

**IMPLEMENTASI ALGORITMA *CONVOLUTIONAL NEURAL NETWORK* (CNN) DALAM KLASIFIKASI JENIS SERANGGA
HAMA PADA GUDANG BERAS MENGGUNAKAN MODEL
MOBILENETV3-LARGE BERBASIS *WEBSITE***

SKRIPSI

RIZKI CHRISMASYADI SIANIPAR

201401063



**PROGRAM STUDI S-1 ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS SUMATERA UTARA
MEDAN
2025**

**IMPLEMENTASI ALGORITMA *CONVOLUTIONAL NEURAL NETWORK*
(CNN) DALAM KLASIFIKASI JENIS SERANGGA HAMA PADA GUDANG
BERAS MENGGUNAKAN MODEL MOBILENETV3-LARGE BERBASIS
*WEBSITE***

SKRIPSI

**Diajukan untuk melengkapi tugas dan memenuhi syarat memperoleh ijazah
Sarjana Ilmu Komputer**

RIZKI CHRISMASYADI SIANIPAR

201401063



**PROGRAM STUDI S-1 ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS SUMATERA UTARA**

MEDAN

2025

PERSETUJUAN

Judul : IMPLEMENTASI ALGORITMA
CONVOLUTIONAL NEURAL NETWORK (CNN)
DALAM KLASIFIKASI JENIS SERANGGA HAMA
PADA GUDANG BERAS MENGGUNAKAN
MODEL MOBILENETV3-LARGE BERBASIS
WEBSITE

Kategori : SKRIPSI

Nama : RIZKI CHRISMASYADI SIANIPAR

Nomor Induk Mahasiswa : 201401063

Program Studi : SARJANA (S-1) ILMU KOMPUTER

Fakultas : ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS SUMATERA UTARA

Komisi Pembimbing :

Pembimbing 2

Amer Sharif S.Si., M.Kom.
NIP. 196910212021011001

Pembimbing 1

Dian Rachmawati S.Si., M.Kom.
NIP. 198307232009122004

Diketahui/Disetujui Oleh

Program Studi S-1 Ilmu Komputer

Dr. Amalia ST., M.T.
NIP. 197812212014042001

PERNYATAAN

**IMPLEMENTASI ALGORITMA CONVOLUTIONAL
NEURAL NETWORK (CNN) DALAM KLASIFIKASI JENIS
SERANGGA HAMA PADA GUDANG BERAS
MENGUNAKAN MODEL MOBILENETV3-LARGE
BERBASIS WEBSITE
SKRIPSI**

Saya mengakui bahwa skripsi ini adalah hasil karya saya sendiri, kecuali beberapa kutipan dan ringkasan yang masing-masing telah disebutkan sumbernya.

Medan, 9 Januari 2025



Rizki Chrismasyadi Sianipar

201401063

PENGHARGAAN

Segala puji syukur dipanjatkan kepada Tuhan Yang Maha Esa atas segala kasih dan karunia-Nya sehingga penulis dapat berada di tahap penyusunan skripsi ini sebagai syarat untuk mendapatkan gelar Sarjana Komputer di Program Studi S-1 Ilmu Komputer, Universitas Sumatera Utara.

Penyusunan skripsi ini tidak terlepas dari bantuan, dukungan, dan bimbingan dari banyak pihak. Oleh karena itu, penulis mengucapkan banyak terima kasih kepada:

1. Bapak Prof. Dr. Muryanto Amin S.Sos., M.Si. selaku Rektor Universitas Sumatera Utara.
2. Ibu Dr. Maya Silvi Lydia B.Sc., M.Sc. selaku Dekan Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara.
3. Ibu Dr. Amalia, S.T., M.T. selaku Ketua Program Studi S-1 Ilmu Komputer Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara.
4. Ibu Dian Rachmawati S.Si., M.Kom. sebagai Dosen Pembimbing I yang telah memberikan banyak masukan, motivasi, serta dukungan kepada penulis selama penyusunan skripsi ini.
5. Bapak Amer Sharif S.Si., M.Kom. selaku dosen Pembimbing II yang telah memberikan banyak bimbingan dan masukan yang berharga kepada penulis selama proses penyusunan skripsi ini.
6. Bapak Dr. Mohammad Andri Budiman S.T., M.Comp.Sc., M.E.M. selaku dosen Pembimbing Akademik penulis.
7. Seluruh Bapak dan Ibu Dosen Program Studi S-1 Ilmu Komputer, yang telah membimbing saya selama masa perkuliahan hingga akhir masa studi.
8. Bapak Marhusor Sianipar dan Ibu Malondang Hutabarat, selaku orang tua penulis yang telah memberikan dukungan mental dan juga materi sampai menyelesaikan skripsi.
9. Teman seperjuangan Kaum Mujahirudin yang telah memberikan banyak motivasi kepada penulis untuk menyelesaikan skripsi.
10. Teman-teman seperjuangan di program studi Ilmu Komputer yang telah memberikan dukungan dan bantuan yang berarti selama proses perkuliahan.

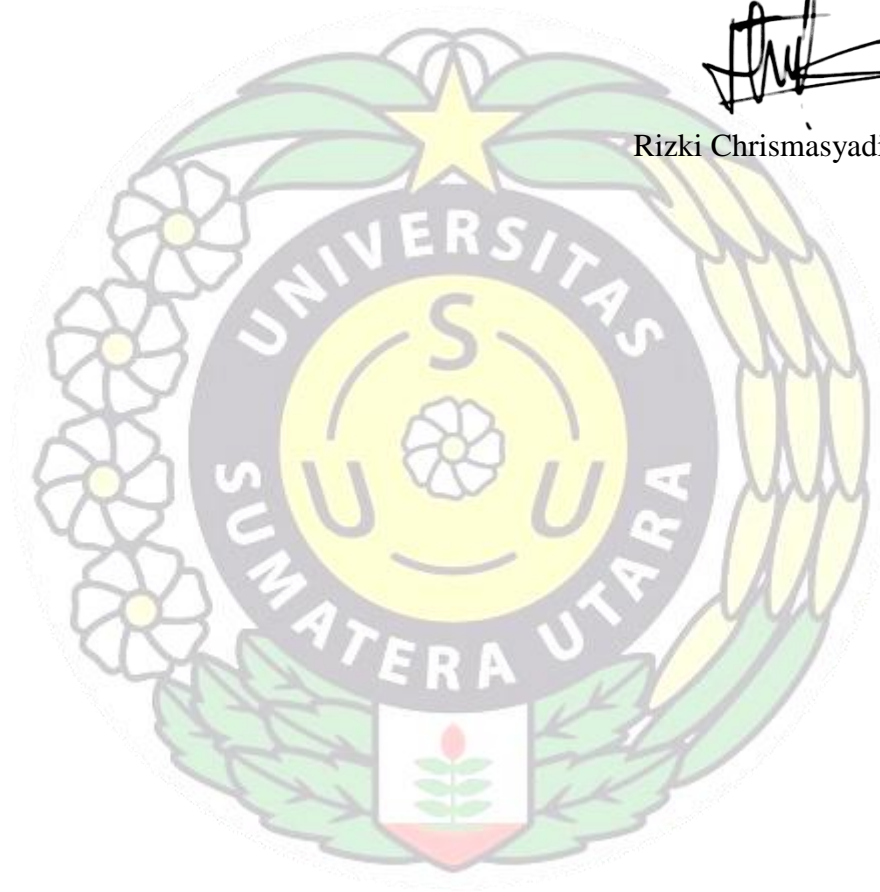
Dan seluruh pihak yang telah memberi dukungan serta doa baik yang tidak dapat penulis sebutkan satu per-satu. Semoga Tuhan Yang Maha Esa senantiasa melimpahkan keberkahan serta kebaikan atas semua dukungan yang telah diberikan kepada penulis dan hasil penelitian ini dapat memberi manfaat maupun inspirasi untuk kedepannya.

Medan, 9 Januari 2025

Penulis,



Rizki Chrismasyadi Sianipar



ABSTRAK

Serangga hama merupakan salah satu permasalahan utama pada gudang beras, karena dapat menyebabkan penurunan kualitas dan kuantitas beras selama penyimpanan. Proses identifikasi serangga hama yang selama ini dilakukan secara manual memiliki kelemahan dari segi akurasi, efisiensi waktu, dan tenaga. Penelitian ini bertujuan untuk mengembangkan sistem klasifikasi jenis serangga hama pada gudang beras berbasis *website* menggunakan algoritma Convolutional Neural Network (CNN) dengan arsitektur MobileNetV3-Large. Model dikembangkan dengan menggunakan 800 *dataset* per kelas, optimizer Adam dengan nilai learning rate 0.0001, dan dilatih selama 20 epoch. Hasil pengujian menunjukkan bahwa model mampu mencapai akurasi sebesar 95% tanpa mengalami overfitting. Evaluasi menggunakan confusion matrix juga menunjukkan performa konsisten dengan nilai precision, recall, dan F1-score yang seimbang untuk setiap kelas. Sistem yang dirancang menyediakan antarmuka *website* yang memudahkan pengguna untuk mengunggah gambar serangga hama yang akan diklasifikasikan. Proses pengolahan gambar mencakup *resize*, normalisasi, dan augmentasi, sebelum dimasukkan ke model untuk klasifikasi. Hasil klasifikasi ditampilkan kepada pengguna dalam bentuk teks dan visual, memberikan identifikasi yang cepat dan akurat. Pengujian dengan gambar dari kondisi dunia nyata menunjukkan sistem tetap bekerja dengan baik. Dengan sistem ini, pengguna dapat mengidentifikasi jenis serangga hama gudang secara efisien, mendukung pengambilan keputusan yang lebih cepat dalam pengendalian hama, dan membantu menjaga kualitas serta kuantitas beras selama penyimpanan.

Kata Kunci: *CNN, MobileNetV3-Large, klasifikasi serangga hama, gudang beras, aplikasi berbasis web*

ABSTRACT

IMPLEMENTATION OF CONVOLUTIONAL NEURAL NETWORK (CNN) ALGORITHM FOR CLASSIFYING PEST SPECIES IN RICE WAREHOUSES USING THE MOBILENETV3-LARGE MODEL IN A WEB-BASED APPLICATION

Pest insects pose a significant challenge in rice storage warehouses, leading to a decrease in both the quality and quantity of rice during storage. The traditional manual identification of pest insects is often inefficient, time-consuming, and lacks accuracy. This study aims to develop a web-based pest insect classification system for rice storage warehouses using the Convolutional Neural Network (CNN) algorithm with the MobileNetV3-Large architecture. The model was trained using 800 datasets per class, with the Adam optimizer (learning rate 0.0001) and 20 training epochs. The experimental results demonstrate that the model achieved an accuracy of 95% without overfitting. Evaluation through a confusion matrix revealed consistent performance with balanced precision, recall, and F1-score across all classes. The proposed system includes a user-friendly web interface that allows users to upload pest insect images for classification. Image preprocessing steps, such as resizing, normalization, and augmentation, ensure the data is suitable for the MobileNetV3-Large model. The classification results are displayed to users in both textual and visual formats, enabling quick and accurate identification. Tests conducted on real-world images confirmed the system's reliability and robustness. This solution enables efficient pest insect identification, facilitating faster decision-making for pest control, and supports maintaining the quality and quantity of rice during storage.

Keywords: *CNN, MobileNetV3-Large, pest classification, rice warehouse, web-based application*

DAFTAR ISI

PERSETUJUAN	ii
PERNYATAAN.....	iii
PENGHARGAAN.....	iv
ABSTRAK	vi
ABSTRACT.....	vii
DAFTAR ISI	viii
DAFTAR TABEL	x
DAFTAR GAMBAR.....	xi
BAB 1 PENDAHULUAN	1
1.1 Latar Belakang.....	1
1.2 Rumusan Masalah.....	3
1.3 Batasan Masalah	3
1.4 Tujuan Penelitian.....	4
1.5 Manfaat Penelitian	4
1.6 Metodologi Penelitian.....	4
1.7 Penelitian Relevan	5
1.8 Sistematika Penulisan	6
BAB 2 LANDASAN TEORI.....	8
2.1 Klasifikasi Gambar	8
2.2 <i>Convolutional Neural Network (CNN)</i>.....	8
2.3 MobileNetV3-Large.....	10
2.4 <i>Pre-processing Data</i>	13
2.5 Serangga Hama Pada Gudang Beras.....	14
BAB 3 ANALISIS DAN PERANCANGAN	16
3.1 Analisis	16
3.1.1 Analisis masalah.....	16
3.1.2 Ishikawa Diagram	16
3.1.3 Analisis kebutuhan	18
3.1.4 Use Case Diagram	19
3.1.5 Activity Diagram	20
3.1.6 Sequence Diagram	21
3.1.7 Flowchart Sistem.....	21
3.2 Perancangan Sistem	24

3.2.1.	<i>Arsitektur Umum Sistem</i>	24
3.2.2.	<i>Pengumpulan Dataset</i>	25
3.2.3.	<i>Pre-processing Dataset</i>	26
3.2.4.	<i>Perancangan Algoritma MobileNetV3-Large</i>	27
3.3	Evaluasi Model	30
3.4	Menyimpan Model	30
3.5	Integrasi Model Pada Website	31
BAB 4 IMPLEMENTASI DAN PENGUJIAN		32
4.1	<i>Pre-processing Dataset</i>	32
4.2	<i>Training dan Implementasi MobileNetV3-Large</i>	34
4.3	Evaluasi	35
4.4	Penyimpanan Model	39
4.5	Implementasi Sistem pada Website	40
4.6	Pengujian	41
BAB 5 PENUTUP		51
5.1	Kesimpulan	51
5.2	Saran	51
DAFTAR PUSTAKA		53

DAFTAR TABEL

Tabel 4. 1 Tabel Hasil Evaluasi	36
Tabel 4. 2 <i>Tabel Nilai TP, TN, FP, dan FN Pada Semua Kelas</i>	48
Tabel 4. 3 Tabel Pengujian.....	42



DAFTAR GAMBAR

Gambar 2. 1 Diagram Blok Arsitektur CNN.....	8
Gambar 2. 2 Operasi Konvolusi	9
Gambar 2. 3 Max-pooling.....	10
Gambar 2. 4 Fully Connected Layer	10
Gambar 2. 5 Arsitektur MobileNetV3-Large (Dyah et al., 2023).....	11
Gambar 2. 6 MobileNetV3 Block (Howard et al., 2019).....	13
Gambar 2. 7 Serangga Hama Pada Gudang Beras	15
Gambar 3. 1 <i>Ishikawa Diagram</i>	17
Gambar 3. 2 <i>Use Case Diagram</i>	20
Gambar 3. 6 <i>Activity Diagram</i>	20
Gambar 3. 7 <i>Sequence Diagram</i>	21
Gambar 3. 8 Flowchart Interaksi Pengguna Dengan Aplikasi Web	22
Gambar 3. 9 Flowchart Sistem Klasifikasi Menggunakan MobileNetV3-Large.....	23
Gambar 4. 1 Membagi <i>Dataset</i> Menjadi Train dan Test	32
Gambar 4. 2 Membagi <i>Dataset</i> Train Menjadi Train dan Validation	32
Gambar 4. 3 <i>Resize</i> Gambar	33
Gambar 4. 4 Augmentasi Gambar.....	33
Gambar 4. 5 Menampilkan Jumlah <i>Dataset</i>	33
Gambar 4. 6 Tampilan Jumlah <i>Dataset</i>	34
Gambar 4. 7 Fungsi Pretrained Model.....	34
Gambar 4. 8 Model Klasifikasi.....	35
Gambar 4. 9 Evaluasi	36
Gambar 4. 10 Pengujian Pada Data Testing.....	37
Gambar 4. 11 Perbandingan Kurva Akurasi dan Loss Pada Data Training dan Validation	37
Gambar 4. 12 Confusion Matrix	38
Gambar 4. 13 Pengambilan Gambar Acak Dari Pengujian.....	39
Gambar 4. 14 Tampilan Contoh 15 Gambar Pengujian dan Hasilnya	39
Gambar 4. 15 Penyimpanan Model.....	40
Gambar 4. 16 Halaman Utama Web	40
Gambar 4. 17 Hasil Prediksi Gambar.....	41

BAB 1

PENDAHULUAN

1.1 Latar Belakang

Serangga hama sering diidentikkan dengan serangga yang merugikan pada tanaman pertanian karena serangga hama tersebut memakan bagian-bagian tanaman yang sering mengakibatkan kerusakan, kematian, dan mengurangi produksi baik kuantitas, maupun kualitas. Serangga hama tersebut juga banyak dijumpai pada gudang beras yang membuat kualitas dan kuantitas beras berkurang.

