DETEKSI TRAY UNTUK *QUALITY CONTROL* DI INDUSTRI MANUFAKTUR MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK*

LAPORAN TUGAS AKHIR

Proposal ini Disusun Guna Memenuhi Salah Satu Syarat Mendaftar Tugas Akhir
Sarjana Strata (S1) pada Program Studi Teknik Informatika
Fakultas Teknologi Industri
Universitas Islam Sultan Agung Semarang

Disusun Oleh:

Zaky Zulfa Maulayya Ahmad
32602000065

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS ISLAM SULTAN AGUNG
SEMARANG
2025

TRAY DETECTION FOR QUALITY CONTROL IN MANUFACTURING INDUSTRY USING CONVOLUTIONAL NEURAL NETWORK

FINAL PROJECT

Proposed to complete the requirement to obtain a bachelor's degree (S1)
at Informatics Engineering Departement of Industrial Technology Faculty
Sultan Agung Islamic University

***Arranged By*:**

Zaky Zulfa Maulayya Ahmad
32602000065

MAJORING OF INFORMATICS ENGINEERING
INDUSTRIAL TECHNOLOGY FACULTY
SULTAN AGUNG ISLAMIC UNIVERSITY
SEMARANG
2025

LEMBAR PENGESAHAN TUGAS AKHIR

DETEKSI TRAY UNTUK *QUALITY CONTROL* DI INDUSTRI MANUFAKTUR MENGGUNAKAN *CONVOLUTIONAL NEURAL NETWORK*

ZAKY ZULFA MAULAYYA AHMAD

Telah dipertahankan di depan tim penguji ujian sarjana tugas akhir

Program Studi Teknik Informatika

Universitas Islam Sultan Agung

Pada tanggal : 19 Agustus 2025

TIM PENGUJI UJIAN SARJANA :

Sam Farisa CH, S.T., M.Kom NIDN. 0628028602 (Ketua Penguji)	.....	02 - 09 - 2025
Badie'ah, S.T., M.Kom NIDN. 0619018701 (Anggota Penguji)	.....	03 - 09 - 2025
Ir. Sri Mulyono, M.Engg NIDN. 0626066601 (Pembimbing)	.....	04 - 09 - 2025

Semarang, 04 - 09 - 2025

Mengetahui,

Kaprodi Teknik Informatika

Universitas Islam Sultan Agung

Moch. Taufik, S.T., MIT

NIDN 0622037502

SURAT PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini :

Nama : Zaky Zulfa Maulayya Ahmad.

NIM : 32602000065

Judul Tugas Akhir : **Deteksi Tray untuk Quality Control di Industri Manufaktur Menggunakan Convolutional Neural Network**

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, 04-07-2025

Yang Menyatakan,

Zaky Zulfa Maulayya Ahmad

PERNYATAAN PERSETUJUAN
PUBLIKASI KARYA ILMIAH

Saya yang bertanda tangan di bawah ini :

Nama : Zaky Zulfa Maulayya Ahmad

NIM : 32602000065

Program Studi : S1 Teknik Informatika

Fakultas : Teknologi industri

Dengan ini menyatakan Karya Ilmiah berupa Tugas akhir dengan Judul : **Deteksi Tray Untuk Quality Control di Industri Manufaktur Menggunakan Convolutional Neural Network**

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, 09-09-2025

Yang menyatakan,

Zaky Zulfa Maulayya Ahmad

KATA PENGANTAR

Dengan mengucapkan syukur atas kehadirat Allah SWT yang telah memberikan rahmat dan karunianya, sehingga dapat menyelesaikan Tugas Akhir dengan judul "Deteksi Tray untuk *Quality Control* di Industri Manufaktur Menggunakan *Convolutional Neural Network*" guna memenuhi salah satu syarat menyelesaikan Program Studi Teknik Informatika Fakultas Teknologi Industri Universitas Islam Sultan Agung Semarang.

Tugas Akhir ini disusun dan dibuat dengan adanya bantuan dari berbagai pihak, peneliti mengucapkan terima kasih kepada:

1. Rektor UNISSULA 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. Novi Marlyana, S.T., M.T., IPU, ASEAN Engg.
3. Dosen Pembimbing Bapak Ir. Sri Mulyono, M.Eng yang telah meluangkan waktu dan memberi ilmu.
4. Orang tua penulis yang telah mendoakan, mendukung, dan mengizinkan untuk menyelesaikan laporan ini.
5. Dan kepada semua pihak yang tidak dapat saya sebutkan satu persatu.

Dengan segala hormat, penulis menyadari dengan sepenuhnya bahwa banyak kekurangan dari segi kualitas ilmu pengetahuan dalam penyusunan laporan, penulis mengharapkan adanya saran dan kritikan yang bersifat membangun demi kesempurnaan laporan ini dan untuk masa mendatang.

Semarang, …….

Zaky Zulfa Maulayya Ahmad

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

ABSTRAK

Dalam industri manufaktur, proses *Quality Control* (QC) memegang peranan penting untuk memastikan kualitas dan kelayakan produk. Namun, dalam praktiknya sering muncul permasalahan, adanya part kecil yang disusun di atas wadah atau tray yang tidak sesuai standar yang masuk dalam kategori tray *not good* (NG) namun tetap lolos proses pengecekan manual. Untuk mengatasi hal tersebut, penelitian ini mengusulkan sistem deteksi tray menggunakan metode *Convolutional Neural Network* (CNN) dengan arsitektur MobileNetV2. Dataset yang digunakan terdiri dari 500 citra tray hasil pengambilan langsung di industri manufaktur otomotif, yang kemudian diperluas melalui teknik augmentasi. Model dilatih menggunakan *optimizer* Adam dengan variasi *learning rate* (0.0001, 0.001, dan 0.01), serta dievaluasi menggunakan metrik *precision*, *recall*, dan *F1-score*. Hasil pengujian menunjukkan bahwa *learning rate* 0.001 menghasilkan performa dengan nilai akurasi mencapai 96.54%. Sistem ini kemudian diimplementasikan dalam bentuk aplikasi berbasis web menggunakan Flask, yang dapat mendeteksi tray melalui tangkap gambar kamera maupun unggahan gambar. Dengan demikian, sistem yang dibangun terbukti mampu meningkatkan akurasi QC dan memiliki potensi untuk diterapkan di lingkungan produksi nyata.

Kata kunci: *Quality Control*, *Tray Detection,* CNN, MobileNetV2, *Image Classification*.

ABSTRACT

In the manufacturing industry, the Quality Control (QC) process plays an important role in ensuring product quality and suitability. However, in practice, problems often arise, such as small parts being arranged in trays that do not meet the standards, which fall into the category of not good (NG) trays but still pass the manual inspection process. To overcome this, this study proposes a tray detection system using the Convolutional Neural Network (CNN) method with the MobileNetV2 architecture. The dataset used consists of 500 tray images taken directly from the automotive manufacturing industry, which were then expanded using augmentation techniques. The model was trained using the Adam optimizer with varying learning rates (0.0001, 0.001, and 0.01) and evaluated using precision, recall, and F1-score metrics. Test results show that a learning rate of 0.001 produces performance with accurate score 96.54%. This system was then implemented in the form of a web-based application using Flask, which can detect trays either through camera capture or image upload. Thus, the system built has been proven to improve QC accuracy and has the potential to be applied in real production environments.

Keywords*: Quality Control*, *Tray Detection*, *CNN, MobileNetV2*, *Image Classification.*

BAB I
PENDAHULUAN

1.1. Latar Belakang

Dalam industri manufaktur otomotif, *quality control (QC)* merupakan tahapan penting yang bertujuan untuk memastikan setiap komponen yang diproduksi sesuai dengan standar kualitas yang telah ditetapkan. QC dilakukan agar produk yang sampai ke pelanggan tidak mengalami cacat atau kesalahan produksi. Salah satu bentuk pengawasan ini adalah proses pengecekan pada tray, yaitu wadah atau tempat yang digunakan untuk menyusun dan mengemas komponen sebelum diproses lebih lanjut atau dikirimkan.

Tray ini berisikan komponen kecil, berupa cincin(*ring*) yang disusun secara teratur agar mudah diperiksa pada tahap *quality control*. Dengan adanya tray, proses pengecekan dapat dilakukan lebih cepat dan sistematis, sehingga meminimalisasi risiko kesalahan dalam pendeteksian komponen cacat maupun dalam perhitungan jumlah komponen. Oleh karena itu, kondisi tray yang baik menjadi faktor pendukung penting dalam menjaga kelancaran proses *quality control* di industri otomotif.

Namun, dalam praktiknya sering muncul permasalahan, misalnya adanya tray yang tidak sesuai standar (*tray not good*/NG) namun tetap lolos proses pengecekan. Kondisi ini dapat menimbulkan masalah serius, karena tray yang tidak sesuai bisa memengaruhi kualitas pengemasan, menimbulkan risiko kerusakan pada komponen, dan pada akhirnya berdampak pada kepuasan pelanggan serta menambah biaya produksi.

Saat ini, sistem deteksi tray yang digunakan di perusahaan manufaktur komponen otomotif, masih mengandalkan metode sederhana berbasis pola warna hitam putih pada gambar. Walaupun praktis, metode ini memiliki banyak keterbatasan, terutama pada kondisi pencahayaan yang bervariasi, posisi tray yang tidak konsisten, dan kualitas citra yang berbeda-beda. Akibatnya, sistem sering kali tidak akurat dalam membedakan tray baik(*good*) dan tray tidak baik(*not good*/NG), sehingga masih ada kemungkinan tray cacat lolos dalam proses QC.

Seiring perkembangan teknologi, Convolutional Neural Network (CNN) semakin banyak digunakan dalam pengolahan citra dan pengenalan pola yang lebih kompleks. CNN mampu mengekstraksi fitur detail dari sebuah gambar sehingga lebih handal dalam menghadapi variasi kondisi nyata dibandingkan metode tradisional berbasis warna. Dengan memanfaatkan CNN pada sistem deteksi tray, diharapkan dapat meningkatkan akurasi QC, mengurangi tingkat kelolosan tray NG, serta mendukung proses produksi yang lebih efisien.

Berdasarkan permasalahan tersebut, penelitian ini bertujuan untuk mengimplementasikan metode CNN dalam sistem deteksi tray di industri manufaktur otomotif. Hasil penelitian diharapkan dapat membantu menurunkan tingkat kesalahan deteksi, meningkatkan kualitas produk, serta memberikan kontribusi dalam pengembangan teknologi deteksi otomatis di dunia industri manufaktur.

1.2. Perumusan Masalah

Bagaimana merancang sistem deteksi tray di industri manufaktur otomotif menggunakan metode *Convolutional Neural Network* (CNN) dengan arsitektur MobileNetV2 agar dapat membedakan tray kategori *GOOD* dan *NOT GOOD* secara akurat?

1.3. Batasan Masalah

Pembatasan masalah yang akan diterapkan dalam "Deteksi Tray untuk *Quality Control* di Industri Manufaktur Menggunakan *Convolutional Neural Network*" adalah:

1. Menggunakan metode *Convolutional Neural Network* (CNN) dengan arsitektur MobileNetV2.
2. Sistem hanya mendeteksi gambar tray hasil produksi di industri manufaktur otomotif.
3. Penelitian ini hanya akan berfokus pada evaluasi akurasi deteksi menggunakan metode *Convolutional Neural Network* (CNN).

4. Implementasi sistem tidak dilakukan di perusahaan secara langsung. Penelitian hanya berupa simulasi menggunakan dataset gambar tray sehingga tidak mencakup uji coba nyata di lingkungan industri.

1.4. Tujuan Tugas Akhir

Tujuan dari penelitian tugas akhir ini adalah merancang sistem pendeteksi tray menggunakan metode *Convolutional Neural Network* (CNN) dan arsitektur MobileNetV2 yang dapat mendeteksi *TRAY GOOD* atau *TRAY NOT GOOD* pada aplikasi deteksi tray di industri manufaktur otomotif.

1.5. Manfaat Tugas Akhir

Berdasarkan permasalahan dan tujuan diatas, penelitian ini diharapkan dapat membantu dalam mengurangi permasalahan yang terjadi di industri manufaktur otomotif yaitu tingginya tingkat kelolosan tray NG pada sistem aplikasi deteksi tray untuk *quality control*.

1.6. Sistematika Penulisan

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

BAB I : PENDAHULUAN

Bab ini memuat latar belakang masalah, rumusan masalah, pembatasan masalah, tujuan penelitian, manfaat penelitian, serta sistematika penulisan skripsi secara menyeluruh.

BAB II : TINJAUAN PUSTAKA DAN DASAR TEORI

Bab ini berisi teori-teori dasar yang mendasari penelitian, termasuk penjelasan terkait deteksi objek, arsitektur MobileNetV2, serta kajian dari penelitian terdahulu yang relevan.

BAB III : METODE PENELITIAN

Bab ini menguraikan metode yang digunakan dalam pengembangan sistem, mulai dari tahapan perancangan sistem,

pengumpulan data, pelatihan model, pemilihan parameter, hingga proses evaluasi dan pengujian model.

BAB IV : HASIL PENELITIAN

Bab ini menyajikan hasil implementasi sistem, performa model berdasarkan pengujian yang dilakukan, serta analisis terhadap kinerja sistem berdasarkan parameter dan data yang digunakan.

