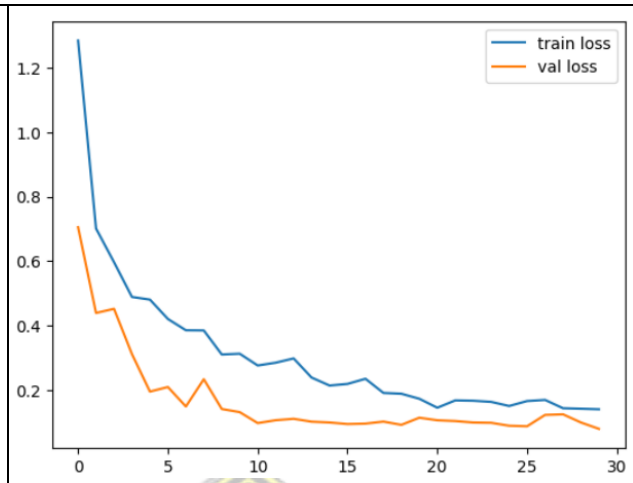
Grafik yang menunjukkan ketepatan dan kehilangan untuk berbagai konfigurasi dapat ditemukan di Tabel 4.2. Sementara grafik pelatihan dan validasi semakin dekat, grafik ketepatan semakin meningkat dan grafik kehilangan semakin menurun. Ini menunjukkan peningkatan kinerja model tersebut.

Tabel 4. 3 Konfigurasi *Optimizer SGD*

<table>
<tr><td colspan="2">

- Learning rate = 0.0001
- Batch size = 16

</td></tr>
<tr><td>Accuracy</td><td> 
</td></tr>
</table>

Loss	

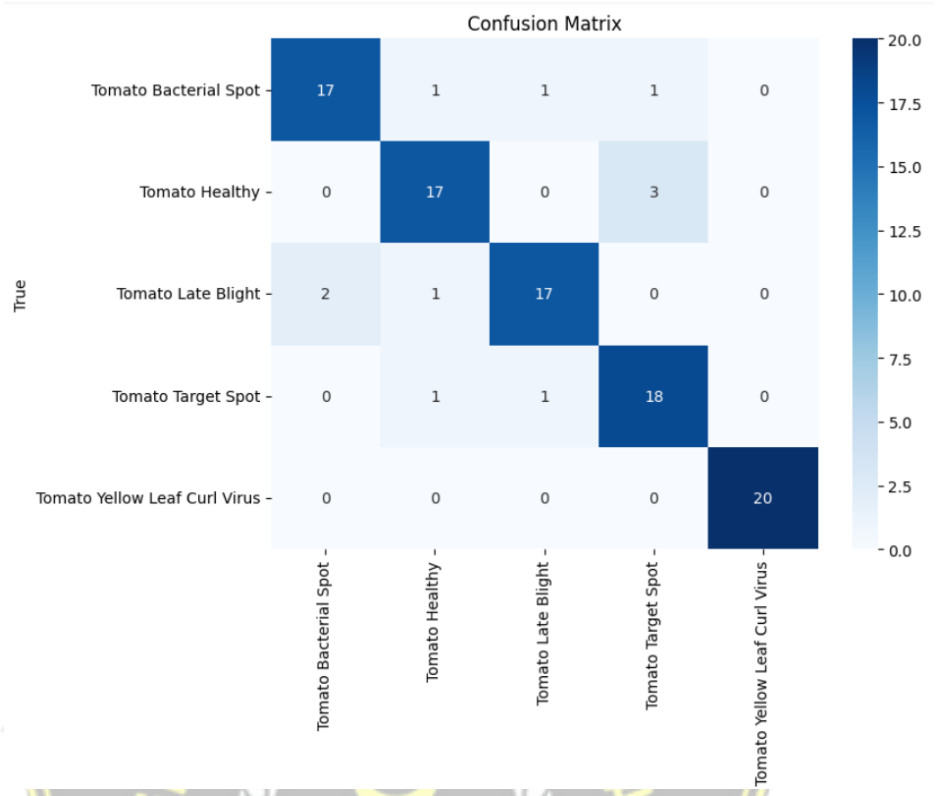
Dalam proses pengujian/*training* didapatkan hasil *accuracy* dan hasil *loss* sesuai gambar 4.2 sebesar secara berturut turut accuracy 0,9412, loss 0,1503, validasi *accuracy* 1,0000, dan validasi *loss* 0,0797 dengan epoch 30 dan batch size 16.