Gudang beras merupakan sentral atau pusat dari rangkaian fungsi rantai pasokan pangan secara nasional untuk menyimpan stok beras dalam jumlah besar sebelum dikirim ke daerah-daerah atau lembaga terkait. Gudang ini berperan penting sebagai penyangga stok beras, baik dalam situasi stabil maupun saat krisis pangan, serta menjaga stabilitas harga pangan dan mendukung ketahanan pangan nasional. Di Indonesia, gudang beras umumnya dikelola oleh lembaga pemerintah seperti Bulog maupun oleh pihak swasta dalam rantai distribusi pangan. Kualitas beras harus diawetkan secara optimal dalam proses penyimpanan untuk memenuhi kebutuhan masyarakat secara berkelanjutan. Meski demikian, salah satu tantangan yang berkembang adalah serangan hama serangga yang dapat merusak kualitas dan kuantitas beras selama proses penyimpanan (Putri & Ludji, 2023). Kondisi lingkungan kurang terkontrol juga menyebabkan perkembangan lebih dahsyat oleh hama. Hama serangga menyebabkan kerusakan pada beras, baik secara langsung melalui konsumsi butiran, maupun secara tidak langsung karena kotoran, telur, atau bangkai yang dapat menurunkan kualitas sehingga produk menjadi tidak layak dikonsumsi. Kerusakan ini dapat mengurangi berat produk hingga puluhan persen. Ada berbagai jenis serangga hama yang terdapat pada gudang beras, dan beberapa jenis serangga hama tersebut memiliki bentuk yang mirip sehingga sulit untuk dibedakan, contohnya seperti *Sitophilus oryzae*, *Rhyzopertha dominica*, *Oryzaephilus surinamensis*, *Tribolium castaneum*, *Cryptolestes pusillus*, *Cadra cautella*, *Liposcelis spp.* dan *Alphitobius laevigatus* (Putri & Ludji, 2023). Setiap jenis hama juga

memiliki cara penanggulangan yang berbeda-beda, agar hama tersebut dapat secara efektif teratasi. Saat ini, identifikasi serangga hama dalam gudang beras sebagian besar masih dilakukan dengan inspeksi secara manual oleh petugas, yang tentu saja sangat memakan waktu, tenaga, serta keahlian khusus. Oleh karena itu, perlunya sebuah kecerdasan buatan untuk mengklasifikasi hama dengan tingkat akurasi yang tinggi.

Artificial intelligence (AI) adalah sistem kompleks yang meniru kemampuan manusia dalam menjalankan berbagai tugas dan dapat meningkatkan kecerdasannya berdasarkan informasi yang diperoleh. (Russell & Norvig, 2010). Salah satu cabang pada *artificial intelligence* adalah *machine learning*. *Machine learning* merupakan fokus pada data dan algoritma supaya dapat berpikir dan belajar seperti manusia. Dan pada *machine learning* terdapat salah satu cabang yaitu *deep learning* yang menggunakan jaringan saraf tiruan dengan banyak lapisan (*deep*) untuk memodelkan dan mempelajari representasi data yang kompleks melalui komputasi matematis.

Salah satu algoritma pada *deep learning* adalah CNN. *Convolutional Neural Networks* atau biasa dikenal dengan CNN umumnya berbentuk algoritma dua dimensi, mengikuti konsep pengembangan *multi-layer perceptrons* (MLPs) yang dirancang untuk mengolah data data seperti gambar dan suara (Aryawan et al., 2023). Algoritma CNN memiliki lapisan jaringan yang mendalam dan sering diterapkan pada data berupa citra, sehingga termasuk salah satu jenis algoritma *deep neural network*.

MobileNet adalah salah satu model pada CNN yang dikembangkan oleh Google. Keunggulan MobileNet terletak pada efisiensinya dalam hal penggunaan sumber daya komputasi (Setiadi & Alam, 2024). Model MobileNet yang digunakan pada penelitian ini adalah MobileNetV3. MobileNetV3 merupakan arsitektur CNN yang termasuk efisien dan hasil peningkatan dari versi sebelumnya, yakni MobileNetV1 dan MobileNetV2. Peningkatan performa dari kedua metode tersebut terletak pada pengurangan kehilangan informasi dari aktivasi linear dan efisiensi penggunaan memori. Arsitektur ini menawarkan keunggulan berupa akurasi tinggi dengan latensi rendah.

(Chu et al., 2020). Model MobileNetV3 yang digunakan pada penelitian ini spesifik pada MobileNetV3-Large yang menawarkan lebih banyak kapasitas dibandingkan MobileNetV3-Small, sehingga lebih baik dalam menangkap perbedaan

halus antara kelas yang mirip pada serangga hama.

Pada sistem yang ingin dirancang, pengguna akan mengakses sistem melalui antarmuka *website*. *Website* ini menyediakan fitur untuk mengunggah gambar serangga hama yang ingin diidentifikasi. Gambar yang diunggah tidak harus diambil secara *real-time*, pengguna dapat mengunggah gambar yang sudah ada atau diambil sebelumnya. Akan tetapi, gambar yang diambil harus diperdekat sampai serangga pada gambar yang diambil tersebut terlihat jelas. Setelah pengguna mengunggah gambar serangga, gambar tersebut akan diproses, seperti *resize*, normalisasi, dan augmentasi untuk memastikan gambar siap digunakan sebagai input model MobileNetV3-Large. Gambar yang telah diproses akan dimasukkan ke dalam model MobileNetV3-Large yang telah dilatih khusus untuk mengenali beberapa jenis serangga hama gudang beras. Model akan menganalisis gambar dan menentukan jenis serangga yang ada dalam gambar berdasarkan fitur-fitur yang diekstraksi. Setelah klasifikasi selesai, hasilnya akan dikembalikan ke pengguna dalam bentuk teks dan gambar pada halaman *website*. Sistem ini memungkinkan pengguna untuk mengidentifikasi jenis serangga hama dengan cepat dan membantu dalam pengambilan keputusan terkait pengendalian hama di gudang atau fasilitas penyimpanan, meningkatkan efisiensi dan mengurangi kerugian yang disebabkan oleh hama.

1.2 Rumusan Masalah

Mengimplementasikan algoritma *deep learning* yaitu CNN menggunakan model MobileNetV3-Large dalam mengklasifikasikan 8 jenis serangga hama pada gudang beras.

1.3 Batasan Masalah

Batasan masalah dalam penelitian ini adalah sebagai berikut:

1. Terdapat 8 jenis serangga yang akan diklasifikasikan, yaitu *Sitophilus oryzae*, *Rhyzopertha dominica*, *Oryzaephilus surinamensis*, *Tribolium castaneum*, *Cryptolestes pusillus*, *Cadra cautella*, *Liposcelis spp.* dan *Alphitobius laevigatus*. Dan sistem hanya dapat digunakan untuk mengklasifikasikan 8 kelas hama serangga tersebut.
2. Program yang dirancang berbentuk *website* dan akan menggunakan bahasa pemrograman *Python* dan IDE berupa *Visual Studio Code*.

3. *Dataset* yang digunakan terdiri dari 800 gambar tiap kelas yang diperoleh dari *website* Kaggle, dan 41 gambar untuk pengujian pada *website* yang diperoleh dari Google.
4. Gambar yang diunggah oleh pengguna harus diperdekat (*zoom-in*) sampai bentuk serangga terlihat dengan jelas.
5. Penelitian ini akan dibatasi pada penggunaan model MobileNetV3-Large sebagai arsitektur CNN yang diterapkan, dan tidak membahas secara mendalam perbandingan dengan model lain.

1.4 Tujuan Penelitian

Adapun tujuan dalam penelitian ini adalah sebagai berikut:

1. Mengimplementasikan Algoritma CNN, khususnya MobileNetV3-Large, untuk mengembangkan sistem klasifikasi berbasis *website* yang dapat mengidentifikasi jenis serangga hama pada gudang beras secara efisien dan akurat.
2. Membangun sebuah sistem berbasis *website* yang memungkinkan pengguna mengunggah gambar serangga hama dan menerima hasil klasifikasi secara langsung melalui platform *online*, sehingga mempermudah identifikasi hama dan melakukan penganggulangan yang efektif.
3. Menguji dan mengevaluasi performa model MobileNetV3-Large dalam mengklasifikasikan jenis serangga hama gudang beras dan menentukan efektivitas MobileNetV3-Large dalam menangani klasifikasi serangga yang memiliki kemiripan karakteristik *visual*.

1.5 Manfaat Penelitian

Adapun manfaat yang diharapkan dari penelitian ini yaitu membantu pengelola gudang dalam mengenali dan mengidentifikasi serangga hama dengan cepat dan akurat hanya melalui gambar yang diunggah ke *website*, sehingga membantu dalam pengelolaan yang tepat dan pencegahan kerugian akibat hama dapat dilakukan lebih dini.

1.6 Metodologi Penelitian

Beberapa metode yang diterapkan dalam penelitian ini adalah sebagai berikut:

1. Studi Pustaka

Pada tahap ini penelitian dimulai dengan mencari referensi dari berbagai sumber terpercaya dan melakukan peninjauan pustaka melalui buku-buku, jurnal, *e-book*, artikel ilmiah, makalah ataupun situs internet yang berhubungan dengan Model MobileNetV3-Large pada CNN.

2. Analisis dan Perancangan Sistem

Berdasarkan ruang lingkup penelitian, penulis melakukan analisa terhadap apa saja yang akan dibutuhkan dalam penelitian untuk segera dirancang dalam sebuah diagram alir (*flowchart*), UML, dan *Ishikawa Diagram*.

3. Implementasi Sistem

Pada tahap ini, membuat sebuah sistem dengan menggunakan bahasa pemrograman *Python* sebagai program dan HTML sebagai antarmuka sesuai dengan diagram alir yang telah dirancang.

4. Pengujian Sistem

Pada tahap ini, sistem yang telah dirancang dilakukan uji coba untuk melakukan klasifikasi jenis serangga hama pada gudang beras menggunakan Model MobileNetV3-Large pada CNN.

5. Dokumentasi Sistem

Pada tahap ini, penelitian yang telah dilakukan, didokumentasikan mulai dari tahap analisa sampai kepada pengujian dalam bentuk skripsi.

1.7 Penelitian Relevan

Beberapa penelitian terdahulu yang relevan dengan penelitian yang dilakukan dalam penelitian ini, antara lain:

1. Penelitian dengan judul *Strawberry Plant Diseases Classification Using CNN Based on MobileNetV3-Large and EfficientNet-B0 Architecture* (Dyah et al., 2023), dilakukan klasifikasi penyakit tanaman stroberi menggunakan CNN berbasis arsitektur MobileNetV3-Large dan EfficientNet-B0. Dari hasil pengolahan *dataset* gambar, diperoleh kesimpulan tingkat akurasi tertinggi pada 30 *epoch* terdapat pada *learning rate* 0,01 (91,43%), pada 50 *epoch* terdapat pada *learning rate* 0,03 dan 0,1 (91,43%), dan pada 70 *epoch* terdapat pada *learning rate* 0,001 (92,14%). Secara keseluruhan, persentase model evaluasi

untuk presisi, *recall*, dan *F1-Score* masing-masing mencapai 92,81%, 92,14%, dan 92,25%.

2. Penelitian dengan judul *Intelligent garbage classification system based on improve MobileNetV3-Large* (Zhang et al., 2022), sebuah sistem klasifikasi sampah cerdas dengan MobileNetV3-Large. Sistem ini menggabungkan perangkat keras (tempat sampah berbasis *Raspberry Pi*) dan perangkat lunak (aplikasi WeChat) untuk mengklasifikasikan sampah. Model MobileNetV3-Large digunakan untuk pengenalan gambar, mengategorikan sampah ke dalam empat kategori utama (dapat didaur ulang, limbah makanan, limbah berbahaya, dan lainnya) serta 158 *subkategori*. Model pengenalan gambar menggunakan MobileNetV3-Large yang memanfaatkan konvolusi terpisah dalam, struktur residual terbalik, struktur perhatian ringan, dan fungsi aktivasi *hard_swish* untuk mengenali gambar sampah dan kurasi pengenalan gambar mencapai 81%.
3. Penelitian dengan judul Peningkatan Performa MobileNetV3 dengan *Squeeze-and-Excitation* (Studi Kasus Klasifikasi Kesegaran Ikan Berdasarkan Mata Ikan) (Galih & Fikri, 2023), sebuah sistem klasifikasi kesegaran ikan berdasarkan mata ikan, yang membandingkan kinerja MobileNetV1, MobileNetV2, MobileNetV3-Large dan MobileNetV3-Small dengan jumlah *epoch* dan *optimizer* yang beragam. Berdasarkan hasil yang didapatkan bahwa model yang memiliki hasil yang terbaik untuk penelitian ini merupakan model dengan arsitektur MobileNetV3-Small dengan *hyperparameter epoch* 100, *learning rate* 0.00001, *batch size train* 10 dan *validation* 10, *optimizer* ADAM, dengan jumlah parameter 1,378,604, menghasilkan akurasi dalam pengujian sebesar 68%, *precision* 69%, *recall* 67%, dan *F1-Score* 68%, dan nilai *loss* pada pengujian 876 data uji sebesar 0,91.

1.8 Sistematika Penulisan

Sistematika penulisan skripsi yang digunakan dalam penelitian ini adalah sebagai berikut:

BAB 1 PENDAHULUAN

Bab ini mencakup penjelasan mengenai latar belakang penelitian, rumusan masalah, batasan masalah, tujuan penelitian, manfaat

penelitian, metodologi penelitian, penelitian relevan, dan sistematika penulisan skripsi.

BAB 2 LANDASAN TEORI

Bab ini menjelaskan beberapa teori yang berkaitan dengan penelitian, seperti klasifikasi gambar, *Convolutional Neural Network* (CNN), dan MobileNetV3-Large.

BAB 3 ANALISIS DAN PERANCANGAN

Bab ini menjelaskan mengenai analisis pada algoritma yang digunakan dan perancangan sistem yang menggunakan model MobileNetV3-Large. Selanjutnya membuat diagram alir (*flowchart*) terkait sistem tersebut .

BAB 4 IMPLEMENTASI DAN PENGUJIAN

Bab ini berisi penjelasan mengenai implementasi algoritma pada sistem dalam melakukan klasifikasi hama serangga pada gudang beras beserta hasil pengujian sistem yang dibangun dan analisis kinerja sistem tersebut.

BAB 5 KESIMPULAN DAN SARAN

Bab ini berisi kesimpulan yang dapat diperoleh berdasarkan pemaparan pada setiap bab serta saran yang diberikan peneliti sebagai masukan untuk penelitian selanjutnya.