BAB V : KESIMPULAN DAN SARAN

Bab ini berisi kesimpulan yang diperoleh dari hasil penelitian serta saran.

BAB II
TINJAUAN PUSTAKA DAN DASAR TEORI

2.1. Tinjauan Pustaka

Convolutional Neural Network (CNN) merupakan salah satu arsitektur dari deep learning yang dirancang khusus untuk mengolah data berbentuk citra. CNN terinspirasi dari cara kerja sistem penglihatan manusia yang mampu mengenali objek berdasarkan pola visual tertentu. Berbeda dengan jaringan saraf tiruan biasa, CNN memiliki lapisan konvolusi yang mampu mengekstraksi fitur penting dari sebuah gambar, seperti garis, tepi, maupun tekstur, secara otomatis tanpa perlu rekayasa fitur secara manual. Hal ini membuat CNN sangat efektif dalam menyelesaikan berbagai permasalahan di bidang pengolahan citra, seperti klasifikasi objek, deteksi, segmentasi, hingga pengenalan wajah.

Arsitektur CNN umumnya terdiri atas beberapa jenis lapisan utama, yaitu lapisan konvolusi (*convolution layer*), lapisan *pooling* (*subsampling*), dan lapisan *fully connected*. Pada lapisan konvolusi, filter atau kernel digunakan untuk mengekstrak ciri visual dari citra dengan menghasilkan *feature map*. Lapisan *pooling* kemudian berfungsi untuk mengurangi dimensi data sehingga mempercepat proses komputasi sekaligus mencegah *overfitting*. Sementara itu, lapisan *fully connected* digunakan untuk melakukan klasifikasi akhir berdasarkan fitur-fitur yang telah dipelajari sebelumnya. Aktivasi *non-linear* seperti ReLU (*Rectified Linear Unit*) umumnya digunakan agar jaringan mampu mempelajari representasi yang lebih kompleks.

Seiring dengan perkembangan teknologi, CNN telah menjadi standar utama dalam pengolahan citra digital, termasuk pada bidang industri manufaktur. CNN terbukti mampu mendeteksi dan mengklasifikasikan objek dengan tingkat akurasi yang tinggi dibandingkan metode konvensional berbasis *machine learning*. Dalam konteks *quality control*, CNN digunakan untuk mengidentifikasi perbedaan antara produk yang baik dan yang cacat melalui analisis citra. Keunggulan CNN terletak pada kemampuannya belajar langsung dari data gambar dalam jumlah besar

sehingga sistem dapat melakukan generalisasi terhadap berbagai variasi kondisi di lapangan.

MobileNetV2 merupakan salah satu arsitektur *Convolutional Neural Network* (CNN) yang dirancang secara khusus untuk menghasilkan model dengan performa tinggi namun tetap efisien dalam penggunaan sumber daya komputasi. Arsitektur ini dikembangkan oleh tim Google pada tahun 2018 sebagai penyempurnaan dari versi sebelumnya, yaitu MobileNetV1. Keunggulan utama MobileNetV2 terletak pada kemampuannya memproses citra dengan cepat, menggunakan parameter yang relatif sedikit, serta tetap mempertahankan akurasi yang kompetitif dibandingkan arsitektur CNN yang lebih kompleks. Hal ini menjadikan MobileNetV2 sangat sesuai digunakan pada perangkat dengan keterbatasan memori dan daya komputasi, seperti *smartphone* atau sistem *embedded* di lingkungan industri.

Secara teknis, MobileNetV2 memperkenalkan konsep *inverted residual* dan linear *bottleneck*. *Inverted residual* adalah mekanisme yang memungkinkan jaringan tetap efisien dengan menggunakan koneksi *shortcut* antara lapisan *input* dan *output*. Pendekatan ini memungkinkan propagasi gradien berjalan lebih baik sehingga pelatihan model menjadi lebih stabil. Sementara itu, *linear bottleneck* digunakan untuk menjaga informasi penting pada representasi fitur dengan menghindari fungsi aktivasi *non-linear* yang berpotensi menghilangkan informasi pada dimensi rendah. Kombinasi kedua konsep ini membuat MobileNetV2 mampu mencapai keseimbangan optimal antara kecepatan komputasi dan akurasi deteksi.

Dalam berbagai penelitian, MobileNetV2 terbukti unggul untuk tugas klasifikasi dan deteksi objek. Misalnya, penelitian oleh (Yong dkk., 2023) yang mengembangkan sistem klasifikasi sampah rumah tangga menggunakan MobileNetV2. Model ini mencapai akurasi sebesar 82,92 %, dan kinerjanya secara signifikan lebih baik sekitar 15,42 % lebih tinggi dibandingkan CNN konvensional. Keunggulan ini sangat relevan dalam konteks industri manufaktur, di mana sistem *quality control* membutuhkan model yang tidak hanya akurat, tetapi juga cepat dalam melakukan inferensi secara *real-time*. Dengan demikian,

MobileNetV2 menjadi pilihan yang tepat untuk mendeteksi kondisi *TRAY GOOD* atau *TRAY NOT GOOD* secara efisien tanpa membebani perangkat keras yang digunakan di lapangan.

Di bidang manufaktur logam, studi "MobileNetV2-AGCA" oleh (Delibasoglu dkk., 2025) fokus pada deteksi cacat permukaan baja. Model ini menggunakan perhatian graf kanal adaptif (AGCA) untuk memperkuat deteksi sambil menjaga kecepatan inferensi tinggi. Evaluasi dilakukan berdasarkan akurasi, presisi, recall, F1-score, FPS, dan FLOPS, dan hasilnya menunjukkan keseimbangan optimal antara akurasi dan efisiensi komputasi.

Dalam konteks inspeksi las di lapangan, (Ding dkk., 2022) mengupayakan penambahan modul CBAM ke MobileNetV2 untuk mendeteksi cacat las secara real-time. Model ini mencetak akurasi tinggi hingga 98,23 % dengan jumlah parameter hanya sekitar 1,4 juta jauh lebih ringan dibandingkan arsitektur seperti ResNet-50 serta sangat efisien untuk deployment di perangkat dengan keterbatasan memori.

Lebih lanjut, penelitian oleh (Jin dkk., 2023) menggunakan MobileNetV2 ditambah dengan EDCGAN untuk memperkaya dataset dalam mendeteksi defect pada permukaan baja. Model ini mencapai akurasi rata-rata 99,11 % meskipun data pelatihan terbatas, menandakan ketahanan arsitektur terhadap variasi data.

2.2. Dasar Teori

2.2.1 Manufaktur

Manufaktur merupakan aktivitas transformasi bahan baku menjadi produk jadi melalui sistem produksi yang terstruktur, melibatkan penggunaan peralatan, tenaga, dan teknologi. Seiring kemajuan era Industri 4.0, sektor manufaktur telah mengalami percepatan digitalisasi yang signifikan. Hal ini diwujudkan melalui integrasi teknologi seperti Internet of Things (IoT), robotika, big data analytics, dan kecerdasan buatan (AI) semuanya bertujuan meningkatkan efisiensi, responsivitas, dan kualitas produk. Teknologi digital ini memungkinkan manufaktur menjadi lebih adaptif terhadap perubahan, mengurangi biaya, dan mempercepat waktu produksi (Darshana dkk., 2022).

Salah satu aspek paling krusial dalam digitalisasi adalah Automated Visual Inspection (AVI) yaitu penggunaan computer vision dan AI untuk menjalankan Quality Control (QC) secara otomatis. Pendekatan ini menggantikan inspeksi manual yang rawan human error dan kurang efisien. Riset terbaru oleh Hütten et al. (2024) menemukan bahwa mayoritas model AVI dalam manufaktur berbasis Convolutional Neural Networks (CNN), yang terbukti unggul dalam klasifikasi citra dan deteksi objek. Penelitian ini juga menunjukkan adanya potensi tren baru seperti Vision Transformers dan pembelajaran tak terawasi sebagai pengembangan selanjutnya (Hütten dkk., 2024).

2.2.2 Tray

Dalam industri manufaktur, tray didefinisikan sebagai wadah atau tempat berbentuk datar yang dirancang untuk menampung, menyusun, serta memindahkan komponen, bahan baku, maupun produk jadi sepanjang proses produksi. Penggunaan tray telah menjadi standar dalam sistem material *handling* karena kemampuannya meningkatkan efisiensi, menjaga kualitas produk, serta memudahkan integrasi dengan peralatan otomatis seperti conveyor, robot *pick and place*, maupun sistem penyimpanan otomatis.

Tray memiliki beberapa fungsi utama, antara lain:

1. Penyimpanan sementara – komponen dapat ditata secara terorganisir sebelum memasuki tahap produksi berikutnya.
2. Proteksi – mencegah terjadinya kerusakan fisik, kontaminasi, atau goresan pada produk yang bernilai tinggi, seperti komponen elektronik dan farmasi.
3. Efisiensi pemindahan – memungkinkan pemindahan banyak produk sekaligus sehingga mempercepat aliran material dalam lini produksi.
4. Standarisasi proses – ukuran dan desain tray yang seragam mendukung penerapan otomasi dalam sistem manufaktur modern.

Bahan pembuat tray bervariasi sesuai kebutuhan industri. Dalam industri elektronik biasanya digunakan ESD(*Electrostatic Discharge*) tray yang terbuat dari plastik antistatis untuk melindungi komponen sensitif terhadap listrik statis. Di sisi lain, industri makanan dan farmasi banyak menggunakan tray berbahan

stainless steel atau *paperboard* karena sifatnya yang higienis dan ramah lingkungan. Untuk industri otomotif dan logam, tray umumnya dibuat dari material plastik kuat atau logam berat yang mampu menahan beban komponen besar.

Secara keseluruhan, penerapan tray dalam manufaktur bukan hanya berfungsi sebagai media penyimpanan, melainkan juga sebagai bagian integral dari sistem logistik internal, yang memengaruhi produktivitas, kualitas, hingga keberlanjutan proses produksi.

2.2.3 *Quality Control*

Quality Control (QC) merupakan proses penting dalam manufaktur yang menjamin produk sesuai standar dengan mengidentifikasi cacat secara dini. Dengan kemajuan teknologi, terutama *computer vision* dan *deep learning*, proses QC kini semakin otomatis dan efisien, menggantikan pemeriksaan manual yang rentan human error.

Survei komprehensif oleh (Hütten dkk., 2024) dalam *Deep Learning for Automated Visual Inspection in Manufacturing and Maintenance* menunjukkan bahwa mayoritas aplikasi QC otomasi berbasis citra menggunakan Convolutional Neural Networks (CNN). Teknik ini efektif dalam mendeteksi cacat visual meski dihadapkan pada variasi lingkungan dan intensitas pencahayaan yang kompleks.

Selain itu, penelitian oleh (Hahn dkk., 2023) mengusulkan sistem *predictive quality* berbasis *deep learning* untuk proses pengelasan di manufaktur, yang dapat memprediksi kualitas secara *real-time* dari data sensor menggunakan *autoencoder* dan model rekuren. Konsep ini menjanjikan pengurangan inspeksi destruktif dan meningkatkan responsivitas QC dalam proses dinamis.

Dalam konteks inspeksi visual *real-time*, ulasan oleh peneliti di (Ettalibi dkk., 2024) menyoroti integrasi *machine vision* (MV) dengan AI untuk QC. Mereka menekankan bahwa kombinasi ini memungkinkan inspeksi produk berkecepatan tinggi dengan hasil akurasi yang tinggi dalam skala industri *ScienceDirect*.

Lebih jauh, studi (Saadallah dkk., 2022) dalam *Explainable Predictive Quality Inspection* mengeksplorasi sistem QC yang tidak hanya akurat, tetapi juga dapat menjelaskan keputusan model. Aplikasi mereka dalam manufaktur elektronik mampu memberikan prediksi kualitas secara real-time, meningkatkan transparansi dan kepercayaan pengguna terhadap sistem QC *ScienceDirect*.

2.2.4 Python 3

Python 3 merupakan generasi penerus dari Python 2 yang secara resmi dirilis pada tahun 2008 dan menjadi standar utama sejak dihentikannya dukungan Python 2 pada Januari 2020. Perbedaan mendasar antara Python 2 dan Python 3 adalah penyempurnaan sintaks serta kompatibilitas dengan kebutuhan komputasi modern. Salah satu perubahan paling penting yaitu adopsi Unicode sebagai standar representasi string, sehingga Python 3 mampu menangani data multibahasa dengan lebih baik. Hal ini sangat relevan untuk pengembangan aplikasi global yang membutuhkan interoperabilitas lintas sistem.

Bahasa Python 3 dikenal memiliki sintaks yang ringkas, mudah dipelajari, namun tetap ekspresif. Keunggulan tersebut menjadikannya populer tidak hanya di kalangan akademisi, tetapi juga di industri. Studi komparatif yang dilakukan menunjukkan bahwa Python secara konsisten berada pada jajaran bahasa pemrograman yang efisien dan mudah digunakan dibandingkan dengan bahasa lain dalam berbagai domain pemrograman. Kejelasan sintaks Python 3 memungkinkan pengembang untuk menulis kode dengan lebih sedikit baris, namun tetap mempertahankan keterbacaan yang tinggi.