Tabel 4. 4 Hasil *Compile Optimizer SGD*

Percobaan	*epochs*	*loss*	*accuracy*	*val-loss*	*val-accuracy*
Konfigurasi (SGD)	30/30	0.1503	0.9412	0.0797	1.0000

4.4.2 *Testing*

Pada tahap *testing*, dilakukan untuk mengevaluasi seberapa akurat sistem dalam mengklasifikasikan *data testing* yang terdiri dari 5 *class* dan berjumlah 100 citra. *Confusion matrix* digunakan untuk membantu dalam menghitung *accuracy, precission, recall* dan *f1 score*. Berikut proses perhitungan konfigurasi matrix.



Gambar 4. 4 *Confusion Matrix*

Tabel 4.4 menampilkan hasil dari *confusion matrix* untuk masing-masing kelas.

Tabel 4. 5 *Precision, recall*, dan *F1-Score* tiap kelas

Kelas	*Precission*	*Recall*	*F1-Score*
Bacterial Spot	0.89	0.85	0.87
Healthy	0.85	0.85	0.85
Late Blight	0.89	0.85	0.87
Target Spot	0.82	0.90	0.86
Yellow Leaf Curl Virus	1.00	1.00	1.00

Tabel 4.4 menunjukkan hasil *precision, recall,* dan *F1-Score* untuk masing-masing kelas.

Tabel 4. 6 *Confusion matrix* tiap kelas

Kelas	TP	FP	FN	TN
Bacterial Spot	17	2	3	78
Healthy	17	3	3	77

Late Blight	17	2	3	78
Target Spot	18	4	2	76
Yellow Leaf Curl Virus	20	0	0	80

Tabel 4.5 menunjukkan hasil Nilai *True Positive* (TP), *False Positive* (FP), *False Negative* (FN), dan *True Negative* (TN) dari masing-masing kelas.

Tabel 4. 7 Perhitungan Manual

	Perhitungan	**Hasil**
$\boldsymbol{Precision\ (P) = \frac{TP}{TP + FP}}$		
P_{bs}	$= \frac{TP_{bs}}{TP_{bs} + FP_{bs}} = \frac{17}{17+2} = \frac{17}{19} = 0.89$	0.89
P_{h}	$= \frac{TP_{h}}{TP_{h} + FP_{h}} = \frac{17}{17+3} = \frac{17}{20} = 0.85$	0.85
P_{lb}	$= \frac{TP_{lb}}{TP_{lb} + FP_{lb}} = \frac{17}{17+2} = \frac{17}{19} = 0.89$	0.89
P_{ts}	$= \frac{TP_{ts}}{TP_{ts} + FP_{ts}} = \frac{18}{18+4} = \frac{18}{22} = 0.82$	0.82
P_{ylcv}	$= \frac{TP_{ylcv}}{TP_{ylcv} + FP_{ylcv}} = \frac{20}{20+0} = \frac{20}{20} = 1.00$	1.00
$\boldsymbol{Recall\ (R) = \frac{TP}{TP + FN}}$		
R_{bs}	$= \frac{TP_{bs}}{TP_{bs} + FN_{bs}} = \frac{17}{17+3} = \frac{17}{20} = 0.85$	0.85
R_{h}	$= \frac{TP_{h}}{TP_{h} + FN_{h}} = \frac{17}{17+3} = \frac{17}{20} = 0.85$	0.85
R_{lb}	$= \frac{TP_{lb}}{TP_{lb} + FN_{lb}} = \frac{17}{17+3} = \frac{17}{20} = 0.85$	0.85
R_{ts}	$= \frac{TP_{ts}}{TP_{ts} + FN_{ts}} = \frac{18}{18+2} = \frac{18}{20} = 0.90$	0.90
R_{ylcv}	$= \frac{TP_{ylcv}}{TP_{ylcv} + FN_{ylcv}} = \frac{20}{20+0} = \frac{20}{20} = 1.00$	1.00
$\boldsymbol{F1 - Score = \frac{2 * Precision * Recall}{Precision * Recall}}$		