BAB 2

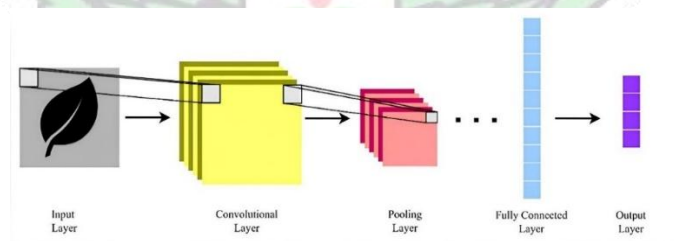
LANDASAN TEORI

2.1 Klasifikasi Gambar

Klasifikasi adalah proses pengelompokan atau penyusunan data ke dalam kategori atau kelas yang berbeda berdasarkan karakteristik tertentu. Dalam konteks *machine learning* dan kecerdasan buatan (AI), klasifikasi merupakan metode di mana suatu model dilatih untuk mengidentifikasi kategori dari data input, salah satunya adalah klasifikasi gambar. Klasifikasi gambar adalah pengenalan dan pengelompokan gambar ke dalam kategori atau kelas yang tepat yang memiliki banyak aplikasi praktis seperti pengenalan wajah, deteksi objek, pengenalan tulisan tangan, dan masih banyak lagi (Setiadi & Alam, 2024).

2.2 *Convolutional Neural Network* (CNN)

Convolutional Neural Network adalah jenis arsitektur jaringan saraf tiruan (*neural network*) yang dirancang khusus untuk bekerja dengan data yang memiliki struktur *grid*, seperti gambar. CNN adalah jenis jaringan saraf dalam yang sangat berguna untuk mengenali visual, seperti objek, wajah, dan rambu lalu lintas. Jaringan ini paling umum digunakan dalam pembelajaran mendalam untuk pengenalan gambar dan video, klasifikasi gambar, analisis pencitraan medis, dan pemrosesan bahasa (Bing & Asad, 2023).



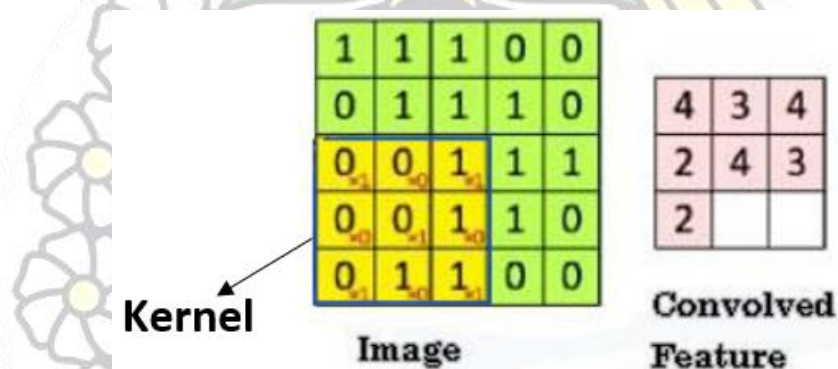
Gambar 2.1 Diagram Blok Arsitektur CNN

CNN memiliki arsitektur seperti pada gambar 2.1, yaitu :

1. *Input layer*, lapisan ini berguna untuk menampung data input dan meneruskannya ke lapisan-lapisan berikutnya. Contohnya berupa gambar yang direpresentasikan sebagai matriks tiga dimensi dengan dimensi tinggi (*height*),

lebar (*width*), dan jumlah *channel* (*depth*), misalnya *channel* bisa berupa warna merah, hijau, dan biru (RGB). Contohnya, gambar berukuran 224x224 piksel dengan tiga *channel* warna RGB akan memiliki dimensi input [224, 224, 3].

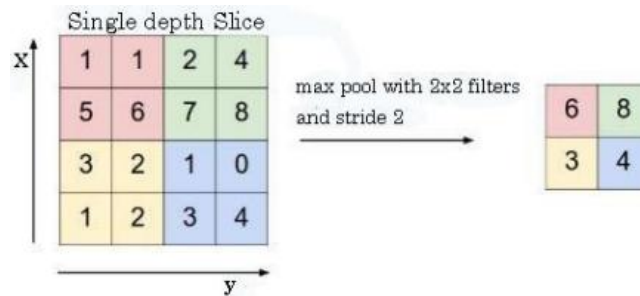
2. *Convolution Layer*, berfungsi untuk mengekstrak fitur dari input data, seperti gambar. Lapisan ini menerapkan operasi konvolusi pada gambar input untuk mendeteksi pola-pola penting, seperti tepi, tekstur, atau objek dalam gambar. Proses konvolusi menggunakan *kernel* dan *stride*, proses konvolusi ini adalah proses kombinasi antara dua buah matriks yang berbeda untuk menghasilkan suatu nilai matriks yang baru. *Kernel* adalah matriks kecil yang digunakan untuk melakukan operasi konvolusi. Setiap filter bertugas mendeteksi fitur tertentu, seperti tepi horizontal, vertikal, atau pola lain dan *stride* berfungsi untuk menentukan seberapa jauh filter melangkah ketika diterapkan pada gambar.



Gambar 2.2 Operasi Konvolusi

Gambar 2.2 merupakan contoh operasi konvolusi, dimana matriks gambar akan dikalikan dengan *kernel* 3x3 dan hasilnya akan dijumlahkan, operasi ini akan terus diulangi setiap kali *kernel* bebrgerak sesuai dengan jumlah *stride* yang digunakan, sampai keseluruhan gambar diproses. Pada akhirnya akan menghasilkan *feature map*.

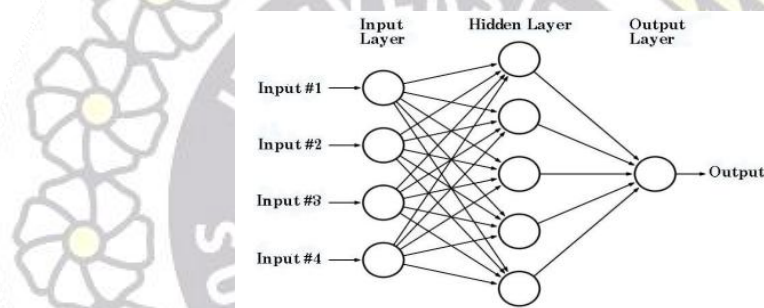
3. *Pooling Layer*, bekerja di setiap tumpukan *feature map* (hasil dari *convolution layer*) dan melakukan pengurangan pada ukurannya. Bentuk lapisan *pooling* umumnya dengan menggunakan filter dengan ukuran 2x2 yang diaplikasikan dengan langkah sebanyak dua dan beroperasi pada setiap irisan dari inputnya.



Gambar 2.3 *Max-pooling*

Gambar 2.3 adalah contoh *max-pooling*, dimana kotak yang berwarna merah, hijau, kuning dan biru pada sisi kiri merupakan kelompok kotak yang akan dipilih nilai maksimumnya. Sehingga hasil dari proses tersebut dapat dilihat pada kumpulan kotak disebelah kanannya.

4. *Fully Connected Layer*



Gambar 2.4 *Fully Connected Layer*

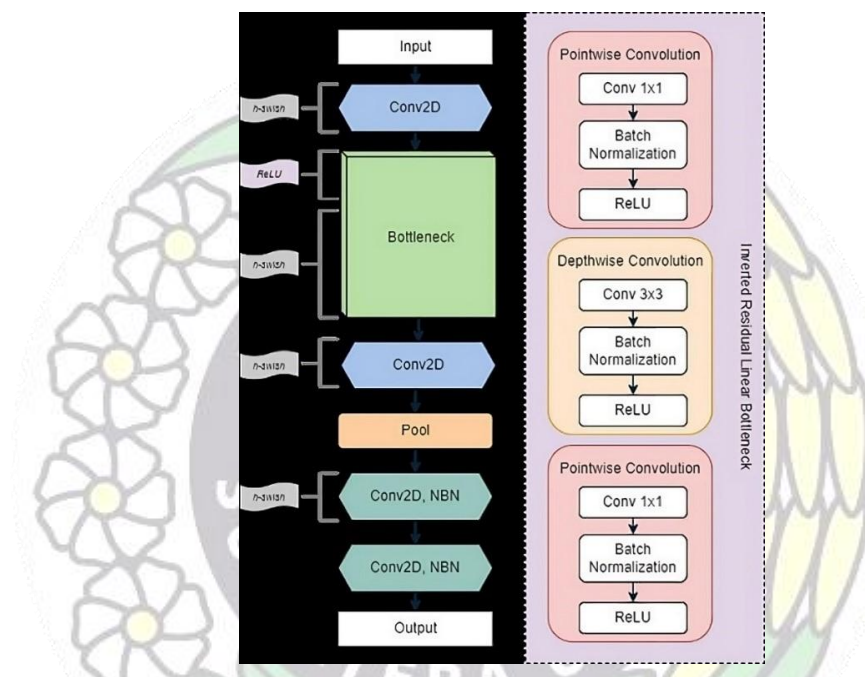
Gambar 2.4 adalah *Fully Connected Layer*, lapisan dimana semua neuron aktivitas dari lapisan sebelumnya terhubung semua dengan neuron di lapisan selanjutnya seperti halnya jaringan syaraf tiruan.

5. *Output Layer*, model menghasilkan prediksi berdasarkan informasi yang telah diproses oleh lapisan-lapisan sebelumnya. Setiap neuron dalam *output layer* mewakili sebuah kelas atau label yang ingin diprediksi oleh model, seperti jenis objek atau kategori tertentu.

2.3 MobileNetV3-Large

MobileNetV3-Large adalah salah satu jenis dari MobileNetV3, dimana MobileNetV3 adalah salah satu arsitektur CNN yang dirancang khusus untuk aplikasi *mobile* dan *embedded* dengan tujuan utama mencapai efisiensi tinggi dalam hal komputasi dan daya tanpa mengorbankan akurasi. MobileNetV3

merupakan arsitektur CNN yang termasuk efisien dan hasil peningkatan dari versi sebelum yakni MobileNetV1 dan MobileNetV2. Peningkatan performa dari kedua metode tersebut ada pada pemangkasan *information loss* dari sebuah aktivasi linear dan efisiensi pemakaian memori. Keunggulan yang ditawarkan oleh arsitektur ini merupakan akurasi yang tinggi dengan latensi yang rendah. MobileNetV3 terbagi menjadi 2 jenis yaitu MobileNetV3-Large dan MobileNetV3-Small yang dimana letak perbedaannya yaitu pada jumlah dan jenis *layer* yang dipakai (Howard et al., 2019).



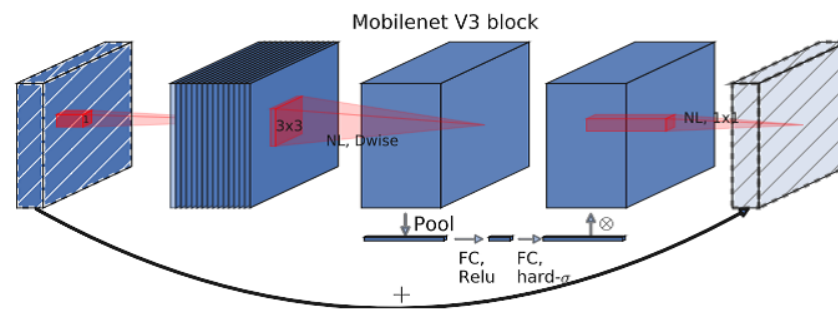
Gambar 2.5 Arsitektur MobileNetV3-Large (Dyah et al., 2023)

Pada Gambar 2.5, seperti halnya MobileNetV2, lapisan MobileNetV3-Large juga memiliki blok *bottleneck*. *Bottleneck* adalah lapisan khusus dalam sebuah jaringan yang dirancang untuk mengkompresi jumlah *channel* (dimensi fitur) pada data dan kemudian mengeksplainsinya kembali setelah beberapa operasi dilakukan. Setiap blok *bottleneck* terdiri dari 3 lapisan yang melakukan operasi konvolusi 1×1 , 3×3 , dan 1×1 . Konvolusi 1×1 akan mengurangi, kemudian mengembalikan dimensi saluran dari gambar input, kemudian konvolusi *depthwise* 3×3 akan menghasilkan ukuran gambar yang lebih kecil tanpa efek pada dimensi saluran (Howard et al., 2019). Konsep *bottleneck* ini dikombinasikan dengan *Squeeze* dan *Excite* yang merupakan sebuah untuk meningkatkan representasi fitur pada jaringan konvolusi atau memperkuat fitur-fitur yang relevan dan melemahkan fitur-

fitur yang kurang penting, sehingga membantu jaringan fokus pada informasi yang lebih signifikan selama proses klasifikasi (Chunguang et al., 2023). *Squeeze* adalah tahapan dimana semua *channel* pada data fitur akan dirangkum menjadi satu nilai statistik representatif (biasanya rata-rata), yang menghasilkan representasi *channel-wise global*. Proses ini menggunakan *global average pooling* untuk merangkum informasi di seluruh *spatial feature map*. Misalnya, jika sebuah data fitur memiliki 256 *channel*, maka *output* dari *squeeze* akan menghasilkan vektor yang terdiri dari 256 elemen, masing-masing mewakili *channel* tersebut.

Excite adalah tahapan yang terdiri dari dua lapisan *fully connected* yang akan menghitung skor penting untuk setiap *channel* berdasarkan informasi yang dirangkum dari tahap *squeeze*. Skor ini nantinya digunakan untuk memberikan bobot lebih pada *channel* penting dan bobot lebih kecil pada *channel* yang kurang penting. Setelah *excitation*, skor yang diperoleh akan digunakan untuk menghitung kembali skala setiap *channel* pada data fitur asli. Dengan demikian, *channel-channel* yang lebih penting (berdasarkan skor) akan dikuatkan, sementara yang kurang penting akan dilemahkan. Oleh karena itu, *Squeeze* dan *Excite* memungkinkan jaringan untuk secara dinamis memberikan bobot pada fitur-fitur penting dan menekan fitur-fitur yang tidak terlalu penting.

Pada MobileNetV3 termasuk MobileNetV3-Large, lapisan ditingkatkan dengan memodifikasi penggunaan fungsi aktivasi nonlinier ReLU menjadi nonlinier *hard-swish* untuk meningkatkan akurasi jaringan saraf (Chunguang et al., 2023). *Hard-Swish* adalah varian dari fungsi aktivasi *Swish* yang digunakan untuk meningkatkan efisiensi komputasi. *Hard-Swish* merupakan fungsi aktivasi non-linear yang dirancang khusus untuk meningkatkan kinerja pada jaringan yang dioptimalkan untuk perangkat seluler atau *edge devices*, seperti pada arsitektur MobileNetV3. Fungsi ini merupakan fungsi *non-linear* dan mulus, serta memiliki sifat memperkuat fitur penting yang bernilai positif, sementara fitur yang kurang penting (biasanya nilai negatif) didorong ke nol atau mendekati nol.



Gambar 2.6 MobileNetV3 Block (Howard et al., 2019)

Tahapan MobileNetV3 seperti pada Gambar 2.6, pertama melibatkan proses input gambar menggunakan konvolusi 2D dengan *kernel* berukuran 3x3, *stride* 2, dan fungsi aktivasi *hard-swish* pada gambar berukuran 224x224 dengan 3 *channel* RGB. Setelah itu, dilakukan proses konvolusi dengan *bottleneck* 3x3, dimulai dengan konvolusi 1x1 pada input untuk mengurangi kedalaman (*depth*), sehingga konvolusi 3x3 menjadi lebih efisien dalam memproses fitur karena kedalamannya lebih rendah. Selanjutnya, konvolusi 1x1 diterapkan pada *output* untuk mengembalikan kedalaman seperti semula, menggunakan fungsi aktivasi ReLU. Proses *bottleneck* ini bertujuan untuk meningkatkan efisiensi model. Kemudian, *layer bottleneck* yang sama diulang 15 kali pada MobileNetV3-Large, dengan perbedaan pada parameter *stride*, fungsi aktivasi, dan ukuran *kernel* di setiap *layer*. Setelah itu, dilakukan *pooling* 7x7 untuk mengurangi dimensi *feature maps*, yang bertujuan mengurangi jumlah parameter yang dipelajari serta beban komputasi, khususnya pada perangkat *mobile*. Proses dilanjutkan dengan konvolusi 2D *kernel* 1x1 tanpa *batch normalization* dan fungsi aktivasi *hard-swish*. Pada *layer* terakhir, konvolusi serupa diterapkan, namun digunakan sebagai *output* dari citra (Howard et al., 2019).