Selain sintaks, Python 3 didukung oleh ekosistem pustaka (*libraries*) yang sangat luas. Pustaka tersebut mencakup bidang komputasi ilmiah, kecerdasan buatan, analisis data, hingga pengembangan perangkat web. Menurut (Balaji dkk., 2021), keberadaan pustaka open-source yang kuat menjadi faktor kunci yang mendorong Python 3 digunakan secara luas dalam *data science* dan *machine learning*. Dengan dukungan komunitas global yang aktif, Python 3 terus mendapatkan pembaruan dan dokumentasi yang memadai, sehingga menambah daya tariknya dibandingkan bahasa lain.

Dari sisi paradigma, Python 3 mendukung beragam gaya pemrograman, mulai dari prosedural, fungsional, hingga berorientasi objek. Fleksibilitas ini memudahkan peneliti maupun praktisi dalam menyesuaikan pendekatan sesuai kebutuhan proyek. Selain itu, Python 3 memiliki integrasi yang baik dengan bahasa lain seperti C, C++, maupun Java, sehingga memperluas ruang lingkup penggunaannya dalam sistem berskala besar.

Popularitas Python 3 juga tercermin dari survei global, di mana bahasa ini menduduki peringkat teratas dalam indeks popularitas bahasa pemrograman. Python 3 dipandang sebagai bahasa yang ramah bagi pemula namun cukup kuat untuk pengembangan sistem kompleks. Dengan dihentikannya dukungan terhadap Python 2, sebagian besar *framework, tools*, serta aplikasi modern kini hanya menyediakan kompatibilitas penuh terhadap Python 3. Hal ini menegaskan bahwa Python 3 merupakan standar industri dan akademik yang akan terus dipertahankan dalam jangka panjang.

Dengan kombinasi sintaks yang sederhana, ekosistem pustaka yang luas, serta dukungan komunitas global yang aktif, Python 3 menjadi salah satu bahasa pemrograman yang dominan di era komputasi modern. Perannya yang besar dalam mendukung riset, industri, dan inovasi teknologi menjadikan Python 3 tidak hanya sebagai alat bantu pengembangan, tetapi juga fondasi penting dalam evolusi teknologi informasi.

2.2.5 *Convolutional Neural Network* (CNN)

Convolutional Neural Network (CNN) adalah salah satu arsitektur jaringan saraf tiruan yang dirancang khusus untuk mengolah data spasial, terutama citra digital. CNN bekerja dengan prinsip operasi konvolusi yang bertujuan mengekstraksi fitur penting dari suatu data secara otomatis. Setiap lapisan pada CNN memiliki fungsi yang berbeda, mulai dari *convolution layer* yang mendeteksi pola sederhana, *pooling layer* yang melakukan reduksi dimensi untuk mempercepat komputasi, hingga *fully connected layer* yang menghubungkan hasil ekstraksi fitur ke proses klasifikasi. Berbeda dengan metode konvensional yang membutuhkan rekayasa fitur secara manual, CNN mampu mempelajari

representasi fitur secara langsung dari data, sehingga lebih efektif dan efisien dalam pengolahan citra.

Keunggulan CNN terletak pada kemampuannya dalam membangun representasi hierarkis dari data. Pada lapisan awal, CNN mampu mengenali fitur sederhana seperti garis atau sudut, sedangkan lapisan yang lebih dalam mampu mengidentifikasi objek yang lebih kompleks. Penelitian terbaru banyak menegaskan bahwa CNN memiliki peran penting dalam *computer vision* modern. Sebagai contoh, penelitian membuktikan bahwa CNN efektif digunakan untuk mendeteksi cacat permukaan pada material logam dalam industri manufaktur dengan tingkat akurasi yang tinggi. Hal ini menegaskan bahwa CNN tidak hanya unggul pada aplikasi pengenalan wajah atau objek umum, tetapi juga relevan untuk mendukung sistem inspeksi kualitas secara otomatis dalam sektor industri.

Dengan sifatnya yang adaptif, akurat, dan mampu menangani data dalam jumlah besar, CNN menjadi salah satu teknologi inti dalam pengembangan sistem berbasis kecerdasan buatan. Arsitektur ini terus dikembangkan dalam penelitian terkini untuk mendukung berbagai aplikasi mulai dari klasifikasi citra, segmentasi objek, deteksi medis, hingga sistem *quality control* di dunia manufaktur.

Gambar 2. 1 Arsitektur *Convolutional Neural Network* (Rahma dkk., 2021)

Arsitektur CNN pada dasarnya terdiri dari beberapa lapisan utama yang bekerja secara berurutan untuk mengekstraksi dan mengklasifikasikan fitur dari data masukan, khususnya citra. Lapisan pertama yang penting adalah *convolutional layer*, di mana filter atau *kernel* digunakan untuk melakukan operasi konvolusi pada citra masukan. Proses ini bertujuan untuk mengekstraksi fitur sederhana seperti tepi, garis, dan tekstur. Selanjutnya, hasil konvolusi

biasanya dilewatkan ke *activation function* seperti *Rectified Linear Unit* (ReLU) untuk memperkenalkan sifat non-linearitas, sehingga jaringan dapat mempelajari pola yang lebih kompleks.

Setelah tahap konvolusi, arsitektur CNN umumnya menggunakan *pooling layer* yang berfungsi mengurangi dimensi data dengan cara mengambil nilai representatif, misalnya menggunakan *max pooling*. Lapisan ini membantu mempercepat proses komputasi sekaligus mengurangi risiko *overfitting*. Pada bagian akhir, CNN memiliki *fully connected layer* yang berfungsi menghubungkan hasil ekstraksi fitur ke dalam bentuk vektor dan kemudian digunakan untuk melakukan klasifikasi. Setiap neuron dalam lapisan ini terhubung penuh dengan neuron pada lapisan sebelumnya, mirip dengan jaringan saraf tiruan konvensional.

Penelitian terbaru juga menunjukkan bahwa variasi arsitektur CNN, seperti ResNet, DenseNet, dan EfficientNet, semakin meningkatkan kinerja dalam pengolahan citra. ResNet misalnya, menggunakan konsep *skip connection* untuk mengatasi masalah *vanishing gradient* pada jaringan yang sangat dalam, sementara EfficientNet menekankan pada skala yang lebih efisien antara kedalaman, lebar, dan resolusi jaringan. Dengan perkembangan ini, CNN tidak hanya menjadi solusi standar untuk klasifikasi citra, tetapi juga mendukung berbagai aplikasi seperti deteksi objek, segmentasi semantik, hingga inspeksi kualitas dalam industri manufaktur.

2.2.6 MobileNetV2

MobileNetV2 merupakan arsitektur CNN ringan yang secara khusus dikembangkan oleh *Google Research* pada tahun 2018 untuk mendukung aplikasi visi komputer pada perangkat dengan keterbatasan komputasi. Arsitektur inilah yang digunakan dalam penelitian ini karena kemampuannya mengoptimalkan *trade-off* antara akurasi dan efisiensi komputasi (Nemavhola dkk., 2025)

Yang membedakan MobileNetV2 dari pendahulunya adalah penggunaan dua mekanisme penting:

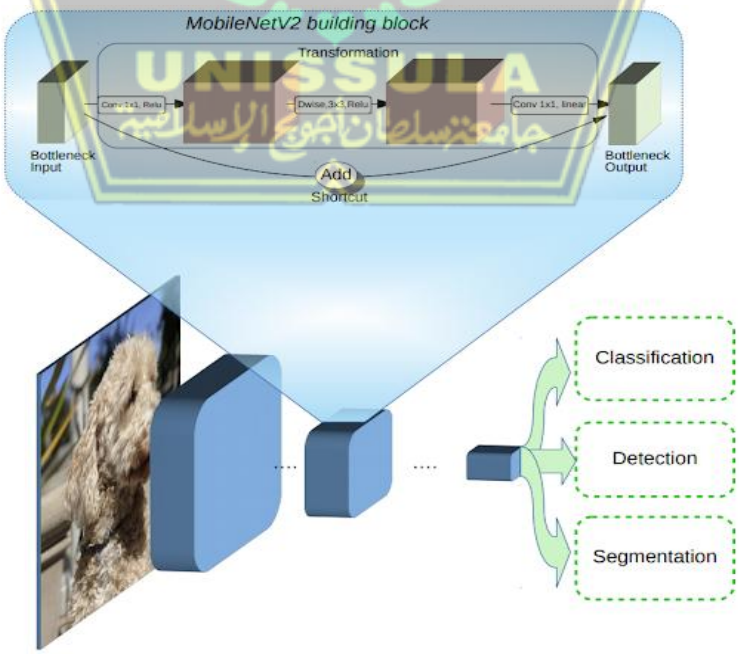
1. *Inverted Residuals*

   Berbeda dengan residual konvensional, blok *inverted residual* meningkatkan efisiensi dengan cara mengekspansi terlebih dahulu ruang fitur (dengan *pointwise convolution*), lalu memprosesnya dengan *depthwise convolution*, kemudian memproyeksikan kembali ke dimensi asli. Struktur ini memungkinkan propagasi informasi tetap optimal sembari menekan biaya komputasi. Hal ini dijelaskan secara rinci oleh studi di *Sensors* (2024) yang menyajikan mekanisme ini dalam konteks CNN efisien.

2. *Linear Bottleneck Layers*

   MobileNetV2 menggunakan lapisan projeksi tanpa aktivasi *non-linear* (seperti ReLU) pada tahap *bottleneck*. Strategi ini penting karena penerapan *non-linear* pada ruang dimensi rendah dapat menghilangkan informasi penting. Penghilangan aktivasi *non-linear* pada tahap ini terbukti menjaga representasi fitur secara efektif.

Secara keseluruhan, MobileNetV2 sering dibangun atas dasar blok-blok ini yang disusun secara berulang, dengan jumlah blok residu mencapai 19 lapisan setelah lapisan konvolusi awal.



Gambar 2. 2 Arsitektur Mobilenetv2

MobileNetV2 merupakan pengembangan dari MobileNet generasi pertama yang didesain untuk menghadirkan efisiensi komputasi tinggi dengan tetap mempertahankan akurasi pada tugas pengenalan visual. Gambar di atas memperlihatkan blok dasar *MobileNetV2 building block* yang terdiri dari beberapa tahap pemrosesan utama.

Blok inti arsitektur ini disebut *Inverted Residual Block*. Pada tahap awal, dilakukan ekspansi dimensi melalui konvolusi 1x1 (*pointwise convolution*) yang menggunakan fungsi aktivasi ReLU6 untuk memperbesar representasi fitur. Selanjutnya, fitur yang diperluas melewati lapisan *depthwise separable convolution* 3x3 dengan aktivasi ReLU6. Teknik ini memisahkan konvolusi spasial dan kanal sehingga secara signifikan mengurangi jumlah parameter dan beban komputasi. Setelah itu, dilakukan *projection layer* menggunakan konvolusi 1x1 linear untuk mengembalikan dimensi fitur ke bentuk lebih kecil (bottleneck).

Ciri khas MobileNetV2 dibandingkan arsitektur sebelumnya adalah penggunaan *shortcut connection* (*skip connection*). Jalur *shortcut* ini memungkinkan informasi dari *input bottleneck* untuk ditambahkan kembali ke *output*, sehingga meminimalkan kehilangan informasi (*information loss*) dan memperbaiki aliran gradien pada proses pelatihan.

Seperti yang tergambar, blok dasar ini dapat ditumpuk berulang kali untuk menghasilkan representasi fitur yang lebih kompleks. Hasil ekstraksi fitur kemudian dapat digunakan untuk berbagai tugas visi komputer, seperti klasifikasi citra, deteksi objek, maupun segmentasi semantik. Dengan struktur tersebut, MobileNetV2 sangat cocok untuk aplikasi berbasis perangkat dengan keterbatasan sumber daya (misalnya *smartphone* atau IoT) karena menawarkan keseimbangan antara efisiensi komputasi dan akurasi.

BAB III
METODE PENELITIAN

3.1 Deskripsi Sistem

Aplikasi yang dikembangkan merupakan sistem deteksi yang memanfaatkan algoritma MobileNetV2, dengan antarmuka pengguna berbasis Flask. Sistem ini dirancang untuk menerima *input* tangkapan gambar dari kamera *index*, yang kemudian diproses dalam bentuk gambar.

Setiap gambar yang diterima akan melalui tahapan seperti pengubahan ukuran dan normalisasi gambar, agar sesuai dengan format *input* yang dibutuhkan oleh model deteksi. Lalu, proses klasifikasi akan dilakukan oleh MobileNetV2 untuk mengidentifikasi tray.