$F1_{bs}$	$\frac{2 * P_{bs} * R_{bs}}{P_{bs} + R_{bs}} = 2 * \frac{0.89 * 0.85}{0.89 + 0.85} = \frac{1.51}{1.74} = 0.87$	0.87
$F1_{h}$	$2 * \frac{P_{h} * R_{h}}{P_{h} + R_{h}} = 2 * \frac{0.85 * 0.85}{0.85 + 0.85} = \frac{1.44}{1.7} = 0.85$	0.85
$F1_{lb}$	$2 * \frac{P_{lb} * R_{lb}}{P_{lb} + R_{lb}} = 2 * \frac{0.89 * 0.85}{0.89 + 0.85} = \frac{1.51}{1.74} = 0.87$	0.87
$F1_{ts}$	$2 * \frac{P_{ts} * R_{ts}}{P_{ts} + R_{ts}} = 2 * \frac{0.82 * 0.90}{0.82 + 0.90} = \frac{1.51}{1.74} = 0.87$	0.87
$F1_{ylcv}$	$2 * \frac{P_{ylcv} * R_{ylcv}}{P_{ylcv} + R_{ylcv}} = 2 * \frac{1.00 * 1.00}{1.00 + 1.00} = \frac{2}{2} = 1.00$	1.00

Keterangan:

bs = Bacterial Spot

h = Healthy

lb = Late Blight

ts = Target Spot

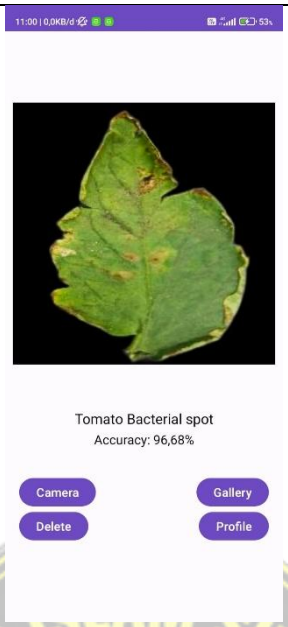
ylcv = Yellow Leaf Curl Virus

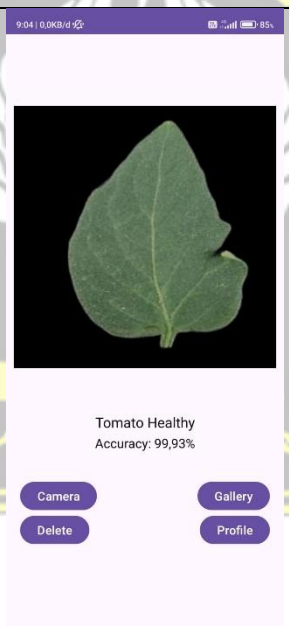
4.4.3 Hasil *Testing* Aplikasi

Pada bagian ini akan menampilkan hasil testing dan analisis Klasifikasi Citra penyakit pada daun tanaman tomat (*Lycopersicum Esculentum Mill.*) menggunakan metode CNN berbasis *Android* serta arsitektur *MobileNetV2* yang telah melakukan uji coba dan beberapa konfigurasi untuk menentukan hasil yang tepat dan baik. Berikut hasil testing menggunakan aplikasi yang dapat dilihat pada tabel 4.7.

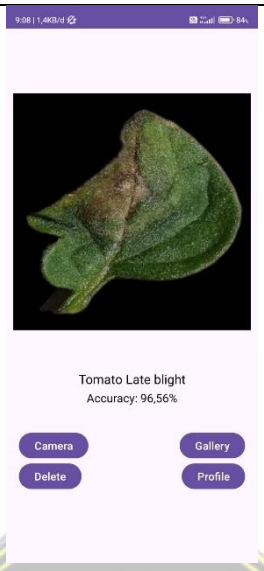
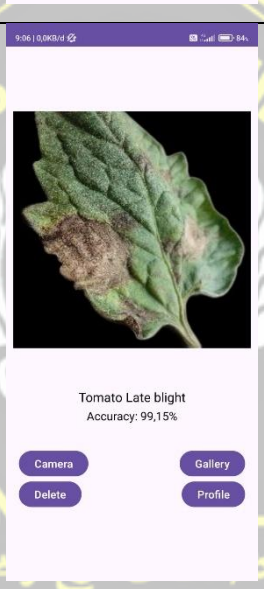
Tabel 4. 8 Hasil Testing Prediksi Aplikasi