2.4 Pre-processing Data

Pre-processing adalah serangkaian langkah atau teknik yang digunakan untuk mempersiapkan *dataset* sebelum dimasukkan ke dalam model *machine learning*. Tujuannya adalah untuk memastikan gambar dalam *dataset* memiliki format yang konsisten, mengurangi *noise*, dan mempermudah model dalam mengekstraksi fitur yang relevan untuk klasifikasi atau tugas lain. Tahapan *pre-processing* data yang akan dilakukan :

1. *Resize*

Resize adalah proses mengubah ukuran gambar menjadi dimensi tertentu yang dibutuhkan oleh model. Model *deep learning* seperti CNN biasanya menerima input dengan ukuran tetap, sehingga semua gambar dalam *dataset* harus diubah ukurannya agar konsisten. Jika ukuran gambar berbeda-beda, model tidak bisa memprosesnya dengan baik.

2. Normalisasi

Normalisasi adalah proses mengubah nilai piksel gambar menjadi rentang yang lebih kecil, biasanya antara 0 hingga 1, atau -1 hingga 1. Nilai piksel gambar biasanya dalam rentang 0 hingga 255 dan normalisasi dilakukan agar model lebih mudah melakukan perhitungan dan mengurangi skala besar dari piksel. Dengan menempatkan nilai piksel dalam rentang kecil, model dapat belajar lebih cepat dan stabil.

3. Augmentasi

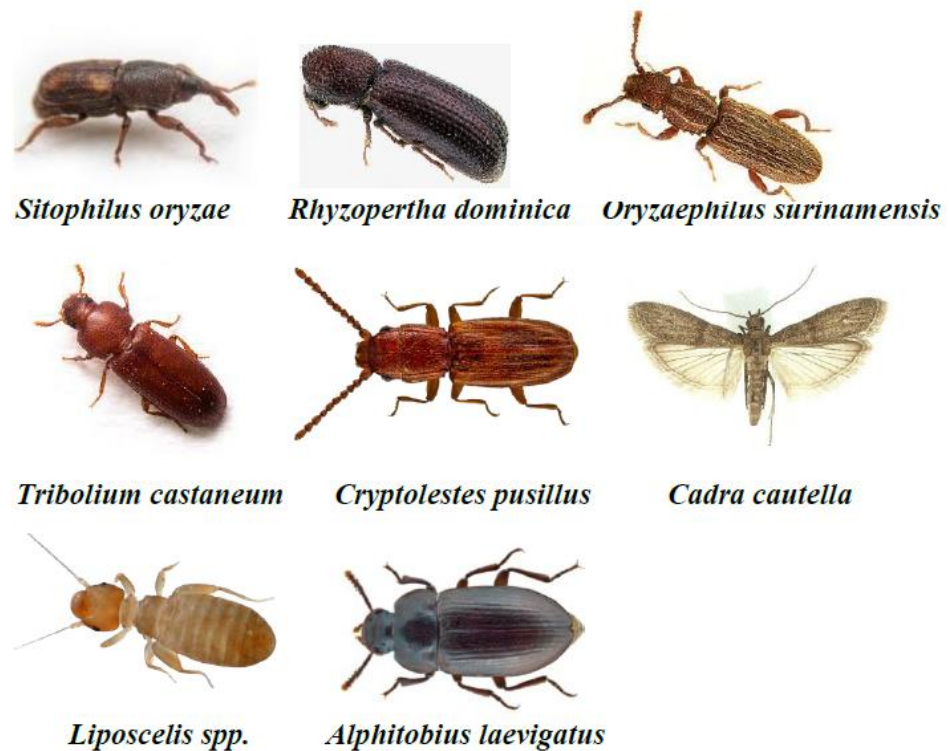
Augmentasi gambar adalah teknik untuk menambah variasi pada *dataset* dengan cara memodifikasi gambar-gambar yang ada. Ini dilakukan untuk mencegah *overfitting* dan membuat model lebih tangguh terhadap variasi nyata di dunia luar. Jika *dataset* terbatas, augmentasi memungkinkan kita membuat variasi dari gambar asli sehingga jumlah data secara efektif bertambah. Metode Augmentasi yang akan digunakan yaitu :

- 1) *Rotation*, yaitu memutar *dataset* gambar beberapa derajat untuk menambah variasi sudut.
- 2) *Height*, yaitu menambah atau mengurangi tinggi atau panjang gambar.
- 3) *Zooming*, yaitu memperbesar atau mengecilkan gambar untuk kebutuhan simulasi variasi jarak antara objek dan kamera.
- 4) *Width*, yaitu menambah atau mengurangi lebar gambar.

2.5 Serangga Hama Pada Gudang Beras

Serangga hama pada gudang beras adalah serangga yang merugikan yang sering mengakibatkan pengurangan produksi baik kuantitas, maupun kualitas (Fajri et al., 2021). Ada berbagai jenis serangga hama yang terdapat pada gudang beras, dan beberapa jenis serangga hama tersebut memiliki bentuk yang mirip sehingga sulit untuk dibedakan, contohnya seperti *Sitophilus oryzae*, *Rhyzopertha dominica*,

Oryzaephilus surinamensis, *Tribolium castaneum*, *Cryptolestes pusillus*, *Cadra cautella*, *Liposcelis spp.* dan *Alphitobius laevigatus* (Putri & Ludji, 2023). Setiap jenis hama juga memiliki cara penanggulangan yang berbeda-beda, agar hama tersebut dapat secara efektif teratasi. Oleh karena itu, perlunya sebuah kecerdasan buatan untuk mengklasifikasi hama dengan tingkat akurasi yang tinggi. Jenis serangga hama gudang beras yang akan diklasifikasi yaitu pada Gambar 2.7.



Gambar 2.7 Serangga Hama Pada Gudang Beras

BAB 3

ANALISIS DAN PERANCANGAN

3.1 Analisis

Tujuan dari analisis adalah untuk menentukan persyaratan komponen selama fase desain sistem dari proses penelitian. Sebagai dasar untuk menciptakan sistem, analisis masalah akan dilakukan pada tahap analisis sistem untuk menentukan sebab dan akibat dari suatu masalah. Proses analisis merupakan langkah penting sebelum melakukan perancangan dan pengembangan sistem tertentu. Hal ini dilakukan agar sistem yang dikembangkan dapat memenuhi kebutuhan yang ada dan menjadi lebih terstruktur dalam mencapai tujuan akhir.

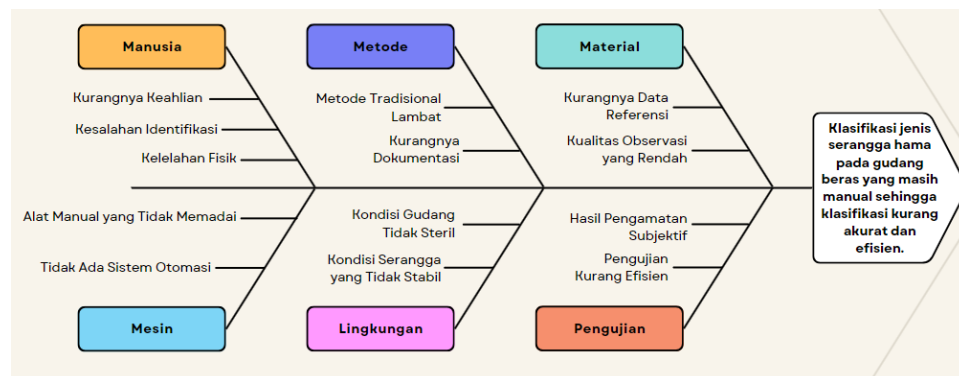
3.1.1 Analisis masalah

Serangan hama pada gudang beras menjadi salah satu tantangan serius dalam menjaga kualitas dan kuantitas hasil panen. Identifikasi jenis serangga hama secara manual sering kali memerlukan waktu yang lama dan memerlukan keahlian khusus, sehingga rentan terhadap kesalahan manusia. Variasi bentuk, ukuran, dan warna serangga menambah kompleksitas proses identifikasi ini. Selain itu, minimnya akses ke teknologi yang mudah digunakan dan efisien menghambat pengelolaan hama secara optimal, terutama di daerah dengan sumber daya terbatas. Oleh karena itu, diperlukan solusi berbasis teknologi yang mampu mengidentifikasi jenis serangga hama dengan cepat, akurat, dan dapat diakses dengan mudah oleh pengguna tanpa keahlian teknis khusus, diharapkan sistem ini mampu menjadi solusi akan permasalahan tersebut.

3.1.2 Ishikawa Diagram

Diagram *Ishikawa* adalah alat yang digunakan untuk menganalisis dan mengidentifikasi penyebab utama suatu masalah atau efek yang terjadi dalam sebuah proses. Diagram ini sering digunakan dalam manajemen kualitas dan pemecahan masalah untuk memahami hubungan antara berbagai faktor yang dapat mempengaruhi hasil suatu proses atau sistem. Diagram *Ishikawa* memiliki bentuk seperti tulang ikan, di mana tulang utama mewakili masalah atau efek yang ingin dianalisis, dan tulang-tulang kecil yang mengarah ke tulang utama mewakili

berbagai kategori penyebab yang mungkin menyebabkan masalah tersebut. Setiap kategori penyebab dapat diperinci lebih lanjut untuk menemukan akar masalah.



Gambar 3. 1 Diagram *Ishikawa*

Berikut penjelasan diagram *Ishikawa* pada Gambar 3.1:

1. Manusia

Faktor-faktor terkait dengan manusia.

- Kurangnya keahlian operator dalam memahami karakteristik masing-masing serangga hama.
- Kesalahan identifikasi dalam membedakan jenis serangga hama, terutama pada spesies yang mirip.
- Kelelahan fisik karena proses identifikasi yang memakan waktu, menyebabkan penurunan akurasi.

2. Metode

Faktor-faktor terkait dengan metode.

- Metode tradisional lambat karena identifikasi dilakukan dengan pengamatan langsung tanpa panduan teknologi.
- Kurangnya dokumentasi atau catatan sistematis mengenai proses dan hasil klasifikasi yang dilakukan.

3. Material

Faktor-faktor terkait dengan material.

- Kurangnya data referensi atau panduan lengkap untuk setiap jenis serangga.
- Kualitas observasi yang rendah pada serangga karena ukurannya kecil atau bersembunyi di tempat sulit dijangkau.

4. Mesin

Faktor-faktor terkait dengan mesin.

- Alat manual yang tidak memadai seperti kaca pembesar atau mikroskop sederhana yang tidak memberikan detail memadai.
- Tidak ada sistem otomasi seperti perangkat atau software yang dapat mendukung identifikasi otomatis.

5. Lingkungan

Faktor-faktor terkait dengan lingkungan.

- Kondisi gudang tidak steril, seperti pencahayaan minim dan kebersihan gudang yang buruk menyulitkan pengamatan.
- Kondisi serangga yang tidak stabil, serangga mungkin bergerak atau terbang, sehingga sulit diidentifikasi secara manual.

6. Pengujian

Faktor-faktor terkait dengan pengujian.

- Hasil pengamatan subjektif karena tidak adanya metode pengujian objektif menyebabkan hasil identifikasi bergantung pada pengetahuan individu.
- Pengujian kurang efisien karena tidak ada alat bantu pengujian seperti perangkat digital yang dapat mempercepat dan meningkatkan akurasi.

3.1.3 Analisis kebutuhan

Analisis kebutuhan digunakan untuk mengidentifikasi dan mendefinisikan hal-hal yang dibutuhkan pengguna dalam merancang sistem. Terdapat 2 jenis kebutuhan yang dapat diidentifikasi dalam perancangan sebuah sistem yaitu kebutuhan fungsional dan kebutuhan non-fungsional.

1. Kebutuhan fungsional

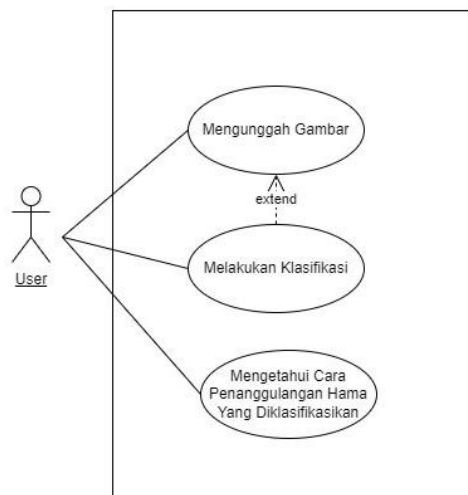
Kebutuhan fungsional adalah kumpulan fitur atau fungsi yang harus ada dalam suatu sistem atau produk untuk memenuhi kebutuhan pengguna atau pemangku kepentingan. Ini mencakup spesifikasi fungsionalitas yang harus ada pada sistem untuk mencapai tujuan utama. Penelitian ini memiliki kebutuhan fungsional utama, yaitu:

- a. Sistem harus mampu melakukan preprocessing pada gambar, seperti normalisasi dan pengubahan ukuran gambar menjadi 224x224 piksel.

- b. Sistem harus mampu menangani pembagian *dataset* menjadi data latih, validasi, dan uji.
 - c. Sistem harus mampu menerapkan model MobileNetV3-Large yang sudah dilatih sebelumnya.
 - d. Pengguna dapat mengunggah gambar melalui antarmuka *website*.
 - e. Sistem harus dapat menerima input berupa gambar serangga hama dalam format umum seperti JPEG atau JPG.
 - f. Sistem harus dapat mengklasifikasikan gambar serangga hama ke dalam salah satu kelas yang telah ditentukan.
2. Kebutuhan non-fungsional
- Kebutuhan non-fungsional adalah spesifikasi tambahan yang mendukung sistem, mencakup aspek kinerja maupun batasan yang dimiliki oleh sistem.. Penelitian ini memiliki beberapa kebutuhan non-fungsional, yaitu:
- a. Sistem harus mampu memberikan hasil klasifikasi dalam waktu kurang dari 5 detik setelah pengguna mengunggah gambar.
 - b. Sistem harus memiliki tingkat akurasi prediksi yang konsisten, minimal 90% pada data uji.
 - c. Sistem menggunakan manipulasi data menggunakan bahasa pemrograman *Python*.
 - d. Kode sumber harus ditulis dengan baik dan terstruktur agar mudah dipelihara dan diperbarui.

3.1.4 Use Case Diagram

Use case diagram adalah diagram yang menggambarkan hubungan aktor tertentu (seperti *user*) dengan sebuah sistem dalam melakukan tugas tertentu. Interaksi yang dilakukan oleh *user* pada sistem ini dapat dilihat pada gambar 3.2.

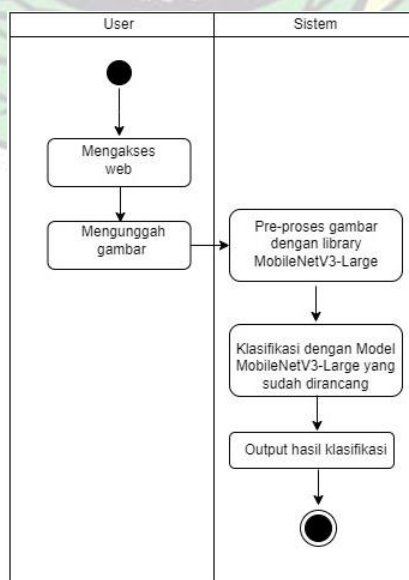


Gambar 3. 2 Use Case Diagram

Pada Gambar 3.2, aktivitas yang dapat dilakukan oleh *user* dimulai dari pengguna mengunggah gambar, melakukan klasifikasi, dan mengetahui cara penanggulangan hama tersebut. Mengunggah gambar adalah aktivitas yang dapat dilakukan pengguna pada web, dan klasifikasi hanya dapat dilakukan pengguna mengunggah gambar.