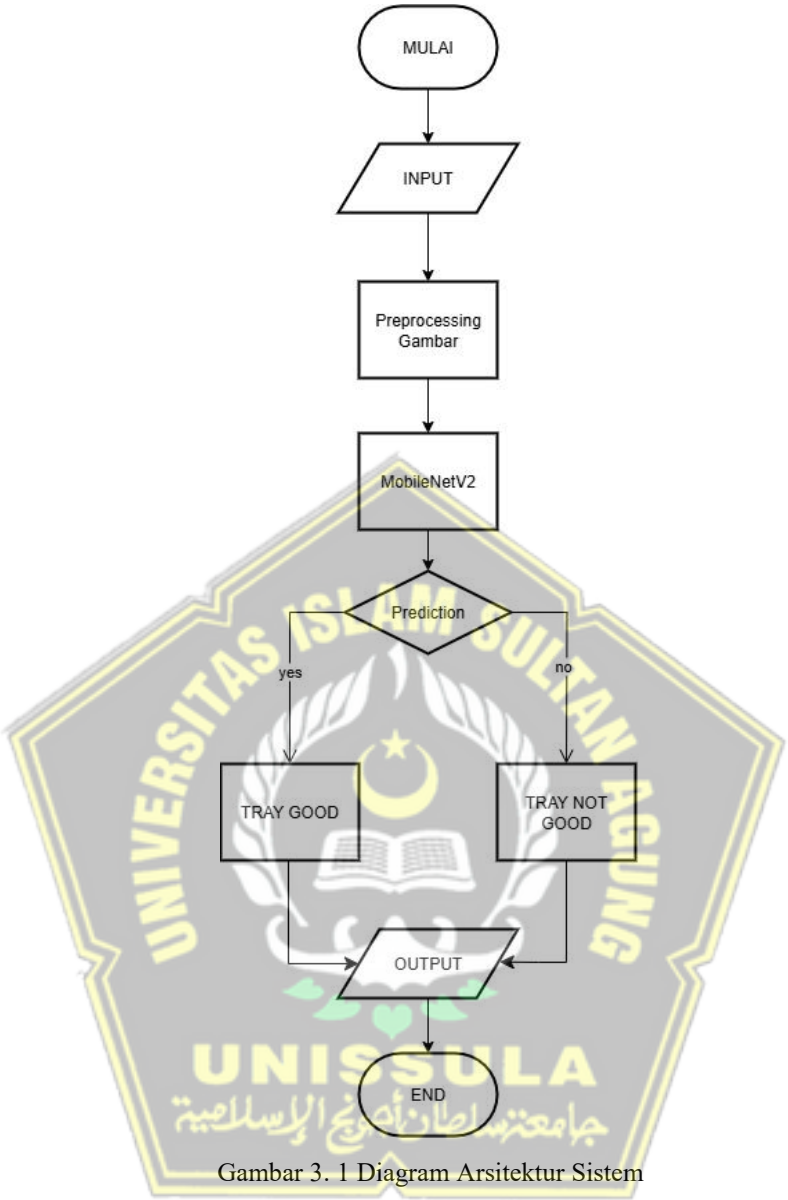
Setelah diidentifikasi, hasil dari deteksi model akan ditampilkan berupa gambar dan informasi dari hasil klasifikasinya, divisualisasikan melalui antarmuka Flask dalam tampilan web.

3.2 Studi Literatur

Penelitian ini diawali dengan kajian terhadap berbagai sumber literatur, baik dari jurnal, skripsi, maupun referensi daring. Yang diambil dari studi kasus di industri manufaktur otomotif. Tujuannya adalah memahami dan mempelajari teori serta pendekatan teknis yang mendasari metode deteksi menggunakan Arsitektur MobileNetV2.

3.3 Arsitektur Sistem

Pada penelitian ini, penulis akan merancang dan membangun sebuah sistem klasifikasi guna mendeteksi kelolosan tray dengan memanfaatkan Algoritma MobileNetV2 sebagai inti dari proses deteksi. Sistem ini dirancang menggunakan arsitektur yang sederhana namun efektif, di mana setiap komponen memiliki peran spesifik dalam mendukung keseluruhan proses deteksi.



Gambar 3. 1 Diagram Arsitektur Sistem

Flowchart pada Gambar 3.1 menunjukkan alur kerja sistem deteksi kualitas tray berbasis *deep learning* dengan menggunakan arsitektur MobileNetV2. Alur ini menjelaskan tahapan yang dilalui mulai dari proses *input* citra hingga keluaran berupa keputusan klasifikasi kualitas tray.

Tahapan perancangan sistem dapat dijelaskan sebagai berikut:

1. Mulai (*Start*)

   Proses diawali dengan inisialisasi sistem yang siap menerima masukan berupa citra tray dari jalur produksi.

2. *Input*

   Data masukan berupa citra tray diperoleh melalui kamera yang terpasang pada lini produksi atau dari dataset uji. *Input* ini menjadi objek utama yang akan dianalisis kualitasnya.

3. *Preprocessing* Gambar

   Citra yang diterima tidak langsung dimasukkan ke dalam model, melainkan diproses terlebih dahulu pada tahap *preprocessing*. Proses ini meliputi:

   - Perubahan ukuran (*resize*) citra ke resolusi tertentu agar sesuai dengan standar input MobileNetV2 (misalnya 224×224 piksel).
   - Normalisasi nilai piksel agar berada pada rentang 0–1 sehingga memudahkan proses perhitungan pada jaringan saraf.
   - Augmentasi data (opsional), seperti rotasi, *flipping*, atau penyesuaian pencahayaan, untuk meningkatkan kemampuan generalisasi model terhadap variasi data.

4. MobileNetV2

   Setelah melalui tahap *preprocessing*, citra dimasukkan ke dalam arsitektur MobileNetV2. Pada tahap ini, model melakukan ekstraksi fitur dan klasifikasi dengan memanfaatkan blok *inverted residual* serta *linear bottleneck* yang menjadi ciri khas arsitektur ini. MobileNetV2 dipilih karena memiliki keunggulan dalam efisiensi komputasi sekaligus tetap mampu menghasilkan akurasi yang kompetitif.

5. *Prediction* (Klasifikasi)

   Berdasarkan hasil perhitungan jaringan, model menghasilkan prediksi berupa kelas citra. Pada sistem ini terdapat dua kemungkinan keluaran, yaitu:

   - *Tray Good*, apabila citra tray teridentifikasi sesuai standar kualitas produksi.
   - *Tray Not Good*, apabila citra tray dinyatakan cacat atau tidak memenuhi kriteria kualitas.

6. *Output*

   Hasil klasifikasi kemudian ditampilkan dalam bentuk keluaran sistem. Informasi ini dapat digunakan oleh operator untuk melakukan tindakan lanjut,

seperti memisahkan tray cacat dari jalur produksi, menyimpan data inspeksi, atau memberikan notifikasi otomatis.

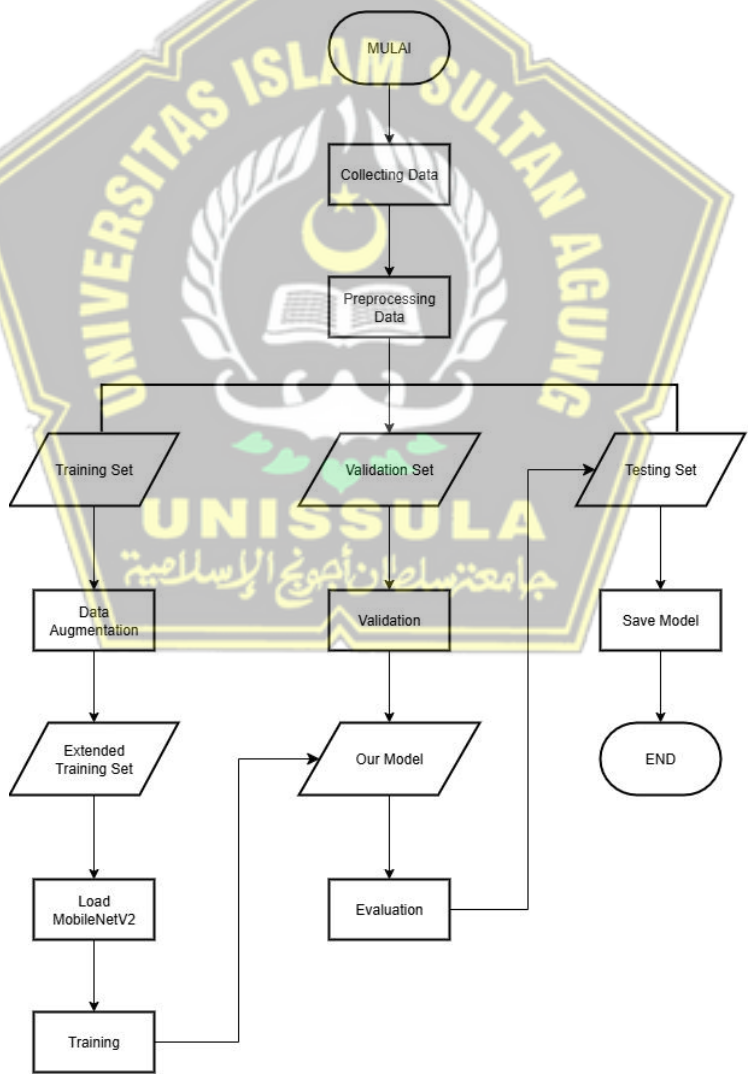
7. Selesai (*End*)

   Proses berakhir setelah hasil klasifikasi ditampilkan. Selanjutnya sistem akan kembali ke kondisi awal untuk memproses *input* berikutnya.

3.4 Perancangan Model

3.4.1 Alur Perancangan Model

Pada tahap perancangan model, peneliti akan membuat alur model dalam bentuk *flowchart* atau diagram alur sebagai berikut:



Gambar 3. 2 Diagram Perancangan Model

Flowchart pada Gambar 3.2 menunjukkan alur perancangan model yang diawali dengan proses pengumpulan data (*collecting* data) yang diperoleh dari *dataset* sesuai dengan kebutuhan penelitian. Setelah data terkumpul, dilakukan tahap *preprocessing* untuk menyiapkan data agar sesuai dengan format *input* model. Proses *preprocessing* meliputi normalisasi citra, pengubahan ukuran citra sesuai standar arsitektur yang digunakan, serta pembersihan data agar tidak terdapat duplikasi maupun data yang tidak relevan.

Selanjutnya, *dataset* dibagi menjadi tiga bagian, yaitu data latih (*training set*), data validasi (*validation set*), dan data uji (*testing set*). Data latih digunakan untuk melatih model, sedangkan data validasi berfungsi untuk memantau kinerja model selama proses pelatihan sehingga dapat menghindari terjadinya *overfitting*. Sementara itu, data uji digunakan untuk mengevaluasi performa model setelah pelatihan selesai.

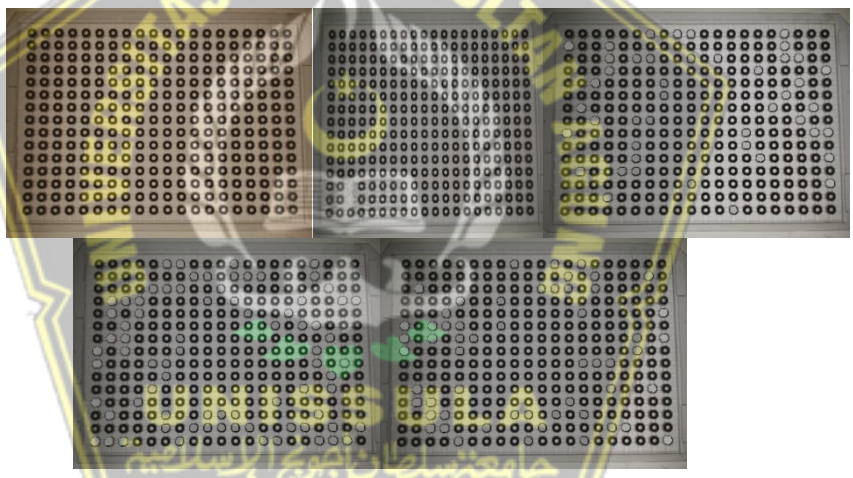
Untuk meningkatkan variasi data, dilakukan data *augmentation* pada *training* set. Teknik ini bertujuan agar model dapat belajar lebih baik terhadap berbagai kemungkinan variasi citra, sehingga menghasilkan performa klasifikasi yang lebih stabil.

Model yang digunakan dalam penelitian ini adalah MobileNetV2, yaitu arsitektur *Convolutional Neural Network (CNN)* yang dikenal ringan dan efisien untuk diterapkan pada perangkat dengan keterbatasan komputasi. Proses pelatihan dilakukan menggunakan data latih yang telah melalui augmentasi, sedangkan proses validasi dilakukan dengan memanfaatkan data validasi untuk menilai akurasi model pada tiap *epoch*.

Tahap selanjutnya adalah evaluasi model, yaitu mengukur performa MobileNetV2 menggunakan data uji dengan metrik evaluasi seperti akurasi, presisi, *recall*, dan F1-*score*. Apabila model telah memenuhi kriteria performa yang diharapkan, maka model terbaik disimpan dan dapat digunakan untuk implementasi lebih lanjut pada sistem klasifikasi citra.

3.4.2 *Collecting Data*

Pada penelitian ini, data yang digunakan berupa citra tray yang diperoleh langsung dari industri manufaktur otomotif. Tray berfungsi sebagai wadah atau tempat penyusunan komponen kecil berbentuk cincin(*ring*) berdiameter kecil yang merupakan salah satu bagian penting dalam perakitan mesin otomotif, sebelum diproses lebih lanjut untuk dilakukan pemeriksaan kualitas dan pengemasan produk. Keberadaan tray ini mempermudah proses *quality control* (QC) karena setiap komponen dapat ditata dengan rapi, sehingga lebih mudah diamati. Dengan demikian, tray tidak hanya berperan sebagai tempat penyimpanan sementara, tetapi juga menjadi bagian penting dalam mendukung akurasi proses inspeksi.