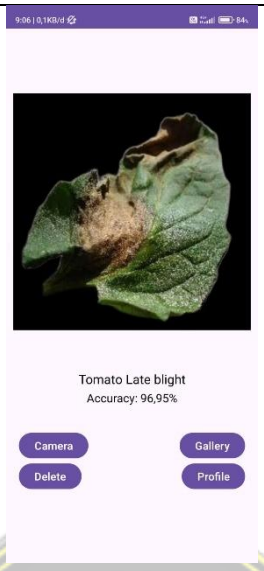
No	Gambar	Keterangan
	Bacterial Spot	

1.		Sesuai
2.		Sesuai

3.		Sesuai
	Healthy	
1.		Sesuai

2.		Sesuai
3.		Sesuai
	Late Blight	

1.		Sesuai
2.		Sesuai

3.		Sesuai
	Target Spot	
1.		Sesuai

2.		Sesuai
3.		Sesuai
	Tomato Yellow Leaf Curl Virus	

1.		Sesuai
2.		Sesuai

3.		Sesuai

Pengujian aplikasi dilakukan dengan memasukkan 3 *input* gambar dari masing-masing kelas penyakit yang berbeda-beda. Dalam pengujian aplikasi ini, *output* dari hasil klasifikasi sama dengan gambar dari kelasnya. Hal ini menunjukkan bahwa aplikasi berjalan dengan baik dalam melakukan klasifikasi serta model *MobileNetV2* yang digunakan memiliki hasil yang baik.

4.4.4 Evaluasi Lapisan *MobileNetV2*

Pada tahap ini dilakukan evaluasi pada lapisan *MobilenetV2* dengan perbedaan jumlah layer tambahan setelah model dasar (base model). Berikut adalah perbedaan akurasi berdasarkan jumlah lapisan dan ukuran dataset:

Tabel 4. 9 Tabel Perbedaan Akurasi Jumlah Layer

Jumlah lapisan	Akurasi Dataset Kecil	Akurasi Dataset Besar
3 lapisan	88%	95%
5 lapisan	84%	87%
6 lapisan	66%	81%

Pada Tabel 4.9 terlihat bahwa:

1. Dengan 3 lapisan, akurasi tinggi baik pada dataset kecil maupun besar.
2. Dengan 5 lapisan, akurasi sedikit turun dibanding 3 lapisan, tetapi masih cukup baik.
3. Dengan 6 lapisan, akurasi turun signifikan pada dataset kecil, tetapi tetap cukup baik pada dataset besar.

Kesimpulannya, semakin banyak lapisan, model cenderung mengalami overfitting pada dataset kecil tetapi tetap berkinerja baik pada dataset besar. Kemudian, diantara jumlah lapisan dengan akurasi terbaik yaitu sebesar 95% terdapat pada penggunaan 3 lapisan dengan dataset besar serta jumlah lapisan dengan akurasi terburuk yaitu sebesar 66% terdapat pada penggunaan 6 lapisan dengan dataset kecil.

BAB V
HASIL DAN ANALISIS PENELITIAN

5.1 Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan, dapat disimpulkan bahwa menggunakan algoritma *Convolutional Neural Network* dengan arsitektur *MobileNetV2* dapat menghasilkan akurasi yang tinggi dalam mengklasifikasikan 4 jenis penyakit daun tomat dan 1 jenis daun tomat. Hal itu telah teruji dengan perbandingan konfigurasi dan model tersebut yang telah diimplementasikan pada aplikasi *mobile* dengan menggunakan *Kotlin language*. Berikut adalah beberapa poin kesimpulan yang penulis peroleh selama penelitian ini:

1. *MobileNetV2* bisa diadaptasi untuk mendeteksi penyakit daun tomat dengan hasil yang baik pada sistem berbasis *Android* berdasarkan akurasi sebesar 94%.
2. Pada tahap *training,* konfigurasi terbaik terdapat pada konfigurasi dengan 30 *epoch* dengan *accuracy* 0.94, *loss* 0.15, val-loss 0.07, dan *val-accuracy* 1.00.
3. Pada penelitian ini semakin banyak lapisan, model cenderung mengalami overfitting pada dataset kecil, tetapi tetap berkinerja baik pada dataset besar. Kemudian, diantara jumlah lapisan dengan akurasi terbaik yaitu sebesar 95% terdapat pada penggunaan 3 lapisan dengan dataset besar, serta jumlah lapisan dengan akurasi terburuk yaitu sebesar 66% terdapat pada penggunaan 6 lapisan dengan dataset kecil.