3.1.5 Activity Diagram

Activity diagram adalah diagram alur kerja yang digunakan untuk menggambarkan proses atau alur aktivitas dalam suatu sistem. Diagram ini menunjukkan urutan langkah-langkah yang diperlukan untuk menyelesaikan suatu tugas. *Activity diagram* yang dirancang dapat dilihat pada Gambar 3.3.

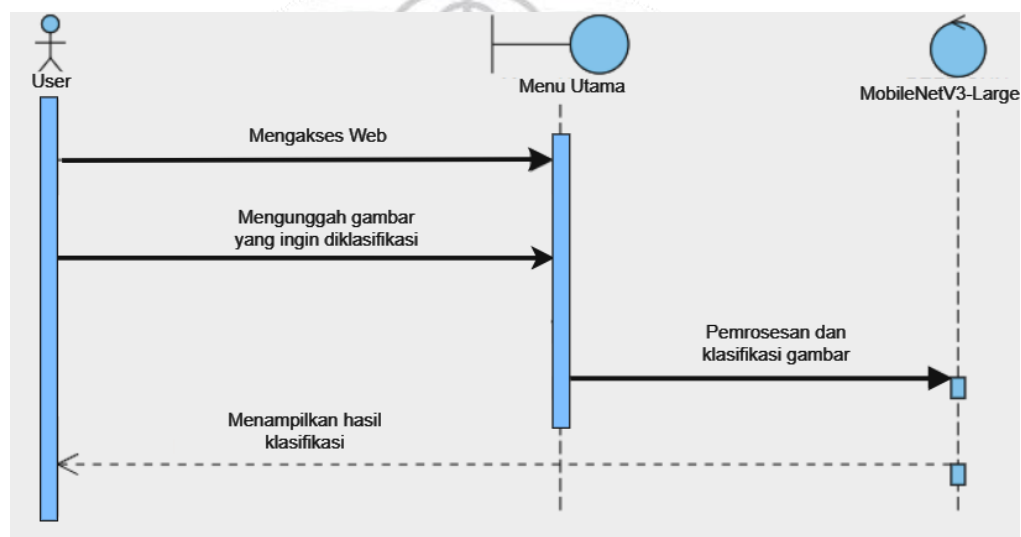


Gambar 3. 3 Activity Diagram

Pada Gambar 3.3, *activity diagram* dimulai dari pengguna mengakses web, kemudian mengunggah gambar yang ingin diklasifikasikan pada web. Setelah itu, sistem akan melakukan pra-pemrosesan terhadap gambar tersebut dan melakukan klasifikasi menggunakan model MobileNetV3-Large dan sistem akan menampilkan klasifikasi gambar tersebut kepada pengguna.

3.1.6 Sequence Diagram

Sequence diagram adalah alat yang digunakan untuk menggambarkan interaksi antar objek dalam sebuah sistem. Diagram ini dapat membantu memahami cara kerja sistem dan mengidentifikasi potensi masalah.

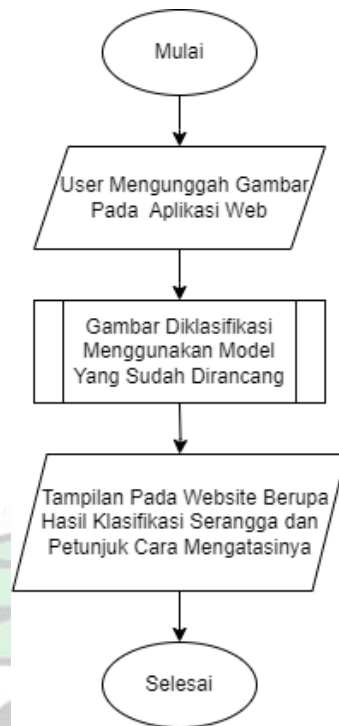


Gambar 3. 4 *Sequence Diagram*

Pada Gambar 3.4, proses dari *sequence diagram* yang dimulai dari *user* mengakses web, kemudian user mengunggah gambar yang ingin diklasifikasikan, setelah itu gambar akan diproses oleh model dan hasilnya akan ditampilkan langsung pada menu utama

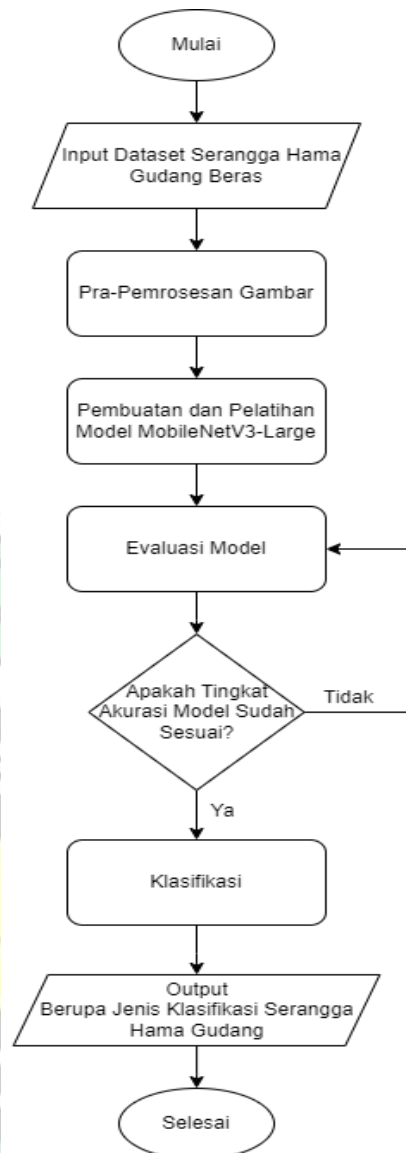
3.1.7 Flowchart Sistem

Flowchart adalah representasi visual dari alur logika sebuah proses atau sistem, menggunakan simbol-simbol standar untuk menampilkan langkah-langkah proses, keputusan, dan aliran informasi dari awal hingga akhir. *Flowchart* juga bermanfaat untuk mendokumentasikan, menganalisis, dan memperbaiki proses-proses yang ada, serta merencanakan proses-proses baru.



Gambar 3.5 *Flowchart* Interaksi Pengguna Dengan Aplikasi Web

Pada sistem yang akan dirancang, interaksi yang dilakukan oleh pengguna dengan *website* akan menggunakan alur seperti pada Gambar 3.5, yaitu pengguna akan mengunggah gambar pada aplikasi web, kemudian gambar akan diproses dan diklasifikasi menggunakan model MobileNetV3-Large, dan hasil klasifikasi dan petunjuk cara mengatasinya akan ditampilkan pada pengguna.



Gambar 3. 6 Flowchart Sistem Klasifikasi Menggunakan MobileNetV3-Large

Pada sistem yang akan dirancang akan menggunakan alur seperti gambar 3.6, yaitu :

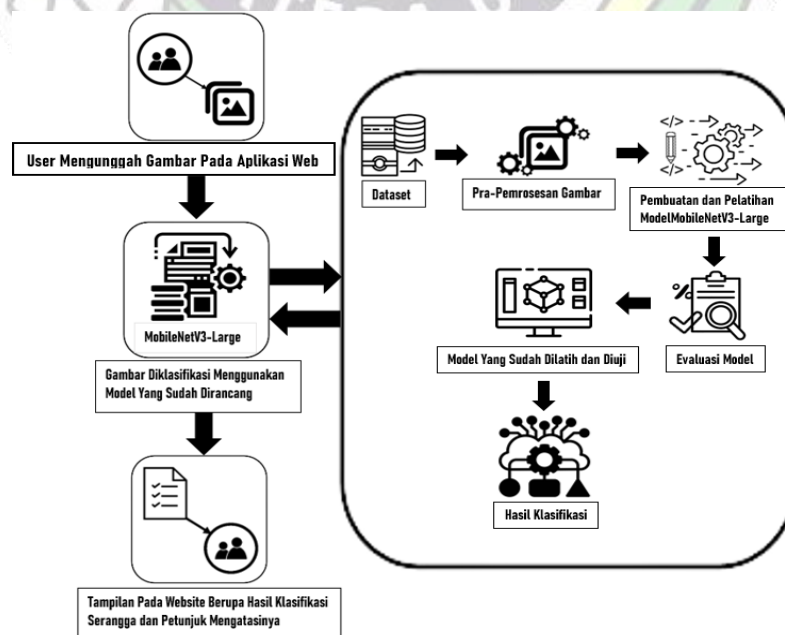
1. Mulai
2. Input *Dataset* Serangga Hama Gudang Beras, *dataset* gambar serangga hama gudang beras dimasukkan ke dalam sistem, yang terdiri dari gambar-gambar yang mewakili berbagai jenis serangga hama, dan setiap gambar sudah diberi label sesuai dengan kelas masing-masing serangga.
3. Pra-Pemrosesan Gambar, yang melibatkan augmentasi, normalisasi, dan pembersihan gambar sebelum dimasukkan ke model.

4. Pembuatan dan Pelatihan Model MobileNetV3-Large, yaitu proses membangun dan melatih model menggunakan arsitektur MobileNetV3-Large untuk klasifikasi.
5. Melakukan evaluasi pada model yang sudah dibuat.
6. Melatih kembali model sampai tingkat akurasi model yang dibuat sudah sesuai.
7. Model yang sudah terlatih digunakan untuk melakukan klasifikasi pada gambar baru. Setiap gambar yang masuk akan dikategorikan ke dalam salah satu kelas serangan hama berdasarkan fitur-fitur yang dipelajari oleh model selama pelatihan.
8. Sistem menampilkan hasil klasifikasi berupa jenis serangan.
9. Selesai.

3.2 Perancangan Sistem

Dalam merancang sistem yang diinginkan, diperlukan beberapa diagram yang menggambarkan secara detail spesifikasi sistem yang direncanakan. Selain itu, perlu dipertimbangkan alur proses dari tahap perancangan program untuk memastikan konsistensi dan keselarasan yang diperlukan.

3.2.1. Arsitektur Umum Sistem



Gambar 3. 7 Arsitektur Umum Sistem

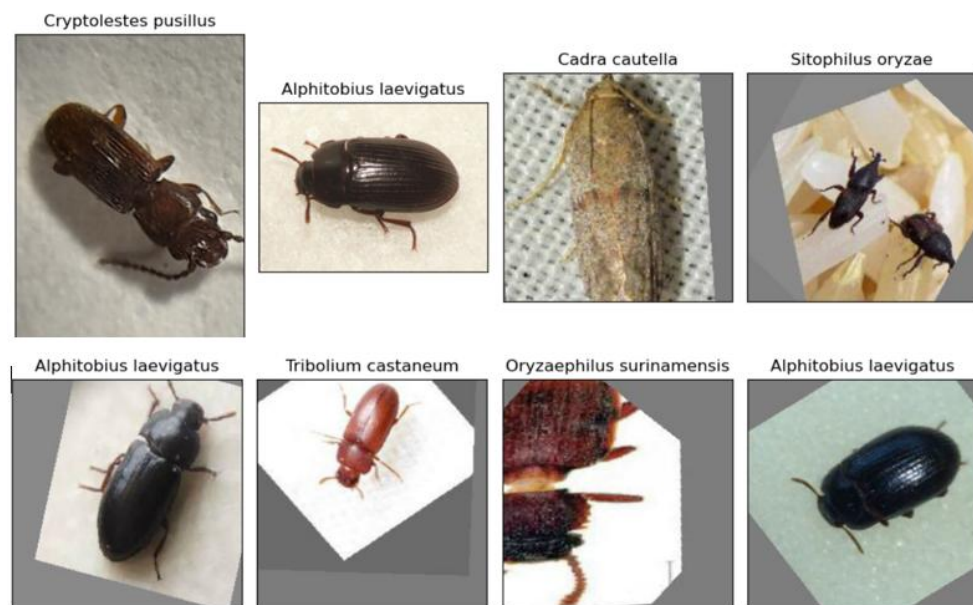
Pada sistem yang ingin dirancang akan mempunyai arsitektur umum seperti pada Gambar 3.7, yaitu :

1. Pengguna akan mengakses sistem melalui antarmuka *website*. *Website* ini menyediakan fitur untuk mengunggah gambar serangga hama yang ingin diidentifikasi. Gambar yang diunggah tidak harus diambil secara real-time, pengguna dapat mengunggah gambar yang sudah ada atau diambil sebelumnya. Perlu diperhatikan, gambar serangga yang diambil haruslah diperdekat (*zoom-in*) sampai bentuk serangga terlihat dengan jelas.
2. Setelah pengguna mengunggah gambar serangga, gambar tersebut akan diproses pada model MobileNetV3-Large. Pembuatan model memiliki beberapa tahapan, yaitu :
 - a. Penyusunan *Dataset* (Pengumpulan dan Pelabelan Gambar)
 - b. Pra-pemrosesan gambar, seperti *resize*, normalisasi, dan augmentasi.
 - c. Pembuatan model MobileNetV3-Large dan pelatihan model tersebut terhadap *dataset* yang dikumpulkan.
 - d. Evaluasi model sampai tingkat akurasi tinggi.
 - e. Jika sudah selesai dievaluasi, model siap untuk digunakan.
3. Model akan menampilkan hasil klasifikasi kepada pengguna sebagai bentuk keluaran dari model. Dan pada *website* akan ditampilkan petunjuk yang sesuai dan efektif untuk mengatasi hama serangga gudang beras berdasarkan hasil klasifikasi dari model.

3.2.2. Pengumpulan Dataset

Proses pengumpulan *dataset* merupakan langkah awal yang penting dalam pengembangan sistem klasifikasi jenis serangga hama pada gudang beras. *Dataset* yang digunakan dalam penelitian ini diperoleh dari situs Kaggle, yang merupakan platform berbagi *dataset* secara global. *Dataset* yang tersedia di Kaggle bersumber dari situs *Global Biodiversity Information Facility* (GBIF), sebuah portal data biodiversitas yang menyediakan koleksi data global tentang spesies, termasuk berbagai jenis serangga. *Dataset* yang dikumpulkan terdiri dari delapan kelas serangga hama yang sering ditemukan di gudang beras, yaitu *Sitophilus oryzae*, *Rhyzopertha dominica*, *Oryzaephilus surinamensis*, *Tribolium castaneum*, *Cryptolestes pusillus*, *Cadra cautella*, *Liposcelis spp.*, dan *Alphitobius laevigatus*. Setiap kelas mewakili spesies serangga tertentu yang

menjadi fokus penelitian ini. Gambar-gambar dalam *dataset* memiliki format .jpg dan telah dilabeli sesuai dengan spesies masing-masing. Label ini sangat penting untuk memastikan data dapat digunakan dalam proses pelatihan model *Convolutional Neural Network* (CNN) dengan akurasi yang tinggi. Contoh visualisasi *dataset* yang sudah dikumpulkan dapat dilihat pada Gambar 3.8.



Gambar 3. 8 Contoh Visualisasi *Dataset*

3.2.3. *Pre-processing Dataset*

Pre-processing dataset adalah serangkaian langkah yang dilakukan untuk membersihkan, mengubah, dan mempersiapkan data mentah menjadi bentuk yang lebih sesuai untuk analisis atau pemodelan. Tujuannya adalah untuk meningkatkan kualitas data, menghilangkan noise atau ketidaksesuaian, dan membuatnya siap untuk dieksploitasi oleh algoritma pemrosesan data atau pembelajaran mesin. Tahapan yang dilakukan pada *pre-processing* penelitian ini adalah :

1. *Resize*

Resize adalah proses mengubah ukuran gambar menjadi dimensi tertentu yang dibutuhkan oleh model. Model *deep learning* seperti CNN biasanya menerima input dengan ukuran tetap, sehingga semua gambar dalam *dataset* harus diubah ukurannya agar konsisten. Jika ukuran gambar berbeda-beda, model tidak bisa memprosesnya dengan baik. Gambar akan diubah menjadi ukuran 224x224

piksel, karena ukuran tersebut adalah ukuran yang didukung oleh model MobileNetV3-Large.