Gambar 3. 3 *Dataset*

Citra tray yang digunakan dalam penelitian ini dikumpulkan secara langsung di area produksi, sehingga mencerminkan kondisi nyata di lapangan. Setiap tray difoto dalam keadaan berisi komponen untuk memperlihatkan susunan serta kondisi aktualnya, baik dalam keadaan normal maupun ketika terdapat penyimpangan. Jumlah keseluruhan citra yang berhasil dikumpulkan mencapai 500 gambar, yang kemudian digunakan sebagai dataset utama dalam proses pelatihan dan pengujian model. Adanya variasi kondisi pada *dataset* ini diharapkan mampu meningkatkan kemampuan model dalam membedakan tray yang baik dan tray yang tidak sesuai standar.

Pada Gambar 3.3 menunjukkan hasil pengumpulan data yang diperoleh. Pada Gambar kiri atas terlihat tray dengan kondisi *good*, di mana seluruh slot terisi secara merata dan tidak terdapat cacat atau kekosongan. Pola lingkaran yang terbentuk dari setiap objek tampak konsisten dan seragam baik dari segi ukuran, posisi, maupun intensitas pencahayaan. Hal ini menandakan bahwa proses produksi atau penempatan objek pada tray berjalan sesuai standar.

Sebaliknya, pada Gambar kanan dan sisanya tray dikategorikan sebagai *not good*. Terlihat adanya ketidaksesuaian di mana beberapa slot kosong dan tidak berisi objek sebagaimana mestinya. Kekosongan tersebut menandakan adanya kesalahan dalam proses pengisian tray atau adanya *defect* pada produk. Selain itu, meskipun sebagian besar pola masih konsisten, keberadaan slot kosong menyebabkan tray secara keseluruhan tidak memenuhi kriteria kualitas yang diharapkan.

Perbedaan antara tray *good* dan *not good* ini menjadi dasar penting dalam implementasi sistem deteksi otomatis berbasis visi komputer. Dengan model *deep learning* atau algoritma pengolahan citra, sistem dapat mengklasifikasikan kondisi tray secara *real-time* sehingga mengurangi potensi *human error* dalam proses inspeksi manual.

3.4.3 Data *PreProcessing*

Tahap awal dalam *preprocessing* gambar adalah *resizing*. Proses ini bertujuan untuk menyeragamkan dimensi atau resolusi seluruh gambar dalam dataset, yang awalnya memiliki resolusi berbeda-beda, sehingga diubah menjadi ukuran yang konsisten pada setiap gambar. Tahap *resizing* ini juga merupakan kebutuhan dari Arsitektur MobileNetV2, karena arsitektur ini memiliki *default input size* dengan ukuran input 224x224x3 piksel, demi mencapai kompatibilitas, efisiensi, dan performa optimal pada proses pelatihan model.

Kemudian dilakukan split data yang merupakan proses pembagian data yang akan dibagi menjadi tiga yaitu data *training*, data *testing*, dan data *validation*.

3.4.4 Augmentasi Data

Pada penelitian ini terdapat beberapa augmentasi data yang dilakukan pada dataset yaitu *rotation, saturation, brightness, exposure,* dan menambahkan efek *noise*.

Rotasi dilakukan untuk membantu model dalam pendeteksian ketika objek deteksi mengalami perubahan orientasi. Mengatur kecerahan dan *exposure* pada gambar dilakukan untuk membantu model dalam mendeteksi objek ketika terjadi perubahan pencahayaan. Menambahkan efek *noise* pada gambar guna membantu model dalam mendeteksi objek dengan kondisi kamera yang digunakan untuk dalam pendeteksian mengalami *noise.*

3.4.5 *Training*

Pada tahap ini, peneliti melatih model deteksi dengan memanfaatkan Arsitektur MobileNetV2. *Dataset* yang digunakan akan diambil dari data yang sudah dikumpulkan dan sudah diaugmentasi sebelumnya.

Proses pelatihan direncanakan berjalan secara online menggunakan GPU *Google Colab*, namun agar kecepatan komputasi lebih optimal diperlukan jaringan dengan kecepatan yang maksimal. Adapun konfigurasi *hyperparameter* pelatihan yang digunakan dalam proses ini dijelaskan sebagai berikut:

- *Number of Epochs* (epochs)

Epochs merupakan jumlah siklus penuh saat seluruh data pelatihan diproses oleh model. Satu *epoch* berarti seluruh *dataset* telah digunakan satu kali dalam proses pelatihan. Jumlah *epoch* yang lebih tinggi memungkinkan model belajar lebih dalam, namun dapat meningkatkan risiko *overfitting.*

- *Image Size* (imgsz)

Gambar yang dimasukkan ke dalam model biasanya disesuaikan ke dalam ukuran tertentu agar konsisten. Penyesuaian ukuran ini bertujuan untuk mengurangi beban komputasi sekaligus memastikan *input* kompatibel dengan arsitektur model.

- *Batch Size* (batch)

Jumlah data yang diproses secara bersamaan dalam satu iterasi pelatihan disebut *Bacth size.* Pemilihan *batch size* sendiri memengaruhi performa komputasi, semakin kecil *batch size* maka cenderung lebih stabil namun lambat, sebaliknya akan menjadi lebih cepat namun membutuhkan memori lebih banyak.

- *Optimizer Type* (optimizer)

Optimizer adalah algoritma yang digunakan untuk memperbarui bobot model berdasarkan hasil evaluasi terhadap data pelatihan. Jenis *optimizer* yang umum digunakan antara lain SGD (*Stochastic Gradient Descent*), Adam, dan *auto.*

- *Early Stopping* (patience)

Digunakan sebagai parameter *early stopping*. Jika performa model tidak mengalami peningkatan selama 5 *epoch* berturut-turut, maka proses pelatihan akan dihentikan lebih awal untuk menghindari pemborosan sumber daya.

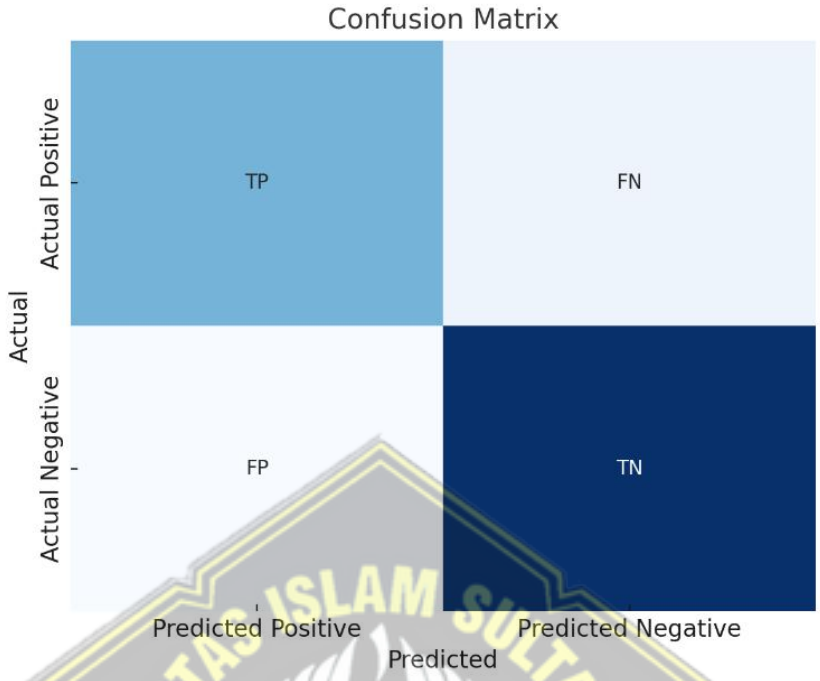
- *Dropout Rate* (dropout)

Dropout merupakan metode regularisasi yang digunakan untuk mencegah *overfitting*. Teknik ini bekerja dengan cara mengabaikan (drop) sebagian neuron secara acak selama pelatihan, sehingga model tidak terlalu bergantung pada fitur tertentu.

3.4.6 Evaluasi Model

Pada penelitian ini, evaluasi model dilakukan menggunakan metrik evaluasi klasifikasi biner, yaitu akurasi, *precision, recall*, dan F1-*score*. Model *Convolutional Neural Network* (CNN) yang telah dilatih digunakan untuk memprediksi kelayakan *tray* berdasarkan citra yang diinputkan.

Evaluasi dilakukan dengan menggunakan *confusion matrix*, yang terdiri dari empat komponen utama:



Gambar 3. 4 *Confusion Matrix*

- *True Positive* (TP): Tray yang benar-benar layak dan berhasil diprediksi sebagai layak.
- *True Negative* (TN): Tray yang tidak layak dan berhasil diprediksi sebagai tidak layak.
- *False Positive* (FP): Tray yang tidak layak tetapi salah diprediksi sebagai layak.
- *False Negative* (FN): Tray yang layak tetapi salah diprediksi sebagai tidak layak.

Berdasarkan nilai-nilai tersebut, beberapa metrik evaluasi dapat dihitung:

1. Akurasi (*Accuracy*)

   Mengukur seberapa sering model membuat prediksi yang benar secara keseluruhan.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.1}$$

2. Presisi (*Precision*)

   Mengukur ketepatan model dalam memprediksi tray yang layak, yaitu seberapa banyak prediksi "layak" yang benar.

$$Precision = \frac{TP}{TP + FP} \tag{3.2}$$

3. *Recall* (Sensitivitas)

   Mengukur seberapa baik model dapat menemukan semua tray yang benar-benar layak.

$$Recall = \frac{TP}{TP + FN} \tag{3.3}$$

4. F1-*Score*

   Merupakan harmonik rata-rata dari *precision* dan *recall*, digunakan untuk menyeimbangkan keduanya.

$$F1\text{-}Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{3.4}$$

Dalam konteks *Quality Control* (QC) pada industri manufaktur, nilai *False Positive* (FP) menjadi perhatian utama. Hal ini karena FP menunjukkan kondisi di mana *tray* yang seharusnya ditolak justru lolos pemeriksaan sistem. Kesalahan ini dapat berdampak langsung terhadap kualitas produk pada tahap produksi berikutnya. Oleh karena itu, selain memperhatikan akurasi, metrik *precision* dan *recall* sangat penting untuk memastikan bahwa sistem deteksi mampu meminimalisasi kesalahan tersebut.

Nilai *confidence* pada klasifikasi *multi-class* akan dihitung berdasarkan hasil penelitian dengan model yang menggunakan fungsi *softmax* untuk mengubah skor keluaran menjadi distribusi probabilitas pada setiap kelas. *Softmax* didefinisikan sebagai:

$$P(y = i|x) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{3.5}$$

Di mana z_i merupakan keluaran untuk kelas ke-i dan K adalah jumlah total kelas. Nilai probabilitas yang dihasilkan menunjukkan tingkat keyakinan model terhadap masing-masing kelas, dan kelas dengan probabilitas terbesar dipilih sebagai prediksi akhir . Misalnya, jika sebuah gambar dievaluasi oleh model dengan hasil *softmax* [0.1][0.7][0.2], maka prediksi jatuh pada kelas kedua dengan *condifence* sebesar 70%.

BAB IV
HASIL DAN ANALISIS PENELITIAN

4.1 Hasil dan Analisis

4.1.1 Proses Pelatihan Model

Proses pelatihan model pada penelitian ini menggunakan Arsitektur MobileNetV2. *Dataset* yang digunakan telah diambil dari industri manufaktur otomotif dikumpulkan dan disesuaikan dengan kebutuhan pelatihan. Seluruh pelatihan dilakukan dengan bahasa pemrograman Python menggunakan GPU untuk mempercepat proses komputasi.

Jumlah *epoch* yang digunakan sebanyak 100, dengan ukuran gambar yang sudah diubah sebelumnya menjadi 224x224x3 *pixel.* Berikut ini merupakan konfigurasi *hyperparameter* yang penulis gunakan untuk proses pelatihan model:

```
inputs = pretrained_model.input
x = resize_and_rescale(inputs)
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(2, activation='softmax')(x)

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

model.compile(
    optimizer=Adam(0.01)/(0.001)/(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=100,
    callbacks=[
        early_stopping,
        create_tensorboard_callback("training_logs","part_classifi
cation"),
        checkpoint_callback,
    ]
)
```

Tabel 4. 1 *List Hyperparameter*

Hyperparameter	Value	Penjelasan
Dense	256, activation='relu'	256: jumlah neuron di hidden layer. Semakin besar, semakin banyak pola kompleks yang bisa dipelajari, tapi juga rawan overfitting. relu (*Rectified Linear Unit*): aktivasi populer untuk *hidden layer*, mempercepat *training* dan mengurangi masalah *vanishing gradient*.
Dropout	0.2	Dropout adalah teknik regularisasi. 0.2 artinya 20% *neuron* akan dimatikan secara acak tiap iterasi untuk mencegah *overfitting*.
Dense	2, activation='softmax'	2: jumlah output class (*binary classification*). *softmax*: mengubah output menjadi probabilitas (total = 1).
optimizer	Adam(0.001)	Adam: algoritma optimasi yang adaptif. 0.001 (*learning rate*): seberapa besar langkah update bobot setiap iterasi. Lebih besar → belajar cepat tapi bisa tidak stabil.