5.2 Saran

Berdasarkan penelitian yang telah dilakukan, hasilnya menunjukkan performa yang cukup memuaskan. Meskipun demikian, untuk penelitian selanjutnya peneliti menyarankan:

1. Disarankan untuk menambahkan jenis penyakit daun tomat sehingga model dapat mempelajari berbagai karakteristik dan kasus yang lebih dalam. Hal ini diharapkan untuk meminimalkan terjadinya *overfitting*.
2. Disarankan pada penelitian selanjutnya sistem dapat dikembangkan, sehingga tidak hanya melakukan klasifikasi pada gambar penyakit daun tomat.
3. Disarankan untuk meningkatkan arsitektur *MobileNetV2* dengan versi terbaru dari *MobileNet* untuk meningkatkan kemampuan pengenalan fitur yang relevan dan performa keseluruhan model.
4. Pada saat ini, sistem hanya dapat memproses *image detection,* diharapkan pada penelitian selanjutnya dapat menggunakan fitur *real time,* untuk mempercepat pekerjaan pengguna.

DAFTAR PUSTAKA

Abdulridha, J., Ampatzidis, Y., Qureshi, J., & Roberts, P. (2020). Laboratory and UAV-based identification and classification of tomato yellow leaf curl, bacterial spot, and target spot diseases in tomato utilizing hyperspectral imaging and machine learning. *Remote Sensing*, *12*(17). https://doi.org/10.3390/RS12172732

Adhikari, P., Adhikari, T. B., Louws, F. J., & Panthee, D. R. (2020). Advances and challenges in bacterial spot resistance breeding in tomato (Solanum lycopersicum L.). *International Journal of Molecular Sciences*, *21*(5), 1–14. https://doi.org/10.3390/ijms21051734

Aini, D. H. N., Kurniasari, D., Nuryaman, A., & Usman, M. (2023). Implementation of Artificial Neural Network With Backpropagation Algorithm for Rating Classification on Sales of Blackmores in Tokopedia. *Jurnal Teknik Informatika (Jutif)*, *4*(2), 365–372. https://doi.org/10.52436/1.jutif.2023.4.2.539

Ali, M. Y., Sina, A. A. I., Khandker, S. S., Neesa, L., Tanvir, E. M., Kabir, A., Khalil, M. I., & Gan, S. H. (2021). Nutritional composition and bioactive compounds in tomatoes and their impact on human health and disease: A review. *Foods*, *10*(1). https://doi.org/10.3390/foods10010045

Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M. A., Al-Amidie, M., & Farhan, L. (2021). Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. In *Journal of Big Data* (Vol. 8, Nomor 1). Springer International Publishing. https://doi.org/10.1186/s40537-021-00444-8

Annur, I. F., Umami, J., Annafii, M. N., Trisnaningrum, N., & Putra, O. V. (2023). Klasifikasi Tingkat Keparahan Penyakit Leafblast Tanaman Padi Menggunakan MobileNetv2. *Fountain of Informatics Journal*, *8*(1), 7–14. https://doi.org/10.21111/fij.v8i1.9419

Badan Pusat Statistik. (2024). *No Title*. https://www.bps.go.id/id/statistics-

table/2/NjEjMg==/produksi-tanaman-sayuran.html

Ding, X., Xia, C., Zhang, X., Chu, X., Han, J., & Ding, G. (2021). *RepMLP: Re-parameterizing Convolutions into Fully-connected Layers for Image Recognition*. *2017*. http://arxiv.org/abs/2105.01883

Dwitomo, A. B., Adi Kristanto, B., & Kusmiyati, F. (2023). Pengaruh Aplikasi Cendawan Mikoriza Arbuskular dan Pemupukan NPK Majemuk dalam Pertumbuhan dan Produksi Tanaman Tomat. *Jurnal Agroplasma*, *10*(1), 1–12.

Febriady Marpaung, N. K. R. M. M. S. (2024). *Penerapan Arsitektur Mobilenetv2 Pada*. 215–225.

Hanif, R. A. (2023). *Program studi diii teknik elektronika politeknik harapan bersama tegal 2023*.

Hendri Butar-Butar, R. J., & Marpaung, N. L. (2023). Deep Learning untuk Identifikasi Daun Tanaman Obat Menggunakan Transfer Learning MobileNetV2. *Jurnal Informatika: Jurnal Pengembangan IT*, *8*(2), 142–148. https://doi.org/10.30591/jpit.v8i2.5217

Jie, H. J., & Wanda, P. (2020). Runpool: A dynamic pooling layer for convolution neural network. *International Journal of Computational Intelligence Systems*, *13*(1), 66–76. https://doi.org/10.2991/ijcis.d.200120.002

Kayhan, O. S., & van Gemert, J. C. (2020). On translation invariance in CNNs: Convolutional layers can exploit absolute spatial location. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *class 2*, 14262–14273. https://doi.org/10.1109/CVPR42600.2020.01428