2. Normalisasi

Normalisasi adalah teknik yang digunakan untuk mengubah nilai-nilai data ke dalam skala atau rentang tertentu, biasanya untuk mempermudah proses analisis atau pemrosesan lebih lanjut oleh model *machine learning*. Tujuan utamanya adalah agar data lebih konsisten dan tidak ada fitur (atau atribut) yang mendominasi proses pembelajaran hanya karena skala atau nilainya yang jauh lebih besar. Proses normalisasi diterapkan melalui fungsi “*preprocessing_function*” yang digunakan dalam *Image Data Generator*.

3. Augmentasi

Augmentasi adalah teknik untuk menambah variasi pada *dataset* dengan cara memodifikasi gambar-gambar yang ada. Ini dilakukan untuk mencegah *overfitting* dan membuat model lebih unggul. Jika *dataset* terbatas, augmentasi memungkinkan kita membuat variasi dari gambar asli sehingga jumlah data secara efektif bertambah.

3.2.4. Perancangan Algoritma MobileNetV3-Large

MobileNetV3-Large adalah salah satu arsitektur model *Convolutional Neural Network* (CNN) yang dirancang untuk perangkat dengan sumber daya terbatas. Implementasi algoritma ini digunakan untuk mengklasifikasikan gambar secara efisien, termasuk dalam konteks klasifikasi jenis serangga hama pada gudang beras. Setelah gambar pada *dataset* melalui pra-pemrosesan data, gambar tersebut akan diolah menggunakan algoritma MobileNetV3-Large. Proses tersebut menggunakan *library* MobileNetV3-Large yang sudah tersedia. Berikut adalah proses implementasi dan bagaimana algoritma ini dengan bantuan *library* yang sudah tersedia tersebut bekerja dalam mengklasifikasikan gambar.

1. Input Data

Data input berupa gambar serangga dalam format RGB dengan resolusi tertentu yang dalam sistem ini dengan ukuran 224x224 piksel. Gambar-gambar ini melewati tahap preprocessing, seperti *resizing*, normalisasi, dan augmentasi data untuk meningkatkan variasi data pelatihan.

2. Ekstraksi Fitur

MobileNetV3-Large memproses gambar melalui beberapa lapisan konvolusi. Lapisan ini mengekstrak fitur dari gambar, seperti pola tekstur, warna, atau bentuk serangga hama. *Depthwise Separable Convolution* digunakan untuk memisahkan pengolahan informasi spasial dan kanal, sehingga mempercepat proses komputasi.

3. Penguatan Fitur Penting

SE Block mengidentifikasi fitur penting dari gambar dan menekannya, sehingga model lebih fokus pada pola-pola yang relevan dengan klasifikasi. *Squeeze* adalah tahapan dimana semua *channel* pada data fitur akan dirangkum menjadi satu nilai statistik representatif (biasanya rata-rata), yang menghasilkan representasi *channel-wise global*. Proses ini menggunakan *global average pooling* untuk merangkum informasi di seluruh *spatial feature map*. Misalnya, jika sebuah data fitur memiliki 256 *channel*, maka *output* dari *squeeze* akan menghasilkan vektor yang terdiri dari 256 elemen, masing-masing mewakili *channel* tersebut. *Excite* adalah tahapan yang terdiri dari dua lapisan *fully connected* yang akan menghitung skor penting untuk setiap *channel* berdasarkan informasi yang dirangkum dari tahap *squeeze*. Skor ini nantinya digunakan untuk memberikan bobot lebih pada *channel* penting dan bobot lebih kecil pada *channel* yang kurang penting. Setelah *excitation*, skor yang diperoleh akan digunakan untuk menghitung kembali skala setiap *channel* pada data fitur asli. Dengan demikian, *channel-channel* yang lebih penting (berdasarkan skor) akan dikuatkan, sementara yang kurang penting akan dilemahkan. Oleh karena itu, *Squeeze* dan *Excite* memungkinkan jaringan untuk secara dinamis memberikan bobot pada fitur-fitur penting dan menekan fitur-fitur yang tidak terlalu penting.

4. Fungsi Aktivasi

Fungsi aktivasi *Hardswish* digunakan dalam *hidden layers* untuk meningkatkan efisiensi komputasi. *Hard-Swish* adalah varian dari fungsi aktivasi *Swish* yang digunakan untuk meningkatkan efisiensi komputasi. *Hard-Swish* merupakan fungsi aktivasi non-linear yang dirancang khusus untuk meningkatkan kinerja pada jaringan yang dioptimalkan untuk perangkat seluler atau *edge devices*, seperti pada arsitektur MobileNetV3. Fungsi ini merupakan fungsi *non-linear* dan mulus, serta memiliki sifat memperkuat fitur penting yang bernilai positif,

sementara fitur yang kurang penting (biasanya nilai negatif) didorong ke nol atau mendekati nol.

$$H - \text{Swish}(x) = \frac{x \cdot \text{ReLU6}(x+3)}{6}$$

$\text{ReLU6}(x)$ membatasi nilai input antara 0 hingga 6. Fungsi ini bekerja dengan mengalikan input x dengan nilai non-linear yang dihasilkan oleh operasi $\text{ReLU6}(x+3)$ yang diskalakan ke rentang 0 hingga 1. Pada rentang $-3 < x < 3$, fungsi ini menghasilkan kurva yang lebih halus dibandingkan ReLU , sementara untuk $x \leq -3$ hasilnya adalah nol, dan untuk $x \geq 3$ fungsi menjadi linier. Keunggulan $H\text{-Swish}$ adalah perhitungannya yang lebih cepat karena tidak memerlukan operasi eksponensial seperti pada Swish , sehingga lebih cocok untuk digunakan pada perangkat dengan sumber daya terbatas.

5. Lapisan *Fully Connected*

Setelah fitur diekstraksi, fitur ini diumpankan ke lapisan *fully connected* untuk mengklasifikasikan gambar ke dalam salah satu kelas dari 8 jenis serangga hama.

6. Fungsi *Softmax*

Lapisan terakhir menggunakan fungsi *Softmax* untuk mengubah keluaran menjadi probabilitas untuk setiap kelas serangga. *Softmax* adalah fungsi aktivasi yang sering digunakan di jaringan saraf, khususnya pada lapisan *output* untuk masalah klasifikasi multi-kelas. Fungsi ini mengubah nilai *output* dari jaringan menjadi probabilitas yang dapat dipahami oleh manusia, dengan cara mengubah skor atau nilai yang tidak terbatas menjadi angka antara 0 dan 1, yang jumlahnya selalu 1. Hal ini memungkinkan model untuk memprediksi kelas yang paling mungkin berdasarkan distribusi probabilitas.

Fungsi *softmax* untuk suatu vektor input $Z = [Z_1, Z_2, \dots, Z_n]$ dihitung dengan rumus :

$$\text{softmax}(Z_i) = \frac{e^{Z_i}}{\sum_{j=1}^n e^{Z_j}}$$

Keterangan :

- Z_i iadalah eksponensial dari skor kelas i

- Z_j adalah eksponensial dari skor kelas tersebut

- Pembagi $\sum_{j=1}^n e^{Z_j}$ adalah jumlah eksponensial dari semua skor kelas.

Hasil dari fungsi *softmax* adalah sebuah vektor probabilitas, di mana setiap elemen dalam vektor ini menunjukkan kemungkinan kelas tertentu. Kelas dengan nilai probabilitas tertinggi adalah kelas yang dipilih sebagai prediksi oleh model. Contoh penggunaannya, misal sebuah model mengeluarkan skor berikut untuk tiga kelas, yaitu $Z_1=2$ $Z_2=1$ $Z_3=0.5$. Dengan menerapkan *softmax*, kita akan mendapatkan probabilitas yang menggambarkan seberapa besar kemungkinan masing-masing kelas adalah hasil prediksi model.

7. Keluaran (*Output*)

Model memberikan keluaran berupa probabilitas untuk masing-masing kelas. Kelas dengan probabilitas tertinggi diambil sebagai prediksi model.

3.3 Evaluasi Model

Setelah model selesai dirancang dan dilatih dengan model MobileNetV3-Large, selanjutnya dilakukan evaluasi untuk mengetahui kemampuan sistem dalam melakukan klasifikasi gambar serangga hama gudang beras. Evaluasi dilakukan dengan menghitung metrik evaluasi seperti *precision*, *recall*, *f1-score*, dan *accuracy* berdasarkan label sebenarnya. Rumus untuk menghitung metrik evaluasi yaitu.

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

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

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

$$\text{F1-Score} = 2 * (\text{Presisi} * \text{Recall}) / (\text{Presisi} + \text{Recall})$$

Evaluasi ini penting untuk memahami kekuatan dan keterbatasan sistem dalam mengklasifikasi gambar serangga hama gudang beras.

3.4 Menyimpan Model

Setelah sistem selesai dilatih dan dievaluasi, maka sistem tersebut akan disimpan dalam bentuk *file* dengan format ".h5". Model yang tersimpan dapat dimuat

kembali untuk digunakan dalam prediksi tanpa perlu melakukan proses pelatihan ulang. Proses ini meningkatkan efisiensi sistem karena model yang sudah dilatih dapat digunakan berulang kali.

3.5 Integrasi Model Pada Website

Tahap terakhir adalah pengembangan antarmuka pengguna berupa *website*. Pembuatan *website* menggunakan bahasa pemrograman *Python* dan *framework Flask*. *Flask* adalah *framework* web minimalis yang ditulis dalam *Python*, yang memungkinkan pengembangan aplikasi web secara cepat dan fleksibel. *Flask* akan berperan sebagai *back-end server* dalam aplikasi web. Ketika pengguna mengunggah gambar serangga hama melalui antarmuka web, *Flask* akan menangani permintaan tersebut (melalui *routing*) dan memproses gambar yang diterima, kemudian menyajikan hasil ke pengguna melalui antarmuka web.



BAB 4

IMPLEMENTASI DAN PENGUJIAN

4.1 Pre-processing Dataset

Tahap pertama adalah membagi *dataset* yang sudah dikumpulkan, menjadi 2 kategori berbeda, yaitu *training* dan *testing*, kemudian data *training* akan dibagi lagi menjadi 2, yaitu *train* dan *validation*. Data *training* akan digunakan untuk melatih model CNN *deep learning* dan parameternya akan disesuaikan dengan data *validation*. Dan kinerja data akan dievaluasi menggunakan data *testing* seperti pada Gambar 4.1 dan Gambar 4.2.

```
# Separate in train and test data
train_df, test_df = train_test_split(image_df, test_size=0.2, shuffle=True, random_state=1)
```

Gambar 4. 1 Membagi *Dataset* Menjadi *Train* dan *Test*

```
# Split the data into three categories.
train_images = train_generator.flow_from_dataframe(
    dataframe=train_df,
    x_col='Filepath',
    y_col='Label',
    target_size=(224, 224),
    color_mode='rgb',
    class_mode='categorical',
    batch_size=32,
    shuffle=True,
    seed=42,
    subset='training'
)

val_images = train_generator.flow_from_dataframe(
    dataframe=train_df,
    x_col='Filepath',
    y_col='Label',
    target_size=(224, 224),
    color_mode='rgb',
    class_mode='categorical',
    batch_size=32,
    shuffle=True,
    seed=42,
    subset='validation'
)

test_images = test_generator.flow_from_dataframe(
    dataframe=test_df,
    x_col='Filepath',
    y_col='Label',
    target_size=(224, 224),
    color_mode='rgb',
    class_mode='categorical',
    batch_size=32,
```

Gambar 4. 2 Membagi *Dataset Train* Menjadi *Train* dan *Validation*

Kemudian *dataset* tersebut diubah ukurannya menggunakan menggunakan perintah *resize* dan *rescaling* pada *library keras* untuk menyamakan ukuran semua gambar pada *dataset*, seperti pada Gambar 4.3.

```
# Resize Layer
resize_and_rescale = tf.keras.Sequential([
    layers.experimental.preprocessing.Resizing(224,224),
    layers.experimental.preprocessing.Rescaling(1./255),
])
```

Gambar 4. 3 *Resize* dan *Rescaling* Gambar

Setelah ukuran gambar disamakan, akan dilakukan augmentasi untuk menambah variasi pada *dataset* dengan cara memodifikasi gambar-gambar yang ada. Ini dilakukan untuk mencegah *overfitting* dan membuat model lebih unggul, seperti pada Gambar 4.4.

```
# Setup data augmentation
data_augmentation = keras.Sequential([
    preprocessing.RandomRotation(0.2),
    preprocessing.RandomZoom(0.2),
    preprocessing.RandomHeight(0.2),
    preprocessing.RandomWidth(0.2),
], name="data_augmentation")
```

Gambar 4. 4 Augmentasi Gambar

Setelah dilakukan augmentasi *dataset* akan bervariasi dan jumlah akan bertambah tiap kelasnya. Jumlahnya dapat kita tampilkan menggunakan fungsi seperti pada Gambar 4.4 dan jumlah tiap kelasnya dapat dilihat pada Gambar 4.5. Total *dataset* yang dimiliki adalah 800 gambar tiap kelas.

```
# Hitung jumlah sampel per kelas dalam data training
train_counts = train_df['Label'].value_counts()
print("Distribusi Data Training:")
print(train_counts)

# Hitung jumlah sampel per kelas dalam data testing
test_counts = test_df['Label'].value_counts()
print("\nDistribusi Data Testing:")
print(test_counts)
```

Gambar 4. 5 Menampilkan Jumlah *Dataset*

```

Distribusi Data Training:
Label
Cadra cautella           664
Alphitobius laevigatus    657
Tribolium castaneum       655
Oryzaephilus surinamensis 648
Rhyzopertha dominica      645
Cryptolestes pusillus     640
Sitophilus oryzae         640
Liposcelis spp            639
Name: count, dtype: int64

Distribusi Data Testing:
Label
Cryptolestes pusillus     174
Sitophilus oryzae         169
Liposcelis spp            166
Oryzaephilus surinamensis 165
Rhyzopertha dominica      162
Tribolium castaneum       160
Alphitobius laevigatus    156
Cadra cautella            146
Name: count, dtype: int64

```

Gambar 4. 6 Tampilan Jumlah *Dataset*

4.2 *Training* dan Implementasi MobileNetV3-Large

Setelah *dataset* siap untuk digunakan, kemudian masuk ke tahapan *training* model menggunakan *dataset train* yang sebelumnya sudah dipisahkan. Pertama adalah membuat sebuah fungsi yang memuat *pretrained model* MobileNetV3-Large yang telah dilatih sebelumnya pada *dataset ImageNet* dan model ini akan digunakan sebagai ekstraktor fitur. Karena bobot model telah dilatih pada *dataset ImageNet*, sehingga fitur dasar seperti deteksi tepi, pola tekstur, dan objek dapat digunakan ulang untuk tugas klasifikasi serangga hama. Adapun fungsi *pretrained model* yang dirancang dapat dilihat pada Gambar 4.7 .

```

# Load the pretrained model
pretrained_model = tf.keras.applications.MobileNetV3Large(
    input_shape=(224, 224, 3),
    include_top=False,
    weights='imagenet',
    pooling='avg'
)
pretrained_model.trainable = False