		Lebih kecil → stabil tapi training lama.
loss	'categorical_crossentropy'	*Loss function* untuk klasifikasi *multi-class* dengan *one-hot encoding*. Cocok karena output menggunakan 2 kelas dengan *softmax*.
metrics	['accuracy']	Untuk *monitoring* akurasi saat *training* dan *validation*.
steps_per_epoch	len(train_images)	Banyaknya batch yang diproses dalam 1 *epoch*. len(train_images) berarti 1 batch = 1 data.
validation_data	val_images	*Dataset* untuk evaluasi model setiap *epoch*, untuk mengecek *overfitting*.
validation_steps	len(val_images)	Sama seperti steps_per_epoch, tetapi untuk *validation set*.
epochs	100	Maksimal siklus *training* (seluruh data diproses ulang). Bisa berhenti lebih cepat jika ada early_stopping.
callbacks	early_stopping, create_tensorboard_callback	early_stopping: berhenti otomatis jika loss/val_loss tidak membaik setelah beberapa *epoch* (hemat waktu & hindari *overfitting*). create_tensorboard_callback: menyimpan *log* ke TensorBoard untuk melihat grafik loss/accuracy.

4.1.2 Perbandingan Penggunaan *Learning Rate* pada *Optimizer*

Untuk mengevaluasi pengaruh jenis algoritma optimasi terhadap performa pelatihan model, peneliti melakukan serangkaian eksperimen dengan menggunakan *optimizer* Adam serta menerapkan tiga nilai *learning rate* yang berbeda, yaitu 0.0001, 0.001, dan 0.01. Tujuan dari variasi ini adalah untuk

mengamati bagaimana perubahan nilai *learning rate* memengaruhi kecepatan konvergensi, stabilitas pelatihan, serta akurasi model. Pengujian dilakukan secara sistematis guna menentukan konfigurasi yang paling optimal dalam meningkatkan performa model CNN dalam mendeteksi kelayakan tray berdasarkan citra.

4.1.3 Hasil Evaluasi Model

Eksperimen dilakukan dengan menggunakan tiga nilai *learning rate* pada *optimizer* Adam(AdamW), yaitu 0.0001, 0.001, 0.01. untuk mengetahui pengaruhnya terhadap performa model MobileNetV2 dalam mendeteksi tray.

Seluruh proses pelatihan dilakukan dengan parameter yang sama, di antaranya: *epoch* sebanyak 100, ukuran gambar 224x224x3 *pixel*. Hanya dibedakan pada *learning rate* yang digunakan, yaitu 0.0001, 0.001, 0.01.

Tabel 4. 2 Hasil Evaluasi Model

Learning Rate	Precision	Recall	F1-Score
0.0001	1.00	1.00	1.00
0.001	1.00	1.00	1.00
0.01	1.00	1.00	1.00

Tabel 4.1 menunjukkan bahwa *learning rate* 0.0001 memberikan hasil terbaik secara umum. Sementara *learning rate* 0.01 menghasilkan nilai terendah pada semua metrik, yang menunjukkan bahwa *learning rate* ini kurang cocok untuk digunakan tanpa penyetelan lanjutan.

4.1.4 Visualisasi Evaluasi Model

Grafik dan Confusion Matrix di bawah ini menggambarkan perbandingan metrik performa model berdasarkan nilai optimizer yang digunakan.

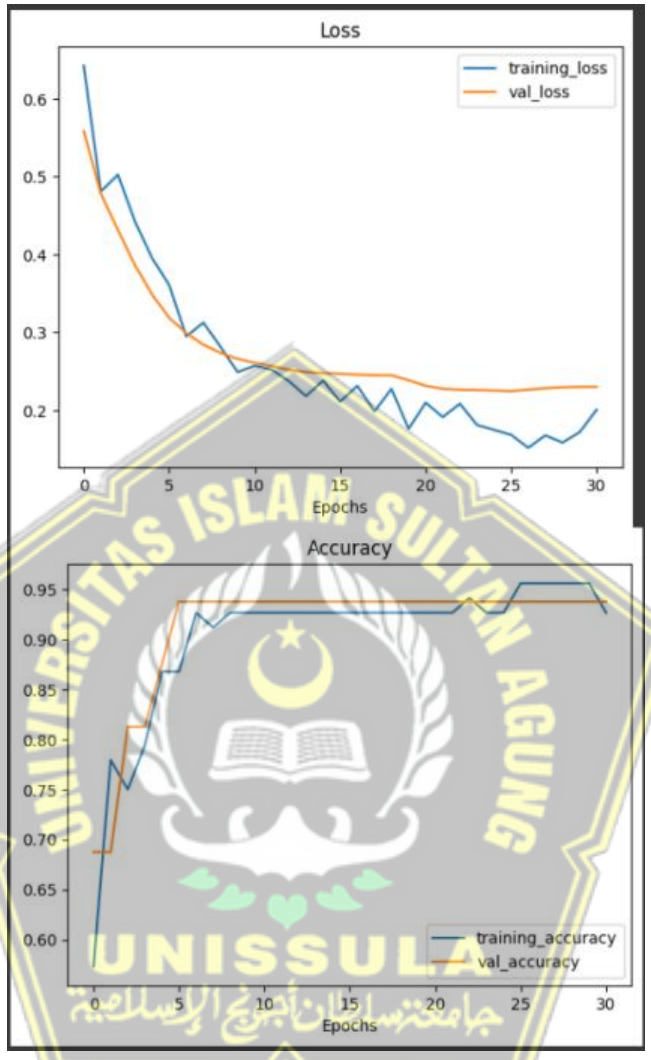
1. *Learning Rate* 0.0001

Berikut merupakan konfigurasi pelatihan yang digunakan:

```
# Compile model
model.compile(optimizer=Adam(0.0001), loss='binary_crossentropy', metrics=['accuracy'])
```

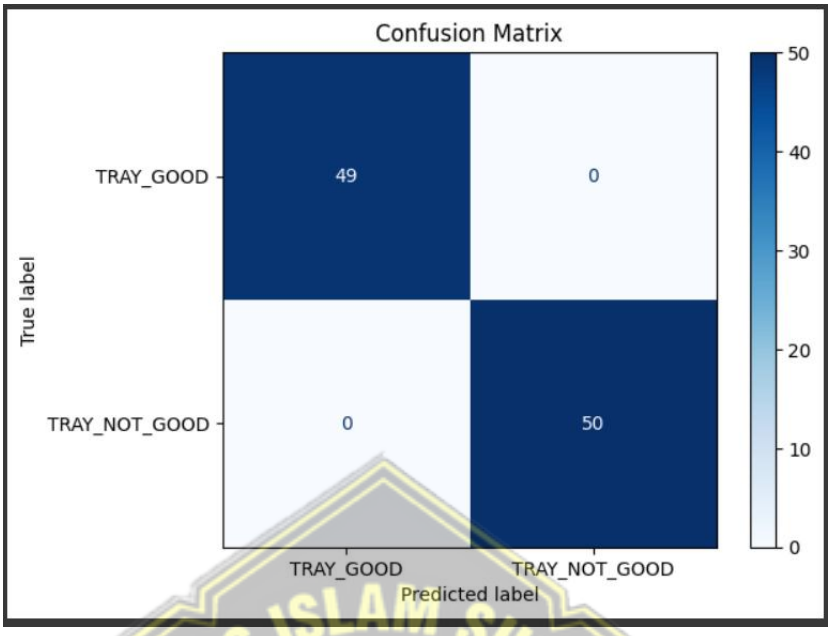
Gambar 4. 1 Konfigurasi Parameter *Learning Rate* 0.0001

Pada Gambar 4.1 parameter *learning rate* yang digunakan pada konfigurasi ini pada baris *optimizer*=Adam(0.0001).



Gambar 4. 2 Grafik Hasil *Learning Rate* 0.0001

Grafik pada Gambar 4.2 menunjukkan bahwa proses pelatihan model dengan *learning rate* 0.0001 sangat stabil dengan grafik *training loss* dan *validation loss* yang secara konsisten menurun dari awal hingga akhir pelatihan. Proses penurunan *loss* terjadi secara perlahan namun mantap, yang merupakan karakteristik umum dari *learning rate* rendah. Meskipun membutuhkan waktu lebih lama untuk mencapai akurasi optimal, stabilitas yang tinggi pada grafik menunjukkan bahwa model mampu menghindari fluktuasi besar dan mengurangi risiko *overfitting*.



Gambar 4. 3 *Confusion Matrix Learning Rate* 0.0001

Perhatikan Gambar 4.2, nilai *training accuracy* 91.90% dan *validation accuracy* 93.75% menunjukkan konsistensi yang baik, di mana kedua kurva saling mendekati dan tidak menunjukkan gejala divergensi. Dari hasil evaluasi Gambar 4.3, *confusion matrix* menunjukkan klasifikasi sempurna untuk kedua kelas (TRAY_GOOD dan TRAY_NOT_GOOD), dengan nilai *precision, recall*, dan *f1-score* sebesar 1.0. Ini menandakan bahwa walaupun lambat, *learning rate* rendah mampu melatih model dengan baik selama jumlah *epoch* yang memadai.

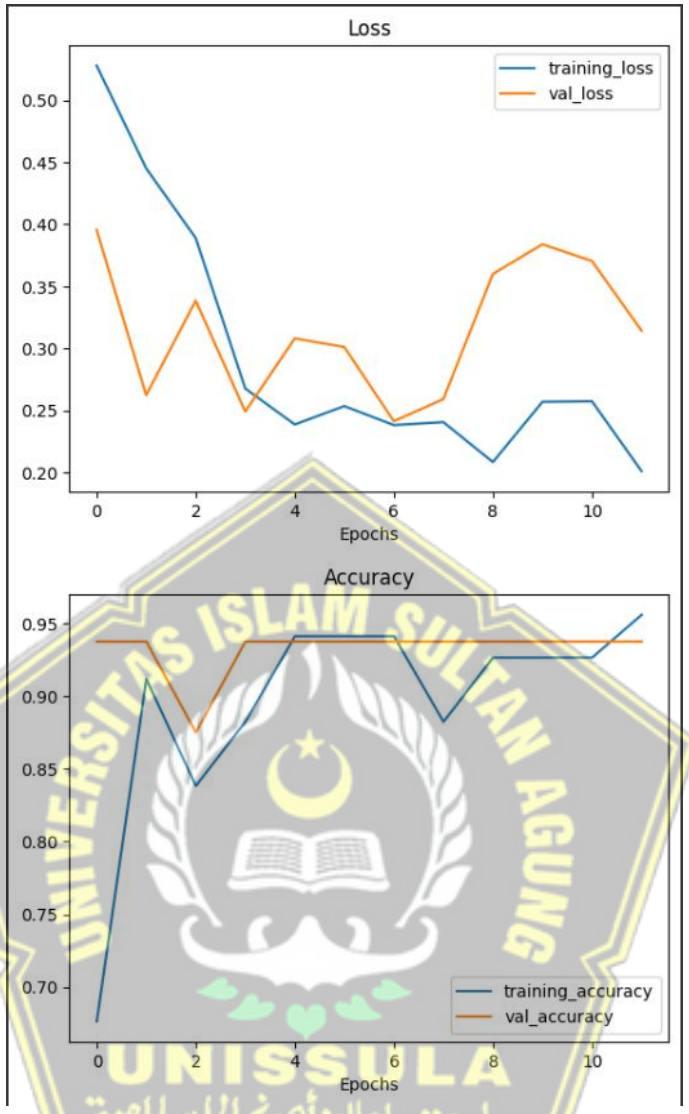
2. *Learning Rate* 0.001

Berikut merupakan konfigurasi pelatihan yang digunakan:

```
# Compile model
model.compile(optimizer=Adam(0.001), loss='binary_crossentropy', metrics=['accuracy'])
```

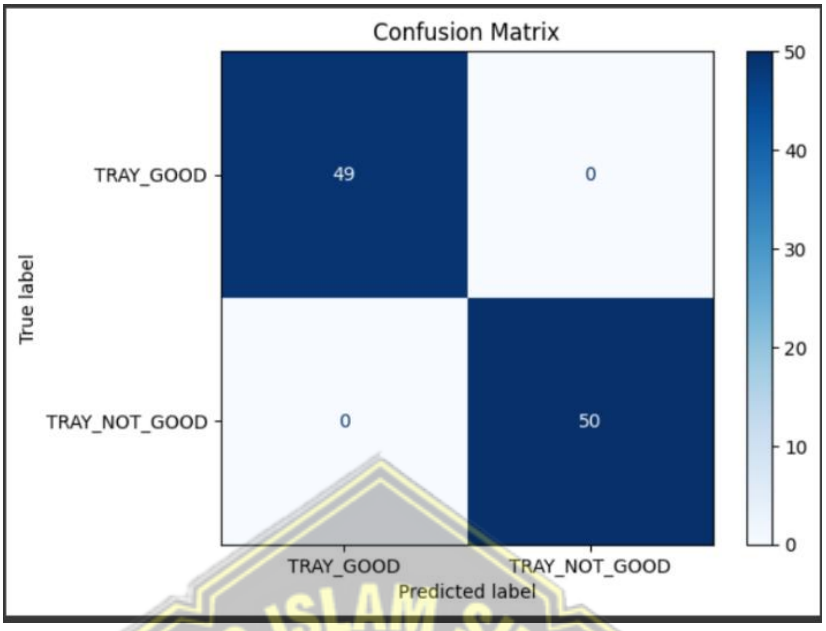
Gambar 4. 4 Konfigurasi Parameter *Learning Rate* 0.001

Pada gambar 4.4 parameter *learning rate* yang digunakan pada konfigurasi ini pada baris *optimizer*=Adam(0.001).