Khultsum, U., & Subekti, A. (2021). Penerapan Algoritma Random Forest dengan Kombinasi Ekstraksi Fitur Untuk Klasifikasi Penyakit Daun Tomat. *Jurnal Media Informatika Budidarma*, *5*(1), 186. https://doi.org/10.30865/mib.v5i1.2624

Monteiro, S., Sokolovas, E., Wittingen, E., Dijk, T. Van, & Huisman, M. (2021). IntelliJML: A JML plugin for IntelliJ IDEA. *FTfJP 2021 - Proceedings of the 23rd ACM International Workshop on Formal Techniques for Java-Like*

Programs, co-located with ECOOP/ISSTA 2021, 39–42. https://doi.org/10.1145/3464971.3468423

Morgan, D., & Jacobs, R. (2020). Opportunities and Challenges for Machine Learning in Materials Science. *Annual Review of Materials Research*, *50*, 71–103. https://doi.org/10.1146/annurev-matsci-070218-010015

Mungki Astiningrum, Arhandi, P. P., & Ariditya, N. A. (2020). Identifikasi Penyakit Pada Daun Tomat Berdasarkan Fitur Warna Dan Tekstur. *Jurnal Informatika Polinema*, *6*(2), 47–50. https://doi.org/10.33795/jip.v6i2.320

Pantang, L. S., Yusnaeni, Y., Ardan, A. S., & Sudirman, S. (2021). Efektivitas Pupuk Organik Cair Limbah Rumah Tangga dalam Meningkatkan Pertumbuhan dan Produksi Tanaman Tomat (Lycopersicum esculentum Mill.). *EduBiologia: Biological Science and Education Journal*, *1*(2), 85. https://doi.org/10.30998/edubiologia.v1i2.8966

Prima, A. (2023). Rancang Bangun Sistem Pendeteksi Aneka Ragam Buah Menggunakan MobileNetv2. *Jurnal Sistim Informasi dan Teknologi*, *5*(2), 208–215. https://doi.org/10.60083/jsisfotek.v5i2.217

Riansyah, B., Kurniawan, D., & Same, M. (2021). Sistem Pakar Diagnosa Penyakit Tanaman Kopi Menggunakan Metode Dempster Shafer. *Jurnal Komputasi*, *9*(1). https://doi.org/10.23960/komputasi.v9i1.2420

Sallang, N. C. A., Islam, M. T., Islam, M. S., & Arshad, H. (2021). A CNN-Based Smart Waste Management System Using TensorFlow Lite and LoRa-GPS Shield in Internet of Things Environment. *IEEE Access*, *9*, 153560–153574. https://doi.org/10.1109/ACCESS.2021.3128314

Saputra, R. A., Wasiyanti, S., Supriyatna, A., & Saefudin, D. F. (2021). Penerapan Algoritma Convolutional Neural Network Dan Arsitektur MobileNet Pada Aplikasi Deteksi Penyakit Daun Padi. *Swabumi*, *9*(2), 184–188. https://doi.org/10.31294/swabumi.v9i2.11678

Soekarta, R., Nurdjan, N., & Syah, A. (2023). Klasifikasi Penyakit Tanaman Tomat Menggunakan Metode Convolutional Neural Network (CNN). *Insect (Informatics and Security): Jurnal Teknik Informatika*, *8*(2), 143–151. https://doi.org/10.33506/insect.v8i2.2356

Vincent, D. P., Elie, K. K., & Gaston, T. N. (2023). Impact of Introduced Tomato Cultivars and Chemical Fungicides on the Occurrence and Intensity of Late Blight in Western Cameroon. *Journal of Experimental Agriculture International*, *45*(9), 71–84. https://doi.org/10.9734/jeai/2023/v45i92177

Winnarto, M. N., Mailasari, M., & Purnamawati, A. (2022). Klasifikasi Jenis Tumor Otak Menggunakan Arsitekture Mobilenet V2. *Jurnal SIMETRIS*, *13*(2), 1–12.

Yan, Z., Wolters, A. M. A., Navas-castillo, J., & Bai, Y. (2021). The global dimension of tomato yellow leaf curl disease: Current status and breeding perspectives. *Microorganisms*, *9*(4), 1–19. https://doi.org/10.3390/microorganisms9040740

Zayd, M. H., Oktavian, M. W., Meranggi, D. G. T., Figo, J. A., & Yudistira, N. (2022). Improvement of garbage classification using pretrained Convolutional Neural Network. *Teknologi*, *12*(1), 1–8. https://doi.org/10.26594/teknologi.v0i0.2403