```

Gambar 4. 7 Fungsi *Pretrained Model*

Selanjutnya membentuk model klasifikasi serangga dengan 8 kelas dan mengoptimalkan performanya menggunakan kombinasi *optimizer Adam*, fungsi *loss categorical_crossentropy*, dan *callback* untuk mengontrol pelatihan. Dengan data pelatihan dan validasi yang digunakan, model belajar pola yang relevan untuk menghasilkan prediksi dengan akurasi tinggi. *Optimizer Adam* dengan *learning rate* 0.0001 digunakan untuk mengatur pembaruan bobot selama pelatihan. *Adam* dipilih karena efisien dalam waktu komputasi dan sering memberikan hasil yang baik pada masalah klasifikasi. Model dilatih selama 20 kali iterasi penuh pada seluruh *dataset* untuk mempelajari pola secara bertahap. Model pun mendapatkan tingkat akurasi 98%. Model klasifikasi yang dirancang dapat dilihat pada Gambar 4.8.

```
#pembuatan model transfer learning untuk klasifikasi
inputs = pretrained_model.input
x = resize_and_rescale(inputs)
x = data_augmentation(x)

x = Dense(256, activation='relu')(pretrained_model.output)
x = Dropout(0.2)(x)
x = Dense(256, activation='relu')(x)
x = Dropout(0.2)(x)

outputs = Dense(8, activation='softmax')(x)

model = Model(inputs=inputs, outputs=outputs)

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

history = model.fit(
    train_images,
    steps_per_epoch=len(train_images),
    validation_data=val_images,
    validation_steps=len(val_images),
    epochs=20,
    callbacks=[
        early_stopping,
        create_tensorboard_callback("training_logs",
                                    "insect_classification"),
        checkpoint_callback,
    ]
)
```

Gambar 4. 8 Model Klasifikasi

4.3 Evaluasi

Setelah perancangan dan pelatihan model dengan MobileNetV3-Large selesai dilakukan, selanjutnya dilakukan evaluasi untuk mengetahui kemampuan sistem dalam melakukan klasifikasi gambar serangga hama gudang beras. Evaluasi

dilakukan dengan menghitung metrik evaluasi seperti *precision*, *recall*, *f1-score*, dan *accuracy* berdasarkan label sebenarnya. Evaluasi ini menggunakan *library* “report”, yang didalamnya menggunakan rumus :

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

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

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

$$\text{F1-Score} = 2 * (\text{Presisi} * \text{Recall}) / (\text{Presisi} + \text{Recall})$$

Evaluasi yang dilakukan dapat dilihat pada Gambar 4.9.

```
# Menghitung prediksi pada data uji
y_pred = model.predict(test_images)
y_pred_classes = np.argmax(y_pred, axis=1)
y_true = test_images.classes # Kelas yang sebenarnya

# Evaluasi model dengan classification report
print(classification_report(y_true, y_pred_classes))
```

Gambar 4. 9 Evaluasi

Setelah evaluasi dijalankan akan dihasilkan nilai evaluasi untuk setiap kelasnya. Nilai evaluasi dari tiap kelas tersebut dapat dilihat pada Tabel 4.1.

Tabel 4. 1 Tabel Hasil Evaluasi

Kelas	Accuracy	Precision	Recall	F1-Score
<i>Alphitobius laevigatus</i>	95,51%	99%	96%	97%
<i>Cadra cautella</i>	98,63%	97%	99%	98%
<i>Cryptolestes pusillus</i>	92,53%	93%	93%	93%
<i>Liposcelis spp</i>	93,37%	99%	93%	97%
<i>Oryzaephilus surinamensis</i>	90,30%	94%	90%	92%
<i>Rhyzopertha dominica</i>	92,59%	96%	93%	94%
<i>Sitophilus oryzae</i>	98,82%	93%	99%	96%
<i>Tribolium castaneum</i>	95%	87%	95%	91%


```

results = model.evaluate(test_images, verbose=0)

print("    Test Loss: {:.5f}".format(results[0]))
print("Test Accuracy: {:.2f}%".format(results[1] * 100))

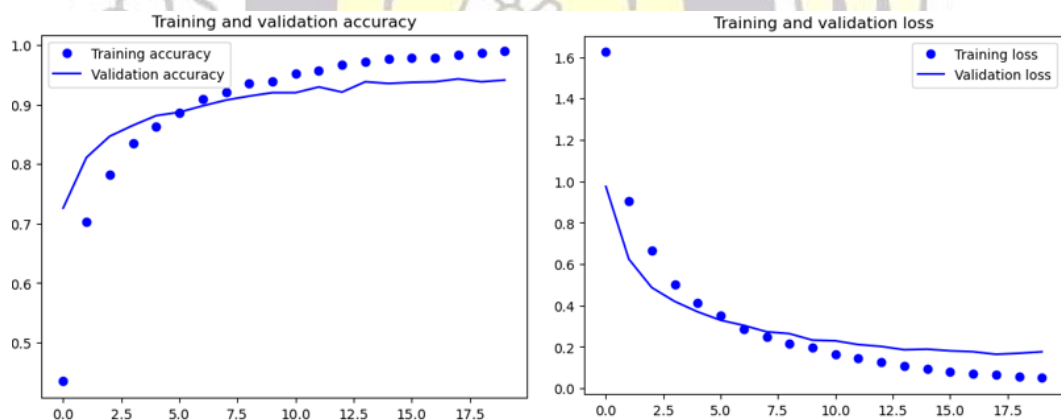
```

✓ 25.5s

Test Loss: 0.15388
Test Accuracy: 94.53%

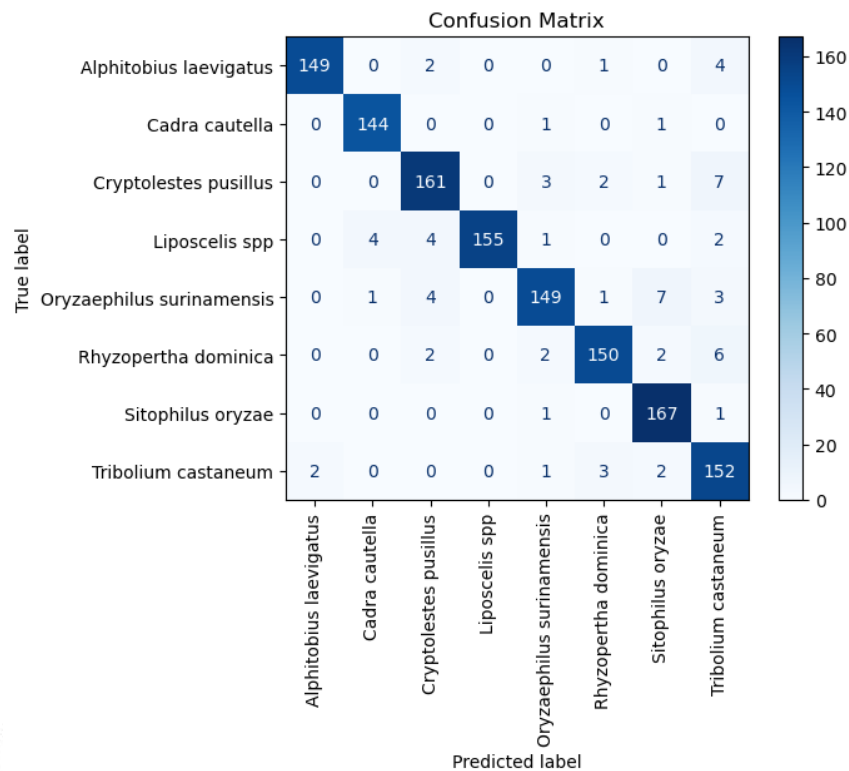
Gambar 4. 10 Pengujian Pada Data *Testing*

Pada Gambar 4.10, dilakukan evaluasi untuk mendeteksi apakah model yang dirancang mengalami *overfitting*, yang merupakan sebuah kondisi dimana model terlalu menghafal data *training* sehingga jika dilakukan pengujian pada data *testing*, tingkat akurasi yang didapat cukup rendah. Evaluasi ini dapat dilakukan dengan melakukan pengujian pada data *testing*. Tampak bahwa nilai akurasi model pada data *testing* juga memiliki nilai 94,53 %, nilai yang masih cukup dekat dengan nilai akurasi pada data *training* yang merupakan 98%, hal ini mengindikasikan bahwa model tidak mengalami *overfitting*.



Gambar 4. 11 Perbandingan Kurva Akurasi dan *Loss* Pada Data *Training* dan *Validation*

Pada Gambar 4.11, kurva akurasi dan *training loss* dan *validation* tampak hampir sama, yaitu ketika *training loss* mengalami penurunan, nilai *validation loss* juga mengalami penurunan, begitu juga pada nilai akurasi, ketika akurasi *training* mengalami kenaikan, akurasi *validation* juga mengalami kenaikan. Hasil ini mengindikasikan bahwa model tidak mengalami *overfitting*, karena kedua nilai tersebut tidak berbanding terbalik.



Gambar 4. 12 *Confusion Matrix*

Berdasarkan gambar 4.12, dapat dibuat tabel untuk melihat *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN) untuk semua kelas serangga berdasarkan *confusion matrix* yang sudah dibuat, dapat dilihat pada Tabel 4.2.

Tabel 4. 2 Tabel Nilai TP, TN, FP, dan FN Pada Semua Kelas

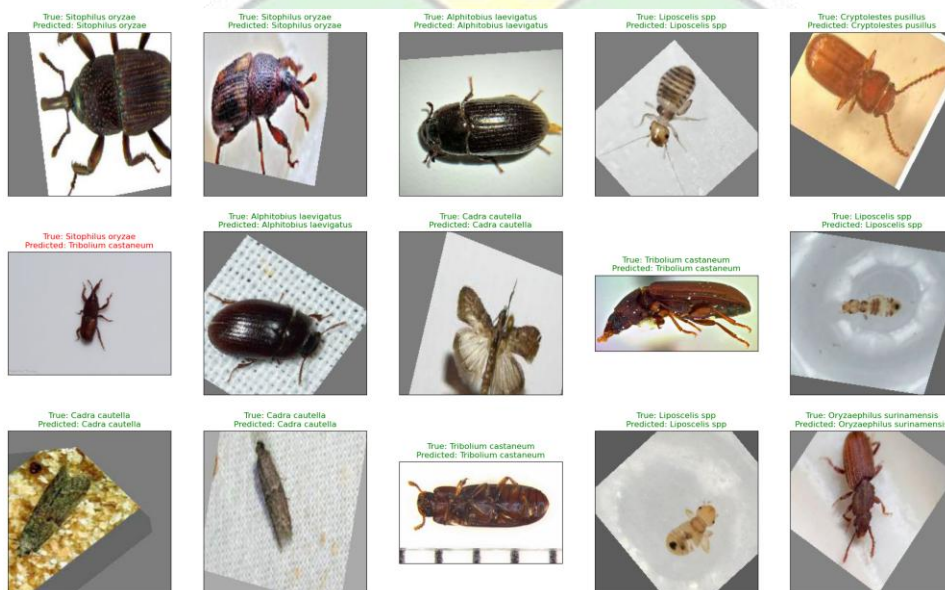
Kelas	TP	TN	FP	FN
<i>Alphitobius laevigatus</i>	149	2	2	7
<i>Cadra cautella</i>	144	5	5	2
<i>Cryptolestes pusillus</i>	161	12	12	13
<i>Liposcelis spp</i>	155	0	0	11
<i>Oryzaephilus surinamensis</i>	149	9	9	16
<i>Rhyzopertha dominica</i>	150	7	7	12
<i>Sitophilus oryzae</i>	167	13	13	2
<i>Tribolium castaneum</i>	152	23	23	8

```
# Menampilkan 15 gambar random dari hasil prediksi
random_index = np.random.randint(0, len(test_df) - 1, 15)
fig, axes = plt.subplots(nrows=3, ncols=5, figsize=(25, 15),
                        subplot_kw={'xticks': [], 'yticks': []})

for i, ax in enumerate(axes.flat):
    ax.imshow(plt.imread(test_df.Filepath.iloc[random_index[i]]))
    if test_df.Label.iloc[random_index[i]] == pred[random_index[i]]:
        color = "green"
    else:
        color = "red"
    ax.set_title(f"True: {test_df.Label.iloc[random_index[i]]}\nPredicted: {pred[random_index[i]]}", color=color)
plt.show()
plt.tight_layout()
```

Gambar 4. 13 Pengambilan Gambar Acak Dari Pengujian

Pada Gambar 4.13, dilakukan pengambilan 15 contoh gambar acak dari pengujian pada *dataset testing*. Kemudian 15 contoh pengujian tersebut ditampilkan dengan keterangan kelas yang sesai dan kelas hasil prediksinya, seperti pada Gambar 4.11.



Gambar 4. 14 Tampilan Contoh 15 Gambar Pengujian dan Hasilnya

Pada Gambar 4.14, tampak ada kesalahan pengujian pada 1 dari 15 contoh pengujian yang dilakukan. Kesalahan ini dapat dilihat pada gambar dengan tulisan berwarna merah, hal ini dapat terjadi karena nilai akurasi dari model dengan data pengujian bernilai 94,5%, sehingga masih ada kemungkinan kesalahan.

4.4 Penyimpanan Model

Setelah sistem selesai dilatih dan dievaluasi, maka sistem tersebut akan disimpan. Model yang tersimpan dapat dimuat kembali untuk digunakan dalam prediksi tanpa perlu melakukan proses pelatihan ulang. Proses ini meningkatkan efisiensi sistem karena model yang sudah dilatih dapat digunakan berulang kali dan dapat

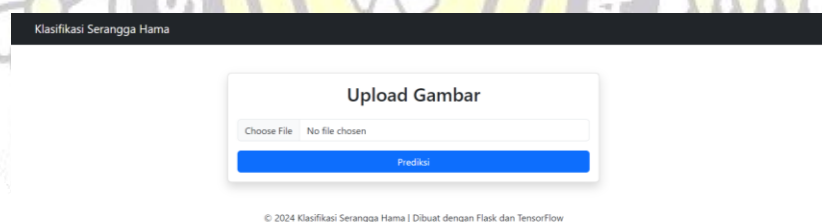
diintegrasikan dengan web yang sudah dirancang. Penyimpanan model dilakukan dalam bentuk *file* dengan format “.h5” seperti pada Gambar 4.15.

```
# Menyimpan model ke dalam file .h5
model.save('bobot/serangga_hama_model2.h5')
```