Gambar 4. 5 Grafik Hasil *Learning Rate* 0.001

Model dengan *learning rate* 0.001 menunjukkan hasil yang sangat kompetitif, perhatikan Gambar 4.5. Kurva *loss* mengalami penurunan lebih cepat dibanding model sebelumnya, menandakan bahwa pembaruan bobot yang dilakukan lebih efisien dalam menemukan jalur menuju minimum fungsi *loss*. Setelah beberapa *epoch* awal, grafik *loss* menjadi stabil dan menunjukkan konvergensi yang baik. *Accuracy* baik pada data pelatihan yang mencapai 95.67% maupun validasi yang mencapai nilai 93.57% lebih cepat dibanding model dengan *learning rate* 0.0001.



Gambar 4. 6 *Confusion Matrix Learning Rate* 0.001

Hasil evaluasi juga mengindikasikan bahwa model ini mampu menjaga keseimbangan antara kecepatan belajar dan stabilitas. Tidak terdapat *overfitting* yang signifikan, ditunjukkan dari keselarasan antara *training accuracy* dan *validation accuracy*. Hasil metrik pada Gambar 4.6, model ini memberikan hasil klasifikasi sempurna seperti model sebelumnya, namun dengan efisiensi pelatihan yang lebih tinggi. Berdasarkan hal tersebut, *learning rate* 0.001 dapat dianggap sebagai nilai optimal dalam konteks dataset dan arsitektur yang digunakan.

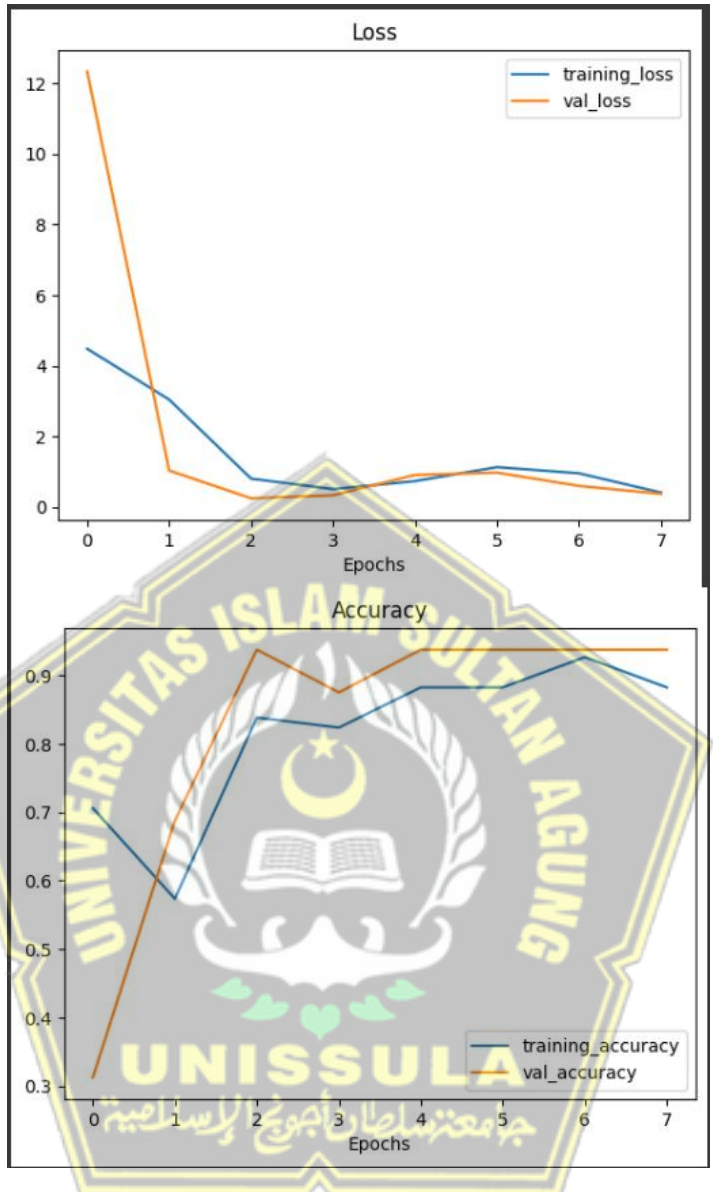
3. *Learning Rate* 0.01

Berikut merupakan konfigurasi pelatihan yang digunakan:

```
# Compile model
model.compile(optimizer=Adam(0.01), loss='binary_crossentropy', metrics=['accuracy'])
```

Gambar 4. 7 Konfigurasi Parameter *Learning Rate* 0.01

Pada Gambar 4.7 menunjukkan parameter *learning rate* yang digunakan pada konfigurasi ini pada baris *optimizer*=Adam(0.01).

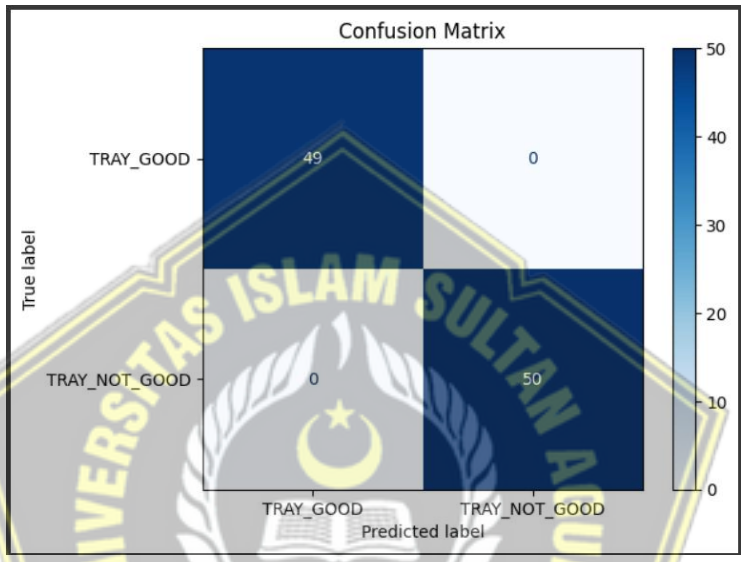


Gambar 4. 8 Grafik Hasil *Learning Rate* 0.01

Berbeda dari dua model sebelumnya, model dengan *learning rate* 0.01 menunjukkan dinamika pelatihan yang tidak stabil, perhatikan Gambar 4.8 . Hal ini terlihat dari fluktuasi tajam pada grafik *validation loss*, terutama pada *epoch-epoch* awal. Meskipun *training loss* menurun secara relatif stabil, *validation loss* beberapa kali mengalami kenaikan yang signifikan. Fluktuasi tersebut menandakan bahwa pembaruan bobot terlalu agresif, menyebabkan model kesulitan beradaptasi terhadap pola data validasi secara konsisten.

Kondisi ini umum terjadi ketika nilai *learning rate* terlalu tinggi, sehingga model cenderung melompati minimum lokal atau bahkan global dari fungsi *loss*.

Akibatnya, alih-alih mencapai titik konvergensi, model dapat terus berpindah-pindah posisi pada *landscape error surface*. Meskipun pada akhirnya akurasi validasi tetap tinggi dengan nilai *training accuracy* mencapai 88.21% dan *validation accuracy* mencapai 93.75% model memberikan hasil evaluasi sempurna, jalannya proses pelatihan menjadi tidak dapat diprediksi dan lebih rentan terhadap gejala *overfitting* maupun *underfitting* tergantung variasi data.



Gambar 4. 9 *Confusion Matrix Learning Rate* 0.01

Stabilitas pelatihan sangat penting dalam konteks pengembangan model untuk *deployment* jangka panjang. Fluktuasi tinggi pada *loss* atau ketidakteraturan dalam kurva akurasi bisa berakibat fatal bila diterapkan pada data dunia nyata yang bervariasi. Oleh karena itu, seperti pada Gambar 4.9 model ini tetap mampu mengklasifikasikan data uji secara sempurna, pendekatan dengan *learning rate* tinggi seperti 0.01 sebaiknya digunakan dengan hati-hati saat pelatihan.

4.1.5 Kesimpulan Pemilihan *Learning Rate* pada *Optimizer*

Berdasarkan hasil evaluasi performa model menggunakan tiga perbandingan *learning rate,* yaitu 0.0001, 0.001, 0.01. Dapat disimpulkan bahwa *learning rate* 0.001 memberikan hasil yang paling optimal dalam pelatihan model MobileNetV2 untuk klasifikasi gambar .

Model dengan *learning rate* 0.0001 sangat stabil namun membutuhkan lebih banyak *epoch* untuk konvergensi. Model dengan *learning rate* 0.001 menjadi pilihan terbaik karena menggabungkan kecepatan konvergensi dan stabilitas

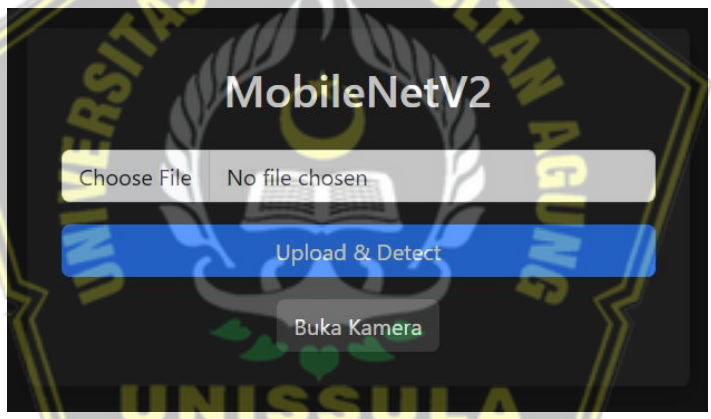
pelatihan dengan tingkat akurasi deteksi yang lebih tinggi dibanding *learning rate* yang lain.

Dengan mempertimbangkan hasil dari seluruh metrik evaluasi, maka dapat disimpulkan bahwa *optimizer* Adam(AdamW) dengan *learning rate* 0.001 merupakan pilihan terbaik untuk digunakan dalam model deteksi kelayakan tray menggunakan MobileNetV2, baik dari sisi akurasi, efisiensi pelatihan, maupun kestabilan proses pembelajaran.

4.2 Hasil Implementasi Aplikasi

4.2.1 Tampilan *User Interface*

Gambar di bawah ini menunjukkan tampilan antarmuka dari aplikasi deteksi tray yang dibangun menggunakan model MobileNetV2.

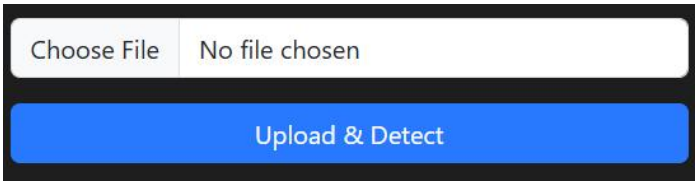


Gambar 4. 10 *User Interface* Aplikasi

Antarmuka pengguna (*user interface*) dari aplikasi deteksi tray berbasis MobileNetV2 dirancang menggunakan *framework* flask, yang memungkinkan pengembangan aplikasi berbasis web secara cepat dan interaktif. Gambar 4.10 menunjukkan tampilan utama aplikasi ini, yang menunjukkan area tampilan utama (main content area) di tengah layar.

Berikut adalah penjelasan dari masing-masing elemen antarmuka:

1. Upload & Detect Button



Gambar 4. 11 *Upload & Detect Button*

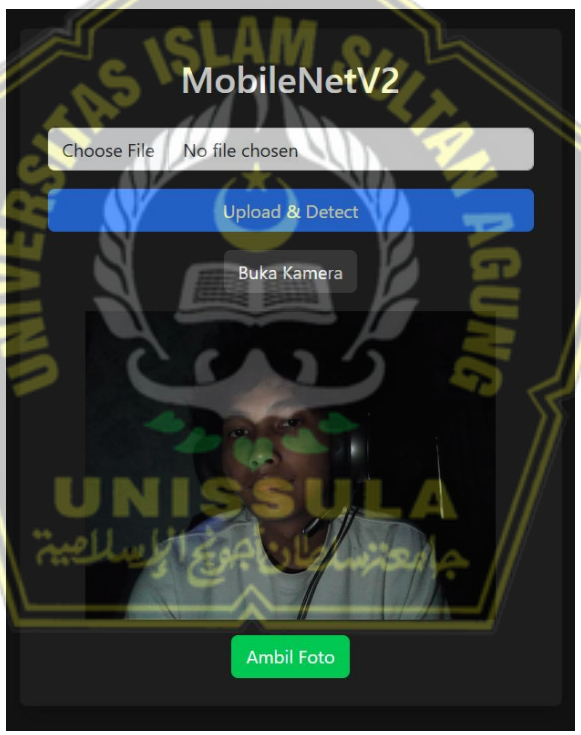
Fitur pada Gambar 4.11 ini berfungsi untuk mengunggah file gambar atau tray yang akan diupload dan dideteksi oleh model MobileNetV2.