Gambar 4. 15 Penyimpanan Model

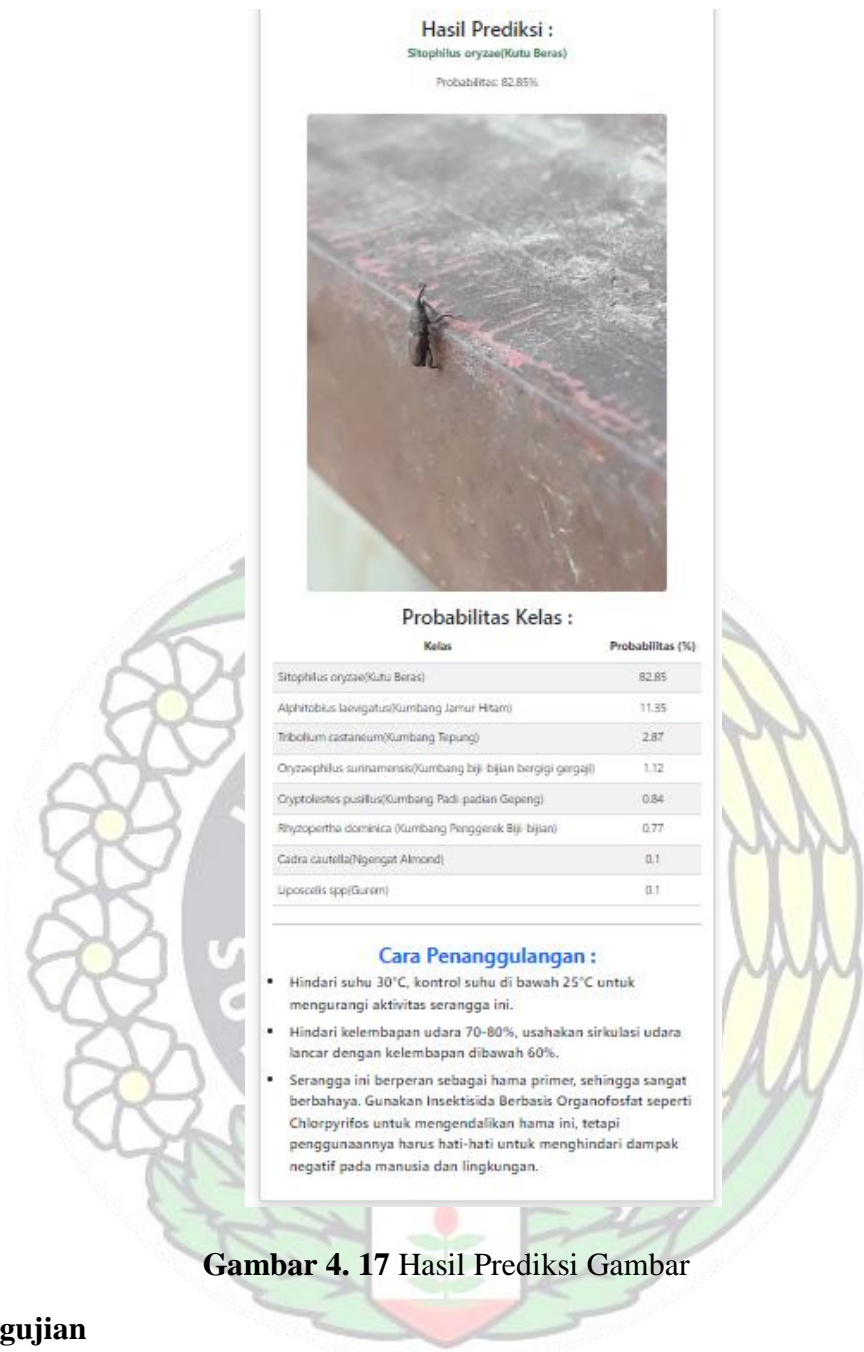
4.5 Implementasi Sistem pada Website

Tahap terakhir adalah pengembangan antarmuka pengguna berupa *website*. Pembuatan *website* menggunakan bahasa pemrograman *Python* dan *framework Flask*. *Flask* adalah *framework* web minimalis yang ditulis dalam *Python*, yang memungkinkan pengembangan aplikasi web secara cepat dan fleksibel. *Flask* akan berperan sebagai *back-end server* dalam aplikasi web. Ketika pengguna mengunggah gambar serangga hama melalui antarmuka web, *Flask* akan menangani permintaan tersebut (melalui *routing*) dan memproses gambar yang diterima, kemudian menyajikan hasil ke pengguna melalui antarmuka web. Halaman utama web yang dirancang dapat dilihat pada Gambar 4.16.



Gambar 4. 16 Halaman Utama Web

Pada halaman ini, pengguna dapat mengunggah gambar serangga hama gudang beras dan menekan tombol prediksi. Kemudian gambar tersebut akan diolah oleh sistem dan akan menampilkan hasil klasifikasinya kepada pengguna beserta cara penanggulangannya, seperti pada Gambar 4.17.














Gambar 4. 17 Hasil Prediksi Gambar






4.6 Pengujian






Pengujian sistem ini dilakukan dengan mengunggah gambar tiap-tiap kelas, untuk menguji model. Pengujian dilakukan pada web yang sudah diintegrasikan dengan model. Hasil pengujian dapat dilihat pada Tabel 4.3.




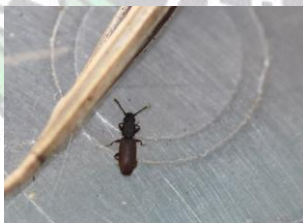

Tabel 4. 3 Tabel Pengujian


Kelas	Gambar	Hasil	Keterangan
<i>Alphitobius laevigatus</i>		<i>Alphitobius laevigatus</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.
		<i>Alphitobius laevigatus</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.
		<i>Alphitobius laevigatus</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.
		<i>Alphitobius laevigatus</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.
		<i>Alphitobius laevigatus</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.






<i>Cadra cautella</i>		<i>Cadra cautella</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.
		<i>Cadra cautella</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.
		<i>Liposcelis spp</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem masih salah, yang seharusnya adalah “ <i>Cadra cautella</i> ”.
		<i>Cadra cautella</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.
		<i>Cadra cautella</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.
		<i>Cryptolestes pusillus</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.






<i>Cryptolestes pusillus</i>		<i>Cryptolestes pusillus</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.
		<i>Cryptolestes pusillus</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.
		<i>Cryptolestes pusillus</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.
		<i>Tribolium castaneum</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem masih salah, yang seharusnya adalah “ <i>Cryptolestes pusillus</i> ”.
		<i>Liposcelis spp</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.

<i>Liposcelis</i> <i>spp</i>		<i>Liposcelis</i> <i>spp</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.
		<i>Liposcelis</i> <i>spp</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.
		<i>Liposcelis</i> <i>spp</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.
		<i>Liposcelis</i> <i>spp</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.
		<i>Oryzaephilus</i> <i>surinamensis</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.

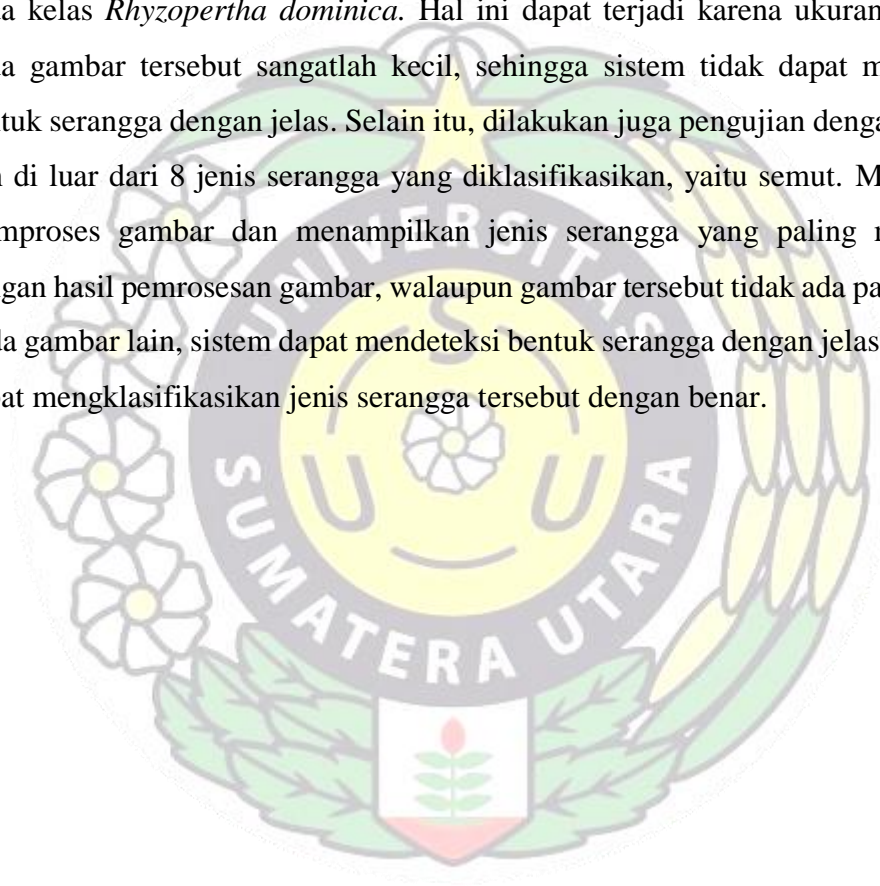
<i>Oryzaephilus surinamensis</i>		<i>Oryzaephilus surinamensis</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.
		<i>Oryzaephilus surinamensis</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.
		<i>Oryzaephilus surinamensis</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.
		<i>Oryzaephilus surinamensis</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.
		<i>Rhyzopertha dominica</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.

<i>Rhyzopertha dominica</i>		<i>Rhyzopertha dominica</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.
		<i>Rhyzopertha dominica</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.
		<i>Liposcelis spp</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem masih salah, yang seharusnya adalah “ <i>Rhyzopertha dominica</i> ”.
		<i>Sitophilus oryzae</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem masih salah, yang seharusnya adalah “ <i>Rhyzopertha dominica</i> ”.
		<i>Sitophilus oryzae</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.

<i>Sitophilus oryzae</i>		<i>Sitophilus oryzae</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.
		<i>Sitophilus oryzae</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.
		<i>Sitophilus oryzae</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.
		<i>Sitophilus oryzae</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.
		<i>Tribolium castaneum</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.

<i>Tribolium castaneum</i>		<i>Tribolium castaneum</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.
		<i>Tribolium castaneum</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.
		<i>Tribolium castaneum</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.
		<i>Tribolium castaneum</i>	Hasil klasifikasi gambar yang dilakukan oleh sistem telah sesuai.
Semut		<i>Sitophilus oryzae</i>	Hasil klasifikasi gambar oleh sistem salah karena gambar tidak termasuk ke dalam 8 jenis serangga yang diklasifikasikan.

Gambar yang digunakan dalam tahap pengujian ini menggunakan gambar yang disesuaikan seperti melakukan pengambilan foto di dunia nyata. Hasil pengambilan foto juga dapat dipotong atau *crop* agar gambar terfokus pada serangga saja. Dapat dilihat gambar pada Tabel 4.3, ukuran serangga pada gambar terlihat kecil, akan tetapi model dapat melakukan klasifikasi dengan benar pada sebagian besar gambar sesuai dengan kelas dari gambar tersebut. Ada kesalahan klasifikasi masing-masing satu gambar pada kelas *Cadra cautella* dan *Cryptolestes pusillus*, dan 2 gambar pada kelas *Rhyzopertha dominica*. Hal ini dapat terjadi karena ukuran serangga pada gambar tersebut sangatlah kecil, sehingga sistem tidak dapat mendeteksi bentuk serangga dengan jelas. Selain itu, dilakukan juga pengujian dengan gambar lain di luar dari 8 jenis serangga yang diklasifikasikan, yaitu semut. Model akan memproses gambar dan menampilkan jenis serangga yang paling mendekati dengan hasil pemrosesan gambar, walaupun gambar tersebut tidak ada pada dataset. Pada gambar lain, sistem dapat mendeteksi bentuk serangga dengan jelas, sehingga dapat mengklasifikasikan jenis serangga tersebut dengan benar.



BAB 5

PENUTUP

5.1 Kesimpulan

Berdasarkan hasil pengujian dan implementasi model MobileNetV3-Large dalam melakukan klasifikasi serangga hama gudang beras, diperoleh kesimpulan sebagai berikut :

1. Dalam mengimplementasikan algoritma MobileNetV3-Large untuk klasifikasi jenis serangga hama pada gudang beras berbasis *website*, model yang dibangun menggunakan algoritma tersebut berjalan sangat baik. Dengan menggunakan 800 *dataset* per kelas, optimizer Adam dengan nilai learning rate 0.0001, serta 20 epoch, model yang dibangun mampu mencapai akurasi sebesar 95% pada pengujian *dataset*, tanpa mengalami *overfitting*. Akurasi tersebut hanya berlaku pada 8 kelas serangga hama gudang beras yang diklasifikasikan.
2. Proses evaluasi menggunakan confusion matrix menunjukkan performa yang konsisten pada tiap kelas dengan nilai *precision*, *recall*, dan *F1-score* yang seimbang, menunjukkan kemampuan model untuk menangani 8 jenis serangga hama tersebut secara akurat.
3. Pengembangan *website* sebagai antarmuka pengguna dan integrasinya dengan model juga berjalan dengan lancar, memungkinkan proses klasifikasi 8 jenis serangga hama gudang beras tersebut dilakukan dengan cepat dan akurat. Ketika dilakukan pengujian menggunakan gambar yang kondisinya bervariasi dan seperti melakukan pengambilan foto di dunia nyata, hasil klasifikasi oleh sistem juga masih cukup bagus dengan hasil klasifikasi yang benar berjumlah 36 gambar dari total 41 gambar yang diuji.

5.2 Saran

Saran yang diberikan untuk penelitian selanjutnya adalah sebagai berikut:

1. Menambah dan menguji *dataset* gambar yang lebih banyak untuk mendapatkan *dataset* yang lebih bervariasi lagi.
2. Meskipun augmentasi data telah diterapkan, eksplorasi teknik augmentasi yang lebih kompleks, seperti pengubahan warna atau rotasi yang lebih luas yang dapat meningkatkan performa model.

3. Disarankan untuk pengembangan lebih lanjut, sistem dapat dilengkapi dengan fitur prediksi tingkat kerusakan beras akibat serangga, rekomendasi langkah mitigasi yang lengkap, dan pencatatan data historis untuk analisis lebih lanjut.



DAFTAR PUSTAKA

- Bing W. & Asad R. S. (2023). Solution for Sports Image Classification Using Modified MobileNetV3 Optimized by Modified Battle Royaloptimization Algorithm, *Heliyon*, 9(11), e21603.
- Chunguang Bi, S. Xu, N. Hu, S. Zhang, Z. Zhu & H. Yu (2023). Identification Method of Corn Leaf Disease Based on Improved Mobilenetv3 Model, *Agronomy*, 13(2), 300.
- Dyah A. P., Fatima A., Ikrar K. A., Rita M. & Sofia S. (2023). Strawberry Plant Diseases Classification Using CNN Based on MobileNetV3-Large and EfficientNet-B0 Architecture. *Jurnal Ilmiah Teknik Elektro Komputer dan Informatika (JITEKI)*, 9(3), 522-534.
- Galih A. R. & M. Fikri H. (2023). Peningkatan Performa MobilenetV3 dengan Squeeze-and-Excitation (Studi Kasus Klasifikasi Kesegaran Ikan Berdasarkan Mata Ikan). *MIND (Multimedia Artificial Intelligent Networking Database) Journal*, 8(1), 27-41.
- Howard A., Mark S., Grace Chu, Liang-Chieh C., Bo Chen, Mingxing Tan, Weijun W., Yukun Zhu, Ruoming Pang, Vijay V., Quoc V. Le & Hartwig A. (2019). Searching for MobileNetV3, *IEEE/CVF International Conference on Computer Vision (ICCV)*, 1314–1324.
- Liquan Zhao & L. Wang (2022). A New Lightweight Network Based on MobileNetV3, *KSH Transactions on Internet and Information Systems*, 16(1), 1–15.
- Nihayah A. & Djarot H. (2024). OPTIMIZATION OF CNN + MOBILENETV3 FOR INSECTIDENTIFICATION: TOWARD HIGH ACCURACY, *Jurnal Teknologi Informasi Universitas Lambung Mangkurat (JTIULM)*, 9(1), 21-28.
- Putri S. R. & Ludji P. A. (2023). KEANEKARAGAMAN DAN KELIMPAHAN HAMA PASCAPANENDI GUDANG BERAS PERUM BULOG KANTOR CABANG CIANJUR. *Jurnal Hama Penyakit Tumbuhan (HPT)*, 11(1), 11-19.
- Russel, S. J. & Norvig P., 2010. Artificial Intelligence A Modern Approach Third Edition 2010.

- Setiadi R. & Alam R. (2024). Implementasi Mobilenet untuk Klasifikasi Gambar dan Deteksi Emosi Menggunakan KERAS, *Jurnal Sistem dan Teknologi Informasi (JUSTIN)*, 12(2), 259-264.
- Waeisul B., Deny N. & M. Qomaruddin (2024). Analisis Perbandingan Klasifikasi Citra Genus Panthera dengan Pendekatan Deep learning Model MobileNet, *Jurnal Informatika dan Rekayasa Perangkat Lunak*, 6(1), 1-9.
- Xiangxiang Chu, Bo Zhang & Ruijun Xu (2020). Searching Beyond MobileNetV3, <https://arxiv.org/abs/1908.01314v4>.
- Yi Zhao, Hancheng H. Zhixiang Li, Huang Y. & Manjie Lu (2022). Intelligent Garbage Classification System Based on Improve MobileNetV3-Large, *CONNECTION SCIENCE*, 34(1), 1299-1321.