2. Tombol Buka Kamera

Gambar 4. 12 Tombol Buka Kamera

Fitur pada Gambar 4.12 ini berfungsi untuk mengaktifkan kamera dan memunculkan frame video di bawah tombol.

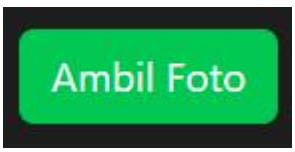
3. Tampilan Frame Video



Gambar 4. 13 Frame Video

Fitur pada Gambar 4.13 ini berfungsi untuk menampilkan gambar asli dari kamera atau video tanpa anotasi.

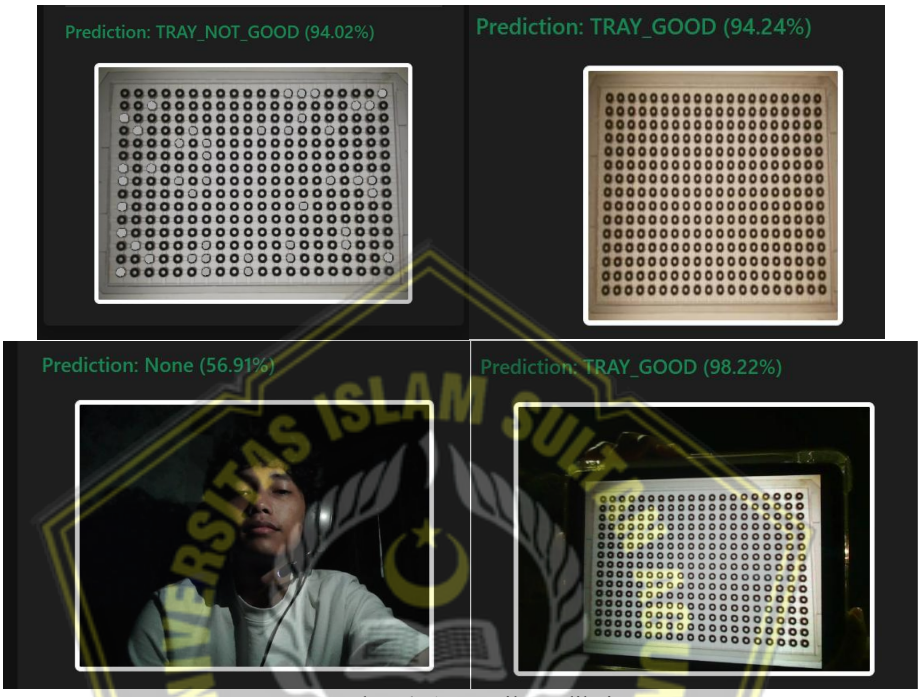
4. Tombol Ambil Gambar



Gambar 4. 14 *Button* Ambil Gambar

Fitur pada Gambar 4.14 ini berfungsi untuk menangkap gambar dari *frame video*. Agar hasil tangkapan gambar dapat diproses oleh model MobileNetV2.

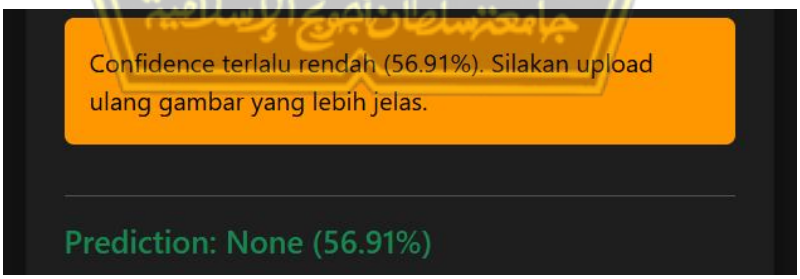
5. Hasil Prediksi



Gambar 4. 15 Hasil Prediksi

Pada Gambar 4.15, bagian ini memungkinkan untuk memunculkan gambar hasil dari tangkapan gambar dan hasil klasifikasi deteksi.

6. Warning



Gambar 4. 16 *Warning*

Seperti pada Gambar 4.16, akan muncul *popup* warning, jika tingkat *confidence* di bawah angka 70%. Sehingga dianjurkan pada user untuk melakukan *input* ulang.

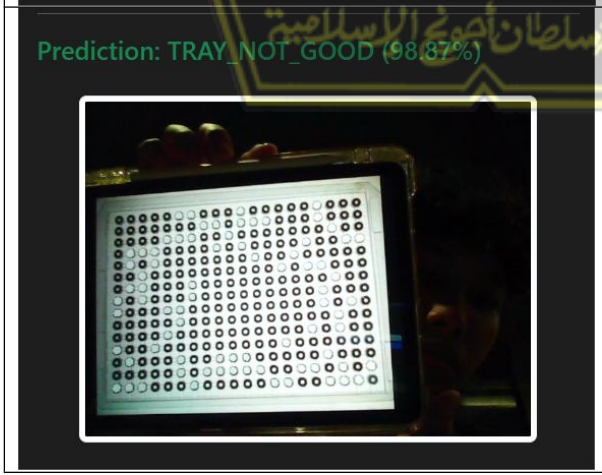
4.2.2 Pengujian Aplikasi

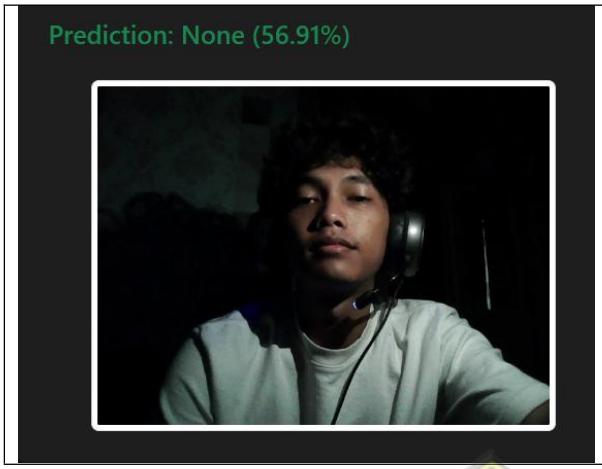
Pengujian aplikasi deteksi tray dilakukan dengan menggunakan 2 jenis sumber input gambar, yaitu tangkap gambar menggunakan kamera *index* dan *upload file* gambar. Pengujian bertujuan untuk mengetahui efektivitas dan akurasi deteksi tray oleh model MobileNetV2 dalam berbagai kondisi *input*.

1. Pengujian Menggunakan Kamera *Index*

Pada pengujian pertama dilakukan menggunakan Kamera *index* bawaan laptop.

Tabel 4. 3 Pengujian Menggunakan Kamera *Index*

Gambar	Hasil Prediksi	*Confidence*
	Berhasil	98.22
	Berhasil	98.87

	None	56.91

Hasil pada Tabel 4.2 menunjukkan bahwa hasil dapat bervariasi berdasarkan kondisi kamera saat proses pengambilan gambar. Pada pengujian kamera peneliti mencoba mengambil gambar dengan kondisi pencahayaan yang berbeda dan model mampu mendeteksi dan mengklasifikasikan gambar dengan baik.

Pengujian juga dilakukan dengan mengambil gambar selain tray sehingga menghasilkan prediksi "None", di mana model tidak mengenali *input* dari gambar tersebut.

2. Pengujian Menggunakan *Upload File*

Pada pengujian kedua, *input* berasal dari *file* gambar yang diunggah ke dalam aplikasi.

Tabel 4. 4 Pengujian Menggunakan *Upload File*

Gambar	Hasil Prediksi	*Confidence*
	Berhasil	95.05

	Berhasil	94.02

Hasil deteksi pada Tabel 4.3 menunjukkan bahwa sistem mampu mengklasifikasi setiap gambar dengan benar dan memperoleh nilai *confidence* yang tinggi. Hal ini menunjukkan bahwa model tetap mampu mendeteksi gambar dari *input upload file* dengan tepat.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan, dapat disimpulkan bahwa sistem deteksi tray menggunakan metode *Convolutional Neural Network* (CNN) dengan arsitektur MobileNetV2 berhasil dibangun dan diimplementasikan dengan baik. Sistem ini mampu mengklasifikasikan tray menjadi dua kategori, yaitu *TRAY GOOD* dan *TRAY NOT GOOD*, dengan tingkat rata-rata akurasi yang tinggi yaitu 96.54% yang diambil dari nilai pengujian dua kategori itu saja. Melalui pengujian menggunakan berbagai *input* gambar, baik dari pengambilan gambar dengan kamera maupun *file* gambar yang diunggah, model menunjukkan performa yang sangat baik dengan nilai *precision, recall*, dan *f1-score* sebesar 1.00. Dengan demikian, sistem ini dapat membantu mengurangi tingkat kelolosan tray NG pada proses *Quality Control* di industri manufaktur otomotif.

5.2 Saran

Untuk pengembangan lebih lanjut, berikut beberapa saran yang dapat dipertimbangkan:

1. Perluasan *dataset* baik dari segi jumlah maupun variasi kondisi pencahayaan, sudut pengambilan gambar, dan jenis tray agar model lebih stabil dalam menghadapi data dunia nyata.
2. Integrasi ke sistem produksi secara otomatis, misalnya dengan menghubungkan kamera ke lini produksi untuk mendeteksi tray secara *real-time* tanpa perlu interaksi manual.
3. Penambahan fitur validasi *confidence*, seperti pengaturan ambang batas yang adaptif atau peringatan otomatis jika *confidence* model terlalu rendah, guna meminimalisir kesalahan klasifikasi.
4. Evaluasi jangka panjang di lingkungan industri, guna memastikan kestabilan performa model saat diterapkan secara terus-menerus dan dalam skala besar.

DAFTAR PUSTAKA

Balaji, S., Abhishek K. Nanavate, Lakshmi Pathi Jakkamputi, N. Raghukiran, and R. Jegadeeshwaran. 2021. "Dynamic Characterisation of Honeycomb CNT Reinforced Sandwich Composite Structure." *Journal of Physics: Conference Series* 1969 (1). https://doi.org/10.1088/1742-6596/1969/1/012016.

Darshana, Nagare, Raundal Amol, and Patil Devesh. 2022. "International Journal of Research Publication and Reviews Review on " Hazards in Manufacturing Industry "" 3 (2): 793–801.

Delibasoglu, Ibrahim, Duygu Bagci Das, and Oguzhan Das. 2025. "Defect Attention-Based Lightweight Real-Time Adaptable Surface Defect Analysis." *Ironmaking & Steelmaking: Processes, Products and Applications*. https://doi.org/10.1177/03019233241296959.

Ding, Kai, Zhangqi Niu, Jizhuang Hui, Xueliang Zhou, and Felix T.S. Chan. 2022. "A Weld Surface Defect Recognition Method Based on Improved MobileNetV2 Algorithm." *Mathematics* 10 (19). https://doi.org/10.3390/math10193678.

Ettalibi, Abdelfatah, Abdelmajid Elouadi, and Abdeljebar Mansour. 2024. "AI and Computer Vision-Based Real-Time Quality Control: A Review of Industrial Applications." *Procedia Computer Science* 231 (2023): 212–20. https://doi.org/10.1016/j.procs.2023.12.195.

Hahn, Yannik, Robert Maack, Guido Buchholz, Marion Purrio, Matthias Angerhausen, Hasan Tercan, and Tobias Meisen. 2023. "Towards a Deep Learning-Based Online Quality Prediction System for Welding Processes." *Procedia CIRP* 120:1047–52. https://doi.org/10.1016/j.procir.2023.09.123.

Hütten, Nils, Miguel Alves Gomes, Florian Hölken, Karlo Andricevic, Richard Meyes, and Tobias Meisen. 2024. "Deep Learning for Automated Visual Inspection in Manufacturing and Maintenance: A Survey of Open- Access Papers." *Applied System Innovation* 7 (1). https://doi.org/10.3390/asi7010011.

Jin, Ge, Yanghe Liu, Peiliang Qin, Rongjing Hong, Tingting Xu, and Guoyu Lu. 2023. “An End-to-End Steel Surface Classification Approach Based on EDCGAN and MobileNet V2.” *Sensors* 23 (4). https://doi.org/10.3390/s23041953.

Nemavhola, Andisani, Colin Chibaya, and Serestina Viriri. 2025. “A Systematic Review of CNN Architectures, Databases, Performance Metrics, and Applications in Face Recognition.” *Information (Switzerland)* 16 (2): 1–41. https://doi.org/10.3390/info16020107.

Rahma, Lusiana, Hadi Syaputra, A.Haidar Mirza, and Susan Dian Purnamasari. 2021. “Objek Deteksi Makanan Khas Palembang Menggunakan Algoritma YOLO (You Only Look Once).” *Jurnal Nasional Ilmu Komputer* 2 (3): 213–32. https://doi.org/10.47747/jurnalnik.v2i3.534.

Saadallah, Amal, Jan Büscher, Omar Abdulaaty, Thorben Panusch, Jochen Deuse, and Katharina Morik. 2022. “Explainable Predictive Quality Inspection Using Deep Learning in Electronics Manufacturing.” *Procedia CIRP* 107 (March): 594–99. https://doi.org/10.1016/j.procir.2022.05.031.

Yong, Liying, Le Ma, Dandan Sun, and Liping Du. 2023. “Application of MobileNetV2 to Waste Classification.” *PLoS ONE* 18 (3 March): 1–10. https://doi.org/10.1371/journal.pone.0282336.